

Mohammed J. Zaki
Jeffrey Xu Yu
B. Ravindran
Vikram Pudi (Eds.)

LNAI 6118

Advances in Knowledge Discovery and Data Mining

14th Pacific-Asia Conference, PAKDD 2010
Hyderabad, India, June 2010
Proceedings, Part I

1
Part I

 Springer

Lecture Notes in Artificial Intelligence 6118

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

Subseries of Lecture Notes in Computer Science

Mohammed J. Zaki Jeffrey Xu Yu
B. Ravindran Vikram Pudi (Eds.)

Advances in Knowledge Discovery and Data Mining

14th Pacific-Asia Conference, PAKDD 2010
Hyderabad, India, June 21-24, 2010
Proceedings
Part I

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

Mohammed J. Zaki
Rensselaer Polytechnic Institute
Troy, NY, USA
E-mail: zaki@cs.rpi.edu

Jeffrey Xu Yu
The Chinese University of Hong Kong
Hong Kong, China
E-mail: yu@se.cuhk.edu.hk

B. Ravindran
IIT Madras, Chennai, India
E-mail: ravi@cse.iitm.ac.in

Vikram Pudi
IIIT, Hyderabad, India
E-mail: vikram@iiit.ac.in

Library of Congress Control Number: 2010928262

CR Subject Classification (1998): I.2, H.3, H.4, H.2.8, I.4, C.2

LNCS Sublibrary: SL 7 – Artificial Intelligence

ISSN 0302-9743
ISBN-10 3-642-13656-7 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-13656-6 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 14th Pacific-Asia Conference on Knowledge Discovery and Data Mining was held in Hyderabad, India during June 21–24, 2010; this was the first time the conference was held in India.

PAKDD is a major international conference in the areas of data mining (DM) and knowledge discovery in databases (KDD). It provides an international forum for researchers and industry practitioners to share their new ideas, original research results and practical development experiences from all KDD-related areas including data mining, data warehousing, machine learning, databases, statistics, knowledge acquisition and automatic scientific discovery, data visualization, causal induction and knowledge-based systems.

PAKDD-2010 received 412 research papers from over 34 countries including: Australia, Austria, Belgium, Canada, China, Cuba, Egypt, Finland, France, Germany, Greece, Hong Kong, India, Iran, Italy, Japan, S. Korea, Malaysia, Mexico, The Netherlands, New Caledonia, New Zealand, San Marino, Singapore, Slovenia, Spain, Switzerland, Taiwan, Thailand, Tunisia, Turkey, UK, USA, and Vietnam. This clearly reflects the truly international stature of the PAKDD conference.

After an initial screening of the papers by the Program Committee Chairs, for papers that did not conform to the submission guidelines or that were deemed not worthy of further reviews, 60 papers were rejected with a brief explanation for the decision. The remaining 352 papers were rigorously reviewed by at least three reviewers. The initial results were discussed among the reviewers and finally judged by the Program Committee Chairs. In some cases of conflict additional reviews were sought. As a result of the deliberation process, only 42 papers (10.2%) were accepted as long presentations (25 mins), and an additional 55 papers (13.3%) were accepted as short presentations (15 mins). The total acceptance rate was thus about 23.5% across both categories.

The PAKDD 2010 conference program also included seven workshops: Workshop on Data Mining for Healthcare Management (DMHM 2010), Pacific Asia Workshop on Intelligence and Security Informatics (PAISI 2010), Workshop on Feature Selection in Data Mining (FSDM 2010), Workshop on Emerging Research Trends in Vehicle Health Management (VHM 2010), Workshop on Behavior Informatics (BI 2010), Workshop on Data Mining and Knowledge Discovery for e-Governance (DMEG 2010), Workshop on Knowledge Discovery for Rural Systems (KDRS 2010).

The conference would not have been successful without the support of the Program Committee members (164), external reviewers (195), Conference Organizing Committee members, invited speakers, authors, tutorial presenters, workshop organizers, reviewers, authors and the conference attendees. We highly appreciate the conscientious reviews provided by the Program Committee

members, and external reviewers. The Program Committee members were matched with the papers using the SubSift system (<http://subsift.ilrt.bris.ac.uk/>) for bid matching; we thank Simon Price and Peter Flach, of Bristol University, for developing this wonderful system. Thanks also to Andrei Voronkov for hosting the entire PAKDD reviewing process on the easychair.org site.

We are indebted to the members of the PAKDD Steering Committee for their invaluable suggestions and support throughout the organization process. We thank Vikram Pudi (Publication Chair), Pabitra Mitra (Workshops Chair), Kamal Karlapalem (Tutorials Chair), and Arnab Bhattacharya (Publicity Chair). Special thanks to the Local Arrangements Committee and Chair R.K. Bagga, and the General Chairs: Jaideep Srivastava, Masaru Kitsuregawa, and P. Krishna Reddy. We would also like to thank all those who contributed to the success of PAKDD 2010 but whose names may not be listed.

We greatly appreciate the support from various institutions. The conference was organized by IIIT Hyderabad. It was sponsored by the Office of Naval Research Global (ONRG) and the Air Force Office of Scientific Research/Asian Office of Aerospace Research and Development (AFOSR/AOARD).

We hope you enjoy the proceedings of the PAKDD conference, which presents cutting edge research in data mining and knowledge discovery. We also hope all participants took this opportunity to share and exchange ideas with each other and enjoyed the cultural and social attractions of the wonderful city of Hyderabad!

June 2010

Mohammed J. Zaki
Jeffrey Xu Yu
B. Ravindran

Program Committee

Osman Abul
Muhammad Abulaish
Arun Agarwal
Hisham Al-Mubaid
Reda Alhajj
Hiroki Arimura
Hideo Bannai
Jayanta Basak
M.M. Sufyan Beg
Bettina Berendt
Fernando Berzal
Raj Bhatnagar
Vasudha Bhatnagar
Arnab Bhattacharya
Pushpak Bhattacharyya
Chiranjib Bhattacharyya
Vivek Borkar
Keith C.C. Chan
Longbing Cao
Doina Caragea
Venkatesan Chakaravarthy
Vineet Chaoji
Sanjay Chawla
Arbee Chen
Phoebe Chen
Jake Yue Chen
Zheng Chen
Hong Cheng
James Cheng
Alok Choudhary
Diane Cook
Alfredo Cuzzocrea
Sanmay Das
Anne Denton
Lipika Dey
Guozhu Dong
Petros Drineas
Tina Eliassi-Rad
Wei Fan
Eibe Frank
Benjamin C.M. Fung
Sachin Garg
Mohamed Gaber

Joao Gama
Jean-Gabriel Ganascia
Gemma Garriga
Ravi Gupta
Mohammad Hasan
Tu Bao Ho
Vasant Honavar
Wynne Hsu
Xiaohua Hu
Akihiro Inokuchi
Shen Jialie
Hasan Jamil
Daxin Jiang
Ruoming Jin
Bo Jin
Hiroyuki Kawano
Tamer Kahveci
Toshihiro Kamishima
Ben Kao
Panagiotis Karras
Hisashi Kashima
Yiping Ke
Latifur Khan
Hiroyuki Kitagawa
Ravi Kothari
Mehmet Koyuturk
Bharadwaja Kumar
Wai Lam
Chung-Hong Lee
Xue Li
Jinyan Li
Tao Li
Chun-hung Li
Ee-Peng Lim
Chih-Jen Lin
Guimei Liu
Tie-Yan Liu
Changtien Lu
Vasilis Megalooikonomou
Tao Mei
Wagner Meira Jr.
Rosa Meo
Dunja Mladenec

Yang-Sae Moon
 Yasuhiko Morimoto
 Tsuyoshi Murata
 Atsuyoshi Nakamura
 J. Saketha Nath
 Juggapong Natwichai
 Richi Nayak
 Wilfred Ng
 Mitsunori Ogihara
 Salvatore Orlando
 Satoshi Oyama
 Prabin Panigrahi
 Spiros Papadimitriou
 Srinivasan Parthasarathy
 Wen-Chih Peng
 Bernhard Pfahringer
 Srinivasa Raghavan
 R. Rajesh
 Naren Ramakrishnan
 Ganesh Ramakrishnan
 Jan Ramon
 Sanjay Ranka
 Rajeev Rastogi
 Chandan Reddy
 Patricia Riddle
 S. Raju Bapi
 Saeed Salem
 Sudeshna Sarkar
 Tamas Sarlos
 C. Chandra Sekhar
 Srinivasan Sengamedu
 Shirish Shevade
 M. Shimbo

Ambuj K. Singh
 Mingli Song
 P. Sreenivasa Kumar
 Aixin Sun
 S. Sundararajan
 Ashish Sureka
 Domenico Talia
 Kay Chen Tan
 Ah-Hwee Tan
 Pang-Ning Tan
 David Taniar
 Ashish Tendulkar
 Thanaruk Theeramunkong
 W. Ivor Tsang
 Vincent S. Tseng
 Tomoyuki Uchida
 Lipo Wang
 Jason Wang
 Jianyong Wang
 Graham Williams
 Xintao Wu
 Xindong Wu
 Meng Xiaofeng
 Hui Xiong
 Seiji Yamada
 Jiong Yang
 Dit-Yan Yeung
 Tetsuya Yoshida
 Aidong Zhang
 Zhongfei (Mark) Zhang
 Zhi-Hua Zhou
 Chengqi Zhang

External Reviewers

Abdul Nizar
 Abhinav Mishra
 Alessandra Raffaeta
 Aminul Islam
 Andrea Tagarelli
 Anitha Varghese
 Ankit Agrawal
 Anuj Mahajan
 Anupam Bhattacharjee

Atul Saroop
 Blaz Novak
 Brian Ruttenberg
 Bum-Soo Kim
 Carlo Mastroianni
 Carlos Ferreira
 Carmela Comito
 Cha Lun Li
 Chandra Sekhar Chellu

Chao Luo
Chen-Yi Lin
Chih jui Lin Wang
Chuancong Gao
Chun Kit Chui
Chun Wei Seah
Chun-Hao Chen
Chung An Yeh
Claudio Silvestri
Da Jun Li
David Uthus
De-Chuan Zhan
Delia Rusu
Dhruv Mahajan
Di Wang
Dinesh Garg
Elena Ikonomovska
En Tzu Wang
Eugenio Cesario
Feilong Chen
Feng Chen
Ferhat Ay
Gokhan Yavas
Gongqing Wu
Hea-Suk Kim
Hideyuki Kawashima
Hui Zhu Su
Ilija Subasic
Indranil Palit
Jan Rupnik
Janez Brank
Jeyashanker Ramamirtham
Jitendra Ajmera
Junjie Wu
Kathy Macropol
Khalil Al-Hussaeni
Kong Wah Wan
Krishna Prasad Chitrapura
Kunpeng Zhang
Kyle Chipman
L. Venkata Subramaniam
Lang Huo
Lei Liu
Lei Shi
Lei Yang

Leting Wu
Li Zheng
Li An
Li Da Jun
Lili Jiang
Ling Guo
Linhong Zhu
Lixin Duan
Lorand Dali
Luka Bradesko
Luming Zhang
Manas Somaiya
Mark Beltramo
Markus Ojala
Masayuki Okabe
Miao Qiao
Michael Barnathan
Min-Ling Zhang
Mingkui Tan
Mitja Trampus
Mohammed Aziz
Mohammed Imran
Muad Abu-Ata
Nagaraj Kota
Nagarajan Krishnamurthy
Narasimha Murty Musti
Narayan Bhamidipati
Ngoc Khanh Pham
Ngoc Tu Le
Nicholas Larusso
Nidhi Raj
Nikolaj Tatti
Ning Ruan
Nirmalya Bandyopadhyay
Nithi Gupta
Noman Mohammed
Palvali Teja
Pannagadatta Shivaswamy
Paolo Trunfio
Parthasarathy Krishnaswamy
Pedro P. Rodrigues
Peipei Li
Prahladavaradan Sampath
Prakash Mandayam Comare
Prasad Deshpande

Prithviraj Sen
Pruet Boonma
Qi Mao
Qiang Wang
Qingyan Yang
Quan Yuan
Quang Khoat Than
Rahul Chougule
Ramanathan Narayanan
Raquel Sebastiao
Rashid Ali
Rui Chen
S. Sathiya Keerthi
Shailesh Kumar
Sai Sundarakrishna
Saikat Dey
J. Saketha Nath
SakethaNath Jagarlapudi
Salim Akhter Chowdhury
Samah Fodeh
Sami Hanhijärvi
Satnam Singh
Sau Dan Lee
Sayan Ranu
Sergio Flesca
Shafkat Amin
Shailesh Kumar
Shalabh Bhatnagar
Shantanu Godbole
Sharanjit Kaur
Shazzad Hosain
Shenghua Gao
Shirish Shevade
Shirish Tatikonda
Shirong Li
Shumo Chu
Shuo Miao
Sinan Erten
Sk. Mirajul Haque
Soumen De
Sourangshu Bhattacharya
Sourav Dutta
Srinivasan Sengamedu
Srujana Merugu

Subhajit Sanyal
Sufyan Beg
Sugato Chakrabarty
Sundararajan Sellamanickam
Tadej Štajner
Takehiko Sakamoto
Thi Nhan Le
Tianrui Li
Timothy DeVries
Toshiyuki Amagasa
Venu Satuluri
Victor Lee
Vikram Pudi
Vishwakarma Singh
Viswanath G
Wang Wen-Chi
Wei Chu
Wei Jin
Wei Peng
Wei Su
Wendell Jordan-Brangman
Wenjun Zhou
Wenting Liu
Xiaofeng Zhu
Xiaogang Wu
Xin Liu
Xing Jiang
Xintian Yang
Xutong Liu
Yan Zhang
Yanchang Zhao
Yang Xiang
Yang Zhou
Yasufumi Takama
Yezhou Yang
Yilin Kang
Yin Zhang
Yong Ge
Yuan Liu
Yukai He
Yuko Itokawa
Zakia Sultana
Zubin Abraham

Organized by

IIIT Hyderabad, India



Sponsoring Institutions

AFOSR, USA



AOARD, Tokyo, Japan



ONRG, USA



Table of Contents – Part I

Keynote Speeches

| | |
|---|---|
| Empower People with Knowledge: The Next Frontier for Web Search . . . <i>Wei-Ying Ma</i> | 1 |
| Discovery of Patterns in Global Earth Science Data Using Data Mining <i>Vipin Kumar</i> | 2 |
| Game Theoretic Approaches to Knowledge Discovery and Data Mining <i>Y. Narahari</i> | 3 |

Session 1A. Clustering I

| | |
|--|----|
| A Set Correlation Model for Partitional Clustering <i>Nguyen Xuan Vinh and Michael E. Houle</i> | 4 |
| iVAT and aVAT: Enhanced Visual Analysis for Cluster Tendency Assessment <i>Liang Wang, Uyen T.V. Nguyen, James C. Bezdek, Christopher A. Leckie, and Kotagiri Ramamohanarao</i> | 16 |
| A Robust Seedless Algorithm for Correlation Clustering <i>Mohammad S. Aziz and Chandan K. Reddy</i> | 28 |
| Integrative Parameter-Free Clustering of Data with Mixed Type Attributes <i>Christian Böhm, Sebastian Goebel, Annahita Oswald, Claudia Plant, Michael Plavinski, and Bianca Wackersreuther</i> | 38 |
| Data Transformation for Sum Squared Residue <i>Hyuk Cho</i> | 48 |

Session 1B. Social Networks

| | |
|---|----|
| A Better Strategy of Discovering Link-Pattern Based Communities by Classical Clustering Methods <i>Chen-Yi Lin, Jia-Ling Koh, and Arbee L.P. Chen</i> | 56 |
| Mining Antagonistic Communities from Social Networks <i>Kuan Zhang, David Lo, and Ee-Peng Lim</i> | 68 |

As Time Goes by: Discovering Eras in Evolving Social Networks 81
*Michele Berlingerio, Michele Coscia, Fosca Giannotti,
 Anna Monreale, and Dino Pedreschi*

Online Sampling of High Centrality Individuals in Social Networks 91
Arun S. Maiya and Tanya Y. Berger-Wolf

Estimate on Expectation for Influence Maximization in Social
 Networks 99
Yao Zhang, Qing Gu, Jun Zheng, and Daoxu Chen

Session 1C. Classification I

A Novel Scalable Multi-class ROC for Effective Visualization and
 Computation 107
*Md. Rafiul Hassan, Kotagiri Ramamohanarao, Chandan Karmakar,
 M. Maruf Hossain, and James Bailey*

Efficiently Finding the Best Parameter for the Emerging Pattern-Based
 Classifier PCL 121
Thanh-Son Ngo, Mengling Feng, Guimei Liu, and Limsoon Wong

Rough Margin Based Core Vector Machine 134
Gang Niu, Bo Dai, Lin Shang, and Yangsheng Ji

BoostML: An Adaptive Metric Learning for Nearest Neighbor
 Classification 142
Nayyar Abbas Zaidi, David McG. Squire, and David Suter

A New Emerging Pattern Mining Algorithm and Its Application in
 Supervised Classification 150
*Milton García-Borroto, José Francisco Martínez-Trinidad, and
 Jesús Ariel Carrasco-Ochoa*

Session 2A. Privacy

Hiding Emerging Patterns with Local Recoding Generalization 158
*Michael W.K. Cheng, Byron Koon Kau Choi, and
 William Kwok Wai Cheung*

Anonymizing Transaction Data by Integrating Suppression and
 Generalization 171
Junqiang Liu and Ke Wang

Satisfying Privacy Requirements: One Step before Anonymization 181
Xiaoxun Sun, Hua Wang, and Jiuyong Li

| | |
|---|-----|
| Computation of Ratios of Secure Summations in Multi-party Privacy-Preserving Latent Dirichlet Allocation | 189 |
| <i>Bin Yang and Hiroshi Nakagawa</i> | |
| Privacy-Preserving Network Aggregation | 198 |
| <i>Troy Raeder, Marina Blanton, Nitesh V. Chawla, and Keith Frikken</i> | |
| Multivariate Equi-width Data Swapping for Private Data Publication . . . | 208 |
| <i>Yidong Li and Hong Shen</i> | |

Session 2B. Spatio-Temporal Mining

| | |
|--|-----|
| Correspondence Clustering: An Approach to Cluster Multiple Related Spatial Datasets | 216 |
| <i>Vadeerat Rinsurongkawong and Christoph F. Eick</i> | |
| Mining Trajectory Corridors Using Fréchet Distance and Meshing Grids | 228 |
| <i>Haohan Zhu, Jun Luo, Hang Yin, Xiaotao Zhou, Joshua Zhexue Huang, and F. Benjamin Zhan</i> | |
| Subseries Join: A Similarity-Based Time Series Match Approach | 238 |
| <i>Yi Lin and Michael D. McCool</i> | |
| TWave: High-Order Analysis of Spatiotemporal Data | 246 |
| <i>Michael Barnathan, Vasileios Megalooikonomou, Christos Faloutsos, Feroze B. Mohamed, and Scott Faro</i> | |
| Spatial Clustering with Obstacles Constraints by Dynamic Piecewise-Mapped and Nonlinear Inertia Weights PSO | 254 |
| <i>Xueping Zhang, Haohua Du, and Jiayao Wang</i> | |

Session 3A. Pattern Mining

| | |
|--|-----|
| An Efficient GA-Based Algorithm for Mining Negative Sequential Patterns | 262 |
| <i>Zhigang Zheng, Yanchang Zhao, Ziyi Zuo, and Longbing Cao</i> | |
| Valency Based Weighted Association Rule Mining | 274 |
| <i>Yun Sing Koh, Russel Pears, and Wai Yeap</i> | |
| Ranking Sequential Patterns with Respect to Significance | 286 |
| <i>Robert Gwadera and Fabio Crestani</i> | |
| Mining Association Rules in Long Sequences | 300 |
| <i>Boris Cule and Bart Goethals</i> | |
| Mining Closed Episodes from Event Sequences Efficiently | 310 |
| <i>Wenzhi Zhou, Hongyan Liu, and Hong Cheng</i> | |

| | |
|---|-----|
| Most Significant Substring Mining Based on Chi-square Measure | 319 |
| <i>Sourav Dutta and Arnab Bhattacharya</i> | |

Session 3B. Recommendations/Answers

| | |
|--|-----|
| Probabilistic User Modeling in the Presence of Drifting Concepts | 328 |
| <i>Vikas Bhardwaj and Ramaswamy Devarajan</i> | |
| Using Association Rules to Solve the Cold-Start Problem in Recommender Systems | 340 |
| <i>Gavin Shaw, Yue Xu, and Shlomo Geva</i> | |
| Semi-supervised Tag Recommendation - Using Untagged Resources to Mitigate Cold-Start Problems | 348 |
| <i>Christine Preisach, Leandro Balby Marinho, and Lars Schmidt-Thieme</i> | |
| Cost-Sensitive Listwise Ranking Approach | 358 |
| <i>Min Lu, MaoQiang Xie, Yang Wang, Jie Liu, and YaLou Huang</i> | |
| Mining Wikipedia and Yahoo! Answers for Question Expansion in Opinion QA | 367 |
| <i>Yajie Miao and Chunping Li</i> | |
| Answer Diversification for Complex Question Answering on the Web . . . | 375 |
| <i>Palakorn Achananuparp, Xiaohua Hu, Tingting He, Christopher C. Yang, Yuan An, and Lifan Guo</i> | |
| Vocabulary Filtering for Term Weighting in Archived Question Search | 383 |
| <i>Zhao-Yan Ming, Kai Wang, and Tat-Seng Chua</i> | |

Session 3C. Topic Modeling/Information Extraction

| | |
|--|-----|
| On Finding the Natural Number of Topics with Latent Dirichlet Allocation: Some Observations | 391 |
| <i>R. Arun, V. Suresh, C.E. Veni Madhavan, and M.N. Narasimha Murthy</i> | |
| Supervising Latent Topic Model for Maximum-Margin Text Classification and Regression | 403 |
| <i>Wanhong Xu</i> | |
| Resource-bounded Information Extraction: Acquiring Missing Feature Values On Demand | 415 |
| <i>Pallika Kanani, Andrew McCallum, and Shaohan Hu</i> | |
| Efficient Deep Web Crawling Using Reinforcement Learning | 428 |
| <i>Lu Jiang, Zhaohui Wu, Qian Feng, Jun Liu, and Qinghua Zheng</i> | |

| | |
|---|-----|
| Topic Decomposition and Summarization | 440 |
| <i>Wei Chen, Can Wang, Chun Chen, Lijun Zhang, and Jiajun Bu</i> | |
| Session 4A. Skylines/Uncertainty | |
| UNN: A Neural Network for Uncertain Data Classification | 449 |
| <i>Jiaqi Ge, Yuni Xia, and Chandima Nadungodage</i> | |
| SkyDist: Data Mining on Skyline Objects | 461 |
| <i>Christian Böhm, Annahita Oswald, Claudia Plant, Michael Plavinski, and Bianca Wackersreuther</i> | |
| Multi-Source Skyline Queries Processing in Multi-Dimensional Space . . . | 471 |
| <i>Cuiping Li, Wenlin He, and Hong Chen</i> | |
| Efficient Pattern Mining of Uncertain Data with Sampling | 480 |
| <i>Toon Calders, Calin Garboni, and Bart Goethals</i> | |
| Classifier Ensemble for Uncertain Data Stream Classification | 488 |
| <i>Shirui Pan, Kuan Wu, Yang Zhang, and Xue Li</i> | |
| Author Index | 497 |

Table of Contents – Part II

Session 4B. Dimensionality Reduction/Parallelism

| | |
|--|----|
| Subclass-Oriented Dimension Reduction with Constraint Transformation and Manifold Regularization | 1 |
| <i>Bin Tong and Einoshin Suzuki</i> | |
| Distributed Knowledge Discovery with Non Linear Dimensionality Reduction | 14 |
| <i>Panagis Magdalinos, Michalis Vazirgiannis, and Dialecti Valsamou</i> | |
| DPSP: Distributed Progressive Sequential Pattern Mining on the Cloud | 27 |
| <i>Jen-Wei Huang, Su-Chen Lin, and Ming-Syan Chen</i> | |
| An Approach for Fast Hierarchical Agglomerative Clustering Using Graphics Processors with CUDA | 35 |
| <i>S.A. Arul Shalom, Manoranjan Dash, and Minh Tue</i> | |

Session 5A. Novel Applications

| | |
|--|----|
| Ontology-Based Mining of Brainwaves: A Sequence Similarity Technique for Mapping Alternative Features in Event-Related Potentials (ERP) Data | 43 |
| <i>Haishan Liu, Gwen Frishkoff, Robert Frank, and Dejing Dou</i> | |
| Combining Support Vector Machines and the t -statistic for Gene Selection in DNA Microarray Data Analysis | 55 |
| <i>Tao Yang, Vojislave Kecman, Longbing Cao, and Chengqi Zhang</i> | |
| Satrap: Data and Network Heterogeneity Aware P2P Data-Mining | 63 |
| <i>Hock Hee Ang, Vivekanand Gopalkrishnan, Anwitaman Datta, Wee Keong Ng, and Steven C.H. Hoi</i> | |
| Player Performance Prediction in Massively Multiplayer Online Role-Playing Games (MMORPGs) | 71 |
| <i>Kyong Jin Shim, Richa Sharan, and Jaideep Srivastava</i> | |
| Relevant Gene Selection Using Normalized Cut Clustering with Maximal Compression Similarity Measure | 81 |
| <i>Rajni Bala, R.K. Agrawal, and Manju Sardana</i> | |

Session 5B. Feature Selection/Visualization

| | |
|---|-----|
| A Novel Prototype Reduction Method for the K -Nearest Neighbor Algorithm with $K \geq 1$ | 89 |
| <i>Tao Yang, Longbing Cao, and Chengqi Zhang</i> | |
| Generalized Two-Dimensional FLD Method for Feature Extraction: An Application to Face Recognition | 101 |
| <i>Shiladitya Chowdhury, Jamuna Kanta Sing, Dipak Kumar Basu, and Mita Nasipuri</i> | |
| Learning Gradients with Gaussian Processes | 113 |
| <i>Xinwei Jiang, Junbin Gao, Tianjiang Wang, and Paul W. Kwan</i> | |
| Analyzing the Role of Dimension Arrangement for Data Visualization in Radviz | 125 |
| <i>Luigi Di Caro, Vanessa Frias-Martinez, and Enrique Frias-Martinez</i> | |

Session 6A. Graph Mining

| | |
|--|-----|
| Subgraph Mining on Directed and Weighted Graphs | 133 |
| <i>Stephan Günnemann and Thomas Seidl</i> | |
| Finding Itemset-Sharing Patterns in a Large Itemset-Associated Graph | 147 |
| <i>Mutsumi Fukuzaki, Mio Seki, Hisashi Kashima, and Jun Sese</i> | |
| A Framework for SQL-Based Mining of Large Graphs on Relational Databases | 160 |
| <i>Sriganesh Srihari, Shruti Chandrashekar, and Srinivasan Parthasarathy</i> | |
| Fast Discovery of Reliable k -terminal Subgraphs | 168 |
| <i>Melissa Kasari, Hannu Toivonen, and Petteri Hintsanen</i> | |
| GTRACE2: Improving Performance Using Labeled Union Graphs | 178 |
| <i>Akihiro Inokuchi and Takashi Washio</i> | |

Session 6B. Clustering II

| | |
|--|-----|
| Orthogonal Nonnegative Matrix Tri-factorization for Semi-supervised Document Co-clustering | 189 |
| <i>Huifang Ma, Weizhong Zhao, Qing Tan, and Zhongzhi Shi</i> | |
| Rule Synthesizing from Multiple Related Databases | 201 |
| <i>Dan He, Xindong Wu, and Xingquan Zhu</i> | |

| | |
|---|-----|
| Fast Orthogonal Nonnegative Matrix Tri-Factorization for Simultaneous Clustering | 214 |
| <i>Zhao Li, Xindong Wu, and Zhenyu Lu</i> | |
| Hierarchical Web-Page Clustering via In-Page and Cross-Page Link Structures | 222 |
| <i>Cindy Xide Lin, Yintao Yu, Jiawei Han, and Bing Liu</i> | |
| Mining Numbers in Text Using Suffix Arrays and Clustering Based on Dirichlet Process Mixture Models | 230 |
| <i>Minoru Yoshida, Issei Sato, Hiroshi Nakagawa, and Akira Terada</i> | |
| Session 7A. Opinion/Sentiment Mining | |
| Opinion-Based Imprecise Query Answering | 238 |
| <i>Muhammad Abulaish, Tanvir Ahmad, Jahiruddin, and Mohammad Najmud Doja</i> | |
| Blog Opinion Retrieval Based on Topic-Opinion Mixture Model | 249 |
| <i>Peng Jiang, Chunxia Zhang, Qing Yang, and Zhendong Niu</i> | |
| Feature Subsumption for Sentiment Classification in Multiple Languages | 261 |
| <i>Zhongwu Zhai, Hua Xu, Jun Li, and Peifa Jia</i> | |
| Decentralisation of ScoreFinder: A Framework for Credibility Management on User-Generated Contents | 272 |
| <i>Yang Liao, Aaron Harwood, and Kotagiri Ramamohanarao</i> | |
| Classification and Pattern Discovery of Mood in Weblogs | 283 |
| <i>Thin Nguyen, Dinh Phung, Brett Adams, Truyen Tran, and Svetha Venkatesh</i> | |
| Capture of Evidence for Summarization: An Application of Enhanced Subjective Logic | 291 |
| <i>Sukanya Manna, B. Sumudu U. Mendis, and Tom Gedeon</i> | |
| Session 7B. Stream Mining | |
| Fast Perceptron Decision Tree Learning from Evolving Data Streams | 299 |
| <i>Albert Bifet, Geoff Holmes, Bernhard Pfahringer, and Eibe Frank</i> | |
| Classification and Novel Class Detection in Data Streams with Active Mining | 311 |
| <i>Mohammad M. Masud, Jing Gao, Latifur Khan, Jiawei Han, and Bhavani Thuraisingham</i> | |

| | |
|--|-----|
| Bulk Loading Hierarchical Mixture Models for Efficient Stream Classification | 325 |
| <i>Philipp Kranen, Ralph Krieger, Stefan Denker, and Thomas Seidl</i> | |
| Summarizing Multidimensional Data Streams: A Hierarchy-Graph-Based Approach | 335 |
| <i>Yoann Pitarch, Anne Laurent, and Pascal Poncelet</i> | |
| Efficient Trade-Off between Speed Processing and Accuracy in Summarizing Data Streams | 343 |
| <i>Nesrine Gabsi, Fabrice Clérot, and Georges Hébrail</i> | |
| Subsequence Matching of Stream Synopses under the Time Warping Distance | 354 |
| <i>Su-Chen Lin, Mi-Yen Yeh, and Ming-Syan Chen</i> | |
| Session 8A. Similarity and Kernels | |
| Normalized Kernels as Similarity Indices | 362 |
| <i>Julien Ah-Pine</i> | |
| Adaptive Matching Based Kernels for Labelled Graphs | 374 |
| <i>Adam Woźnica, Alexandros Kalousis, and Melanie Hilario</i> | |
| A New Framework for Dissimilarity and Similarity Learning | 386 |
| <i>Adam Woźnica and Alexandros Kalousis</i> | |
| Semantic-Distance Based Clustering for XML Keyword Search | 398 |
| <i>Weidong Yang and Hao Zhu</i> | |
| Session 8B. Graph Analysis | |
| OddBall: Spotting Anomalies in Weighted Graphs | 410 |
| <i>Leman Akoglu, Mary McGlohon, and Christos Faloutsos</i> | |
| Robust Outlier Detection Using Commute Time and Eigenspace Embedding | 422 |
| <i>Nguyen Lu Dang Khoa and Sanjay Chawla</i> | |
| EigenSpokes: Surprising Patterns and Scalable Community Chipping in Large Graphs | 435 |
| <i>B. Aditya Prakash, Ashwin Sridharan, Mukund Seshadri, Sridhar Machiraju, and Christos Faloutsos</i> | |
| BASSET: Scalable Gateway Finder in Large Graphs | 449 |
| <i>Hanghang Tong, Spiros Papadimitriou, Christos Faloutsos, Philip S. Yu, and Tina Eliassi-Rad</i> | |

Session 8C. Classification II

| | |
|--|-----|
| Ensemble Learning Based on Multi-Task Class Labels | 464 |
| <i>Qing Wang and Liang Zhang</i> | |
| Supervised Learning with Minimal Effort | 476 |
| <i>Eileen A. Ni and Charles X. Ling</i> | |
| Generating Diverse Ensembles to Counter the Problem of Class Imbalance | 488 |
| <i>T. Ryan Hoens and Nitesh V. Chawla</i> | |
| Relationship between Diversity and Correlation in Multi-Classifer Systems | 500 |
| <i>Kuo-Wei Hsu and Jaideep Srivastava</i> | |
| Compact Margin Machine | 507 |
| <i>Bo Dai and Gang Niu</i> | |
| Author Index | 515 |

Empower People with Knowledge: The Next Frontier for Web Search

Wei-Ying Ma

Microsoft Research Asia, China
wyma@microsoft.com

The Web has continued to evolve quickly. With the emergence of cloud computing, we see a new opportunity of creating a cloud platform to leverage developer ecosystem and enabling the development of millions of micro-vertical services and applications to serve users' various information need. In this new world, there is an opportunity to build a more powerful and intelligent search engine that understands what users are trying to accomplish and helps them learn, decide and take actions. In this talk, I will first discuss a few new trends from cloud computing that will impact web search, and then I will share my thoughts on possible directions to tap into this new wave and develop not only innovative but also potentially disruptive technologies for Web search.

Discovery of Patterns in Global Earth Science Data Using Data Mining

Vipin Kumar

University of Minnesota, Minneapolis, USA
kumar@cs.umn.edu

The climate and earth sciences have recently undergone a rapid transformation from a data-poor to a data-rich environment. In particular, climate and ecosystem related observations from remote sensors on satellites, as well as outputs of climate or earth system models from large-scale computational platforms, provide terabytes of temporal, spatial and spatio-temporal data. These massive and information-rich datasets offer huge potential for understanding and predicting the behavior of the Earth's ecosystem and for advancing the science of climate change.

However, mining patterns from Earth Science data is a difficult task due to the spatio-temporal nature of the data. This talk will discuss various challenges involved in analyzing the data, and present some of our work on the design of algorithms for finding spatio-temporal patterns from such data and their applications in discovering interesting relationships among ecological variables from various parts of the Earth. A special focus will be on techniques for land cover change detection (and their use in assessing the impact on carbon cycle) and finding teleconnections between ocean and land variables.

Game Theoretic Approaches to Knowledge Discovery and Data Mining

Y. Narahari

Indian Institute of Science, Bangalore, India
hari@csa.iisc.ernet.in

Game theory is replete with brilliant solution concepts such as the Nash equilibrium, the core, the Shapley value, etc. These solution concepts and their extensions are finding widespread use in solving several fundamental problems in knowledge discovery and data mining. The problems include clustering, classification, discovering influential nodes, social network analysis, etc. The first part of the talk will present the conceptual underpinnings underlying the use of game theoretic techniques in such problem solving. The second part of the talk will delve into two problems where we have recently obtained some interesting results:

- (a) Discovering influential nodes in social networks using the Shapley value, and
- (b) Identifying topologies of strategically formed social networks using a game theoretic approach.

A Set Correlation Model for Partitional Clustering

Nguyen Xuan Vinh^{1,2,3} and Michael E. Houle³

¹ School of Electrical Engineering and Telecommunications
The University of New South Wales, Sydney, NSW 2052, Australia

² National ICT Australia (NICTA)

³ National Institute of Informatics, Tokyo, Japan
`n.x.vinh@unsw.edu.au`, `meh@nii.ac.jp`

Abstract. This paper introduces GlobalRSC, a novel formulation for partitional data clustering based on the Relevant Set Correlation (RSC) clustering model. Our formulation resembles that of the K -means clustering model, but with a shared-neighbor similarity measure instead of the Euclidean distance. Unlike K -means and most other clustering heuristics that can only work with real-valued data and distance measures taken from specific families, GlobalRSC has the advantage that it can work with any distance measure, and any data representation. We also discuss various techniques for boosting the scalability of GlobalRSC.

Keywords: Clustering, correlation, shared neighbor, RSC, SASH.

1 Introduction

Clustering is the art of partitioning a data set into groups such that objects from the same group are as similar as possible, and objects from different groups are well differentiated. To support clustering, a measure of similarity (or distance) between data objects is needed. Popular distance measures for clustering include the class of general L_p norms (which includes the Euclidean distance L_2), and the cosine similarity measure. The k objects most similar to a data item v are often referred to as the k -nearest-neighbor (k -NN) set of v .

An interesting and appealing class of ‘secondary’ similarity measures, the so-called *shared-neighbor* (SN) measures, can be derived from any other (‘primary’) similarity measure. SN measures typically are expressed as a function of the intersection size of the k -NN sets of the two objects whose similarity is to be computed, where the neighborhoods are computed using the primary similarity measure. The use of SN-based similarity in clustering can be traced back to the merge criterion of the agglomerative algorithm due to Jarvis and Patrick [1]. Other agglomerative clustering methods with SN-based merge criteria include the hierarchical algorithm ROCK [2] and the density-based algorithm SNN (Shared Nearest Neighbor) [3]. SNN is essentially an improved version of the well-known DBSCAN [4] clustering algorithm; like DBSCAN, SNN is able

to produce clusters of different sizes, shapes and densities. However, the performance of SNN greatly depends on the tuning of several non-intuitive parameters by the user. In practice, it is difficult (if not impossible) to determine appropriate values for these parameters on real datasets.

A common requirement of most SN-based clustering methods is that a fixed neighborhood size — either in terms of the number of neighbors k as in SNN and in Jarvis & Patrick’s method, or in terms of the radius r of the neighborhood ball as in ROCK — needs to be chosen in advance, and then applied equally to all items of the data set. However, fixed choices of neighborhood sizes k or radius r are known to lead to bias in the clustering process [5], the former with respect to the sizes of the clusters discovered, and the latter with respect to the density of the regions from which the clusters are produced.

Recently, an SN-based clustering model was proposed that allows the sizes of the neighborhoods to vary. The *Relevant Set Correlation* (RSC) model [5] defines the relevance of a data item v to a cluster C in terms of a form of ‘set correlation’ between the memberships of $|C|$ and the $|C|$ -nearest-neighbor set of v . RSC quality measures can be used to evaluate the relative importance of cluster candidates of various sizes, avoiding the problems of bias found with other shared-neighbor methods that use fixed neighborhood sizes or radii. The same paper introduced a clustering algorithm based on RSC, called GreedyRSC, that generates cluster candidates in the vicinity of every object of the dataset, evaluates the quality of the candidates according to the model, and greedily selects them in decreasing order of their quality. GreedyRSC is a ‘soft’ clustering algorithm, in that the clusters produced are allowed to overlap. It does not require that the user choose the neighborhood size or specify a target number of clusters; instead, the user simply specifies the minimum allowable cluster size, and the maximum allowable correlation between any two clusters. Unlike many other clustering algorithms, GreedyRSC uses only local criteria for the formation of cluster candidates — the clustering process is not guided by a global optimization criterion.

In this paper, we present a new RSC-based partitional clustering algorithm, *GlobalRSC*, that allows the user to specify the number of clusters to be generated, K . Unlike GreedyRSC, GlobalRSC emulates the well-known K -means clustering algorithm in that it seeks to optimize a global objective function. Given an initial clustering configuration, both K -means and GlobalRSC attempt to optimize their objective function through an iterative hill-climbing improvement process. GlobalRSC, however, replaces the Euclidean distance of K -means by a shared-neighbor similarity measure, and can therefore be applied (in principle) to any form of data, and using any appropriate similarity measure.

This paper is organized as follows. In Section 2 we review those elements of the RSC model upon which GlobalRSC is based, and introduce the GlobalRSC clustering criterion. In Section 3, we give the details of the GlobalRSC clustering algorithm. Experimental results are presented in Section 4, and concluding remarks appear in Section 5.

2 Shared Neighbor Similarity and the RSC Model

In this section, we present a brief introduction to the Relevant Set Correlation (RSC) model for clustering, and the set correlation similarity measure upon which it is based.

Let S be a dataset of $|S| = n$ data items $\{s_1, s_2, \dots, s_n\}$. Any subset A of S can then be represented as a n -dimensional zero-one characteristic vector, where the value of the i -th coordinate is 1 if and only if $s_i \in A$. The simplest SN-based similarity measure between the two sets A and B is the ‘overlap’ or intersection size $|A \cap B|$, which can be expressed as the inner product between the two characteristic vectors. Another popular measure, the cosine similarity $\cos(A, B) \triangleq \frac{|A \cap B|}{\sqrt{|A||B|}}$, is the inner product of the normalized characteristic vectors for A and B , which in turn equals the cosine of the angle between them. Values of the cosine measure lie in the range $[0, 1]$, with 1 attained whenever A is identical to B , and 0 whenever A and B are disjoint.

In [5], the *set correlation* measure was proposed as the value of the Pearson correlation between the coordinate pairs of characteristic vectors of A and B . After derivation and simplification, this expression becomes:

$$R(A, B) \triangleq \frac{|S|}{\sqrt{(|S| - |A|)(|S| - |B|)}} \left(\cos(A, B) - \frac{\sqrt{|A||B|}}{|S|} \right). \quad (1)$$

Values of the set correlation lie in the range $[-1, 1]$. A value of 1 indicates that A and B are identical, and a value of -1 indicates that A and B are complements of each other in S .

Despite their simplicity and their popularity in practice, the overlap and cosine measures both have the disadvantage of bias relative to the sizes of the two sets A and B . To see this, let A be fixed with size $|A| = a$, and let B be selected uniformly at random from the items of S with the constraint that $|B|$ equals some fixed value $b > 0$. Let $a = |A|$. Under these assumptions, the overlap is known to be hypergeometrically distributed with expected value $\mathbf{E}[|A \cap B|] = \frac{ab}{n}$, and the expected value of the cosine measure is therefore $\mathbf{E}[\cos(A, B)] = \frac{\mathbf{E}[|A \cap B|]}{\sqrt{ab}} = \frac{\sqrt{ab}}{n}$. When used to rank the similarity of sets with respect to A , both measures are biased towards sets B of larger sizes. On the other hand, the expected value of the set correlation under the same assumptions can be shown to be $\mathbf{E}[R(A, B)] = 0$, indicating no bias with respect to the sizes of A and B . Therefore, of these three SN-based similarity measures, the set correlation measure is best suited for those applications in which the neighborhood size is variable.

Under the RSC model, the quality of a given cluster candidate set A is assessed in terms of the set correlation between the candidate and neighborhood sets (the ‘relevant sets’) based at its members. Let Q_k^v denote the set of k -nearest neighbors of v with respect to S . The RSC model uses the set correlation $R(Q_k^v, A)$ between $Q_{|A|}^v$ and A as a measure of relevance of item v to the cluster candidate A . Note that in this formulation, the neighborhood size is taken to be

the cardinality of A . This definition eliminates the need for specifying a fixed neighborhood size, and avoids the bias associated with such choices.

For more details concerning the quality measures of the RSC model, see [5].

3 The GlobalRSC Clustering Algorithm

3.1 GlobalRSC and K -Means

The GlobalRSC clustering criterion has the same general form as that of K -means. In the standard K -means formulation, a partition $\mathcal{A} = \{A_1, A_2, \dots, A_K\}$ of the data set is sought which maximizes the following objective function:

$$\mathcal{D}(\mathcal{A}) = \frac{1}{n} \sum_{i=1}^K \sum_{v \in A_i} D(v, c(A_i)), \quad (2)$$

where $c(A)$ is a function which returns the center of mass of a cluster A (computed as $c(A) = \frac{1}{|A|} \sum_{v \in A} v$), and the distance measure D is generally taken to be the square of the Euclidean distance. The proposed formulation of GlobalRSC replaces the distance measure $D(v, c(A_i))$ by the average set correlation between cluster A_i and the neighborhood $Q_{|A_i|}^v$ based at v , as follows:

$$\mathcal{R}(\mathcal{A}) = \frac{1}{n} \sum_{i=1}^K \sum_{v \in A_i} R(Q_{|A_i|}^v, A_i). \quad (3)$$

Both D and R serve as measures of the relevance of an item to its assigned cluster. However, unlike R , D can only be computed when the data can be represented as real-valued vectors. As discussed earlier, the use of set correlation in the formulation of GreedyRSC is preferred over that of the overlap or cosine measure, due to the bias of the latter measures with respect to set sizes.

3.2 A Hill-Climbing Heuristic

The problem of optimizing the GlobalRSC objective function (3) greatly resembles that of optimizing the K -means objective function (2). Despite its simple appearance, the K -means clustering problem with squared Euclidean distance is known to be NP-hard even for $K = 2$ [6]. Although the hardness of the GlobalRSC optimization problem is still an open question, a heuristic approach seems to be indicated.

In this section, we propose an iterative hill-climbing solution, which we simply refer to as *GlobalRSC*. The core idea is as follows: at each round the algorithm iterates through the items of S looking for items whose reassignment to a different cluster leads to an improvement in the value of the objective function \mathcal{R} . As is the case with K -means, two reassignment schemes can be employed for GlobalRSC: *incremental update*, in which reassignment is performed as soon as an improvement is detected, and *batch update*, in which all the membership changes are

applied only at the end of each round, when the algorithm has completed a full iteration through all data items. The advantage of the batch update scheme is that the recomputation of \mathcal{R} can be performed very efficiently through the use of *inverted neighborhood sets* (as defined in Fig. 1). However, it is possible that delaying the reassignment of items until the end of each round could result in a decrease in the value of the objective function, even if each reassignment would have led to an increase if it were applied individually. In practice we often observe that first few rounds of the batch update scheme quickly improves the objective value. It would therefore be beneficial to begin with several rounds of batch updating, followed by an incremental update phase to further refine the clustering.

It should be noted that in an incremental reassignment of v from cluster A_i to cluster A_j , the contributions $\mathcal{R}(A) = \sum_{w \in A} R(Q_{|A|}^w, A)$ to \mathcal{R} for an individual cluster A do not need to be recomputed except for $A = A_i$ and $A = A_j$. To verify whether the reassignment would increase the value of \mathcal{R} , it suffices to perform the test $\mathcal{R}(A_j \cup \{v\}) + \mathcal{R}(A_i \setminus \{v\}) - \mathcal{R}(A_j) - \mathcal{R}(A_i) > 0$.

The recomputation of \mathcal{R} after the reassignment of a single item v would be relatively expensive if all $K - 1$ possible reassignments were considered. We therefore limit the tentative reassignment of v to those candidate clusters found in the vicinity of v ; that is, those clusters containing at least one element in the neighborhood $Q_{|A_i|}^v$, where A_i is the cluster to which v currently belongs. If one of these tentative reassignments of v would result in an increase in the value of \mathcal{R} , then the reassignment that results in the greatest such increase is applied. Otherwise, v is not reassigned. If the size of the new cluster A_j is larger than that of the currently-stored neighborhood of v , then that neighborhood would need to be expanded. Accordingly, whenever it is necessary to recompute a neighborhood for item $v \in A_j$, we choose the size to be $\min\{[(1 + b)|A_j|], m\}$ for some fixed real parameter values $b > 0$ and $m > 0$. In our implementation of GlobalRSC, b is set to a default value of 0.5, and m is set to 50.

A pseudocode description of the basic GlobalRSC heuristic is shown in Fig. 1. The heuristic can easily be shown to converge within a finite number of steps.

3.3 Complexity Analysis

The algorithm requires storage for the neighbor lists of all n data items, each of which has size proportional to that of the cluster to which it has been assigned. The total space required is of order $\sum_{i=1}^K |A_i|^2$. Let μ and σ be respectively the mean and standard deviation of the cluster sizes; in terms of μ and σ , the space required is proportional to $K(\sigma^2 + \mu^2)$.

At the initialization step, the neighborhood list for each data item must be calculated. A straightforward implementation requires the computation of $O(n^2)$ distances. Once computed, these distances can also be used to generate an initial clustering. Since the neighborhoods must be constructed in sorted order, the total time required for preparing neighborhoods is $\sum_{i=1}^K |A_i|(|A_i| + |A_i| \log |A_i|) = O((\sigma^2 + \mu^2)K \log n)$ using linear-time methods for determining order statistics [7]. The total time required for initialization is thus $O(dn^2 + (\sigma^2 + \mu^2)K \log n)$, where d is the cost of computing a single distance.

Input: The data set S ; the number of desired clusters K ; (optionally) a hard initial clustering $\mathcal{A} = \{A_1, A_2, \dots, A_K\}$ on S ; minimum neighborhood set size m ; neighborhood set buffer size b .

0. **Initialization:** If no initial clustering was provided, compute an initial clustering $\mathcal{A} = \{A_1, A_2, \dots, A_K\}$ as follows:
 - (a) Select K items of S uniformly at random as the seeds of the K clusters.
 - (b) Assign each data item v to the cluster corresponding to the seed closest to v .
 - (c) Build the neighborhood Q^v for each item $v \in A_i$, with size equal to $\min\{\lceil(1+b)|A_i|\rceil, m\}$.
 1. **Batch phase:**
 - (a) Calculate $\mathcal{R}(A_1), \mathcal{R}(A_2), \dots, \mathcal{R}(A_K)$ and set the termination flag $halt \leftarrow \text{FALSE}$.
 - (b) Repeat until $halt = \text{TRUE}$:
 - i. Set $halt \leftarrow \text{TRUE}$.
 - ii. For each $v \in S$, build the *inverted neighborhood* I^v , where $r \in I^v$ if and only if $v \in Q^r$.
 - iii. For each data item v currently in cluster A_i :
 - A. Build the list \mathfrak{C} of clusters (other than A_i) containing at least one item of Q^v .
 - B. Tentatively reassign v to each of the A_j in \mathfrak{C} .
 - C. Using the inverted neighborhood sets, calculate the index j for which the improvement value $\mathcal{R}(A_j \cup \{v\}) + \mathcal{R}(A_i \setminus \{v\}) - \mathcal{R}(A_j) - \mathcal{R}(A_i)$ is maximized.
 - D. If the improvement value is positive, record (v, j) for future reassignment.
 - iv. Tentatively apply all the recorded reassignments for this round, and let $\mathcal{A}' = \{A'_1, A'_2, \dots, A'_K\}$ be the resulting clustering. Calculate $\mathcal{R}(\mathcal{A}')$. If $\mathcal{R}(\mathcal{A}') > \mathcal{R}(\mathcal{A})$, set $halt \leftarrow \text{FALSE}$ and $\mathcal{A} \leftarrow \mathcal{A}'$. Otherwise, proceed to the incremental phase.
 2. **Incremental phase:**
 - (a) Set $halt \leftarrow \text{FALSE}$.
 - (b) Repeat until $halt = \text{TRUE}$:
 - i. Set $halt \leftarrow \text{TRUE}$.
 - ii. For each data item v currently in cluster A_i :
 - A. Build the list \mathfrak{C} of clusters (other than A_i) that contribute items to Q^v .
 - B. Tentatively reassign v to each of the A_j in \mathfrak{C} , and calculate the index j for which the improvement value $\mathcal{R}(A_j \cup \{v\}) + \mathcal{R}(A_i \setminus \{v\}) - \mathcal{R}(A_j) - \mathcal{R}(A_i)$ is maximized.
 - C. If the improvement value is positive, reassign v to A_j immediately, adjust the values of $\mathcal{R}(A_j)$ and $\mathcal{R}(A_i)$, and set $halt \leftarrow \text{FALSE}$.
-

Fig. 1. A pseudocode description of the basic GlobalRSC variant

During each round of the batch phase, building the inverted neighbor sets requires that the values of $K(\sigma^2 + \mu^2)$ integer variables be copied. Recalculating \mathcal{R} for the tentative reassignment of item v from cluster A_i to cluster A_j requires time proportional to $|A_i| + |A_j| + 2|I_v|$ when using the inverted neighbor lists. Assuming that v needs to be tentatively reassigned to each of the other $K - 1$ clusters, the cost of reassignment is $O(n)$. Adding the cost over all choices of v , the total cost of reassignment per phase is at most $O(n^2)$. The neighbor list can be reconstructed in $O(n \log n + nd)$ time if required; the total cost of reconstruction will be no worse than that of initialization, which is $O(dn^2 + (\sigma^2 + \mu^2)K \log n)$. The worst case complexity of each batch phase iteration through the data set is therefore $O(dn^2 + (\sigma^2 + \mu^2)K \log n)$. However, in practice we expect a much lower time cost, since there are typically only a limited number of nearby clusters for each data item, and few if any neighborhoods require reconstruction.

In the incremental phase, tentatively moving an item v from a cluster A_i to another A_j requires $O(|A_i|^2 + |A_j|^2)$ operations for the direct recalculation of the objective function. If v is tentatively reassigned to each of the other $K - 1$ clusters, the cost is $O(K(\sigma^2 + \mu^2))$; over all possible choices of v (one full round), the total cost of reassignment becomes $O(nK(\sigma^2 + \mu^2))$. Reconstruction of the neighborhood for each item, if required, can be performed in

$O(dn^2 + n^2 \log n)$ total time. The worst case complexity of a full round is therefore $O(dn^2 + n^2 \log n + nK(\sigma^2 + \mu^2))$. However again, the observed cost should be much lower in practice. Since the incremental phase is often considerably more expensive than the batch phase, to improve time efficiency for large data sets, we employ a ‘reduced’ variant of the incremental phase, in which a data item v is considered for reassignment to another cluster if and only if that cluster already contains the majority of the neighbors of v . This variant scheme focuses on items with neighborhoods of low consistency, with the worst case complexity of each round being reduced to that of the batch phase.

In practice, the standard deviation of the cluster size, σ , is typically of the same order of the mean cluster size $\mu = n/K$, leading to an overall space complexity of $\tilde{O}(n^2/K)$, and a time complexity (for the reduced incremental phase variant) of $\tilde{O}(dn^2 + n^2(\log n)/K)$. Optionally, if a full distance matrix is to be stored in order to speed up neighborhood list computation, then the space complexity would attain its worst-case value of $\Theta(n^2)$.

3.4 Scalability

In this section we present several techniques that can boost the scalability of GlobalRSC for large, high dimensional data sets. The challenges faced by GlobalRSC (and other SN-based clustering algorithms) as the dimensionality increases are: (i) the construction of neighborhoods becomes more expensive, due to an effect known as the ‘curse of dimensionality’ [8]; (ii) the optimization of the objective function becomes more difficult, as local optimization approaches such as hill-climbing are more easily trapped at local maxima that may be far from the global optimum.

In order to accelerate the construction of neighborhoods, we propose the use of the Spatial Approximation Sample Hierarchy (SASH) developed in [8].

If the data set contains many large clusters, the calculation of set correlation scores with respect to these clusters may be prohibitively expensive. One of the simplest ways of avoiding the high costs associated with large cluster candidates is through the restriction of neighborhood sizes. In our implementation of GlobalRSC, we restricted the maximum size of the neighborhood to be 1000. Only the average relevance score for items in cluster of size smaller than this threshold is calculated exactly, while the membership of larger clusters are ‘frozen’. Items are permitted to be reassigned from smaller clusters to frozen clusters and vice-versa, as long as such a movement increases the average relevance score of the smaller clusters.

4 Experimental Results

In this section we report the results of our experiments on various real data sets taken from several domains. GlobalRSC, implemented in C++ and tested on a Pentium IV 3.2GHz workstation equipped with 4Gb of main memory, was compared against a MATLAB implementation of ‘Fast’ K -means [9] (available from

the author’s website), and the ‘bisecting’ K -means algorithm available as part of the CLUTO clustering toolkit [10]. CLUTO was run with its default distance measure, the cosine similarity, using repeated bisecting clustering followed by a final global optimization phase. For large data sets, we used GreedyRSC to initialize GlobalRSC, as well as the SASH to speed up neighborhood construction. We report the mean and standard deviation values of the quality metrics, plus the average execution time and the number of iterations performed by each algorithm (if known). We did not include the SNN clustering algorithm, due to the difficulty in tuning its parameters, and since it often leaves a large number of items unassigned to any cluster. For interested readers, a comparison between SNN, K -means and GreedyRSC on several data sets can be found in [5].

For each of the data sets considered, the clustering results are assessed against a ground-truth classification according to 5 different quality measures: the well-known Adjusted Rand Index (ARI) from statistics [11]; the recently developed Adjusted Mutual Information (AMI) [12] from information theory; the Expected Precision (EPrec), Recall (ERec) and Cosine (ECos) measures [5] from information retrieval. For all these measures, higher values represent better clusterings, with a maximum possible value of 1. A low expected precision score is an indication of cluster fusion, occurring when too few clusters are produced, whereas a low expected recall indicates cluster fragmentation, occurring when too many clusters are generated. A high expected cosine score can be taken as evidence that the clustering avoids extremes of cluster fusion and cluster fragmentation. Due to lack of space, we do not give details of these measures here. Interested readers are referred to the original publications for more information.

4.1 Biological Data

We tested the algorithms on several gene expression microarray data sets:

- **B1:** This set consists of 384 genes whose expression level peak at different time points corresponding to the five phases of a cell cycle [13].
- **B2:** This set consists of 237 genes corresponding to four categories in the MIPS database. The four categories (DNA synthesis and replication, organization of centrosome, nitrogen and sulphur metabolism and ribosomal proteins) were shown to be reflected in clusters from the yeast cell cycle data [13]. These four functional categories form the four classes in the external criterion for this data set.
- **B3:** A subset of 205 genes from the yeast galactose data set [14]. The expression patterns reflect four functional categories in the Gene Ontology (GO) listings.

As is popular practice in microarray data analysis, the data was row-normalized to have zero mean and unit variance. Under this normalization scheme, the Euclidean distance and cosine similarity are equivalent. From the experiment results shown in Table 1, averaged over 100 runs, CLUTO appears to perform best, closely followed by GlobalRSC and then K -means.

Table 1. Experimental results (the highest quality index values are in bold)

| Data | Algorithm | ARI | AMI | EPrec | ERec | ECos | Loops | Time (s) |
|------|-------------|------------------|------------------|------------------|------------------|------------------|-------|-----------|
| B1 | K -means | 0.42±0.05 | 0.48±0.02 | 0.55±0.02 | 0.56±0.04 | 0.55±0.03 | 11±4 | 0±0 |
| | CLUTO | 0.50±0.00 | 0.52±0.00 | 0.59±0.00 | 0.61±0.00 | 0.60±0.00 | N/A | 0±0 |
| | GlobalRSC | 0.48±0.01 | 0.50±0.01 | 0.56±0.01 | 0.62±0.02 | 0.58±0.01 | 11±3 | 2±1 |
| B2 | K -means | 0.49±0.03 | 0.33±0.01 | 0.65±0.01 | 0.54±0.02 | 0.59±0.01 | 8±3 | 0±0 |
| | CLUTO | 0.51±0.00 | 0.34±0.00 | 0.65±0.00 | 0.56±0.00 | 0.60±0.00 | N/A | 0±0 |
| | GlobalRSC | 0.50±0.02 | 0.29±0.01 | 0.69±0.01 | 0.48±0.01 | 0.56±0.01 | 11±3 | 1±1 |
| B3 | K -means | 0.83±0.09 | 0.78±0.08 | 0.92±0.02 | 0.83±0.09 | 0.86±0.06 | 7±4 | 0±0 |
| | CLUTO | 0.96±0.00 | 0.91±0.00 | 0.95±0.00 | 0.96±0.00 | 0.96±0.00 | N/A | 0±0 |
| | GlobalRSC | 0.92±0.04 | 0.84±0.07 | 0.92±0.05 | 0.96±0.01 | 0.93±0.03 | 3±0 | 0±1 |
| I1 | K -means | 0.48±0.01 | 0.72±0.00 | 0.66±0.01 | 0.56±0.01 | 0.56±0.01 | 20±4 | 1278±347 |
| | K -means* | 0.49±0.01 | 0.74±0.00 | 0.61±0.01 | 0.59±0.01 | 0.55±0.01 | 23±5 | 993±416 |
| | CLUTO | 0.53±0.00 | 0.75±0.00 | 0.67±0.00 | 0.61±0.00 | 0.59±0.00 | N/A | 1015±51 |
| | CLUTO* | 0.53±0.00 | 0.75±0.00 | 0.62±0.00 | 0.62±0.00 | 0.58±0.00 | N/A | 847±9 |
| | GreedyRSC | 0.59±0.00 | 0.77±0.00 | 0.64±0.00 | 0.68±0.00 | 0.62±0.00 | N/A | 328±24 |
| | GlobalRSC | 0.61±0.01 | 0.78±0.00 | 0.66±0.00 | 0.71±0.00 | 0.64±0.00 | 17±3 | 417±29 |
| I2 | CLUTO | 0.41±0.00 | 0.73±0.00 | 0.48±0.00 | 0.52±0.00 | 0.48±0.00 | N/A | 7893±16 |
| | CLUTO* | 0.48±0.00 | 0.69±0.00 | 0.80±0.00 | 0.35±0.00 | 0.50±0.00 | N/A | 27302±192 |
| | GreedyRSC | 0.56±0.00 | 0.72±0.00 | 0.82±0.00 | 0.42±0.00 | 0.56±0.00 | N/A | 4352±53 |
| | GlobalRSC | 0.59±0.00 | 0.74±0.00 | 0.85±0.00 | 0.46±0.00 | 0.59±0.00 | 28±3 | 5002±127 |
| T2 | CLUTO* | 0.02±0.00 | 0.24±0.00 | 0.60±0.00 | 0.02±0.00 | 0.10±0.00 | N/A | 1201±14 |
| | GreedyRSC | 0.04±0.00 | 0.26±0.00 | 0.61±0.00 | 0.04±0.00 | 0.12±0.00 | N/A | 495±2 |
| | GlobalRSC | 0.09±0.00 | 0.29±0.00 | 0.64±0.01 | 0.07±0.00 | 0.16±0.00 | 17±4 | 1616±167 |
| T1 | GreedyRSC | 0.01±0.00 | 0.25±0.00 | 0.71±0.00 | 0.01±0.00 | 0.06±0.00 | N/A | 10657±334 |
| | GlobalRSC | 0.02±0.00 | 0.26±0.00 | 0.72±0.00 | 0.02±0.00 | 0.08±0.00 | 18±5 | 19044±756 |

*: K -means and CLUTO run using the number of clusters K as determined by GreedyRSC

4.2 Image Data

We tested the clustering algorithms on the Amsterdam Library of Object Images (ALOI) [15], which consists of 110,250 images of 1000 common objects. Each image is represented by a dense 641-dimensional feature vector based on color and texture histograms (see [16] for details on how the vectors were produced). The following data sets were used:

- **I1-ALOI-var:** A subset of 13943 images, generated by selecting objects unevenly from among the classes, with the i -th object class having approximately $40000/(400+i)$ image instances selected.
- **I2-ALOI-full:** The entire ALOI library.

Since the appearance of individual objects varies considerably with the vantage points of the images, almost every class would be expected to generate several natural clusters. Over 20 runs, GreedyRSC estimated the number of clusters to be 843 ± 8 for the I1 data set, and 3724 ± 25 for the I2 data set. Fast K -means and CLUTO were executed twice with random initialization, for both the true numbers of clusters ($K = 1000$) and the number of clusters as determined by GreedyRSC. GlobalRSC was initialized using GreedyRSC, and a SASH was used to construct the neighborhood sets. All runs except those involving CLUTO were conducted using the Euclidean distance measure. Over 20 runs, GreedyRSC consistently achieves good clustering quality which is then further refined by GlobalRSC, as observed in Table 1. For the ALOI-full data set, the execution of Fast K -means failed to terminate due to insufficient main memory.

It can be observed that when a good initialization is provided and SASH is used, the execution time of GlobalRSC is significantly shorter, comparable to that of K -means and CLUTO.

4.3 Text Data

We tested the clustering algorithms on the Reuters Corpus Volume I (RCV1), an archive of over 800,000 manually categorized newswire stories recently made available by Reuters, Ltd. for research purposes [17]. The document class structure was simplified into 57 distinct classes. We then selected a subset T1 consisting of 200,000 documents classified to either exactly one subtopic or exactly one meta topic. We also constructed a smaller data set T2 consisting of 20,000 documents selected uniformly at random from T1. For both T1 and T2, TF-IDF weighting was used to construct the feature vectors, resulting in sparse vectors of length 320,648. Fast K -means was excluded in this experiment due to its lack of support for sparse numerical data. Since the number of external classes of these data sets was not as reliable as of the ALOI image data set, we first ran GreedyRSC to estimate the number of natural clusters K . The clustering result of GreedyRSC was used for the initialization of GlobalRSC, while CLUTO was run with the desired number of clusters also set to K . The cosine similarity measure was used for all runs. The clustering scores, averaged over 20 runs, are reported in Table 1. While all the algorithms successfully processed the small T2 set, on T1 CLUTO gave a memory failure message after a few hours of execution, leaving only the results of GreedyRSC and GlobalRSC available for evaluation. The observed low agreement between the clustering result and the class information in this experiment can be attributed to natural fragmentation of the classes within the data domain.

4.4 Categorical Data

The mushroom data set, drawn from the UCI machine learning repository [18], contains 8124 varieties of mushrooms, each recorded with 22 different categorical physical attributes (such as color, odor, size, and shape). Each record is classified as to whether its associated mushroom is poisonous or edible. The distance measure for this data set is taken as the straightforward mismatch count, with missing values treated as contributing to the count.

The mushroom data set was previously analyzed with ROCK [2], which in their paper was reported as finding 21 clusters. Most of the clusters consist of only one type of mushroom, either edible or poisonous. Only 32 mushrooms were misclassified by ROCK. On 20 runs with random initialization, GreedyRSC produced 22 ± 1 clusters, with 87 ± 120 mushroom instances misclassified. The result is further refined with GlobalRSC, which brought the number of mushroom species misclassified down to 46 ± 97 . The classification errors of the 20 runs are reported in table 2. All the algorithms greatly outperformed the traditional hierarchical clustering implemented in [2], which produced 20 clusters within which 3432 out of 8124 items were misclassified. K -means was excluded from

Table 2. Experimental results on the Mushroom data set: classification errors over 20 runs

| Classification errors | | | | | | | | | | | | | | | | | | | | |
|-----------------------|----|----|----|----|-----|-----|----|----|----|----|----|----|-----|----|----|----|----|----|----|---|
| GreedyRSC | 32 | 69 | 69 | 32 | 289 | 484 | 69 | 33 | 44 | 33 | 32 | 32 | 278 | 32 | 32 | 52 | 32 | 32 | 32 | |
| GlobalRSC | 1 | 33 | 33 | 1 | 257 | 289 | 33 | 1 | 1 | 1 | 1 | 1 | 257 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |

this experiment as it can not handle categorical data. A clustering algorithm from CLUTO, which operates on the similarity matrix, was tested but did not yield competitive results, with 1339 misclassified species. It should be noted that whereas ROCK required an estimate of the number of clusters, GreedyRSC automatically determined this number.

5 Conclusion

In this paper we have introduced a novel shared-neighbor clustering algorithm based on the Relevant Set Correlation (RSC) model. The key difference in our approach to clustering, compared to other shared-neighbor-based approaches, is that it requires the setting of only one main parameter — the number of clusters. The objective function greatly resembles that of K -means, and like K -means, the GlobalRSC method aims to discover compact, globular clusters. While this class of clusters appears to be restrictive, Dasgupta [19] has shown that for high dimensional data, random projection can transform highly eccentric clusters into more spherical ones, which in turn can be discovered by K -means or GlobalRSC. The techniques we presented for improving the scalability of our proposed GlobalRSC algorithm allow for practical application of the method for large, high-dimensional generic data sets under any reasonable measure of similarity.

Acknowledgement

This work was partially supported by NICTA. NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

References

1. Jarvis, R.A., Patrick, E.A.: Clustering using a similarity measure based on shared near neighbors. *IEEE Trans. Comput.* 22(11), 1025–1034 (1973)
2. Guha, S., Rastogi, R., Shim, K.: Rock: A robust clustering algorithm for categorical attributes. *Information Systems*, 512–521 (1999)
3. Ertöz, L., Steinbach, M., Kumar, V.: Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data. In: *Proc. 3rd SIAM Intern. Conf. on Data Mining, SDM* (2003)

4. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proc. 2nd Int. Conf. on Knowl. Discovery and Data Mining (KDD), pp. 226–231. AAAI Press, Menlo Park (1996)
5. Houle, M.E.: The relevant-set correlation model for data clustering. *Stat. Anal. Data Min.* 1(3), 157–176 (2008)
6. Dasgupta, S.: The hardness of k-means clustering. Technical Report CS2007-0890, University of California, San Diego (2008)
7. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: *Introduction to Algorithms*, 2nd edn. MIT Press, McGraw-Hill Book Company (2000)
8. Houle, M.E., Sakuma, J.: Fast approximate similarity search in extremely high-dimensional data sets. In: ICDE 2005: Proceedings of the 21st International Conference on Data Engineering, Washington, DC, USA, pp. 619–630. IEEE Computer Society, Los Alamitos (2005)
9. Elkan, C.: Using the triangle inequality to accelerate k-means. In: Proceedings of the Twentieth International Conference on Machine Learning, ICML 2003 (2003)
10. Karypis, G.: CLUTO – a clustering toolkit (2002)
11. Lawrence, H., Phipps, A.: Comparing partitions. *Journal of Classification* 2(1), 193–218 (1985)
12. Vinh, N.X., Epps, J., Bailey, J.: Information theoretic measures for clusterings comparison: Is a correction for chance necessary? In: ICML 2009: Proceedings of the 26th international conference on Machine learning (2009)
13. Yeung, K.Y.: Cluster analysis of gene expression data. PhD thesis, University of Washington, Seattle, WA (2001)
14. Yeung, K., Medvedovic, M., Bumgarner, R.: Clustering gene-expression data with repeated measurements. *Genome Biology* 4(5), R34 (2003)
15. Geusebroek, J.M., Burghouts, G.J., Smeulders, A.W.M.: The amsterdam library of object images. *Int. J. Comput. Vision* 61(1), 103–112 (2005)
16. Boujemaa, N., Fauqueur, J., Ferecatu, M., Fleuret, F., Gouet, V., LeSaux, B., Sahbi, H.: Ikona: Interactive specific and generic image retrieval. In: International workshop on Multimedia ContentBased Indexing and Retrieval, MMCBIR 2001 (2001)
17. Lewis, D.D., Yang, Y., Rose, T.G., Li, F.: RCV1: A new benchmark collection for text categorization research. *J. Mach. Learn. Res.* 5, 361–397 (2004)
18. Asuncion, A., Newman, D.: UCI machine learning repository (2007)
19. Dasgupta, S.: Experiments with random projection. In: UAI 2000: Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence, pp. 143–151. Morgan Kaufmann Publishers Inc., San Francisco (2000)

iVAT and aVAT: Enhanced Visual Analysis for Cluster Tendency Assessment

Liang Wang¹, Uyen T.V. Nguyen², James C. Bezdek²,
Christopher A. Leckie², and Kotagiri Ramamohanarao²

¹ Department of Computer Science
University of Bath, BA2 7AY, United Kingdom
lw356@cs.bath.ac.uk

² Department of Computer Science and Software Engineering
The University of Melbourne, Victoria 3010, Australia
{thivun, caleckie, rao}@csse.unimelb.edu.au

Abstract. Given a pairwise dissimilarity matrix \mathbf{D} of a set of n objects, visual methods (such as VAT) for cluster tendency assessment generally represent \mathbf{D} as an $n \times n$ image $I(\tilde{\mathbf{D}})$ where the objects are reordered to reveal hidden cluster structure as dark blocks along the diagonal of the image. A major limitation of such methods is the inability to highlight cluster structure in $I(\tilde{\mathbf{D}})$ when \mathbf{D} contains highly complex clusters. To address this problem, this paper proposes an improved VAT (iVAT) method by combining a path-based distance transform with VAT. In addition, an automated VAT (aVAT) method is also proposed to automatically determine the number of clusters from $I(\tilde{\mathbf{D}})$. Experimental results on several synthetic and real-world data sets have demonstrated the effectiveness of our methods.

Keywords: Visual cluster analysis, cluster tendency assessment, VAT, path-based distance, chamfer matching.

1 Introduction

A general question in the pattern recognition and data mining community is how to organize observed data into meaningful structures or taxonomies. As such, cluster analysis aims at grouping objects of a similar kind into their respective categories. Given a data set \mathcal{O} comprising n objects $\{o_1, o_2, \dots, o_n\}$, (crisp) clustering partitions the data into c groups C_1, C_2, \dots, C_c , so that $C_i \cap C_j = \emptyset$, if $i \neq j$ and $C_1 \cup C_2 \cup \dots \cup C_c = \mathcal{O}$. There have been a large number of clustering algorithms reported in the recent literature [1]. In general, clustering of unlabeled data poses three major problems: (1) assessing cluster tendency, *i.e.*, how many groups to seek or what is the value of c ? (2) partitioning the data into c groups; and (3) validating the c clusters discovered. Given a pairwise dissimilarity matrix $\mathbf{D} \in \mathcal{R}^{n \times n}$ of \mathcal{O} , this paper addresses the problem of determining the number of clusters *prior* to clustering.

Most clustering algorithms require the number of clusters c as an input, so the quality of the resulting clusters is largely dependent on the estimation of c . Various attempts have been made to estimate c . However, most existing methods are *post-clustering* measures of cluster validity [2,1,3,4,5,6,7]. In contrast, tendency assessment attempts to estimate c before clustering occurs. Visual methods for cluster tendency assessment [8,9,10,11,12,13,14,15] generally represent pairwise dissimilarity information about a set of n objects as an $n \times n$ image, where the objects are reordered so that the resulting image is able to highlight potential cluster structure in the data. A “useful” reordered dissimilarity image (RDI) highlights potential clusters as a set of “dark blocks” along the diagonal of the image, and can be viewed as a visual aid to tendency assessment.

Our work is built upon one method for generating reordered dissimilarity images, namely VAT (Visual Assessment of cluster Tendency) of Bezdek and Hathaway [8]. Several algorithms extend VAT for related assessment problems. For example, bigVAT [13] and sVAT [11] offer different ways to approximate the VAT reordered dissimilarity image for very large data sets. CCE [16] and DBE [17] use different schemes to automatically estimate the number of clusters in the VAT images. In addition, Havens *et al.* [18] perform data clustering in ordered dissimilarity images, and coVAT [10] extends the VAT idea to rectangular dissimilarity data. Naturally, the performance of these VAT-based methods is greatly dependent of the quality of the VAT images. However, while VAT has been widely used for cluster analysis, it is usually only effective at highlighting cluster tendency in data sets that contain compact well-separated clusters. Many practical applications involve data sets with highly irregular structure, which invalidate this assumption. In this paper, we propose an improved VAT (iVAT) approach to generating RDIs that combines VAT with a path-based distance transform. The resulting iVAT images can clearly show the number of clusters and their approximate sizes for data sets with highly complex cluster structures. We also propose a new strategy for automated determination of the number of clusters c from RDIs, by detecting and counting dark blocks along the main diagonal of the image. Experimental results on both synthetic and real-world data sets validate our methods.

The remainder of the paper is organized as follows: Section 2 briefly reviews the VAT algorithm. Section 3 illustrates our iVAT algorithm. Section 4 presents our strategy for automatically determining the number of clusters c . The experimental results on both synthetic and real-world data sets are given and analyzed in Section 5, prior to conclusion in Section 6.

2 VAT

Let $\mathcal{O} = \{o_1, o_2, \dots, o_n\}$ denote n objects in the data and \mathbf{D} a pairwise matrix of dissimilarities between objects, each element of which $d_{ij} = d(o_i, o_j)$ is the dissimilarity between objects o_i and o_j , and generally, satisfies $1 \geq d_{ij} \geq 0$; $d_{ij} = d_{ji}$; $d_{ii} = 0$, for $1 \leq i, j \leq n$. Let $\pi()$ be a permutation of $\{1, 2, \dots, n\}$ such that $\pi(i)$ is the new index for o_i . The reordered list is thus $\{o_{\pi(1)}, \dots, o_{\pi(n)}\}$. Let \mathbf{P}

be the permutation matrix with $p_{ij} = 1$ if $j = \pi(i)$ and 0 otherwise, then the matrix $\tilde{\mathbf{D}}$ for the reordered list is a similarity transform of \mathbf{D} by \mathbf{P} , *i.e.*,

$$\tilde{\mathbf{D}} = \mathbf{P}^T \mathbf{D} \mathbf{P}.$$

The reordering idea is to find \mathbf{P} so that $\tilde{\mathbf{D}}$ is as close to a block diagonal form as possible. The VAT algorithm [8] reorders the row and columns of \mathbf{D} with a modified version of Prim’s minimal spanning tree algorithm, and displays a reordered dissimilarity matrix $\tilde{\mathbf{D}}$ as a gray-scale image. If an object is a member of a cluster, then it should be part of a sub-matrix with low dissimilarity values, which appears as one of the dark blocks along the diagonal of the VAT image $\mathbf{I}(\tilde{\mathbf{D}})$, each of which corresponds to one potential cluster.

Figure 1(a) is a scatter plot of 2000 data points in \mathcal{R}^2 . The 5 visually apparent clusters are reflected by the 5 distinct dark blocks along the main diagonal in Figure 1(c), which is the VAT image of the data. Given the image of \mathbf{D} in the original input order in Figure 1(b), reordering is necessary to reveal the underlying cluster structure of the data. VAT reordering produces neither a partition nor a hierarchy of clusters. It merely reorders the data to (possibly) reveal its hidden structure, which can be viewed as an illustrative data visualization for estimating c . Sometimes, hierarchical structure can be detected by the presence of diagonal sub-blocks within larger diagonal blocks.

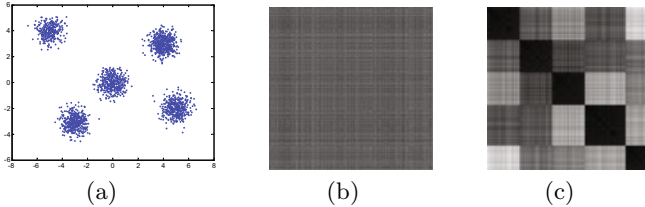


Fig. 1. An example of the VAT algorithm

3 Improved VAT (iVAT)

At a glance, a viewer can estimate the number of clusters c from a VAT image by counting the number of dark blocks along the diagonal if these dark blocks possess visual clarity. However, this is not always possible. Note that a dark block appears only when a compact group exists in the data. For complex-shaped data sets where the boundaries between clusters become less distinct due to either significant overlap or irregular geometries, the resulting VAT images may fail to produce dark blocks even when cluster structure is clearly present. See Figures 4(b) and 5(a) for examples. Different viewers may deduce different numbers of clusters from such poor-quality images, or worse, not be able to estimate c at all. This raises the question of whether we can transform \mathbf{D} into a new form \mathbf{D}' so that the VAT image of \mathbf{D}' is clearer and more informative about the cluster structure.

In [12], SpecVAT combines VAT with graph embedding [19,20] to solve this problem. SpecVAT first embeds the data into a k -dimensional subspace spanned by the eigenvectors of the normalized Laplacian matrix and then re-computes a new pairwise dissimilarity matrix in the embedding subspace as the input of the VAT algorithm. However, this method depends on two main parameters, one of which is r for the r -th nearest neighbor based local scale computation when constructing the affinity matrix from \mathbf{D} [21] (for deriving the Laplacian matrix), and the other is k , the number of eigenvectors used. In particular, k largely depends on the number of clusters c . Figure 2 gives an example of SpecVAT images with respect to different values of k . Since c is unknown, a range of k values need to be used for generating a series of SpecVAT images to find the ‘best’ SpecVAT image that is truly informative of the real structure in the data (e.g., $k = 3$ in this case). In contrast, this work adopts a *parameter-free* method (called iVAT) by combining VAT with a path-based distance transform.

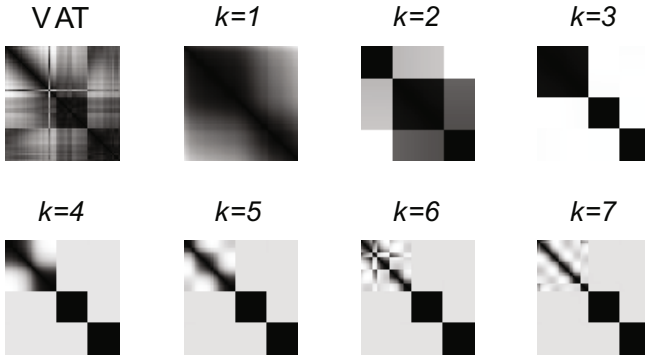


Fig. 2. An example of SpecVAT on synthetic three-circle data set S-3 ($c = 3$)

The path-based dissimilarity measure was introduced in [22]. The intuitive idea is that if two objects o_i, o_j are very far from each other (reflected by a large distance value d_{ij}), but there is a path connecting them consisting of other objects such that the distances between any two successive objects are small, then d_{ij} should be adjusted to a smaller value to reflect this connection. The adjustment of d_{ij} reflects the idea that no matter how far the distance between two objects may be, they should be considered as coming from one cluster if they are connected by a set of successive objects forming dense regions. This reflects the characteristic of elongated clusters.

Let us treat \mathbf{D} as a fully connected graph \mathcal{G} , where each vertex corresponds to an object and the edge weight between vertices i and j is the distance d_{ij} . Suppose that P_{ij} is the set of all possible paths from o_i to o_j , then for each path $p \in P_{ij}$, the effective dissimilarity between objects o_i and o_j along p is the maximum of all edge weights belonging to this path. The path-based distance d'_{ij} is defined as

$$d'_{ij} = \min_{p \in P_{ij}} \left\{ \max_{1 \leq h < |p|} d_{p[h]p[h+1]} \right\}$$

where $p[h]$ denotes the object at the h -th position in path p and $|p|$ denotes the length of path p . After obtaining $\mathbf{D}' = [d'_{ij}]$, we reorder it using the VAT algorithm to obtain the iVAT image (see examples in Figures 4(c) and 5(b)). The iVAT images are almost always clearer and more informative than the original VAT images in revealing the data structure.

4 Automated VAT (aVAT)

A viewer can simply estimate the number of clusters c (*i.e.*, count the number of dark blocks along the diagonal of a RDI image if these dark blocks possess visual clarity). However, as the boundaries between different clusters become less distinct, the RDI image will degrade considerably with confusing boundaries between potential dark blocks. Accordingly, different viewers may deduce different numbers of clusters from such poor-quality images. Can we automatically determine the number of clusters c , as suggested by $I(\tilde{\mathbf{D}}')$, in an objective manner, without viewing the visual display? To answer this interesting question, two methods have been developed, *i.e.*, DBE [17] and CCE [16] (see the algorithm details and comparison in Section 5.3). Here we propose an alternative method, called aVAT, using some image processing techniques. The process of aVAT is illustrated in Figure 3. The individual steps are all well known in the field of image processing, so we do not describe their underlying theories.

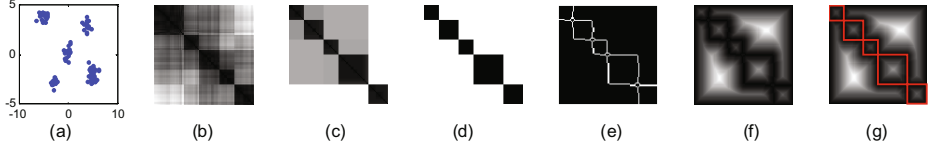


Fig. 3. Illustration of the aVAT algorithm. From left to right: scatter plot ($n = 100, c = 5$), VAT image of (a), iVAT image of (a), binarized image of (c), edge map of (d), DT image of (e), and detected squares in (f) imposed using red lines.

Since information about possible cluster structure in the data is embodied in the *square dark blocks* along the diagonal of a RDI, we propose to detect and count them using shape-based Chamfer matching [23]. As a preprocessing step, the RDI is firstly binarized to extract regions of interest (Figure 3(d)). Otsu’s method [24] is used to automatically choose a global threshold. To make within-cluster distances smaller and between-cluster distances larger (*i.e.*, increasing contrast) to obtain a more reliable threshold, we transform the image intensities using a “monotonic” function

$$f(t_{xy}) = 1 - \exp(-t_{xy}^2/\sigma^2)$$

where t_{xy} denotes the intensity value of the image pixel on the location (x, y) , and σ is empirically set as the mean value of all pixel intensities.

Chamfer matching was first proposed by Barrow *et al.* [25]. Assume that two point sets are $\mathcal{U} = \{\mathbf{u}_i\}_{i=1}^N$ and $\mathcal{V} = \{\mathbf{v}_i\}_{i=1}^M$, the chamfer distance is defined as

$$d_{cham}(\mathcal{U}, \mathcal{V}) = \frac{1}{N} \sum_{\mathbf{u}_i \in \mathcal{U}} \min_{\mathbf{v}_j \in \mathcal{V}} \|\mathbf{u}_i - \mathbf{v}_j\|.$$

The symmetric chamfer distance can be obtained by adding $d_{cham}(\mathcal{V}, \mathcal{U})$. The chamfer distance between two shapes can be efficiently computed using a distance transform (DT, Figure 3(f)), which takes a binary image as input, and assigns to each pixel in the image the distance to its nearest feature. We use Canny edges as image feature points (Figure 3(e)) and the Euclidean distance for DT, and the model points are the projected contours of a 2D (rigid) square template. The distance between the template and the edge map can then be computed as the mean of the DT values at the template point coordinates.

In general, matching consists of translating, rotating and scaling the template shape at various locations of the distance image. Fortunately, in the RDI, we just need to search for squares along the diagonal axis and scale the template to different sizes to adapt to various cluster sizes. There is no need for template rotation, because there are no orientation changes in VAT RDIs. This greatly reduces the complexity of common shape detection using Chamfer matching. The exact matching cost is ideally 0, but in practice the edges in an image are slightly displaced from their ideal locations. Thus in our experiments, when the matching cost lies below a certain threshold τ , the target shape is considered to have been detected (Figure 3(g)).

5 Experimental Results

In order to evaluate our methods, we have carried out a number of experiments on 6 artificially generated data sets, as well as 6 real-world data sets. Unless otherwise mentioned, in the following experiments the (Euclidean) distance matrix \mathbf{D} was computed in the attribute space (if the object vectorial representation is available). All experiments were implemented in a Matlab 7.2 environment on a PC with an Intel 2.4GHz CPU and 2GB memory running Windows XP.

5.1 Test Datasets

Six synthetic data sets were used in our experiments, whose scatter plots are shown in Figure 4(a). These data sets involve irregular data structures, in which an obvious cluster centroid for each group is not necessarily available. Six real-world data sets were also considered to evaluate our algorithms, 3 of which were taken from the UCI Machine Learning Repository, *i.e.*, Iris, Vote and Multiple Features. The *Face* data set [26] contains 1755 images of 3 individuals, each of which was down-sampled to 30×40 pixels. The *Gene* data set [27] is a 194×194 matrix consisting of pairwise dissimilarities of a set of gene products from 3 protein families. The *Iris* data set contains 3 types of iris plants, 50 instances each. The *Vote* data set consists of 435 vote records (267 democrats and 168

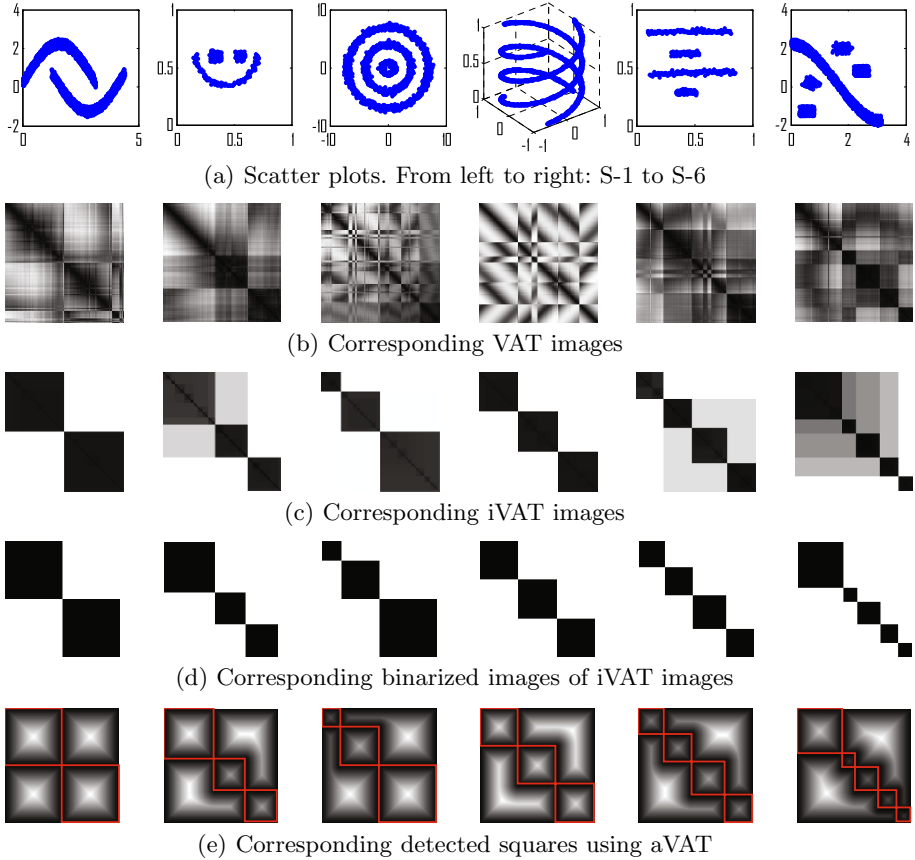


Fig. 4. Visual analysis on 6 synthetic data sets

republicans). Votes were numerically encoded as 0.5 for “yea”, -0.5 for “nay” and 0 for “unknown disposition”. The *Action* data set [28] is an 198×198 pairwise dissimilarity matrix derived from 198 human action clips. The *Multiple-Features* (MF) data set consists of binary image features of 10 handwritten numerals, 200 patterns per class. The characteristics of these synthetic and real data sets are summarized in Table 1.

5.2 Results and Analysis

For each of the data sets, we first applied the VAT algorithm. The VAT images are shown in Figure 4(b) for synthetic data and Figure 5(a) for real data, respectively. It can be seen that the cluster structure of the data in the VAT images is not necessarily clearly highlighted, especially for complex-shaped data. Accordingly, viewers have difficulties in giving a consistent result about the number of clusters, and different viewers may deduce different estimates of c . Next

Table 1. Summary of results of estimating c using different methods

| Data | | | | VAT | | | | iVAT | | | |
|--------|-------|----------|------|----------|----------|----------|----------|-----------|----------|----------|----------|
| Name | c_p | # attri. | n | Manual | DBE | CCE | aVAT | Manual | DBE | CCE | aVAT |
| S-1 | 2 | 2 | 2000 | ≥ 1 | 6 | 3 | 6 | 2 | 2 | 2 | 2 |
| S-2 | 3 | 2 | 266 | ≥ 2 | 5 | 3 | 3 | 3 | 3 | 3 | 3 |
| S-3 | 3 | 2 | 1800 | ≥ 1 | 10 | 11 | 11 | 3 | 3 | 3 | 3 |
| S-4 | 3 | 3 | 3000 | - | 8 | 9 | 7 | 3 | 3 | 3 | 3 |
| S-5 | 4 | 2 | 512 | ≥ 1 | 5 | 5 | 5 | 4 | 4 | 4 | 4 |
| S-6 | 5 | 2 | 2500 | ≥ 5 | 8 | 6 | 5 | 5 | 5 | 5 | 5 |
| Vote | 2 | 16 | 435 | ≥ 2 | 2 | 2 | 3 | 2 | 2 | 3 | 2 |
| Iris | 3 | 4 | 150 | 2 | 3 | 2 | 2 | 2 | 2 | 2 | 2 |
| Gene | 3 | - | 194 | ≥ 3 | 3 | 4 | 3 | 3 | 5 | 3 | 4 |
| Face | 3 | 1200 | 1755 | 3 or 4 | 4 | 3 | 6 | 4 | 4 | 5 | 5 |
| Action | 10 | - | 198 | ≥ 9 | 8 | 7 | 7 | 10 | 7 | 7 | 7 |
| MF | 10 | 649 | 2000 | ≥ 8 | 9 | 5 | 12 | 8 | 7 | 6 | 9 |
| AAE | | | | - | 2.17 | 2.25 | 2.25 | 0.33 | 0.83 | 0.92 | 0.67 |
| ARE | | | | - | 0.73 | 0.59 | 0.66 | 0.07 | 0.16 | 0.18 | 0.14 |

we carried out our iVAT algorithm for each of the data sets used. The resulting iVAT images are shown in Figure 4(c) for synthetic data and Figure 5(b) for real data, respectively. In contrast to the original VAT images, the iVAT images have generally clearer displays in terms of block structure, thus better highlighting the hidden cluster structure.

Table 1 summarizes the number of clusters determined from iVAT images automatically, along with the results estimated from the VAT and iVAT images using manual inspection by the authors for comparison. From Table 1, we can see that

1. The results estimated from the iVAT images by manual inspection are clearly better than those estimated from the original VAT images by manual inspection, whether for synthetic or real-world data sets.
2. The results of cluster number estimation from the iVAT images for all synthetic data sets are accurate in terms of the number of real physical classes (c_p), whether it was estimated automatically by our aVAT algorithm or by manual inspection.
3. For real-world data sets, some estimates deviate slightly from the number of real physical classes using our aVAT algorithm.

Overall, these results highlight the benefits of converting \mathbf{D} to \mathbf{D}' by the path-based distance transform for obtaining a good estimation of c (whether automatically or manually). We would like to note several points:

1. Though some estimates are imperfect for the real data, the aVAT algorithm correctly detected all squares in the binarized images (see Figure 5(c)). This suggests that we may need to seek more sophisticated methods of image binarization (*e.g.*, multiple local thresholds) for avoiding the loss of some

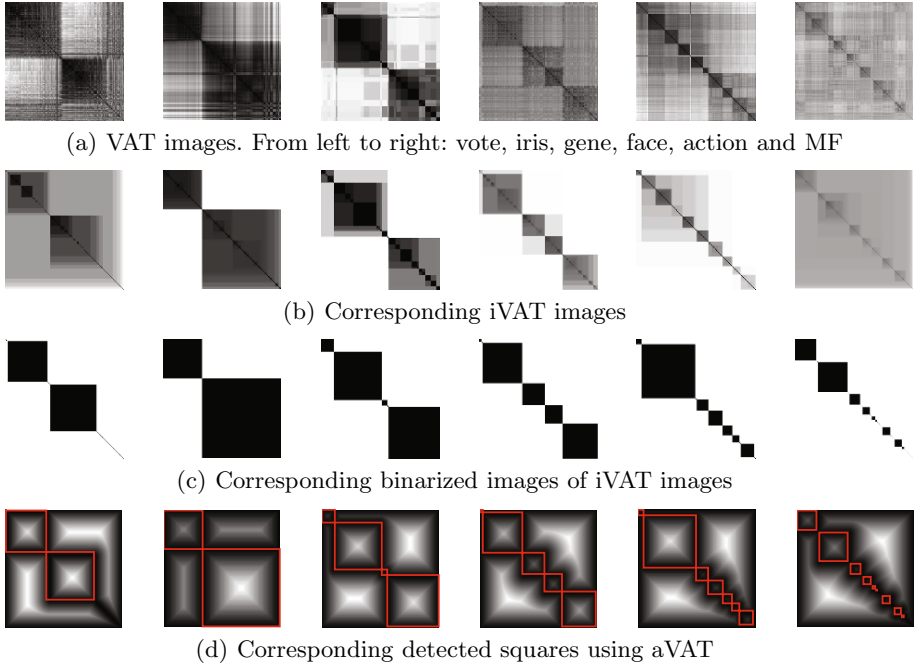


Fig. 5. Visual analysis on 6 real data sets

physically meaningful blocks (*e.g.*, for Action data, some small blocks corresponding to different action classes were transferred into a bigger block after binarization).

2. We could use a ‘square size’ threshold to filter some detected very ‘small’ blocks corresponding to either noise/inliers or subtle sub-structures (*e.g.*, for Face and Gene data sets).
3. As a side product, for ‘perfect’ iVAT images (such as those of the 6 synthetic data sets), the exactly detected squares may be directly used to retrieve data partitions (*i.e.*, each square corresponds to one potential cluster and its size corresponds to the cluster size).

5.3 Algorithm Comparison

We compared our aVAT algorithm with CCE [16] and DBE [17] in terms of estimating the number of clusters. Note that we did not compare aVAT to indexed methods for post-clustering assessment of cluster validity, as our interest is in estimating the number of clusters *before* clustering. The major steps for CCE are summarized as follows: 1) Threshold the VAT image with Otsu’s algorithm; 2) Apply the FFT to the segmented VAT and a correlation filter of size s and multiply the transformed image with the complex conjugate of the transformed filter; 3) Compute the inverse FFT for the filtered image; 4) Take

the q -th off-diagonal pixel values of the back-transformed image and compute its histogram; and 5) Cut the histogram at a horizontal line $y = b$, and count the number of spikes. The major steps of DBE are summarized as follows: 1) Perform intensity transform and segmentation of the VAT image, followed by directional morphological filtering with size of αn ; 2) Apply a distance transform to the filtered image and project the pixel values onto the main diagonal axis to form a projection signal; 3) Smooth the projection signal by an average filter with a length of $2\alpha n$, compute its first-order derivative, and then detect the number of major peaks by ignoring minor ones using a filter with size of $2\alpha n$.

As suggested in [16,17], we used $s = 20$, $q = 1$ and $b = 0$ for CCE and $\alpha = 0.03$ for DBE. We used both VAT and iVAT images to make our comparisons, and the results are summarized in Table II, in which we used *bold* figures to show that the estimate is equal to the number of real physical classes c_p and *italic* figures to show results that are relatively closer to c_p . AAE and ARE represent average absolute error and average relative error between the number of estimated clusters and the number of real physical classes, respectively. From Table II, it can be seen that:

1. For synthetic data sets, all of these three methods give correct results when using the iVAT images, while aVAT and CCE are slightly better than DBE when using the original VAT images (*i.e.*, 2 correct and 2 closer for aVAT, 1 correct and 3 closer for CCE, and 2 closer to DBE).
2. For real-world data sets plus the use of VAT images, DBE performs best, then CCE and finally aVAT (*i.e.*, 3 correct and 3 closer for DBE, 2 correct and 2 closer for CCE, and 1 correct and 2 closer for aVAT); while for real-world data sets plus the use of iVAT images, aVAT is a little better than both CCE and DBE (*i.e.*, 1 correct and 4 closer for aVAT, and 1 correct and 3 closer for both CCE and DBE).
3. Specifically, when using iVAT images, aVAT, CCE and DBE yield the same estimate for the Iris and Action data sets. They all yield acceptable (but different) estimates for the Gene and Face data sets. They disagree for the Vote and MF data sets.

Overall, these three methods are comparable to each other and there is no clear winner (at least based on the results on these data sets used currently). However, we can see that the positions of peaks and valleys in the projection signal in DBE *implicitly* correspond to centers and ranges of sub-blocks (or clusters). It is hard to see a similar phenomena from the CCE histograms. In contrast, aVAT is better in this aspect because it *explicitly* shows the number of clusters, positions and ranges of each block (or clusters) within the image itself, in a more intuitive manner.

6 Conclusion

This paper has presented a new visual technique for cluster tendency assessment. Our contributions include: 1) The VAT algorithm was enhanced by using

a path-based distance transform. The iVAT algorithm can better reveal the hidden cluster structure, especially for complex-shaped data sets. 2) Based on the iVAT image, the cluster structure in the data can be reliably estimated by visual inspection. As well, the aVAT algorithm was proposed for automatically determining the number of clusters c . 3) We performed a series of primary and comparative experiments on 6 synthetic data sets and 6 real-world data sets, and our methods obtained encouraging results.

In addition to further performance evaluation on more data sets with various structures, future work will mainly focus on increasing the robustness of our algorithms, *e.g.*, exploring more sophisticated image thresholding methods [29] and robust path-based distance computation [30].

References

1. Xu, R., II, D.W.: Survey of clustering algorithms. *IEEE Transactions on Neural Networks* 16(3), 645–678 (2005)
2. Maulik, U., Bandyopadhyay, S.: Performance evaluation of some clustering algorithms and validity indices. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24(12), 1650–1654 (2002)
3. Hu, X., Xu, L.: A comparative study of several cluster number selection criteria. In: Liu, J., Cheung, Y.-m., Yin, H. (eds.) *IDEAL 2003*. LNCS, vol. 2690, pp. 195–202. Springer, Heidelberg (2003)
4. Bezdek, J.C., Pal, N.R.: Some new indices of cluster validity. *IEEE Transactions on System, Man and Cybernetics* 28(3), 301–315 (1998)
5. Tibshirani, R., Hastie, G.W., T.: Estimating the number of clusters in a dataset via the gap statistics. *Journal of the Royal Statistical Society. Series B, Statistical Methodology* 63(2), 411–423 (2001)
6. Calinski, R.B., Harabasz, J.: A dendrite method for cluster analysis. *Communications in Statistics* 3(1), 1–27 (1974)
7. Dunn, J.C.: Indices of partition fuzziness and the detection of clusters in large sets. *Fuzzy Automata and Decision Processes* (1976)
8. Bezdek, J.C., Hathaway, R.J.: VAT: A tool for visual assessment of (cluster) tendency. In: *International Joint Conference on Neural Networks*, vol. 3, pp. 2225–2230 (2002)
9. Tran-Luu, T.: *Mathematical Concepts and Novel Heuristic Methods for Data Clustering and Visualization*. PhD Thesis, University of Maryland, College Park, MD (1996)
10. Bezdek, J.C., Hathaway, R., Huband, J.: Visual assessment of clustering tendency for rectangular dissimilarity matrices. *IEEE Transactions on Fuzzy Systems* 15(5), 890–903 (2007)
11. Hathaway, R., Bezdek, J.C., Huband, J.: Scalable visual assessment of cluster tendency. *Pattern Recognition* 39(7), 1315–1324 (2006)
12. Wang, L., Geng, X., Bezdek, J., Leckie, C., Kotagiri, R.: SpecVAT: Enhanced visual cluster analysis. In: *International Conference on Data Mining*, pp. 638–647 (2008)
13. Huband, J., Bezdek, J.C., Hathaway, R.: bigVAT: Visual assessment of cluster tendency for large data sets. *Pattern Recognition* 38(11), 1875–1886 (2005)
14. Ling, R.: A computer generated aid for cluster analysis. *Communications of the ACM* 16(6), 355–361 (1973)

15. Rousseeuw, P.J.: A graphical aid to the interpretations and validation of cluster analysis. *Journal of Computational and Applied Mathematics* 20(1), 53–65 (1987)
16. Sledge, I., Huband, J., Bezdek, J.C. (Automatic) cluster count extraction from unlabeled datasets. In: *Joint International Conference on Natural Computation and International Conference on Fuzzy Systems and Knowledge Discovery*, vol. 1, pp. 3–13 (2008)
17. Wang, L., Leckie, C., Kotagiri, R., Bezdek, J.: Automatically determining the number of clusters in unlabeled data sets. *IEEE Transactions on Knowledge and Data Engineering* 21(3), 335–350 (2009)
18. Havens, T.C., Bezdek, J.C., Keller, J.M., Popescu, M.: Clustering in ordered dissimilarity data. *International Journal of Intelligent Systems* 24(5), 504–528 (2009)
19. Belkin, M., Niyogi, P.: Laplacian eigenmaps and spectral techniques for embedding and clustering. *Advances in Neural Information Processing Systems* 14, 585–591 (2002)
20. Chung, F.: Spectral graph theory. In: *CBMS Regional Conference Series in Mathematics*, American Mathematical Society, vol. 92 (1997)
21. Zelnik-Manor, L., Perona, P.: Self-tuning spectral clustering. *Advances in Neural Information Processing Systems* 17, 1601–1608 (2004)
22. Fisher, B., Zoller, T., Buhmann, J.: Path based pairwise data clustering with application to texture segmentation. In: *Figueiredo, M., Zerubia, J., Jain, A.K. (eds.) EMMCVPR 2001. LNCS*, vol. 2134, pp. 235–250. Springer, Heidelberg (2001)
23. Thayananthan, A., Stenger, B., Torr, P., Cipolla, R.: Shape context and chamfer matching in cluttered scenes. In: *International Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 127–133 (2003)
24. Otsu, N.: A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics* 9(1), 62–66 (1979)
25. Barrow, H.G., tenenbaum, J.M., Bolles, R.C., Wolf, H.C.: Parametric correspondence and chamfer matching: Two new techniques for image matching. In: *International Joint Conference on Artificial Intelligence*, vol. 2, pp. 659–663 (1977)
26. Breitenbach, M., Grudic, G.: Clustering through ranking on manifolds. In: *International Conference on Machine Learning*, vol. 119, pp. 73–80 (2005)
27. Pal, N., Keller, J., Popescu, M., Bezdek, J.C., Mitchell, J., Huband, J.: Gene ontology-based knowledge discovery through fuzzy cluster analysis. *Journal of Neural, Parallel and Scientific Computing* 13(3-4), 337–361 (2005)
28. Wang, L., Leckie, C., Wang, X., Kotagiri, R., Bezdek, J.: Tensor space learning for analyzing activity patterns from video sequences. In: *ICDM Workshop on Knowledge Discovery and Data Mining from Multimedia Data and Multimedia Applications*, pp. 63–68 (2007)
29. Sezgin, M., Sankur, B.: Survey over image thresholding techniques and quantitative performance evaluation. *Journal of Electronic Imaging* 13(1), 146–165 (2004)
30. Chang, H., Yeung, D.Y.: Robust path-based spectral clustering with application to image segmentation. In: *International Conference on Computer Vision*, vol. 1, pp. 278–285 (2005)

A Robust Seedless Algorithm for Correlation Clustering

Mohammad S. Aziz and Chandan K. Reddy

Department of Computer Science,
Wayne State University, Detroit, MI, USA
maziz@wayne.edu, reddy@cs.wayne.edu

Abstract. Finding correlation clusters in the arbitrary subspaces of high-dimensional data is an important and a challenging research problem. The current state-of-the-art correlation clustering approaches are sensitive to the initial set of seeds chosen and do not yield the optimal result in the presence of noise. To avoid these problems, we propose RObust SEedless Correlation Clustering (ROSECC) algorithm that does not require the selection of the initial set of seeds. Our approach incrementally partitions the data in each iteration and applies PCA to each partition independently. ROSECC does not assume the dimensionality of the cluster beforehand and automatically determines the appropriate dimensionality (and the corresponding subspaces) of the correlation cluster. Experimental results on both synthetic and real-world datasets demonstrate the effectiveness of the proposed method. We also show the robustness of our method in the presence of a significant noise levels in the data.

Keywords: Correlation clustering, principal component analysis.

1 Introduction

Clustering is one of the most popular techniques in the field of data mining [9]. The basic idea of clustering is to partition the data in such a way that the members of a partition are closer to each other and the members of different partitions are far apart. But many real-world applications often suffer from the “*curse of dimensionality*” and the measure of nearness becomes meaningless. In such scenarios, many feature selection and dimensionality reduction methods have been proposed to aid clustering [7]. However, these methods reduce the dimensionality by optimizing a certain criterion function and do not address the problem of data clustering directly. The result of using dimensionality reduction techniques for clustering high-dimensional data is far from satisfactory and rarely used in such scenarios. Moreover, it is possible that different clusters lie in different subspaces and thus cannot be identified using any dimensionality reduction or feature selection method (see Fig. 1(a)). To deal with such cases, subspace clustering methods such as CLIQUE [4], ENCLUS [6], SUBCLU [10] etc. have been proposed in the literature (see Fig. 1(b)). These methods attempt to find clusters in different subspaces. Searching every possible subspace is a computationally intensive task due to exponentially large search space. Thus, most of these methods use some form of an Apriori-based approach to identify the most interesting subspaces. Projected clustering [3,12] is one form of subspace clustering that uses the concept of projection. However, these subspace clustering or projected clustering methods have the following two limitations:

1. They are only able to separate clusters that are oriented in axis-parallel manner. They cannot find cluster in arbitrary-oriented subspace (see Fig. 1(c)).
2. They can only find clusters in the sense of locality.

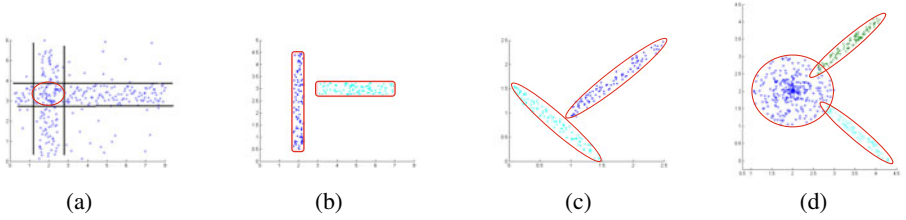


Fig. 1. Different kinds of correlation clusterings. (a) Projected clustering (b) Axis-Parallel correlation clusters. (c) Arbitrary oriented subspace clusters. (d) Arbitrary oriented subspace clusters with different dimensionality.

In subspace clustering, the clusters are formed in the subspaces rather than the full-dimensional space [4,6]. Finding arbitrary-oriented subspace cluster (Fig. 1(c)) involves exponential search space. To efficiently address the problem, most correlation based approaches use PCA to avoid searching unwanted regions of the search space. ORCLUS [2], which uses the same idea of axis-parallel PROCLUS [3], is the first PCA based method to find correlation clusters. 4C [5] also uses PCA, however it incorporates a density-based approach and hence, the number of clusters need not be pre-specified. Although these methods are able to keep the computational complexity low and do not suffer from a potentially infinite search space, their success in finding correlation clusters is highly dependent on the initial choice of the seeds. In addition, they usually do not produce optimal results in the presence of noise. Another problem with these methods is that the dimensionality of the correlation has to be pre-defined which is usually difficult for the end-user from the practical viewpoint. Yip et al. [13] proposed an algorithm called HARP [13], which exploits the data to adjust the internal threshold values dynamically at the runtime. However, this method faces difficulties in obtaining a low-dimensional cluster [11]. To avoid the problem associated with PCA-based methods such as choosing the initial set of seeds and susceptibility to noise, CASH algorithm [1] uses Hough transform to find correlations. Though this method does not use any initial seeds, its worst case time complexity is exponential, thus making it impractical for high-dimensional data. To solve these issues, we propose a novel PCA based algorithm which eliminates the problem of susceptibility to noise and the need for initial seeds to find correlation clusters. It can also simultaneously find correlation clusters of different dimensionality (see Fig. 1(d)) and the computational complexity is relatively low.

In this paper, we propose RObust SEedless Correlation Clustering (ROSECC) algorithm to find the correlation clusters in high-dimensional data. The main advantages of the ROSECC algorithm compared to the state-of-the-art methods proposed in the literature are:

1. It does not require initial seeds for the clustering and hence it is deterministic.
2. It can simultaneously identify correlation clusters with different number of dimensions.

3. It is robust to handle noise in the data and can obtain the desired correlation clusters despite the presence of significant noise levels in the data.

The rest of this paper is organized as follows: Section 2 describes the necessary definitions and notations. The proposed algorithm along with its computational complexity is described in Section 3. Section 4 outlines the experimental results on both synthetic and real-world datasets. Finally, Section 5 concludes our discussion.

2 Preliminaries

In this section, we will introduce some definitions that are needed to comprehend our algorithm. Our method is based on the projection distance of a point onto the principal vectors. Let $dp \in D$ is a datapoint and v is the principal vector of a member P_i of Partitionset P , then the projection distance of dp and P_i is given by the following equation:

$$PDist(dp, P_i) = \begin{cases} dp.v & \text{if } |P_i| > 1. \\ dist(dp, x) & \text{if } |P_i| = 1. \end{cases}$$

where $dist(dp, x) = \sqrt{\sum_j (dp_j - x_j)^2}$. dp_j and x_j corresponds to the j^{th} feature of the data points dp and x respectively. At the beginning of the generation of partitions, there will be only one single point x without any principal component. In such cases, the Euclidean distance from the point x will be used as the projection distance. A “Partitionset” (denoted by P_i) is defined as a set of datapoints which have lower projection distance to a particular component compared to the projection distance to any other component. A “Partition” is defined as a set of Partitionsets.

Definition 1. (Nearest Component): Let $dp \in D$ and $P = \{P_i\}$ where P_i 's are components, the nearest component of dp from P , denoted by $NComp(dp, P)$, is defined as the component for which the projection distance from dp is minimum. $NComp(dp, P) = \underset{P_i \in P}{\operatorname{argmin}}(PDist(dp, P_i))$. The corresponding distance is the “minimum projection”, denoted by $MProj(dp, P) = \min_{P_i \in P}(PDist(dp, P_i))$

Definition 2. (Farthest Candidate): Let $P = \{P_i\}$ where P_i is a Partitionset. The farthest candidate of P , denoted by $FCand(P)$, is defined as the data point for which the minimum projection distance is maximum. $FCand(P) = \underset{dp \in D}{\operatorname{argmax}}(MProj(dp, P))$

Theorem 1. Let a new component consisting of a single element $\{dp'\}$ is added to the component list P to become P' i.e. if $P' = P \cup \{dp'\}$. If $MProj(dp, P) > dist(dp', dp)$ then $MProj(dp, P') = dist(dp', dp)$

Proof

From the definition of the projection distance, we have $PDist(dp, \{dp'\}) = dist(dp', dp)$. Therefore, $MProj(dp, P) > dist(dp', dp) \Rightarrow MProj(dp, P) > PDist(dp, \{dp'\})$. Also, from the definition of the Minimum Projection, we get, $MProj(dp, P) \leq \forall_{P_i \in P} PDist(dp, P_i) \Rightarrow PDist(dp, \{dp'\}) < \forall_{P_i \in P} PDist(dp, P_i)$

$$\Rightarrow PDist(dp, \{dp'\}) \leq \forall_{P_i \in P'} PDist(dp, P_i).$$

Hence, from the definition of the projection distance, we get

$$MProj(dp, P') = PDist(dp, \{dp'\}) = dist(dp', dp). \quad \square$$

This theorem helps in reducing some of the unnecessary computations during the partitionset assignment step after obtaining a farthest candidate and the corresponding new component of a single datapoint. We can use the minimum projection value directly from the last iteration and compare it to the Euclidean distance from the farthest candidate. This is an important step that saves a lot of computation time.

Definition 3. (*Dimensionality of a cluster*): Let $P = \{P_i\}$ where P_i is a Partitionset and $\lambda_1, \lambda_2, \dots, \lambda_m$ be the eigen values of the Eigen decomposition of that partition set in descending order. The dimensionality of the cluster is defined as $Dim(P_i) = Min_k(\sum_1^k \lambda_i > \tau)$ where τ is a threshold.

When $\tau = 0.9$, 90% of the original variance of the data is preserved and the corresponding Partitionset is obtained as a cluster. The dimensionality of the cluster is the number of eigenvalues that cover most of the variance in the data of the Partitionset.

Definition 4. (*Acceptable Partitionset*): A Partitionset P_i is said to be an acceptable Partitionset, if $|P_i| \geq n_0$ and $\frac{1}{|P_i|} \sum_{dp \in P_i} PDist(dp, P_i) \leq \epsilon$

For a Partitionset to be acceptable, it must satisfy the following two conditions:

1. There must be sufficient number of data points (n_0) in the Partitionset.
2. The points in the correlation cluster must be closer ($< \epsilon$) to its principal component.

3 The ROSECC Algorithm

In this section, we will describe the details of the proposed ROSECC algorithm and also analyze the effect of different parameter values in the algorithm. The overall procedure is shown in Algorithm 1.

3.1 Algorithm Description

The five different steps in the ROSECC algorithm are described below.

Step 1: Finding k_0 principal components: For a given set of data points (*Data*), this step will find k_0 number of components in the data by iteratively generating a sufficient number of partitions and their corresponding principal components. The details of this step are described in Algorithm 2. Initially, the principal components are computed with the entire dataset. Then, the *Farthest Candidate* from the first principal component is taken and the dataset is partitioned into two different subsets depending on the distance of the data point to the previous principal component vector and the new seedpoint. PCA is independently applied on these two subsets, thus generating the two corresponding principal component vectors. The membership is updated and this process is repeated until k_0 number of components are obtained (see Fig. 2(a)). At this point, one might encounter any of these following scenarios:

1. Some principal components are obtained for data points that are not correlated at all. These are outliers or noise points.
2. There might be some components that contain more than 1-dimensional correlation.
3. More than one component and corresponding points might be obtained for the same cluster.

In the next three steps, these three situations are handled.

Step 2: Removing non-informative principal components: The components that may not represent a correlation cluster are removed in this step. A component is accepted for further consideration only if it satisfies the definition of acceptability given in the previous section. When the unacceptable components are removed, the corresponding set of points become members of the outlier set (O), which can then become a member of any other cluster in the later stages (see Fig. 2(b)).

Algorithm 1. $ROSECC(Data, k_0, k_1, n_0)$

- 1: **Input:** Data matrix ($Data$), Number of Partitionsets (k_0), Number of iterations (k_1), Threshold for datapoints (n_0)
 - 2: **Output:** Set of Correlation Clusters (P)
 - 3: **Procedure:**
 - 4: $P \leftarrow Generate_Components(Data, k_0, k_1)$
 - 5: $[P, O] \leftarrow Remove_Unacceptable_Components(P, n_0)$
 - 6: $[P, O] \leftarrow Include_Possible_Datapoints(O, P)$
 - 7: $P \leftarrow Merge_Possible_Components(P)$
 - 8: Return P
-

Step 3: Finding the dimensionality of the correlation: Though Step 1 successfully produces the set of components that possibly represent some correlation, identifying the dimensionality of the correlation is one of the key aspects of the algorithm. Practically, the correlation cluster lies in a low-dimensional space. Hence, we can assume that a high-dimensional correlation in the data will contain a one dimensional correlation with at least a fewer number of those datapoints. Correlation dimensionality of an acceptable component P (denoted by $Dim(P)$) is the number of eigenvectors needed for preserving the desired amount of variance.

Step 4: Adding points to the correlation cluster: This step assumes that there are some acceptable components and a set of outliers obtained from step 1. It will also consider each outlier and tries to assign them to any one of the current components. If a datapoint cannot be assigned to any of the correlation clusters, then it remains to be an outlier. A data point dp in the outlier set O is includable to P_i , if $PDist(dp, P_i)$ is less than a threshold η (see Fig. 2(c)). *At this point, the mutual exclusiveness of the component does not hold and hence the algorithm will produce overlapping clusters.*

Step 5: Merging the correlation cluster: In this step, each component is checked for any possibility of merging with another component. Two components will be merged if

Algorithm 2. *Generate_Components(Data, k_0, k_1)*

```

1: Input: Data matrix (Data), Number of Components ( $k_0$ ), Number of iterations ( $k_1$ )
2: Output: Partition Set (P)
3: Procedure:
4:  $P = Data, k = 0$ 
5: repeat
6:    $k = k + 1; dp' = FCand(P); newp = \{dp'\}; P = P \cup newp$ 
7:   for each  $dp \in D$  do
8:     if  $MProj(dp, P) > dist(dp', dp)$  then
9:        $Comp(dp) = Comp(dp) - \{dp\}$  //Reassignment of cluster membership
10:       $newp = newp \cup \{dp\}$  //when the new partition is created
11:     end if
12:   end for
13:   for  $i = 1$  to  $k_1$  do
14:     for each  $dp \in D$  do
15:       for each  $P_i \in P$  do
16:         if  $MProj(dp, P) > PDist(dp, P_i)$  then
17:            $Comp(dp) = Comp(dp) - dp$ 
18:            $P_i = P_i \cup dp$ 
19:         end if
20:       end for
21:     end for
22:   end for
23: until  $k = k_0$  //Each iteration generates a new partition

```

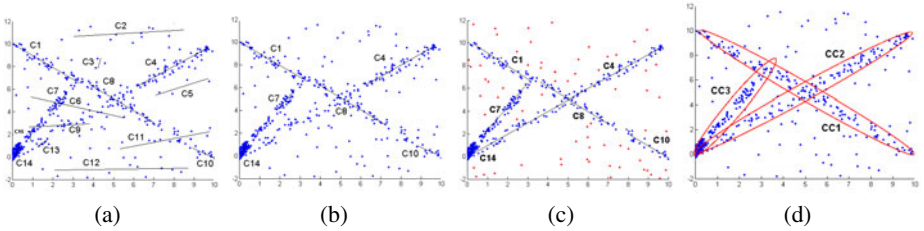


Fig. 2. Different steps of the proposed ROSECC algorithm. (a) Generation of a Partition-set: 15 partitions (C1-C15) are generated. (b) Removal of unacceptable Partitionsets: Only C1, C4, C7, C8, C10 and C14 remain after this step. (c) Addition of datapoints: the remaining points (points colored in red) are outliers. (d) Merging of Partitionsets: C1, C8 and C10 are merged to CC1; C4 and C14 are merged to CC2.

the coefficient of their representative vectors does not differ significantly. i.e. two components P_i and P_j are merged, if $|c_{il} - c_{jl}| < \delta, \forall l$ (see Fig. 2(d)).

The time complexity of our algorithm is $O(k_0^3 + Nd^2)$, which is typically dominated by the Nd^2 term for large-scale and/or high-dimensional data. This time complexity of the ROSECC algorithm is competitive to the other state-of-the-art techniques.

Table 1. Effect of the change in parameter values for the DS1 dataset

| k_1 | 3 | | | | 5 | | | |
|----------|--------|--------|-----|--------|--------|-----|------|------|
| n_0 | 1%-3% | | | | 1%-3% | | | |
| k_0 | 5 | 10 | 15 | 20 | 5 | 10 | 15 | 20 |
| Accuracy | 51.45% | 70.34% | 87% | 87.35% | 61.32% | 95% | 100% | 100% |

3.2 Tuning of the Parameters

The parameter k_0 is the number of components that are to be generated in the first step of the algorithm. This value should be at least twice the number of correlation clusters present in the data. The second parameter k_1 is the number of iterations that the algorithm runs for convergence with the new partition. From our experiments, we observed that very few iterations (fewer than 6) are necessary for convergence. The third parameter n_0 is required to check the acceptability of a component and should be atleast 1% of the number of data points. In Table 1 we show the effect of different parameter values for a synthetic dataset (DS1). Since we know the ground truth, we presented the accuracy as the percentage of correctly clustered datapoints. We can see that k_1 is good with value as low as 5, while the value of n_0 can be between 1% and 3%. Results are not optimal when $k_0 = 5$, but $k_0 = 15$ or more gives an accurate result. Hence, it is important to generate more number of components in the first step of the algorithm.

4 Results and Discussion

4.1 Synthetic Datasets

Our algorithm was tested successfully on various synthetic datasets that were generated to see the different aspects of the ROSECC algorithm. We discuss two different synthetic datasets and explain the data generation along with the corresponding results.

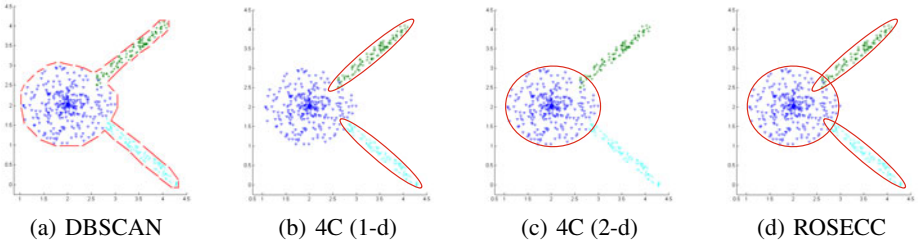
(1) **DS1** : In this dataset, two one-dimensional correlations containing 200 data points generated from the origin with a slope of 1 and -1. Another correlation cluster (with 100 data points) is generated from the origin with a slope of 1.4. 100 random noise points are also added. The ROSECC algorithm found all the correlation clusters, even in the presence of sufficient noise and significant overlapping of the clusters (see Fig. 2(d)). Thus it shows that our algorithm is robust and only starts to break when the noise is greater than 20% in the data (see Table 2).

(2) **DS2**: In this dataset, one two-dimensional correlation cluster (circular shape) with 300 data points is generated. Two 1-dimensional correlation clusters that start from the edge of that cluster in two different directions, one with 150 datapoints and the other with 100 datapoints is added to the first cluster (see Fig. 3). This dataset is created to test the ability of the ROSECC algorithm to simultaneously identify the correlation clusters with different dimensionality which is one of key challenges in the correlation clustering problem.

Table 2. Robustness of ROSECC: Accuracy is measured in presence of different noise levels

| Noise percentage | 5% | 10% | 15% | 20% | 25% | 30% | 35% |
|------------------|------|------|------|------|-----|-----|-----|
| Accuracy | 100% | 100% | 100% | 100% | 93% | 85% | 70% |

We compared the result of the ROSECC algorithm with the well-studied DBSCAN [8] and 4C [5] algorithms. In this dataset, all the datapoints are density connected (see Fig. 3). There are two 1-dimensional and one 2-dimensional correlation cluster in this dataset. As shown in the result, DBSCAN recognizes all the data points as a single cluster (Fig. 3(a)). Algorithm 4C recognizes either 1-dimensional clusters (Fig. 3(b)) or the 2-dimensional cluster (Fig. 3(c)) depending on the choice of the parameter corresponding to the number of dimensionality. On the other hand, the ROSECC algorithm was able to successfully identify all the three clusters simultaneously in a single run (Fig. 3(d)).

**Fig. 3.** Comparison results: ROSECC(d) finds all the desired correlation clusters in a synthetic dataset (DS2) where both DBSCAN(a) and 4C(b and c) fail

4.2 Real-World Datasets

We also show the performance of the ROSECC algorithm in finding correlation clusters in the following three real-world datasets.

(1) **Breast Cancer dataset:**¹ This dataset measures nine biomedical parameters characterizing breast cancer type in 683 humans. The algorithm found one 3-dimensional cluster and two 2-dimensional clusters. When the result is projected in a 3-D space using PCA, we found that the 3-dimensional correlation cluster is a pure cluster of benign cancer (see Fig. 4) and no malignant cancerous patient belongs to any correlation cluster and are considered to be outliers by our algorithm.

(2) **Wages dataset :** The wages dataset contains the statistics of the determinants of Wages from the 1985 Current Population Survey. It contains 534 observations on 11 features sampled from the original Current Population Survey of 1985 and can be

¹ <http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/>

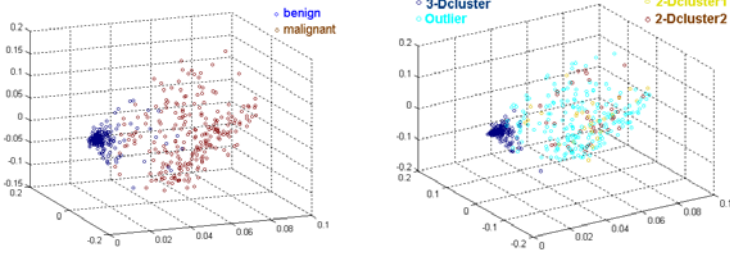


Fig. 4. Projection of the breast cancer dataset onto the 3-dimensional space using PCA. (a) Original class labels and (b) Result of the ROSECC algorithm.

downloaded from StatLib Data archive². ROSECC gives one 2-dimensional correlation (see Fig. 5) which basically gives $YE(\text{Years of Education}) + WE(\text{Years of work experience}) + 6 = \text{Age}$. It also gives four 1-dimensional correlation clusters;

- (i) $\text{Age} = WE + 18, YE=12$
- (ii) $\text{Age} = WE + 20, YE=16$
- (iii) $\text{Age} = WE + 24, YE=18$
- (iv) $\text{Age} = WE + 24, YE=14$.

However, all of the data points of these four clusters are also the members of the 2-dimensional clusters, which suggests that these four 1-dimensional correlations are actually on the 2-dimensional plane of the 2-dimensional cluster. This is an interesting example because there are quite a few data points in the 1-dimensional clusters and our algorithm identifies those as separate clusters as well.

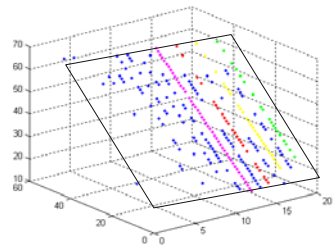


Fig. 5. ROSECC identifies the clusters in the Wages dataset

(3) **Glass Identification dataset:** The glass identification dataset³ consists of 214 samples and 9 attributes measuring the refractive index and weight percentage of different metals in the corresponding oxide in the glass. We ran our algorithm to find the clusters in this dataset and the wdbc dataset. We measure the performance of the ROSECC algorithm by using the class label as ground truth and calculating the F1 measure. Table 3 shows the results of the ROSECC algorithm in both the datasets are better than the other state-of-the-art methods.

Table 3. Experimental results of F1 measure for real-world datasets

| Dataset | CLIQUE [4] | SUBCLU [10] | PROCLUS [3] | ROSECC |
|---------|------------|-------------|-------------|--------|
| Glass | 0.45 | 0.49 | 0.57 | 0.58 |
| WDBC | 0.42 | 0.43 | 0.47 | 0.49 |

² http://lib.stat.cmu.edu/datasets/CPS_85_Wages

³ <http://archive.ics.uci.edu/ml/datasets/Glass+Identification/>

5 Conclusion

We proposed ROSECC, a novel PCA based robust seedless correlation clustering algorithm. ROSECC does not require the initial set of seeds for clustering and is robust to noise in the data compared to the other PCA based approaches which typically require initial seeds and a pre-defined dimensionality parameter. It incrementally partitions the data space and eventually finds the correlation clusters in any arbitrary subspace even in the presence of overlapping clusters. Using several synthetic and real-world datasets, we demonstrated the advantages of the ROSECC algorithm compared to the other methods available in the literature for identifying correlation clusters.

References

1. Achtert, E., Böhm, C., David, J., Kröger, P., Zimek, A.: Robust clustering in arbitrarily oriented subspaces. In: Proceedings of SIAM International Conference on Data Mining (SDM), pp. 763–774 (2008)
2. Aggarwal, C., Yu, P.: Finding generalized projected clusters in high dimensional spaces. In: Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 70–81 (2000)
3. Aggarwal, C.C., Wolf, J.L., Yu, P.S., Procopiu, C., Park, J.S.: Fast algorithms for projected clustering. In: Proceedings of the ACM SIGMOD international conference on Management of data, pp. 61–72 (1999)
4. Agrawal, R., Gehrke, J., Gunopulos, D., Raghavan, P.: Automatic subspace clustering of high dimensional data for data mining applications. In: Proceedings of ACM SIGMOD International Conference on Management of Data, pp. 94–105 (1998)
5. Bohm, C., Kailing, K., Kroger, P., Zimek, A.: Computing clusters of correlation connected objects. In: Proceedings of the ACM SIGMOD international conference on Management of data, pp. 455–466 (2004)
6. Cheng, C., Fu, A.W., Zhang, Y.: ENCLUS: Entropy-based subspace clustering for mining numerical data. In: Proceedings of the ACM conference on Knowledge Discovery and Data Mining (SIGKDD), pp. 84–93 (1999)
7. Ding, C.H.Q., He, X., Zha, H., Simon, H.D.: Adaptive dimension reduction for clustering high dimensional data. In: Proceedings of the IEEE International Conference on Data Mining, pp. 147–154 (2002)
8. Ester, M., Kriegel, H., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of the ACM conference on Knowledge Discovery and Data Mining (SIGKDD), pp. 226–231 (1996)
9. Jain, A.K., Dubes, R.C.: Algorithms for Clustering Data. Prentice Hall, Englewood Cliffs (1988)
10. Kailing, K., Kriegel, H., Kröger, P.: Density-connected subspace clustering for high-dimensional data. In: Proceedings of SIAM International Conference on Data Mining (SDM), pp. 246–257 (2004)
11. Kriegel, H., Kröger, P., Zimek, A.: Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 3(1), 1–58 (2009)
12. Yip, K.Y., Cheung, D.W., Ng, M.K.: On discovery of extremely low-dimensional clusters using semi-supervised projected clustering. In: Proceedings of the 21st International Conference on Data Engineering (ICDE), pp. 329–340 (2005)
13. Yip, K.Y., Ng, M.K.: Harp: A practical projected clustering algorithm. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 16(11), 1387–1397 (2004)

Integrative Parameter-Free Clustering of Data with Mixed Type Attributes

Christian Böhm¹, Sebastian Goebel¹, Annahita Oswald¹, Claudia Plant²,
Michael Plavinski¹, and Bianca Wackersreuther¹

¹ University of Munich

² Technische Universität München

{boehm,oswald,wackersreuther}@dbs.ifi.lmu.de,

{goebel,plavinsk}@cip.ifi.lmu.de,

plant@lrz.tum.de

Abstract. Integrative mining of heterogeneous data is one of the major challenges for data mining in the next decade. We address the problem of integrative clustering of data with mixed type attributes. Most existing solutions suffer from one or both of the following drawbacks: Either they require input parameters which are difficult to estimate, or/and they do not adequately support mixed type attributes. Our technique INTEGRATE is a novel clustering approach that truly integrates the information provided by heterogeneous numerical and categorical attributes. Originating from information theory, the Minimum Description Length (MDL) principle allows a unified view on numerical and categorical information and thus naturally balances the influence of both sources of information in clustering. Moreover, supported by the MDL principle, parameter-free clustering can be performed which enhances the usability of INTEGRATE on real world data. Extensive experiments demonstrate the effectiveness of INTEGRATE in exploiting numerical and categorical information for clustering. As an efficient iterative algorithm INTEGRATE is scalable to large data sets.

1 Introduction

Integrative data mining is among the top 10 challenging problems in data mining identified in [1]. Moreover it is essential for solving many of the other top 10 challenges, including data mining in social networks and data mining for biological and environmental problems. In this paper, we focus on *integrative clustering*. Clustering aims at finding a natural partitioning of the data set into meaningful groups or clusters. Thus, clustering provides an overview on major patterns in the data without requiring much previous knowledge. During the last decades, clustering has attracted a lot of attention as reflected in a huge volume of research papers, e.g. [2,3,4,5,6,7], to mention a few. We address the question of how to find a natural clustering of data with mixed type attributes. In everyday life, huge amounts of such data are collected, for example from credit assessments. The collected data include numerical attributes (e.g. credit amount, age), as well as categorical attributes (e.g. personal status). A cluster analysis of credit assessment data is interesting, e.g., for target marketing. However, finding a natural clustering of such data is a non-trivial task. We identified two major problems: Either much previous knowledge is required, or there is no adequate support of mixed type attributes.

To cope with these two major problems, we propose INTEGRATE, a parameter-free technique for integrative clustering of data with mixed type attributes. The major benefits of our approach, which to the best of our knowledge no other clustering method meets all of them, can be summarized as follows:

- Natural balance of numerical and categorical information in clustering supported by information theory;
- Parameter-free clustering;
- Making most effective usage of numerical as well as categorical information;
- Scalability to large data sets.

The rest of this paper is organized as follows: Section 2 gives a brief survey of the large previous work. Section 3 presents a detailed derivation of $iMDL$, an information-theoretic clustering quality criterion suitable for integrative clustering. Section 4 presents our effective and efficient iterative algorithm INTEGRATE optimizing $iMDL$. Section 5 documents that INTEGRATE makes most effective usage of numerical as well as categorical information by comparing it to well-known and state-of-the-art clustering algorithms on synthetic and real data sets. Section 6 summarizes the paper.

2 Related Work

The algorithm k -prototypes [3] combines k -means [2] for clustering numerical data with k -modes for categorical data in order to cluster mixed type data. The attribute weights and the number of clusters have to be determined a priori. CFIKP [8] can process large data sets by k -prototypes in combination with a CF*-tree, which pre-clusters the data into dense regions. The problem for selecting the number of clusters remains. The algorithm CAVE [9] is an incremental entropy-based method which first selects k clusters, parametrized by the user, and then assigns objects to these clusters based on variance and entropy. Knowledge of the similarity among categorical attributes is needed in order to construct the distance hierarchy for the categorical attributes. The cluster ensemble approach CEBMDC [10] overcomes the problem of selecting k but requires a threshold parameter that defines the intra-cluster similarity between objects. The CBC algorithm [11] is an extension of BIRCH [4] for clustering mixed type data. It uses a weight-balanced tree that needs two parameters, defining the number of entries for (non)-leaf nodes. Furthermore, all entries in a leaf node must satisfy a particular threshold requirement. Ahmad and Dey [12] propose a k -means-based method for mixed type attributes, but the process of solving the optimization of the cost function is very complex and thus not scalable to large data sets. [13] uses standard fuzzy c -means on a set of features which is mapped to a set of feature vectors with only real valued components. This mapping is computationally intensive and is designed rather for low dimensional data. An extension of the cost function of entropy weighting k -means [14] to more efficiently specify the inter- and intra-cluster similarities is proposed by the IWEKM approach [15]. Some papers have focused on avoiding the choice of k in partitioning clustering, e.g. X-Means [5], RIC [6] and OCI [7]. However, these clustering methods are designed for numerical vector data only.

3 Minimum Description Length for Integrative Clustering

Notations. In the following we consider a data set DS with n objects. Each object x is represented by d attributes. Attributes are denoted by capital letters and can be either numerical features or categorical variables with two or more values. For a categorical attribute A , we denote a possible value of A by a . The result of our algorithm is a disjoint partitioning of DS into k clusters C_1, \dots, C_k .

Likelihood and Data Compression. One of the most challenging problems in clustering data with mixed attribute type is selecting a suitable distance function, or unifying clustering results obtained on the different representations of the data. Often, the weighting between the different attribute types needs to be specified by parameter settings, cf. Section 2. The minimum description length (MDL) principle provides an theoretical foundation for parameter-free integrative clustering avoiding this problem. Regarding clustering as a data compression problem allows us a unifying view, naturally balancing the influence of categorical and numerical attributes in clustering. Probably the most important idea of MDL which allows integrative clustering is relating the concepts of likelihood and data compression. Data compression can be maximized by assigning short descriptions to regular data objects which exhibit the characteristic patterns and longer descriptions to the few irregular objects or outliers.

3.1 Coding Categorical Data

Assume a data set, where each object is represented by one categorical attribute A with two possible values. It can be shown that the code length to encode this data is lower bounded by the entropy of A . Thus, the coding costs CC of A are provided by:

$$CC(A) = - \sum_{a \in A} p(a) \cdot \log_2 p(a).$$

By the application of the binary logarithm we obtain the code length in bits. If we have no additional knowledge on the data we have to assume that the probabilities for each value are equal. Hence, we need one bit per data object. Clustering, however, provides high-level knowledge on the data which allows for a much more effective way to reduce the costs. Even if the probabilities for the different outcomes of the attributes are approximately equal w.r.t. the whole data set, often different clusters with non-uniform probabilities can be found. As an example, refer to Figure 1. The data are represented by two numerical attributes (which we ignore for the moment) and one categorical attribute which has two possible values, *red* and *blue*. Considering all objects, the probabilities for *red* and *blue* are equal. However, it is evident that the outcomes are not uniformly distributed. Rather, we have two clusters, one preliminarily hosts the red objects, and the other the blue ones. In fact, the data has been generated such that in the left cluster, we have 88% of blue objects and 12% of red objects. For the right cluster, the ratio has been selected reciprocally. This clustering drastically reduces the entropy and hence $CC(A) = 0.53$ bits per data object, which corresponds to the entropy of A in both clusters.

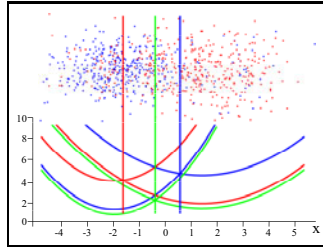


Fig. 1. Top: Example data set with two numerical and one categorical attribute with the outcomes *red* and *blue*. Bottom: Cost curves assuming two clusters: Considering the numerical information only (green), integrating numerical and categorical information (red, blue). For each outcome and each cluster, we have a unique cost curve. Intersection points mark the resulting cluster borders.

3.2 Coding Numerical Data

To specify the probability of each data object considering an additional numerical attribute B , we assign a probability density function (PDF) to B . In this paper, we apply a Gaussian PDF for each numerical attribute. However, let us note that our ideas can be straightforwardly extended to other types PDF, e.g. Laplacian or Generalized Gaussian. Thus, the PDF of a numerical attribute B is provided by:

$$p(b) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(b - \mu_B)^2}{2\sigma_B^2}\right).$$

If the data distribution of B is Gaussian with mean μ_B and standard deviation σ_B , we minimize the costs of the data by a coding scheme which assigns short bit strings to objects with coordinate values that are in the area of high probability and longer bit strings to objects with lower probability. This principle is also commonly referred to as Huffman-Coding. The coding costs CC of B are provided by:

$$CC(B) = - \int p(b) \log_2 p(b) \mathbf{d}b.$$

Again, if we have no knowledge on the data, we would have to assume that each attribute is represented by a single Gaussian with mean and standard deviation determined from all data objects. As discussed for categorical data, clustering can often drastically reduce the costs. Most importantly, relating clustering to data compression allows us a unified view on data with mixed type attributes. Consider again the data displayed in Figure 1. In addition to the categorical attribute we now consider the numerical x -coordinate, denoted by X . To facilitate presentation, we ignore the y -coordinate which is processed analogously. The two green curves at the bottom represent the coding costs of the two clusters considering X . For both curves, the cost minimum coincides with the mean of the Gaussian which generated the data. The cluster on the right has been generated with slightly larger variance, resulting in slightly higher coding costs. The intersection of both cost curves represents the border between the two clusters provided by X , indicated by a green vertical line. In addition, for each cluster and each outcome

of the categorical attribute, we have included a cost curve (displayed in the corresponding colors). Again, the intersection points mark the cluster borders provided by the categorical attribute. Consider, e.g., the red vertical line. Red objects with a value in X beyond that point are assigned to the cluster on the right. Thus, in the area between the red and the blue vertical line, the categorical value is the key information for clustering. Note that all borders are not fixed but optimized during the run of our algorithm.

3.3 A Coding Scheme for Integrative Clustering

We also need to elaborate a coding scheme describing the clustering result itself. The additional costs for encoding the clustering result can be classified into two categories: the parameter costs PC required to specify the cluster model and the id-costs IDC required to specify the cluster-id for each object, i.e. the information to which cluster the object belongs.

For the parameter costs, let's focus on the set of objects belonging to a single cluster \mathcal{C} . To specify the cluster model, we need for each categorical attribute A to encode the probability of each value or outcome a . For a categorical attribute with $|A|$ possible values, we need to encode $|A| - 1$ probabilities since the remaining probability is implicitly specified. For each numerical attribute B we need to encode the parameters μ_B and σ_B of the PDF. Following a central result from the theory of MDL [16], the parameter costs to model the $|\mathcal{C}|$ objects of the cluster can be approximated by $p/2 \cdot \log_2 |\mathcal{C}|$, where p denotes the number of parameters. The parameter costs depend logarithmically on the number of objects in the cluster. The considerations behind this are that for clusters with few objects, the parameters do not need to be coded with very high precision. To summarize, the parameter costs for a cluster \mathcal{C} are provided by

$$PC(\mathcal{C}) = \frac{1}{2} \cdot \left(\left(\sum_{A_{cat}} |A| - 1 \right) + |B_{num}| \cdot 2 \right) \cdot \log_2 |\mathcal{C}|.$$

Here A_{cat} stands for all categorical attributes and B_{num} for all numerical attributes in the data. Besides the parameter costs, we need for each object to encode the information to which of the k clusters it belongs. Also for the id-costs, we apply the principle of Huffman coding which implies that we assign shorter bitstrings to the larger clusters. Thus, the id-costs of a cluster \mathcal{C} are provided by:

$$IDC(\mathcal{C}) = \log_2 \frac{n}{|\mathcal{C}|}.$$

Putting it all together, we are now ready to define $iMDL$, our information-theoretic optimization goal for integrative clustering.

$$iMDL = \sum_{\mathcal{C}} \left(\sum_A |\mathcal{C}| \cdot CC(A) \right) + PC(\mathcal{C}) + IDC(\mathcal{C}).$$

For all clusters \mathcal{C} we sum up the coding costs for all numerical and categorical attributes A . To these costs we need to add the parameter costs and the id-cost of the cluster, denoted by $PC(\mathcal{C})$ and $IDC(\mathcal{C})$, respectively. Finally, we sum up these three terms for all clusters.

4 The Algorithm INTEGRATE

Now we present the highly effective algorithm INTEGRATE for clustering mixed type attributes that is based on our new MDL criterion $iMDL$, defined in Section 3. INTEGRATE is designed to find the optimal clustering of a data set DS , where each object x comprises both numerical and categorical attributes by optimizing the overall compression rate. First, INTEGRATE builds an initial partitioning of k clusters. Each cluster is represented by a Gaussian PDF in each numerical dimension B with μ_B and σ_B , and a probability for each value of the categorical attributes. All objects are then assigned to the k clusters by minimizing the overall coding costs $iMDL$. In the next step, the parameters of each cluster are recalculated according to the assigned objects. That implies the μ and the σ in each numerical dimension and the probabilities for each value of the categorical attributes, respectively. After initialization the following steps are performed repeatedly until convergence. First, the costs for coding the actual cluster partition are determined. Second, assignment of objects to clusters is performed in order to decrease the $iMDL$ value. Third, the new parameters of each cluster are recalculated. INTEGRATE terminates if no further changes of cluster assignments occur. Finally, we receive the optimal clustering for DS represented by k clusters according to minimum coding costs.

4.1 Initialization

The effectiveness of an algorithm often heavily depends on the quality of the initialization, as it is often the case that the algorithm can get stuck in a local optimum. Hence, we propose an initialization scheme to avoid this effect. We have to find initial cluster representatives that correspond best to the final representatives. An established method for partitioning methods is to initialize with randomly chosen objects of DS . We adopt this idea and take the μ of the numerical attributes of k randomly chosen objects as cluster representatives. During initialization, we set $\sigma = 1.0$ in each numerical dimension. The probabilities of the values for the categorical attributes are set to $\frac{1}{|a|}$. Then a random set of $\frac{1}{z}n$ objects is selected, where n is the size of DS and $z = 10$ turned out to give satisfying results. Finally, we chose the clustering result that minimizes $iMDL$ best, within m initialization runs. Typically $m = 100$ runs suffice for an effective result. As only a fraction of DS is used for the initialization procedure, our method is not only effective but also very efficient.

4.2 Automatically Selecting the Number of Clusters k

Now we propose a further improvement of the effectiveness of INTEGRATE. Using $iMDL$ for mixed type data we can avoid the parameter k . As an optimal clustering that represents the underlying data structure best has minimum coding costs, $iMDL$ can also be used to detect the number of clusters. For this purpose, INTEGRATE uses $iMDL$ no longer exclusively as selection criterion for finding the correct object to cluster assignment. Rather we now estimate the coding costs for each k where k is selected in a range of $1 \leq k \leq n$. For efficiency reasons INTEGRATE performs this iteration step on a $z\%$ sample of DS . The global minimum of this cost function gives the optimal k and thus the optimal number of clusters.

5 Experimental Evaluation

Since INTEGRATE is a hybrid approach combining the benefits of clustering methods using only numeric attributes and those for categorical attributes we compare algorithms of both categories and algorithms that can also handle mixed type attributes. In particular, we selected the popular k -means algorithm, the widely used method k -modes, the k -means-based method by Ahmad and Dey denoted by KMM and k -prototypes (cf. Section 2). For k -means and k -modes the numerical and categorical attributes were ignored. For evaluation we used the validity measure by [17] referred to as DOM in the following (smaller values indicate a better cluster quality), which has the advantage that it allows for clusterings with different numbers of clusters and integrates the class labels as “ground truth”. We report in each experiment the average performance of all clustering algorithms over 10 runs.

5.1 Synthetic Data

If not otherwise specified the artificial data sets include three Gaussian clusters with each object having two numerical attributes and one categorical attribute. To validate the results we added a class label to each object which was not used for clustering.

Varying Ratio of Categorical Attribute Values. In this experiment we generated three-dimensional synthetic data sets with 1,500 points including two numerical and one two-valued categorical attribute. We varied the ratio for each of the values of the categorical attributes from 1:0 to 0:1 clusterwise in each data set. Without need for difficult parameter setting INTEGRATE performs best in all cases (cf. Figure 2). Even in the case of equally (5:5) distributed values, where the categorical attribute gives no information for separating the objects, the cluster quality of INTEGRATE is best compared to all other methods. As k -means does not take the categorical attributes into account the performance is relatively constant.

Varying Variance of Clusters. This experiment aims at comparing the performance of the different methods on data sets with varying variances. In particular, we generated synthetic data sets each comprising 1,500 points including two numerical and one two-valued categorical attribute that form three Gaussian clusters with a variance ranging from 0.5 to 2.0. Figure 3 shows that INTEGRATE outperforms all competitors in all cases, in which each case reflects different degree of overlap of the three clusters. Even at a variance of 2.0 where the numerical attributes carry nearly no cluster information our proposed method shows best cluster quality as in this case the categorical attributes are used to separate the clusters. On the contrary, k -modes performs worst as it can only use the categorical attribute as single source for clustering.

Varying Clustersize. In order to test the performance of the different methods on data sets with unbalanced clustersize we generated three Gaussian clusters with different variance and varied the ratio of number of points per cluster from 1:10:1 to 10:1:10 in five steps. It is obvious from Figure 4 that INTEGRATE separates the three clusters best even with highly unbalanced cluster sizes. Only in the case of two very small clusters and one big cluster (1:10:1) k -modes shows a slightly better cluster validity.

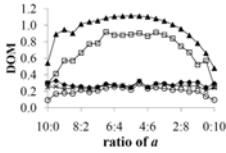


Fig. 2. Varying ratio of categorical attribute values

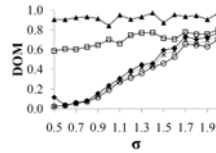


Fig. 3. Varying variance of clusters

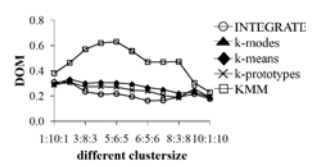


Fig. 4. Varying Clustersize

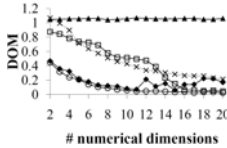


Fig. 5. Numerical dimensions

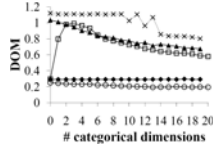


Fig. 6. Categorical dimensions

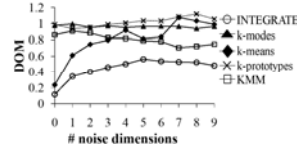


Fig. 7. Noise dimensions

Varying Number of Numerical Dimensions. In this experiment we leave the number of categorical attributes to a constant value and successively add numerical dimensions to each object that are generated with a variance of $\sigma=1.8$. INTEGRATE shows best performance in all cases (cf. Figure 5). All methods show a slight increase in cluster quality when varying the numerical dimensionality, except k -modes that performs constantly as it does not consider the numerical attributes.

Varying Number of Categorical Dimensions. For each object we added three-valued categorical attributes where we set the probability of the first value to 0.6 and the probability of the two remaining values to 0.2, respectively. Figure 6 illustrates that our proposed method outperforms the other methods and even k -modes by magnitudes which is a well-known method for clustering categorical data. Whereas KMM shows a heavy decrease in clustering quality in the case of two and four additional categorical attributes, our method performs relatively constant. Taking the numerical attributes not into account the cluster validity of k -means remains constant.

Noise Dimensions. Figure 7 illustrates the performance of the different methods on noisy data. It is obvious that INTEGRATE outperforms all compared methods when adding non-clustered noise dimensions to the data. k -means shows a highly increase in the DOM values which refers to decreasing cluster validity. Even in the case of nine noise dimensions INTEGRATE leads to the best clustering result.

5.2 Real Data

Finally, we show the practical application of INTEGRATE on real world data, available at the UCI repository <http://archive.ics.uci.edu/ml/>. We chose two different data sets with mixed numerical and categorical attributes. An additional class attribute allows for an evaluation of the results. Table 1 reports the μ and σ of all methods within 10 runs. For all compared methods we set k to the number of classes.

Table 1. Results on real data

| | INTEGRATE | k -means | k -modes | kMM | k -prototypes | |
|-----------------|-----------|------------|------------|------|-----------------|------|
| Heart Disease | μ | 1.23 | 1.33 | 1.26 | 1.24 | 1.33 |
| | σ | 0.02 | 0.01 | 0.03 | 0.02 | 0.00 |
| Credit Approval | μ | 0.61 | 0.66 | 0.70 | 0.63 | 0.66 |
| | σ | 0.03 | 0.00 | 0.00 | 0.09 | 0.00 |

Heart Disease. The Heart-Disease data set comprises 303 instances with six numerical and eight categorical attributes each labeled to an integer value between 0 and 4 which refers to the presence of heart disease. Without any prior knowledge on the data set we obtained best cluster validity of 1.23 with INTEGRATE. KMM performed slightly worse. However, the runtime of INTEGRATE is 0.1 seconds compared to KMM which took 2.8 seconds to return the result.

Credit Approval. The Credit Approval data set contains results of credit card applications. It has 690 instances, each being described by six numerical and nine categorical attributes and classified to the two classes ‘yes’ or ‘no’. With a mean DOM value of 0.61 INTEGRATE separated the objects best into two clusters in only 0.1 seconds without any need for setting input parameters.

5.3 Finding the Optimal k

On the basis of the data set illustrated in Figure 8(a) we highlight the benefit of INTEGRATE for finding the correct number of clusters that are present in the data set. The data set comprises six Gaussian clusters with each object having two numerical attributes and one categorical attribute with two different values that are marked in “red” and “blue”, respectively. Figure 8(b) shows the $iMDL$ of the data model for different values of k . The cost function has its global minimum, which refers to the optimal number of clusters, at $k = 6$. In the range of $1 \leq k \leq 4$ the plotted function shows an intense decrease in the coding costs and for $k > 6$ a slight increase of the coding costs as in these cases the data does not optimally fit into the data model and therefore causes high coding costs. Note, that there is a local minimum at $k = 4$ which would also refer to a meaningful number of clusters.

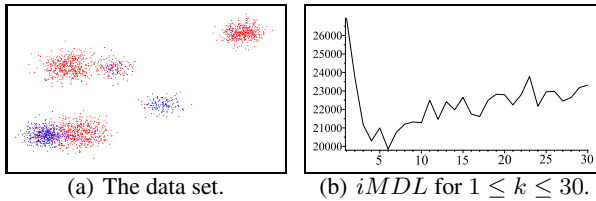


Fig. 8. Coding costs for different k according to a data set that consists of $k = 6$ clusters

6 Conclusion

We have introduced a new information-theoretic clustering method — INTEGRATE. We gave a solution to avoid difficult parameter settings guided by the information-theoretic idea of data compression. We have shown with extensive experiments that INTEGRATE uses the numerical and categorical information most effectively. And finally, INTEGRATE shows high efficiency and is therefore scalable to large data sets.

References

1. Yang, Q., Wu, X.: 10 challenging problems in data mining research. *IJITDM* 5(4), 597–604 (2006)
2. Macqueen, J.B.: Some methods of classification and analysis of multivariate observations. In: *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pp. 281–297 (1967)
3. Huang, Z.: Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data Min. Knowl. Discov.* 2(3), 283–304 (1998)
4. Zhang, T., Ramakrishnan, R., Livny, M.: Birch: An efficient data clustering method for very large databases. In: *SIGMOD Conference*, pp. 103–114 (1996)
5. Pelleg, D., Moore, A.W.: X-means: Extending k-means with efficient estimation of the number of clusters. In: *ICML*, pp. 727–734 (2000)
6. Böhm, C., Faloutsos, C., Pan, J.Y., Plant, C.: Robust information-theoretic clustering. In: *KDD*, pp. 65–75 (2006)
7. Böhm, C., Faloutsos, C., Plant, C.: Outlier-robust clustering using independent components. In: *SIGMOD Conference*, pp. 185–198 (2008)
8. Yin, J., Tan, Z.: Clustering mixed type attributes in large dataset. In: *ISPA*, pp. 655–661 (2005)
9. Hsu, C.C., Chen, Y.C.: Mining of mixed data with application to catalog marketing. *Expert Syst. Appl.* 32(1), 12–23 (2007)
10. He, Z., Xu, X., Deng, S.: Clustering mixed numeric and categorical data: A cluster ensemble approach. *CoRR abs/cs/0509011* (2005)
11. Rendon, E., Sánchez, J.S.: Clustering based on compressed data for categorical and mixed attributes. In: *SSPR/SPR*, pp. 817–825 (2006)
12. Ahmad, A., Dey, L.: A k-mean clustering algorithm for mixed numeric and categorical data. *Data Knowl. Eng.* 63(2), 503–527 (2007)
13. Brouwer, R.K.: Clustering feature vectors with mixed numerical and categorical attributes. *IJCIS* 1-4, 285–298 (2008)
14. Jing, L., Ng, M.K., Huang, J.Z.: An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data. *IEEE Trans. Knowl. Data Eng.* 19(8), 1026–1041 (2007)
15. Li, T., Chen, Y.: A weight entropy k-means algorithm for clustering dataset with mixed numeric and categorical data. In: *FSKD 2008*, vol. (1), pp. 36–41 (2008)
16. Rissanen, J.: An introduction to the mdl principle. Technical report, Helsinki Institute for Information Technology (2005)
17. Dom, B.: An information-theoretic external cluster-validity measure. In: *UAI*, pp. 137–145 (2002)

Data Transformation for Sum Squared Residue

Hyuk Cho

Computer Science, Sam Houston State University,
Huntsville, TX 77341-2090, USA
hyukcho@shsu.edu

<http://www.cs.shsu.edu/~hxc005/>

Abstract. The sum squared residue has been popularly used as a clustering and co-clustering quality measure, however little research on its detail properties has been performed. Recent research articulates that the residue is useful to discover shifting patterns but inappropriate to find scaling patterns. To remedy this weakness, we propose to take specific data transformations that can adjust latent scaling factors and eventually lead to lower the residue. First, we consider data matrix models with varied shifting and scaling factors. Then, we formally analyze the effect of several data transformations on the residue. Finally, we empirically validate the analysis with publicly-available human cancer gene expression datasets. Both the analytical and experimental results reveal column standard deviation normalization and column Z-score transformation are effective for the residue to handle scaling factors, through which we are able to achieve better tissue sample clustering accuracy.

Keywords: Data Transformation, Sum Squared Residue, Z-score Transformation, Scaling Pattern, Shifting Pattern.

1 Introduction

Hartigan's pioneering work, *direct clustering* [13], stimulated a vast amount of research on co-clustering. Co-clustering aims at identifying homogeneous local patterns, each of which consists of a subset of rows and a subset of columns in a given two dimensional matrix. This idea has attracted genomic researchers, because it is compatible with our understanding of cellular processes, where a subset of genes are coregulated under a certain experimental conditions [5]. Madeira and Oliveira [15] surveyed biclustering algorithms and their applications to biological data analysis.

Cheng and Church [7] are considered to be the first to apply co-clustering, *biclustering*, to gene expression data. The greedy search heuristic generates bi-clusters, one at a time, which satisfy a certain homogeneity constraint, called *mean squared residue*. Since then, several similar approaches have been proposed to enhance the original work. For example, Cho *et al.* [8] developed two minimum sum squared co-clustering (MSSRCC) algorithms: one objective function is based on the partitioning model proposed by Hartigan [13] and the other one

is based on the squared residue formulated by Cheng and Church [7]. The later is the residue of interest and defined in the next chapter.

Recently, Aguilar-Ruiz [1] shows that the mean squared residue depends on the scaling variance in the considered data matrix. This finding issues the weakness of the residue and the need of new approaches to discover scaling patterns. Motivated by the work, we propose a simple remedy to find scaling patterns, still using the same residue measure. We suggest to take specific data transformations so as to handle hidden scaling factors. In this paper, we apply several data transformations to data matrix models derived from varied scaling and shifting factors and analyze the effect of data transformations on the the second residue, RESIDUE(II) [8]. Furthermore, using MSSRCC, we empirically demonstrate the advantage of the data transformations with publicly available human cancer microarrays. Both analysis and experimental results reveal that column standard deviation normalization and column Z-score transformation are effective.

The rest of this paper is organized as follows: In Section 2 we introduce some definitions and facts used in this paper. We describe the considered data transformations in Section 3. Then, we formally analyze the effects of data transformations and summarize the analysis results in Section 4. We discuss the experimental results with human cancer gene expression datasets in Section 5. Finally, the paper is concluded with some remark.

2 Definition

We adapt the following definitions in Aguilar-Ruiz [1], Cheng and Church [7], and Cho *et al.* [8] to fit for our context.

Data matrix. A data matrix is defined as a real-valued rectangular matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, whose (i, j) -th element is denoted by a_{ij} .

For example, a microarray can be defined with two finite sets, the set of genes and the set of experimental conditions. Note that Aguilar-Ruiz [1] describes the microarray whose rows represent experimental condition and columns represent genes. However, in this paper, we will consider a microarray which consists of examples of genes in rows and attributes as experimental conditions in columns.

Co-cluster. Let $I \subseteq \{1, 2, \dots, m\}$ and $J \subseteq \{1, 2, \dots, n\}$ denote the set of indices of the rows in a row cluster and the set of indices of the columns in a column cluster. A submatrix of \mathbf{A} induced by the index sets I and J is called a co-cluster and denoted as $\mathbf{A}_{IJ} \in \mathbb{R}^{|I| \times |J|}$, where $|I|$ and $|J|$ denote the cardinality of index set I and J , respectively. In reality, rows and columns in a co-cluster are not necessary to be consecutive. However, for brevity we consider the co-cluster, \mathbf{A}_{IJ} , whose entries consist of first $|I|$ rows and first $|J|$ columns in \mathbf{A} .

2.1 Sum Squared Residue

In order to evaluate the coherence of such a co-cluster, we define RESIDUE(II) of an element a_{ij} in the co-cluster determined by index sets I and J as below.

Residue. *RESIDUE(II)* is defined as $h_{ij} = a_{ij} - a_{iJ} - a_{Ij} + a_{IJ}$, where the mean of the entries in row i whose column indices are in J is computed by $a_{iJ} = \frac{1}{|J|} \sum_{j \in J} a_{ij}$, the mean of the entries in column j whose row indices are in I by $a_{Ij} = \frac{1}{|I|} \sum_{i \in I} a_{ij}$, and the mean of all the entries in the co-cluster whose row and column indices are in I and J by $a_{IJ} = \frac{1}{|I||J|} \sum_{i \in I, j \in J} a_{ij}$.

Sum squared residue (SSR). Let $\mathbf{H}_{IJ} \in \mathbb{R}^{|I| \times |J|}$ be the residue matrix whose entries are described by *RESIDUE(II)*. Then, the sum squared residue of \mathbf{H}_{IJ} is defined as $SSR = \|\mathbf{H}_{IJ}\|^2 = \sum_{i \in I, j \in J} h_{ij}^2$, where $\|\mathbf{X}\|$ denotes the Frobenius norm of matrix \mathbf{X} , i.e., $\|\mathbf{X}\|^2 = \sum_{i,j} x_{ij}^2$.

2.2 Patterns

We assume \mathbf{A} contains both scaling and shifting factors. We borrow the concepts of “local” and “global” scaling and shifting from Cheng and Church [7], Cho et al. [8], and Aguilar-Ruiz [1] and generalize the definition of data patterns in [1].

Global/local scaling and global/local shifting patterns. A bicluster contains both scaling and shifting patterns when it expresses $a_{ij} = \pi_i \times \alpha_j + \beta_j$, where π_i is the base value for row (e.g., gene) i , α_j is the scaling factor for column (e.g., experimental condition) j , and β_j is the shifting factor for column (e.g., experimental condition) j . We classify the expression into the following four patterns: global scaling (gsc) and global shifting pattern (gsh) when $a_{ij} = \pi_i \times \alpha + \beta$; global scaling (gsc) and local shifting pattern (lsh) when $a_{ij} = \pi_i \times \alpha + \beta_j$; local scaling (lsc) and global shifting pattern (gsh) when $a_{ij} = \pi_i \times \alpha_j + \beta$; and local scaling (lsc) and local shifting pattern (lsh) when $a_{ij} = \pi_i \times \alpha_j + \beta_j$.

3 Data Transformations

Raw data values have a limitation that raw values do not disclose how they vary from the central tendency of the distribution. Therefore, transformation of the raw data is considered one of the most important steps for various data mining processes since the variance of a variable will determine its importance in a given model [16]. In this study, we investigate the following data transformations.

No transformation (NT). No centering or scaling is taken. In other words, $a'_{ij} = a_{ij}$, $\forall i$ and $\forall j$, i.e., the raw matrix is directly input to MSSRCC.

Double centering (DC). DC is defined as $a'_{ij} = a_{ij} - a_{i.} - a_{.j} + a_{..}$, $\forall i$ and $\forall j$. Through DC, each entry of a data matrix \mathbf{A} becomes $a'_{ij} = (\pi_i - \mu_\pi)(\alpha_j - \mu_\alpha)$. Note that we have $a'_{i.} = a'_{.j} = 0$ and consequently $a'_{..} = 0$, since DC transforms the data matrix to have both row means and column means to be 0.

Column/row mean centering (MC). Column MC is defined as $a'_{ij} = a_{ij} - a_{.j}$, $\forall i$ and $\forall j$. Through column MC, each entry becomes $a'_{ij} = \pi_i \alpha_j + \beta_j - \mu_{.j}$. Therefore, row mean, column mean, and whole mean become $a'_{i.} = \pi_i \mu_\alpha + \mu_\beta - a_{.}$, $a'_{.j} = \mu_\pi \alpha_j + \beta_j - a_{.j}$, and $a'_{..} = \mu_\pi \mu_\alpha + \mu_\beta - a_{..}$, respectively. Similarly, row MC is defined similarly with $a_{i.}$. Therefore, row mean, column mean, and whole mean become $a'_{i.} = \pi_i \mu_\alpha + \mu_\beta - a_{i.}$, $a'_{.j} = \mu_\pi \alpha_j + \beta_j - a_{.}$, and $a'_{..} = \mu_\pi \mu_\alpha + \mu_\beta - a_{..}$.

Column/row standard deviation normalization (SDN). Column SDN is defined as $a'_{ij} = \frac{a_{ij}}{\sigma_{\cdot j}}$, $\forall i$ and $\forall j$. Similarly, row SDN is defined with σ_i^2 . Through column and row SDN each column and row has a unit variance, respectively.

Column/row Z-score transformation (ZT). Column ZT is defined as $a'_{ij} = \frac{a_{ij} - a_{\cdot j}}{\sigma_{\cdot j}}$, $\forall i$ and $\forall j$. Similarly, row ZT is defined with a_i and σ_i^2 . It is also called “autoscaling”, where the measurements are scaled so that each column/row has a zero mean and a unit variance [14]. Through ZT, the relative variation in intensity is emphasized, since ZT is a linear transformation, which keeps the relative positions of observations and the shape of the original distribution.

4 Analysis

Now, we analyze the effect of the data transformations on the sum squared residue, RESIDUE(II). Because of space limitation, we focus on analysis on the three data transformations including NT, column SDN, and column ZT, which clearly demonstrate the effect of the specific data transformation.

4.1 No Transformation (NT)

(i, j) -th entry of row $i \in I$ and column $j \in J$ of co-cluster \mathbf{A}_{IJ} is described as $a_{ij} = \pi_i \alpha_j + \beta_j$. Then, the mean of the base values of \mathbf{A}_{IJ} is computed by $\mu_{\pi_I} = \frac{1}{|I|} \sum_{i \in I} \pi_i$ and the mean of the scaling factors by $\mu_{\alpha_J} = \frac{1}{|J|} \sum_{j \in J} \alpha_j$, and the mean of the shifting factors by $\mu_{\beta_J} = \frac{1}{|J|} \sum_{j \in J} \beta_j$. Also, the mean of row i is obtained by $a_{iJ} = \pi_i \mu_{\alpha_J} + \mu_{\beta_J}$, the mean of column j by $a_{IJ} = \mu_{\pi_I} \alpha_j + \beta_j$, and the mean of all the elements by $a_{IJ} = \mu_{\pi_I} \mu_{\alpha_J} + \mu_{\beta_J}$. Using these values, we obtain RESIDUE(II), $h_{ij} = (\pi_i - \mu_{\pi_I})(\alpha_j - \mu_{\alpha_J})$. Consequently, the sum squared residue (SSR) can be computed as $SSR = \|\mathbf{H}_{IJ}\|^2 = \sum_{i \in I, j \in J} h_{ij}^2 = \sum_{i \in I, j \in J} (\pi_i - \mu_{\pi_I})^2 (\alpha_j - \mu_{\alpha_J})^2 = |I||J| \sigma_{\pi_I}^2 \sigma_{\alpha_J}^2$, where $\sigma_{\pi_I}^2 = \frac{1}{|I|} \sum_{i \in I} (\pi_i - \mu_{\pi_I})^2$ and $\sigma_{\alpha_J}^2 = \frac{1}{|J|} \sum_{j \in J} (\alpha_j - \mu_{\alpha_J})^2$.

In fact, SSR shown above is a revisit of Theorems in Aguilar-Ruiz [1]. It shows with no data transformation that SSR is dependent on both the variance of base values and the variance of scaling factors, but independent from shifting factors. Accordingly, any shifting operations such as DC and MC to the given data matrix should not contribute to RESIDUE(II). As also shown in [1], RESIDUE(II) itself has an ability to discover shifting patterns.

4.2 Column Standard Deviation Normalization (SDN)

Column SDN transforms \mathbf{A} to have the constant global scaling factor, i.e., 1, and the local shifting factors, i.e., $\frac{\beta_j}{\alpha_j}$. To be more specific, (i, j) -th entry is transformed as $a_{ij} = \frac{1}{\sigma_{\cdot j}} (\pi_i \alpha_j + \beta_j) = \frac{1}{\sigma_{\cdot j}} \left(\pi_i + \frac{\beta_j}{\alpha_j} \right)$. Then, row mean, column mean,

and whole mean of co-cluster \mathbf{A}_{IJ} are computed by $a_{iJ} = \frac{1}{|J|} \sum_{j \in J} \frac{1}{\sigma_j} (\pi_i \alpha_j + \beta_j)$, $a_{Ij} = \frac{1}{|I|} \sum_{i \in I} \frac{1}{\sigma_j} (\pi_i \alpha_j + \beta_j)$, and $a_{IJ} = \frac{1}{|I||J|} \sum_{i \in I} \sum_{j \in J} \frac{1}{\sigma_j} (\pi_i \alpha_j + \beta_j)$, respectively. Therefore, using RESIDUE(II), we can capture the perfect co-cluster, i.e., zero RESIDUE(II), for all the four expression patterns.

4.3 Column Z-Score Transformation (ZT)

Column ZT transforms \mathbf{A} to have the constant global scaling factor, i.e., 1, and the constant global shifting factor, i.e., $-\mu_\pi$. To be more specific, (i, j) -th entry is transformed as $a_{ij} = \frac{1}{\sigma_j} (\pi_i \alpha_j + \beta_j - a_{.j}) = \frac{1}{\sigma_\pi} (\pi_i - \mu_\pi)$. Then, row mean of co-cluster \mathbf{A}_{IJ} is obtained by $a_{iJ} = \frac{1}{\sigma_\pi} (\mu_{\pi_i} - \mu_\pi) = a_{ij}$, and column mean and whole mean by $a_{Ij} = \frac{1}{\sigma_\pi} (\mu_{\pi_I} - \mu_\pi) = a_{IJ}$. Like column SDN, we obtain zero REDIDUE(II) for all the possible combinations of scaling and shifting patterns.

5 Experimental Results

Now, we empirically show the effect of data transformations on the four publicly available human cancer microarray datasets including Colon cancer [2], Leukemia [12], Lung cancer [3], and MLL [3]. With MSSRCC [8][9], we generate 100×2 or 100×3 co-clusters with random and spectral initializations, setting $\tau = 10^{-3} \|\mathbf{A}\|^2$ and $\tau = 10^{-6} \|\mathbf{A}\|^2$ for batch and local search, respectively. Detailed algorithmic strategies and their contributions are discussed in [9].

Data preprocessing. Since utilizing sophisticated feature selection algorithms is not a main focus, we just apply the simple preprocessing steps usually adopted in microarray experiments as in [6][10][11] to detect differential expression. Details are summarized in Table 1. Further, the gene expression values in Colon dataset were transformed by taking the base-10 logarithm.

Table 1. Description of microarray datasets used in our experiments

| | Colon Leukemia | | Lung | MLL |
|--------------------|-------------------------|--------------------|----------------------|-------------------------------|
| # original genes | 2000 | 7129 | 12533 | 12582 |
| # samples | 62 | 72 | 181 | 72 |
| # sample classes | 2 | 2 | 2 | 3 |
| Sample class names | Normal(20) Tumor(42) | ALL(47) AML(25) | ADCA(150) MPM(31) | ALL(24) AML(25) MLL(23) |
| $ max/min $ | 15 | 5 | 5 | 5 |
| $ max - min $ | 500 | 500 | 600 | 5500 |
| # remaining genes | 1096 | 3571 | 2401 | 2474 |

Abbreviations: ALL – Acute Lymphoblastic Leukemia; AML – Acute Myeloid Leukemia; ADCA – Adenocarcinoma; MPM – Malignant Pleural Mesothelioma; and MLL – Mixed-Lineage Leukemia. The number after each sample class name denotes the number of samples.

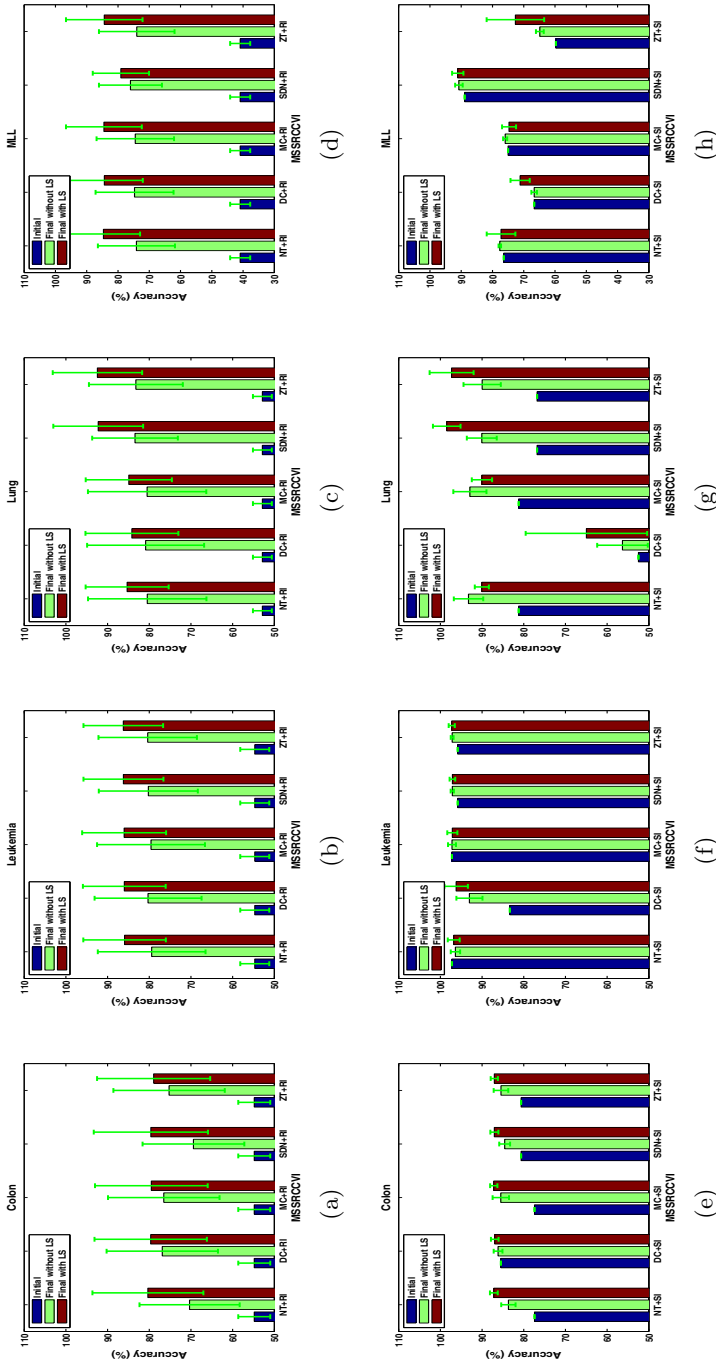


Fig. 1. Average tissue sample clustering accuracy using MSSRCC with RESIDUE(II). The accuracy values are averaged over 1 to 100 gene clusters. (a)-(d) are averaged over 10 random runs and (e)-(h) are obtained with deterministic spectral initialization. Abbreviations: RI - Random Initialization; SI - Spectral Initialization; NT - No Transformation; DC - Double Centering; MC - (column) Mean Centering; SDN - (column) Standard Deviation Normalization; ZT - (column) Z-score Transformation; and LS - Local Search.

Tissue sample clustering evaluation measure. To evaluate the performance of sample clusterings, we quantify tissue sample clustering performance using the following clustering accuracy measure: $accuracy(\%) = \frac{1}{T} \left(\sum_{i=1}^l t_i \right) \times 100$, where T denotes the total number of samples, l the number of sample clusters, and t_i the numbers of the samples correctly clustered into a sample class i .

Performance comparison. Figure 11 illustrates the average tissue sample accuracy using MSSRCC with RESIDUE(II). As reported in [9], spectral initialization and local search strategy play a significant role in improving MSSRCC performance. However, in this paper, we are more interested in how data transformations affect the tissue sample accuracy performance.

NT, DC, and MC with random initialization ((a)-(d)) and NT and MC with spectral initialization ((e)-(h)) result in nearly similar accuracy. Note that RESIDUE(II) is not affected by shifting factors, but still affected by the scaling factors as first articulated in [11] and also revisited in the analysis. To be more specific, the residue with NT on data matrices with local scaling factors is $(\pi_i - \mu_{\pi_I})(\alpha_j - \mu_{\alpha_J})$, on which interestingly the residue with DC or MC is also dependent. In our experiment, DC with random initialization generates compatible accuracy with that of other data transformations ((a)-(d)), however it is relatively less effective with spectral initialization ((f)-(h)). For all the considered datasets, MC presents compatible performance that of NT, but not better than that of either SDN or ZT.

As analyzed in the previous section, both column SDN and column ZT help MSSRCC with RESIDUE(II) capture perfect co-clusters, thus MSSRCC with column SDN or column ZT is supposed to generate similar accuracy and also better accuracy than those with NT, DC, or MC. Accordingly, they lead to the best accuracy values for most cases ((a)-(h)).

6 Conclusion and Remark

Aguilar-Ruiz [11] issues the need of a new metric to discover both scaling and shifting patterns, showing that the sum squared residue can discover any shifted patterns but may not capture some scaled patterns. To answer this need, we propose a simple remedy that helps the residue resolve its dependency on scaling variances. We suggest to take specific data transformations through which the hidden scaling factors are implicitly removed. We analyze the effect of various data transformation on RESIDUE(II) [8] for data matrices with global/local scaling and global/shifting factors.

Both analysis and experimental results reveal that column standard deviation normalization and column Z-score transformation are effective for RESIDUE(II). To be more specific, through MSSRCC with RESIDUE(II) and the two data transformations, we are able to discover coherent patterns with both scaling and shifting factors. The transformed matrix contains the constant global scaling factor 1 and local shifting factors and gives the perfect residue score, *i.e.*, zero RESIDUE(II).

Note that RESIDUE(II) is a special case (scheme 6) of the six Euclidean co-clustering schemes in Bregman co-clustering algorithms [4]. Our formal

analysis can be applicable to any clustering/co-clustering algorithm that has a closed-form of objective function, thus our potential future direction is to apply the proposed analysis to the remaining co-clustering models in Bregman co-clustering algorithms and formally characterize each of Bregman co-clustering algorithms.

References

1. Aguilar-Ruiz, J.S.: Shifting and scaling patterns from gene expression data. *Bioinformatics* 21(20), 3840–3845 (2005)
2. Alon, U., et al.: Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *PNAS* 96(12), 6745–6750 (1999)
3. Armstrong, S.A., et al.: MLL translocations specify a distinct gene expression profile that distinguishes a unique leukemia. *Nature Genetics* 30, 41–47 (2002)
4. Banerjee, A., Dhillon, I.S., Ghosh, J., Merugu, S., Modha, D.: A Generalized maximum entropy approach to Bregman co-clustering and matrix approximation. *Journal of Machine Learning Research* 8, 1919–1986 (2007)
5. Ben-Dor, A., Chor, B., Karp, R., Yakhini, Z.: Discovering Local Structure in Gene Expression Data: The Order-Preserving Submatrix Problem. *Journal of Computational Biology* 10(3–4), 373–384 (2003)
6. Bø, T.H., Jonassen, I.: New feature subset selection procedures for classification of expression profiles. *Genome Biology* 3(4) (2002)
7. Cheng, Y., Church, G.M.: Biclustering of expression data. In: *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology (ISMB)*, vol. 8, pp. 93–103 (2000)
8. Cho, H., Dhillon, I.S., Guan, Y., Sra, S.: Minimum sum squared residue based co-clustering of gene expression data. In: *Proceedings of the Fourth SIAM International Conference on Data Mining (SDM)*, pp. 114–125 (2004)
9. Cho, H., Dhillon, I.S.: Co-clustering of human cancer microarrays using minimum sum-squared residue co-clustering (MSSRCC) algorithm. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (IEEE/ACM TCBB)* 5(3), 114–125 (2008)
10. Dettling, M., Bühlmann, P.: Supervised clustering of genes. *Genome Biology* 3(12) (2002)
11. Dudoit, S., Fridlyand, J.: A Prediction-based Resampling Method for Estimating the Number of Clusters in a Dataset. *Genome Biology* 3(7) (2002)
12. Golub, T.R., et al.: Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring. *Science* 286, 531–537 (2002)
13. Hartigan, J.A.: Direct clustering of a data matrix. *Journal of the American Statistical Association* 67(337), 123–129 (1972)
14. Kowalski, B.R., Bender, C.F.: Pattern recognition: A powerful approach to interpreting chemical data. *Journal of the American Chemical Society* 94(16), 5632–5639 (1972)
15. Madeira, S.C., Oliveira, A.L.: Biclustering algorithms for biological data analysis: a survey. *IEEE Transactions on Computational Biology and Bioinformatics (IEEE ACM/TCBB)* 1(1), 24–45 (2004)
16. Sánchez, F.C., Lewi, P.J., Massart, D.L.: Effect of different preprocessing methods for principal component analysis applied to the composition of mixtures: detection of impurities in HPLC-DAD. *Chemometrics and Intelligent Laboratory Systems* 25(2), 157–177 (1994)

A Better Strategy of Discovering Link-Pattern Based Communities by Classical Clustering Methods

Chen-Yi Lin¹, Jia-Ling Koh², and Arbee L.P. Chen³

¹ Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan
d9562820@oz.nthu.edu.tw

² Department of Computer Science and Information Engineering,
National Taiwan Normal University, Taipei, Taiwan
jlkoh@csie.ntnu.edu.tw

³ Department of Computer Science, National Chengchi University, Taipei, Taiwan
alpchen@cs.nccu.edu.tw

Abstract. The definition of a community in social networks varies with applications. To generalize different types of communities, the concept of link-pattern based community was proposed in a previous study to group nodes into communities, where the nodes in a community have similar intra-community and inter-community interaction behaviors. In this paper, by defining centroid of a community, a distance function is provided to measure the similarity between the link pattern of a node and the centroid of a community. The problem of discovering link-pattern based communities is transformed into a data clustering problem on nodes for minimizing a given objective function. By extending the partitioning methods of cluster analysis, two algorithms named G-LPC and KM-LPC are proposed to solve the problem. The experiment results show that KM-LPC outperforms the previous work on the efficiency, the memory utilization, and the clustering result. Besides, G-LPC achieves the best result approaching the optimal solution.

Keywords: Social Network, Link-Pattern based Community, Clustering Algorithms.

1 Introduction

Social network analysis is an established field in sociology. A social network is mostly modeled by a graph in which a node represents an individual and an edge between two nodes denotes a social interaction between the corresponding individuals. In recently years, because of the increasing availability of social network data on the Web 2.0 platform, the study of social network analysis has emerged into an active research field. The community structure is an important topological characteristic of social networks, which provides a basis for further analysis of social networks. Accordingly, discovering the communities from a social network has become an essential problem on social network analysis.

The definitions of a community in social networks vary with applications. In most studies, finding groups of nodes within which the interconnections are dense but

between which the interconnections are sparse is attractive to users. In earlier papers, the graph partitioning techniques were adopted to divide nodes into subsets by discovering the various kinds of cuts in a graph such as average cuts [1], normalized cuts [9], min-max cuts [2], and maximum flow/minimum cuts [3, 5]. However, in some applications such as those in blogosphere, a group of individuals linking to the same set of blogs indicates a set of latent friends with common interests even though they sparsely link to each other [6, 8]. Therefore, to generalize the different types of communities, the concept of *link-pattern based community* was proposed in [7].

A link-pattern based community is a group of nodes which have a similar link patterns, i.e., the nodes in the same community have similar intra-community and inter-community interaction behaviors. For example, the individuals, denoted by the nodes in Figure 1, are grouped into three communities: $C_1 = \{v_1, v_2, v_3, v_4\}$, $C_2 = \{v_5, v_6, v_7, v_8\}$, and $C_3 = \{v_9, v_{10}, v_{11}, v_{12}\}$. The nodes in C_1 link densely to each other, link moderately to the nodes in C_2 , and link sparsely to the nodes in C_3 . On the other hand, the nodes in C_2 link moderately to the nodes in C_1 and link sparsely to the nodes in both C_2 and C_3 . The nodes in C_3 also link to other nodes within the community and between the communities in similar ways. The concept of a *community prototype graph* was also proposed, which consists of a set of *community nodes* (i.e., C_1 , C_2 and C_3 in Figure 1) and a set of edges among community nodes to represent the community structures. Accordingly, the graph and its community prototype graph are represented as affinity matrices in which each entry represents the weight of an edge between two corresponding nodes. An iterative algorithm named CLGA was developed to find the optimal community prototype graph from the graph by solving the optimization problem of matrix approximation.

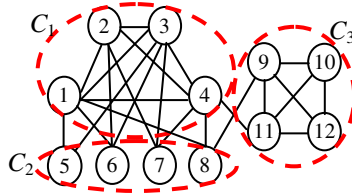


Fig. 1. An example of the link-pattern based communities

The number of individuals in a social network is enormous in most cases, and the size of the affinity matrix of the original graph is determined by the number of individuals. Consequently, the algorithms for solving the problem of discovering the link-pattern based communities are challenging on the requirement of memory usage and the performance efficiency. However, in [7], in order to find the optimal community prototype graph, it requires exhaustive search by moving a node from one community to another community. In each time of iteration, the affinity matrix of the corresponding community prototype graph has to be recomputed. As a result, CLGA is computationally infeasible. Besides, CLGA has to maintain the affinity matrix of the community prototype graph. Consequently, the memory requirement of CLGA is at least double of the one required by the affinity matrix of the original graph.

According to the concept of the link-pattern based community, the edges with weights incident to a node are essential features which imply the link-pattern of the

node. In this paper, we reformulate the problem based on the proximity of the links of nodes to discover the link-pattern based community structures, and evaluate the quality of the community structures according to the similarity of the weights of the intra-links within a community and that of the weights of the inter-links between the community and every other community. It is proved that the reformulated problem of communities discovering is equivalent to the problem defined in [7]. In order to get a good clustering result, two different strategies are provided to select sample nodes for determining the initial the communities. Based on the extracted initial community structures, two algorithms, named G-LPC and KM-LPC, based on the classical clustering methods, are provided to discover the communities. The experiment results show that KM-LPC outperforms CLGA not only on the efficiency and memory utilization, but also on the clustering result. Although G-LPC requires the most computational cost than the others, it achieves the best result approaching the optimal solution.

The remaining sections of this paper are organized as follows. The reformulated problem and the proposed algorithms are described in Sections 2 and 3, respectively. The performance study is reported in Section 4, which shows the effectiveness, efficiency, and memory usage of the proposed methods. Finally, in Section 5, we conclude this paper and discuss directions for our future studies.

2 Preliminaries

In this section, the problem proposed in [7] for discovering the link-pattern based communities is introduced briefly. Then we reformulate the problem and provide solutions to the problem in Section 3.

2.1 Problem of Matrix Approximation Optimization

Suppose an undirected weighted graph $G = (V, E, A)$ is given, where V is a set of nodes $\{v_1, v_2, \dots, v_n\}$, E is a set of edges (v_i, v_j) , and A is the affinity matrix of G . A is an $n \times n$ symmetric matrix, in which $A[i, j]$ is a positive value representing the weight of the edge between nodes v_i and v_j if $(v_i, v_j) \in E$; otherwise, $A[i, j]$ is set to be 0.

A community prototype graph defined in [7] consists of a set of community nodes and a set of edges among community nodes associated with weights to represent the community structures. Let K denote the number of communities specified by the users. The *community structure matrix* B is a $K \times K$ matrix for representing the weights of intra-links and inter-links of the community nodes. Besides, an $n \times K$ matrix C with binary values denotes the community membership of each node, where each node belongs to exact one community and there is no empty community. The affinity matrix of a community prototype graph, denoted as A' , is an $n \times n$ matrix which is the result of CBC^T . Accordingly, the challenge of discovering the link-pattern based communities is how to find C and B such that $\|A - A'\|^2$ is minimized.

[Example 2.1]. The nodes of the social network shown in Figure 2(a) are required to be grouped into two link-pattern based communities. The affinity matrix of the corresponding graph is shown as Figure 2(b). When the two communities are constructed as $C_1 = \{v_1, v_2, v_3, v_4\}$ and $C_2 = \{v_5, v_6, v_7, v_8\}$, the corresponding community prototype graph shown in Figure 2(c) is the optimal solution of this case. Accordingly, the

corresponding matrices C and B of the constructed community structures are shown in Figure 2(d). Besides, the affinity matrix of the community prototype graph is shown as Figure 2(e). Therefore, the difference between the affinity matrix of the graph and the affinity matrix of the community prototype graph is 3.5.

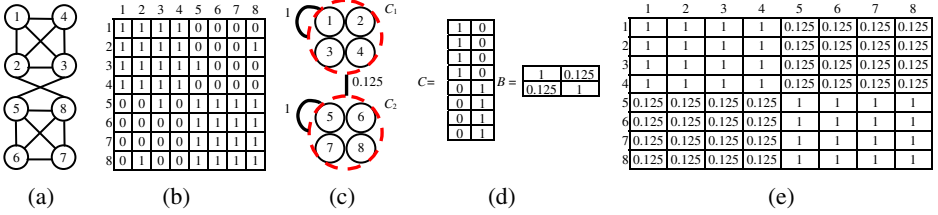


Fig. 2. An example of a graph and its optimal community prototype graph

2.2 Problem Definition

Based on the definition of the link-pattern based community, a good solution of the problem tends to group the nodes with similar intra-community and inter-community interaction behaviors into the same community. The link pattern of a node is characterized by its edges linked to other nodes, and the link pattern of a community by the aggregate link patterns of the nodes in the community. An object function is designed to evaluate the quality of the communities by the distance between the link patterns of each community and its nodes.

Suppose the members in each community have been assigned. Let $C_u = \{v_{u_1}, v_{u_2}, \dots, v_{u_{n_u}}\}$ and $C_v = \{v_{v_1}, v_{v_2}, \dots, v_{v_{n_v}}\}$ denote two communities, where n_u and n_v denote the number of nodes in C_u and C_v , respectively. The affinity matrix for the nodes in C_u , denoted A_{C_u} , is an $n_u \times n_u$ sub-matrix of A which is the affinity matrix of the original graph. The intra-community link pattern of v_{u_i} is represented by the i^{th} row in A_{C_u} with the weights of the intra-community edges of v_{u_i} . Moreover, the affinity matrix A_{C_u, C_v} for the nodes in C_u with the nodes in C_v is an $n_u \times n_v$ sub-matrix of A , in which the i^{th} row represents the inter-community edges of v_{u_i} with the nodes in C_v .

Consequently, the intra-community pattern of the community C_u is represented by a vector of n_u dimensions where each dimension contains the average weight of the intra-community link patterns of all the nodes in C_u as the following formula shows:

$$AVG_{C_u} = \frac{\sum_{i=1}^{n_u} \sum_{j=1}^{n_u} A_{C_u} [i, j]}{n_u \times n_u} \quad (1)$$

The inter-community pattern of the community C_u with C_v is represented by a vector of n_v dimensions where each dimension contains the average weight of the inter-community edges of all the nodes in C_u with C_v as the following formula shows:

$$AVG_{C_u, C_v} = \frac{\sum_{i=1}^{n_u} \sum_{j=1}^{n_v} A_{C_u, C_v} [i, j]}{n_u \times n_v} \quad (2)$$

Therefore, the *intra-distance* of C_u and the *inter-distance* between C_u and C_v are defined by the following formulas:

$$SSD_{C_u} = \sum_{i=1}^{n_u} \sum_{j=1}^{n_u} (A_{C_u} [i, j] - AVG_{C_u})^2 \tag{3}$$

$$SSD_{C_u, C_v} = \sum_{i=1}^{n_u} \sum_{j=1}^{n_v} (A_{C_u, C_v} [i, j] - AVG_{C_u, C_v})^2 \tag{4}$$

[Example 2.2]. The nodes of the graph shown in Figure 2(a) are grouped into C_1 and C_2 . Consequently, the corresponding matrices A_{C_1} , A_{C_2} , A_{C_1, C_2} , and A_{C_2, C_1} are shown in Figure 3(a), (b), (c), and (d), respectively. Accordingly, the values of AVG_{C_1} , AVG_{C_2} , AVG_{C_1, C_2} , and AVG_{C_2, C_1} are 1, 1, 0.125, and 0.125, respectively. Besides, the values of SSD_{C_1} , SSD_{C_2} , SSD_{C_1, C_2} , and SSD_{C_2, C_1} are 0, 0, 1.75, and 1.75, respectively.

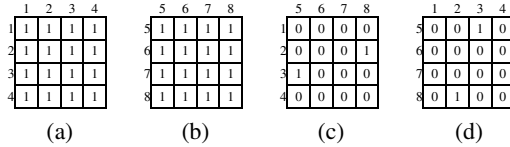


Fig. 3. The affinity matrices A_{C_1} , A_{C_2} , A_{C_1, C_2} , and A_{C_2, C_1}

Consequently, the problem of discovering the link-pattern based communities from a social network is formulated by minimizing the sum of the intra- and inter-distances of the communities as the following. According to a given positive integer K which denotes the number of communities, the optimized communities are given by minimizing the following objective function:

$$\arg \min_{C_1, C_2, \dots, C_K} \left(SSD_{C_1} + \sum_{j=1 \wedge j \neq i}^K SSD_{C_i, C_j} \right) \tag{5}$$

where the n nodes in the social network are separated into C_1, \dots, C_K such that each node belongs to exact one community. Besides, there is no empty community allowed.

It is deducible that the optimal solution discovered according to the proposed objective function is the same as the one discovered by CLGA. Suppose the members in each community have been assigned. Based on our observation on matrix B , the diagonal entry $B[i, i]$ is equal to AVG_{C_i} and the non-diagonal entry $B[i, j]$ in B is equal to AVG_{C_i, C_j} . Therefore, for any pair of v_x and v_y in C_i , the entry $A[x, y]$ in A' of the community prototype graph is equal to AVG_{C_i} . On the other hand, the entry $A[x, y]$ is equal to AVG_{C_i, C_j} for any node v_x in C_i and node v_y in C_j . Since each node belongs to exact one community, $\|A - A'\|^2$ is equal to $\sum_{i=1}^K \left(SSD_{C_i} + \sum_{j=1 \wedge j \neq i}^K SSD_{C_i, C_j} \right)$.

3 The Proposed Algorithms

According to the objective function defined in Section 2, the task of discovering the link-pattern based communities is an optimization problem of minimizing the

objective function. However, it is computationally infeasible to exhaustively search the global optimum solution of this problem. To provide a heuristic algorithm for solving this problem, a greedy based algorithm and a K-Means based algorithm are proposed to discover the disjoint clusters of data nodes which correspond to the link-pattern based communities of these nodes.

3.1 Basic Idea

In order to get a value of the objective function as small as possible, one effective way is to adopt the greedy-based algorithm in which each node is iteratively assigned to a community such that the obtained value of the objective function is minimal. The K-Means algorithm [4] is a typical method of cluster analysis. The goal of K-Means is to minimize the sum of squared distances between data and the mean of the corresponding cluster, which is similar to the goal of the objective function defined in this paper. Therefore, a K-Means based algorithm is also proposed to discover the communities.

According to the given information of graph G , row i in the affinity matrix A of G provides the information of the link pattern of node v_i , which forms the feature vector of v_i . For a community C_u , the feature vector of C_u is an n -dimensional vector. The j^{th} dimension in the feature vector of C_u contains the value of AVG_{C_u} if node v_j belongs to C_u ; otherwise, it contains the value of AVG_{C_u, C_v} if node v_j belongs to another community C_v . During the progressive process of clustering, the feature vector of a community will be used to be its centroid. Therefore, the value of the objective function defined in formula (5) corresponds to the sum of squared distance between the feature vector of each node and the centroid of its community.

To adopt the clustering methods for discovering the communities, first, K initial centroids are determined by performing a clustering on the sample nodes selected from the social network. Next, each node in the social network is assigned to a community by executing one of the proposed two algorithms. The centroid of a community will be updated according to the nodes assigned to the community. Relative to the new centroids, the above process is repeated until there is no change in communities.

3.2 Determining Initial Centroids

In terms of the quality of the clustering result, determining a set of appropriate initial centroids of clusters is the key step of clustering algorithms. However, it is not easy to determine a 'good' set of initial centroids of clusters without knowing the connectivity among the nodes. Therefore, some sample nodes are chosen from the graph. Then the agglomerative hierarchical clustering method is adopted to separate these sample nodes into K disjoint clusters. Finally, the mean of the feature vectors of the sample nodes assigned to a cluster is set as the initial centroid of the cluster.

A straightforward method is to choose the sample nodes randomly. However, a good quality of the clustering result is obtained by chance based on which sample nodes are chosen. In order to understand the characteristics of the link-pattern based communities in a real dataset, the Enron email dataset is analyzed by CLGA. It is observed that the nodes which are distributed to the same community usually have the similar degrees. In other words, it is more possible that two nodes belong to the same community as their degrees are closer. Accordingly, in our second strategy, the sample

nodes are selected based on the degrees of the nodes. Moreover, the number of chosen sample nodes is determined by $K \times U$, where K is the number of expected communities and U is a user-specified integer which is at least one and less than $\lfloor n/K \rfloor$.

By summarizing the above considerations, the following two strategies are used to select sample nodes from the given graph G .

(1) Picking by random

$K \times U$ sample nodes are chosen from G randomly without replacement.

(2) Picking by node degree

The nodes with the identical degree are assigned to the same group. Within each group, U nodes are chosen randomly without replacement.

After selecting sample nodes by one of the above-mentioned strategies, the sample nodes are separated into K clusters by the agglomerative hierarchical clustering. At the beginning, each sample node is considered as an individual cluster. Let c_x and c_y denote the centroids of two clusters C_x and C_y , respectively. The distance between C_x and C_y is decided by calculating the Euclidean distance between c_x and c_y :

$$\text{dist}(C_x, C_y) = \|c_x - c_y\| \quad (6)$$

Iteratively, two nearest clusters are chosen to be merged into a cluster until K clusters remain. Whenever two clusters are merged to generate a new cluster C_l , the centroid of C_l , which is denoted as c_l , is obtained according to the following formula:

$$c_l = \frac{1}{n_l} \sum_{SN \in C_l} SN.fv \quad (7)$$

where n_l denotes the number of sample nodes in C_l , SN denotes a sample node in C_l , and $SN.fv$ denotes the feature vector of the sample node.

[Example 3.1]. In the graph shown in Figure 2(a), the number of distinct degree values of nodes is 2. By using the picking by node degree strategy to select sample nodes, the nodes in the graph are separated into two groups $\{v_1, v_4, v_6, v_7\}$ and $\{v_2, v_3, v_5, v_8\}$. When U is set as 1, the number of chosen sample nodes from each group is 1. Suppose nodes v_4 and v_3 are chosen as the sample nodes. Accordingly, $\langle 1, 1, 1, 1, 0, 0, 0, 0 \rangle$ and $\langle 1, 1, 1, 1, 1, 0, 0, 0 \rangle$ of v_4 and v_3 are used as the initial centroids of C_1 and C_2 .

3.3 Communities Discovering

After the initial centroids of K clusters are obtained, each node in G is then assigned to the closest cluster, and the centroid of each cluster is updated according to the nodes assigned to the cluster. Then two algorithms are proposed to reassign each node in G to the clusters iteratively until the result converges.

(1) Discovering Initial Communities

According to the initial centroids, each node in the graph G is assigned to the closest cluster one by one. The distance between a node v and C_u is determined:

$$\text{dist}(v, C_u) = \|v.fv - c_u\| \quad (8)$$

where $v.fv$ denotes the feature vector of node v .

When the assignment of all the nodes to the clusters completes, the initial structures of communities are constructed. Accordingly, the feature vector of each initial community is computed to update its centroid.

[Example 3.2]. By continuing the result of Example 3.1, the nodes in the graph are separated into two initial clusters $C_1 = \{v_1, v_2, v_4\}$ and $C_2 = \{v_3, v_5, v_6, v_7, v_8\}$. The values of AVG_{C_1} , AVG_{C_2} , and AVG_{C_1, C_2} are 1, 0.76, and 0.2667, respectively. Therefore, the centroids of C_1 and C_2 are $\langle 1, 1, 0.2667, 1, 0.2667, 0.2667, 0.2667, 0.2667 \rangle$ and $\langle 0.2667, 0.2667, 0.76, 0.2667, 0.76, 0.76, 0.76, 0.76 \rangle$, respectively.

(2) The Clustering Algorithms

The two clustering algorithms proposed to discover the community structures according to the initial communities are introduced.

(A) The Greedy based Algorithm

The process of the Greedy based algorithm for discovering Link Pattern-based Communities (abbreviated as G-LPC) aims to distribute a node to the cluster such that the objective function is minimized locally. In each time of iteration, one by one, each node is checked to decide the cluster which the node is assigned to. The node is moved from the cluster, which it was assigned to in last time of iteration, to another cluster if the value of objective function will be reduced after the movement. The above process is repeated until there is no change in clusters.

[Example 3.3]. According to the result of Example 3.2, the two initial clusters are $C_1 = \{v_1, v_2, v_4\}$ and $C_2 = \{v_3, v_5, v_6, v_7, v_8\}$. If we move node v_1 from C_1 to C_2 , AVG_{C_1} , AVG_{C_2} , and AVG_{C_1, C_2} are recomputed to be 1, 0.6111, and 0.4167, respectively. Also, the centroids of C_1 and C_2 are updated. Finally, the new value of the objective function is obtained, which is 14.3889. The value is larger than the previous one, i.e. 10.4267; hence, v_1 is remained in C_1 .

(B) The K-Means based Algorithm

The process of the K-Means based algorithm for discovering Link Pattern-based Communities (abbreviated as KM-LPC) assigns each node to the cluster whose centroid is nearest to the feature vector of the node. At the end of each time of iteration, the centroid of a cluster is recomputed according to the nodes which are assigned to the cluster. The above process is repeated until no member of any cluster changes.

[Example 3.4]. By continuing the result of Example 3.2, the initial centroids of C_1 and C_2 are $\langle 1, 1, 0.2667, 1, 0.2667, 0.2667, 0.2667, 0.2667 \rangle$ and $\langle 0.2667, 0.2667, 0.76, 0.2667, 0.76, 0.76, 0.76, 0.76 \rangle$, respectively. The distances between the feature vector $\langle 1, 1, 1, 1, 0, 0, 0, 0 \rangle$ of v_1 and the centroids of C_1 and C_2 are 0.9068 and 1.9953; consequently, v_1 is remained in C_1 . Similarly, other nodes are assigned to the closest clusters. In the end of this loop, the members of C_1 and C_2 are $\{v_1, v_2, v_3, v_4\}$ and $\{v_5, v_6, v_7, v_8\}$. By KM-LPC, the final community result is $C_1 = \{v_1, v_2, v_3, v_4\}$ and $C_2 = \{v_5, v_6, v_7, v_8\}$, which is the optimal solution of the link-pattern based communities with 2 communities.

4 Performance Evaluation

In order to evaluate the effectiveness, efficiency, and memory requirement of the proposed algorithms, G-LPC and KM-LPC are implemented by MATLAB ver. 7.0.1. Furthermore, CLGA [7] is also implemented for comparison. All the experiments are performed on a personal computer with the Intel Pentium Core 2 Quad CPU, 2 GB of main memory, and running the Microsoft Windows XP.

4.1 Datasets

Two real datasets: Enron email DB¹ and DBLP Bibliography DB² are used in the following experiments. The Enron email DB contains the emails of 151 employees. The dataset is modeled by a graph in which the weight of the edge between two nodes is set to be 1 if any one of the corresponding employees has ever sent an email to the other; otherwise, the edge weight is set to be 0. On the other hand, the DBLP dataset contains the publication information of approximate 700,000 authors. In order to reduce the size of the dataset, we select the authors who have at least 75 coauthors, and the authors who publish more than 10 papers with one of the previously selected authors to run the experiments. As a result, only 7,356 authors are selected. Then an undirected weighted graph is constructed, in which a node represents one of the selected 7,356 authors; in addition, the weight of an edge between two nodes is set to be the number of collaborations between the two corresponding authors normalized by the maximum number of collaborations between any two authors.

4.2 Results and Discussions

There are three parts of experiments to be performed. In the first part, the Enron email dataset is used to evaluate the quality of obtained communities, the execution time, and the memory requirement of the proposed algorithms and the previous work. Next, by using the DBLP dataset, the detailed comparisons of the parameters setting in KM-LPC are observed in the second part of experiments. At last, the properties of the discovered link pattern-based communities from the DBLP dataset are analyzed.

4.2.1 Comparison between the Proposed Algorithms and CLGA

[Exp. 1]. The Enron email dataset is used in this part of experiments. The parameter U is set to be 1. In addition, the strategy of picking by node degree is adopted.

By varying the value of K , the values of the objective function for the communities discovered by the proposed two algorithms and CLGA are shown in Figure 4(a). It is indicated that the community structures obtained by the two proposed algorithms are both better than the one obtained by CLGA. Besides, G-LPC gets the best result. In CLGA, if users have no prior knowledge about the social network, the initial setting of the community structures is determined by random. The result shows that the random setting adopted in CLGA usually results in a poorer result by comparing with our algorithms. Figure 4(b) shows the execution time of the algorithms, in which the execution time of KM-LPC is much less than the time of the others. It shows that KM-LPC provide a significant improvement for discovering the communities efficiently. Moreover, the sizes of memory requirement of the algorithms are shown in Figure 4(c). In our algorithms, in addition to the affinity matrix of the given graph, only the centroids of the communities have to be maintained in main memory instead of storing another affinity matrix of the community prototype graph adopted by CLGA. Therefore, both the proposed algorithms require less memory than CLGA.

¹ <http://www.cs.cmu.edu/~enron/>

² <http://www.informatik.uni-trier.de/~ley/db/>

To decide the community of a node, G-LPC computes the new value of the objective function for each possible movement of the node. On the other hand, in KM-LPC, it only computes the distance between the feature vector of a node and all the centroids of communities to find the community with the nearest centroid. As a result, although G-LPC requires the most computational cost than the others, it achieves the better result than the others when the value of K is increasing. Besides, when the strategy of picking by random is adopted, the performances of our algorithms are also better than that of CLGA. Due to space restrictions, the details of this part of experiment are not shown here.

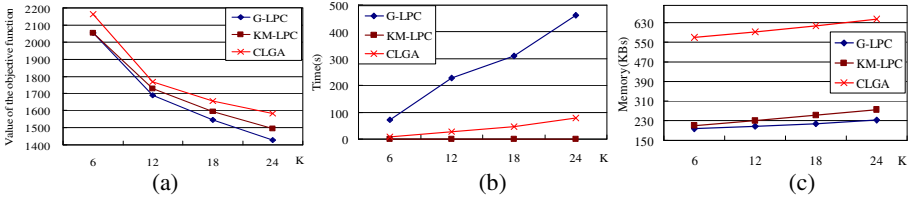


Fig. 4. Comparisons between our algorithms and CLGA in the Enron email dataset

4.2.2 Comparison of the Parameters Setting in KM-LPC

Because the number of authors chosen from the DBLP dataset is 7,356, the graph constructed for the dataset is large and complicated. CLGA cannot run on the DBLP dataset under limited memory. Thus, in this part of experiments, we will observe the effect of the parameters U and K on KM-LPC. Two versions of the algorithm are implemented where the strategy for selecting the sample nodes adopts the picking by random and picking by node degree, individually.

[Exp. 2]. The parameter U is set to be 1 and the value of K is varied from 100 to 220. The results of the objective function of the 2 different versions of KM-LPC are shown in Table 1. The bold-faced values shown in the table indicate the better result in the two versions of KM-LPC. In most cases, it obtains the better result of the communities by using the picking by node degree strategy than using the picking by random strategy. However, since most of the edge weights in the graph are very small, the difference between the obtained objective function values of these two strategies is not obvious. Table 2 shows the execution time of the 2 versions of KM-LPC, in which the bold-faced values represent the same meaning as used in Table 1. According to the results, for KM-LPC, the version by adopting the picking by node degree strategy runs faster than the one by adopting the picking by random strategy in most cases.

[Exp. 3]. For a setting of K , the value of U is varied from 1 to 3. In addition, KM-LPC is performed by combined with the picking by node degree strategy. Table 3 shows that a larger value of U gets smaller value of the objective function in most cases. That is, for KM-LPC, picking more sample nodes from the graph to determine the initial centroids of the clusters is a good strategy to get better result. However, when the value of U increases, the number of selected sample nodes increases. The cost of performing the hierarchical clustering to determine the initial centroids of clusters also increases tremendously. Therefore, as the results shown in Figure 5, the execution time of KM-LPC substantially increases when the value of U increases.

Table 1. The values of the objective function obtained by Exp. 2

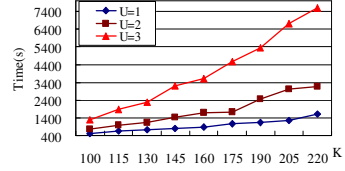
| Algo. \ K | 100 | 115 | 130 | 145 | 160 | 175 | 190 | 205 | 220 |
|-----------------|---------------|---------------|---------------|--------|---------------|---------------|---------------|---------------|---------------|
| KM-LPC (Random) | 7212.5 | 7207.7 | 7186.1 | 7178.1 | 7162.6 | 7149.0 | 7132.1 | 7119.4 | 7101.3 |
| KM-LPC (Degree) | 7211.0 | 7206.8 | 7191.6 | 7177.1 | 7154.8 | 7144.1 | 7130.2 | 7118.5 | 7098.9 |

Table 2. The running time (sec.) obtained by Exp. 2

| Algo. \ K | 100 | 115 | 130 | 145 | 160 | 175 | 190 | 205 | 220 |
|-----------------|--------------|--------------|--------------|--------------|--------------|-------------|---------------|---------------|-------------|
| KM-LPC (Random) | 915.1 | 653.5 | 935.5 | 1320.2 | 1396.1 | 1459.5 | 1576 | 1182.4 | 1431 |
| KM-LPC (Degree) | 518.3 | 643.6 | 743.8 | 811.4 | 879.1 | 1048 | 1152.2 | 1258.7 | 1604.8 |

Table 3. The values of the objective function by varying the values of K and U

| U \ K | 100 | 115 | 130 | 145 | 160 | 175 | 190 | 205 | 220 |
|-------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|
| 1 | 7211.0 | 7206.8 | 7191.6 | 7177.1 | 7154.8 | 7144.1 | 7130.2 | 7118.5 | 7098.9 |
| 2 | 7210.9 | 7205.5 | 7189.6 | 7174.6 | 7159.7 | 7143.7 | 7130.1 | 7115.7 | 7099.0 |
| 3 | 7210.7 | 7205.3 | 7190.3 | 7173.4 | 7159.3 | 7143.4 | 7127.8 | 7112.1 | 7097.2 |

**Fig. 5.** The running time of Exp. 3

4.2.3 Property Study of the Discovered Link Pattern-Based Communities

[Exp. 4]. The parameter U is set to be 1 and the value of K is varied from 25 to 200. In addition, KM-LPC is performed by combined with the picking by node degree strategy. Suppose a node v_x belongs to a community C_i . The *intra-community-interaction* of node v_x is defined to be the sum of the weights of the edges between v_x and all the other nodes belonging to C_i divided by the sum of the weights of all edges of v_x . A node with a high value of the intra-community-interaction implies that the edges of the node mainly connect to the other nodes within the same community. In this experiment, the property of the discovered communities is studied by measuring the average intra-community-interaction of the nodes in the graph. When the values of K are set to be 200, 100, 50, and 25, the values of the obtained average intra-community-interaction are 0.70, 0.79, 0.84, and 0.91, respectively. It is indicated that the value of the average intra-community-interaction tends upwards by decreasing the value of K .

By analyzing the discovered communities by KM-LPC in detail, when a small value of K is given, most of the communities have dense connections within the community and sparse connections with other communities. However, when the value of K becomes as large as 200, two different types of communities are observed. The first type of communities has dense connections within the community. Although the second type of communities has sparse connections within the community, the nodes in each community consistently connect to the nodes in a certain set of dense communities.

According to the semantics of the dataset, the authors assigned to one of the first type of communities have strong co-author relationship within the community. Thus, the members of a community in first type have common research topics. On the other hand, in the second type of communities, although the authors cooperate seldom with each other, they co-work with the authors in the same set of the first type communities. Therefore, the authors assigned to a second type community also have the similar research interests, who are potential partners with each other. This interesting finding

is useful for authors to find possible cooperators. Consequently, depending on the needs of users, KM-LPC can discover the different meaningful communities by varying the value of K .

5 Conclusions and Future Work

In this paper, we reformulate the problem of discovering link-pattern based communities from a social network based on the similarity of link patterns of the nodes within each community. The problem of discovering link-pattern based communities is transformed to a classical clustering problem. Two algorithms named G-LPC and KM-LPC are proposed based on the classical clustering methods. The experiment results with the real datasets demonstrate that KM-LPC is better than CLGA not only on the discovered communities but also on the efficiency and memory utilization. Although the computational cost of G-LPC is higher than the others, its result is the best approaching the optimal solution. Finally, in most cases, picking by node degree is a good strategy to select the sample nodes for deciding the initial community centroids.

In some social networks, it is allowed that an individual belongs to multiple communities. To extend the concept of link-pattern based communities in this environment for identifying the communities is under our investigation. Moreover, how to determine a proper number of communities for discovering a set of semantically meaningful communities is another important issue for our future study.

References

1. Chan, P.K., Schlag, M.D.F., Zien, J.Y.: Spectral K-Way Ratio-Cut Partitioning and Clustering. In: Proceedings of the 30th Design Automation Conference, pp. 749–754 (1993)
2. Ding, C.H.Q., He, X., Zha, H., Gu, M., Simon, H.D.: A Min-Max Cut Algorithm for Graph Partitioning and Data Clustering. In: Proceedings of the 1st IEEE International Conference on Data Mining, pp. 107–114 (2001)
3. Flake, G., Lawrence, S., Giles, C.: Efficient Identification of Web Communities. In: Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 150–160 (2000)
4. Hartigan, J.A.: Clustering algorithms. John Wiley & Sons, New York (1975)
5. Ino, H., Kudo, M., Nakamura, A.: Partitioning of Web Graphs by Community Topology. In: Proceedings of the 14th International Conference on World Wide Web, pp. 661–669 (2005)
6. Kumar, R., Raghavan, P., Rajagopalan, S., Tomkins, A.: Trawling the Web for Emerging Cyber-Communities. *Journal of Computer Networks* 31(11-16), 1481–1493 (1999)
7. Long, B., Wu, X., Zhang, Z.M., Yu, P.S.: Community Learning by Graph Approximation. In: Proceedings of the 7th IEEE International Conference on Data Mining, pp. 232–241 (2007)
8. Reddy, P., Kitsuregawa, M.: Inferring Web Communities through Relaxed Cocitation and Dense Bipartite Graphs. In: Proceedings of the 2nd International Conference on Web Information Systems Engineering, pp. 301–310 (2001)
9. Shi, J., Malik, J.: Normalized Cuts and Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22(8), 888–905 (2000)

Mining Antagonistic Communities from Social Networks

Kuan Zhang, David Lo, and Ee-Peng Lim

School of Information Systems, Singapore Management University
{kuang.zhang.2008,davidlo,eplim}@smu.edu.sg

Abstract. During social interactions in a community, there are often sub-communities that behave in opposite manner. These antagonistic sub-communities could represent groups of people with opposite tastes, factions within a community distrusting one another, etc. Taking as input a set of interactions within a community, we develop a novel pattern mining approach that extracts for a set of antagonistic sub-communities. In particular, based on a set of user specified thresholds, we extract a set of pairs of sub-communities that behave in opposite ways with one another. To prevent a blow up in these set of pairs, we focus on extracting a compact lossless representation based on the concept of closed patterns. To test the scalability of our approach, we built a synthetic data generator and experimented on the scalability of the algorithm when the size of the dataset and mining parameters are varied. Case studies on an Amazon book rating dataset show the efficiency of our approach and the utility of our technique in extracting interesting information on antagonistic sub-communities.

1 Introduction

We form opinions and at times strong convictions on various issues and questions. Based on similarity in opinions and ideals, it is common that sub-groups or communities of users are formed. As members support or uphold a particular view or even conviction, we also observe the dynamics of human social interaction of *antagonistic groups*, i.e., two groups that consistently differ in opinions.

Opposing groups and their nature have been studied in the sociology domain [17,5,4,14,10,6]. Understanding the formation of these groups and wide-spreadness of opposing communities are of research interest. They could potentially signify signs of disunity in the larger community and point to sub-communities that oppose one another. If these issues could be detected early, unwanted tensions between communities could potentially be averted. Identification of antagonistic communities is also the first step to further studies on: e.g., how the antagonistic communities are formed, why they are formed, how does the antagonistic communities grow over time, when do the communities stop being antagonistic, etc.

Aside from enriching studies on dynamics of social interactions, information on groups of people having opposing opinions could potentially be used for: designing better marketing/product survey strategy by deeper understanding

on the nature of each sub-community and potentially an opposing one, better recommendation of friends, or even recommendation of “non-friends”, e.g., those whose reviews one could ignore, etc.

In this study, our goal is to discover antagonistic communities automatically from their history of interactions. We design a novel pattern mining algorithm to directly mine antagonistic communities. We take as input a database of user opinions/views over things, bucketized into high/medium/low or positive/neutral/negative. From this database, we extract every two sets of users that are antagonistic over enough number of common items/issues with a high likelihood. Each mined pattern identifies a group of users that oppose another group over a sufficient number of common issues/items of interest (i.e., enough *support*) with a high likelihood (i.e., enough *confidence*).

Our approach explores the search space of all possible opposing communities and prunes those that appear with not enough frequency/support. An apriori-like anti-monotonicity property is used to prune the search space. Eventually the patterns mined are checked if the confidence is sufficient. If it is, it would then be reported. As a frequent antagonistic pattern would have many sub-patterns that are frequent we only report patterns that are *closed*. A pattern is closed if there exists no super-pattern having the same support and confidence.

To show the scalability of our approach, we developed a synthetic data generator in a similar fashion as the IBM data generator used for mining association rules [2]. The data generator is used to test the scalability of our approach on several dimensions of interest. The result shows that our algorithm is able to run well on various parameter settings. We also investigate a rating dataset from Amazon. Our algorithm is able to run on the real dataset and extract antagonistic communities. A few hundred communities are mined from the dataset.

The contributions of our work are as follows:

1. We propose a new problem of mining antagonistic communities from social network. Mined antagonistic communities could potentially be used to shed better light on social interactions, prevent unwanted tensions in the communities, improve recommendations and marketing strategies, etc.
2. We propose a new algorithm to mine for antagonistic communities that is shown to be scalable.
3. We extract antagonistic communities from real datasets shedding light to the extent of consistent antagonistic behavior in real rating datasets.

The structure of this paper is as follows. Section 2 describes some related work. Section 3 formalizes some concepts and the semantics of antagonistic communities. Section 4 describes our mining algorithm. Experiments and case studies are described in Section 5. We finally conclude and describe future work in Section 6.

2 Related Work

There have been a number of studies on finding communities in social network [3,9,8]. In this study we enrich past studies by discovering not cohesive

communities but ones with opposing sub-communities. We believe these two source of information could give more light to the social interactions among users in Web 2.0.

Antagonistic communities is also related to the concept of homophily. Members of a pair of antagonistic communities, intuitively share more preferences with those in the same community and share less preferences with others from the opposing community. There have been a number of studies on homophily in social networks [16]. In this work, our mined communities not only express similar preferences but also opposing preferences. Homophily and trust are closely related as users with similar preferences are more likely to trust each other [11]. In this sense, our work enriches existing studies on homophily and trust [12,15,13].

In the sociology, economics, and psychology communities, the concept of inter-group antagonism has been studied by various work [17,5,4,14,10,6]. We extend this interesting research question by providing a computation tool to automatically identify opposing communities from a history of their behaviors. We believe our tool could potentially be used to help sociologist in understanding the behaviors of communities from the wealth of available data of user interactions in Web 2.0.

Our algorithm belongs to a family of pattern mining algorithms. There have been a number of pattern mining algorithms including those mining association rules (e.g., [2]), frequent sequences (e.g., [18]), frequent repetitive sequences (e.g., [7]), frequent graphs, etc. The closest to our study is the body of work on association rule mining [2]. Association rule mining also employs the concept of support and confidence like us. However, association rule mining extracts frequent transactions, and relationship between transactions. On the other hand, we extract two sets of opposing users that share many common interests/form opinions/commonly rated items but oppose each other with high likelihood. This problem is inherently different from association rule mining. We show that a similar apriori-like anti-monotonicity property holds but we employ a different algorithm to mine for antagonistic communities. Similar to the work in [18], we do not report all frequent and confident groups rather only the *closed* ones.

3 Antagonistic Group

We formalize past histories of user social interactions in terms of ratings to items which can be objects, views, or even ideas. Hence there is a bipartite graph between users and objects where the arrows are labeled with rating scores. We divide all rating scores to be high, medium, low **rating polarities** depending on the score ranges. For example in Epinions where there is a 5-point scale assigned to an item by a user, we bucketize rating scores of 1 – 2 to be of low rating polarity, 3 to be of medium rating polarity, and 4 – 5 to be of high rating polarity. We formalize our input as a database of ratings, defined in Definition 1. We refer to the size of a rating database DB_R as $|DB_R|$ which is equal to the number of mapping entries in the database.

Definition 1. Consider a set of users U and a set of items I . A database of ratings consists of a set of mappings of item identifiers to a set of pairs, where each pair consists of user identifier and rating score. There are three types of rating scores considered: high (*hi*), medium (*mid*), and low (*lo*). The database of ratings could be formally represented as:

$$DB_R = \{it_{id} \mapsto \{(us_{id}, rtg), \dots\} \mid it_{id} \in I \wedge us_{id} \in U \wedge rtg \in \{hi, mid, lo\} \wedge us_{id} \text{ gives } it_{id} \text{ a rating of } rtg\}$$

Two ratings are said to be common between two users if the ratings are assigned by the two users on the same item. A set of ratings is said to be common among a set of users if these ratings are on a common set of items rated by the set of users.

Definition 2 (Antagonistic Group): Let U_i and U_j be two disjoint sets of users. (U_i, U_j) is called an antagonistic group (or simply, a-group) if at least σ of their common ratings satisfy all the following conditions:

- Users from U_i share the same rating polarity p_i ;
- Users from U_j share the same rating polarity p_j ; and
- p_i and p_j are opposite polarities.

The number of common ratings between two sets of users U_i and U_j is known as their **support count** and is denoted by $count(U_i, U_j)$. The **support** of the two user sets $support(U_i, U_j)$ is defined as $\frac{count(U_i, U_j)}{|I|}$ where I represents the set of all items.

The number of common ratings between U_i and U_j that satisfy the three conditions in the antagonistic group definition (see Definition 2) is called the **antagonistic count**, denoted by $antcount(U_i, U_j)$. Obviously, $antcount(U_i, U_j) \leq count(U_i, U_j)$. The **antagonistic support** of the two user sets $asupport(U_i, U_j)$ is defined as $\frac{antcount(U_i, U_j)}{|I|}$. We also define the **antagonistic confidence** of a a-group (U_i, U_j) to be $aconf(U_i, U_j) = \frac{antcount(U_i, U_j)}{count(U_i, U_j)}$.

Definition 3 (Frequent Antagonistic Group): An antagonistic group (U_i, U_j) is frequent if $support(U_i, U_j) \geq \lambda$ and $asupport(U_i, U_j) \geq \lambda \times \sigma$ where λ is the support threshold ($\in (0, 1)$), and σ is the **antagonistic confidence threshold** ($\in (0, 1)$).

We consider (U_i, U_j) to **subsume** (U'_i, U'_j) if: (a) $U'_i \subset U_i$ and $U'_j \subseteq U_j$; or (b) $U'_i \subseteq U_i$ and $U'_j \subset U_j$. We denote this by $(U'_i, U'_j) \subset (U_i, U_j)$.

Frequent a-groups satisfy the important Apriori property as stated below. Due to space constraint, we move the proof to [1].

Property 1 (Apriori Property of Freq. A-group): Every size $(k-1)$ a-group (U'_i, U'_j) subsumed by a size- k frequent a-group (U_i, U_j) is frequent.

Definition 4 (Valid Antagonistic Group): An a-group (U_i, U_j) is valid if it is frequent and $aconf(U_i, U_j) \geq \sigma$.

Definition 5 (Closed Antagonistic Group): A valid a-group (U_i, U_j) is closed if $\neg\exists(U'_i, U'_j).(U_i, U_j) \subset (U'_i, U'_j)$, $count(U'_i, U'_j) = count(U_i, U_j)$ and $antcount(U'_i, U'_j) = antcount(U_i, U_j)$.

Example 1. Consider the example rating database in Table 1 (left). Suppose $\lambda = 3$ and $\sigma = 0.5$. Both (a, d) and (a, bd) are valid a-groups. However, since $count(a, d) = count(a, bd) = 3$ and $antcount(a, d) = antcount(a, bd) = 2$, (a, d) is not a closed a-group and is subsumed by (a, bd) . Hence, (a, d) is considered as redundant. On the other hand, both (a, b) and (a, bc) are closed a-groups even though both $aconf(a, b)$ and $aconf(a, bc)$ has the same value which is $\frac{2}{3}$. This is so as $count(a, b) \neq count(a, bc)$ and $antcount(a, b) \neq antcount(a, bc)$.

Table 1. Example Rating Database 1 (DB_{EX1}), 2, and 3

| Item | User ratings |
|-------|---------------------------|
| i_1 | a -hi, b -lo, d -lo |
| i_2 | a -hi, b -lo, d -lo |
| i_3 | a -hi, b -hi, d -hi |
| i_4 | a -hi, b -lo, c -lo |
| i_5 | a -hi, b -lo, c -lo |
| i_6 | a -hi, b -hi, c -lo |

| Item | User ratings |
|-------|---------------------------|
| i_1 | a -hi, b -lo, c -lo |
| i_2 | a -hi, b -lo, c -lo |
| i_3 | a -hi, b -lo, c -hi |
| i_4 | d -hi, e -lo, f -lo |
| i_5 | d -hi, e -hi |

| Item | User ratings |
|-------|---------------------------|
| i_1 | a -hi, b -lo, d -lo |
| i_2 | a -hi, b -lo, d -lo |
| i_3 | a -hi, b -hi, d -hi |

Note that $count(U_i, U_j) = count(U'_i, U'_j)$ does not imply that $antcount(U_i, U_j) = antcount(U'_i, U'_j)$ for any $(U_i, U_j) \subset (U'_i, U'_j)$, and vice versa. We can show this using the rating database example in Table 1 (middle). In this example, we have $count(a, b) = count(a, bc) = 3$ but $(antcount(a, b) = 3) > (antcount(a, bc) = 2)$. We also have $antcount(d, e) = antcount(d, ef) = 1$ but $(count(d, e) = 2) > (count(d, ef) = 1)$.

Definition 6 (Antagonistic Group Mining Problem): Given a set of items I rated by a set of users U , the antagonistic group mining problem is to find all closed antagonistic groups with the given support threshold λ and antagonistic confidence threshold σ .

4 A-Group Mining Algorithm

We develop a new algorithm to mine for antagonistic groups from a database of rating history. The database could be viewed as a cleaned representation of people opinions or views or convictions on various items or issues. Our algorithm systematically traverses the search space of possible antagonistic groups using a search space pruning strategy to remove unfruitful search spaces.

The a-group mining algorithm runs for multiple passes. In the initialization pass, we calculate the *count* and *antcount* of all the size-2 a-group candidates and determine which of them are frequent. In the next pass, with the set of frequent a-groups found in the previous pass, we generate new potential frequent a-groups, which are called *candidate set*. We then count the actual *count* and

antcount values for these candidates. At the end of this pass, we determine the frequent candidates, and they are used to generate frequent a-groups for the next pass. After that, we filter the previous frequent a-group set with the newly generated frequent a-group set by removing non-closed a-groups. Then we move on to the next pass. This process continues until no larger a-groups are found. After successful mining of all frequent a-groups, we derive the valid a-groups from them.

Input: λ ; σ ; rating database

Output: valid and closed a-group of all size

```

1  $L_1 =$  frequent user set;
2  $C_2 = \{(\{u_i\}\{u_j\}) \mid i < j, u_i \in L_1, u_j \in L_1\}$ ;
3 for  $k = 2; k \leq |U|$  and  $|L_{k-1}| \neq 0; k++$  do
4   if  $k > 2$  then
5      $C_k = \text{antGrpMining-gen}(L_{k-1})$ ;
6   end
7    $\text{root} \leftarrow \text{buildHashTree}(k, C_k)$ ;
8   foreach item  $t \in \mathcal{D}$  do
9      $C_t = \text{subset}(t, \text{root})$ ;
10    foreach candidate  $c$  in  $C_t$  do
11      update count and antcount of  $c$ ;
12    end
13  end
14   $L_k = \{g_k \in C_k \mid \frac{\text{count}(g_k)}{|I|} \geq \lambda \text{ and } \frac{\text{antcount}(g_k)}{|I|} \geq \lambda \times \sigma\}$ ;
15   $L_{k-1} = \text{prune}(L_{k-1}, L_k)$ ;
16 end
17  $G = \{g \in \bigcup_k L_k \mid \frac{\text{antcount}(g)}{\text{count}(g)} \geq \sigma\}$ ;
18 Output  $G$ ;

```

Algorithm 1. Mining Algorithm – Clagmine(λ, σ, DB_R)

Algorithm 1 shows the a-group mining algorithm known as Clagmine. Two basic data structures are maintained namely L_k the intermediary set of frequent a-groups of size k and C_k a candidate set of size k for valid a-groups checking. The first two lines of the algorithm derives size-2 candidates to get the frequent size-2 a-groups. It forms the base for subsequent processing. A subsequent pass, say pass k , consists of three phases. First, at line 5, the a-groups in L_{k-1} found in $k-1$ pass are used to generate the candidate a-group set C_k , using the `antGrpMining-gen` method in Algorithm 2. Next, the database is scanned and the count and antcount of candidates in C_k is updated (lines 7 to 13). We make use of the hash-tree data structure described in 2 to hold C_k and we then use a subset function to find the candidates overlap with the raters of an item. After we marked all the overlapped candidates, we update the count and antcount of them. Frequent a-groups can be determined by checking count and antcount against the support threshold and $\lambda \times \sigma$ thresholds respectively. Following that, L_{k-1} is filtered with the newly generated a-groups to remove non-closed a-groups (line 15). After all the passes, the valid a-group is determined from the frequent

a-group set (line 17). The following subheadings zoom into the various components of the mining algorithm in more detail.

Input: size- $(k - 1)$ frequent a-group set L_{k-1}
Output: size- k candidate frequent a-group set

```

1 foreach  $p, q \in L_{k-1}$  do
2    $g_k \leftarrow \text{merge}(p, q)$ ;
3   add  $g_k$  to  $C_k$ ;
4   forall  $(k - 1)$ -subsets  $s$  of  $g_k$  do
5     if  $s \cap \in L_{k-1}$  then
6       delete  $g_k$  from  $C_k$ ;
7   end
8 end
9 end
10 return  $C_k$ ;
```

Algorithm 2. antGrpMining-gen(L_{k-1})

Candidate Generation and Pruning. The antGrpMining-gen function described in Algorithm 2 takes L_{k-1} , the set of all frequent size- $(k - 1)$ a-groups as input. It returns a superset of all frequent size- k a-groups. It works as below. First, we merge all the elements in L_{k-1} that share the same sub-community of size- $(k-2)$. Each of them can be merged into a size- k candidate a-group consisting of the common sub-community and the two differing members. We add the candidate a-groups to C_k . Next, in the pruning stage, we delete $g_k \in C_k$ if some $(k - 1)$ subset of g_k is not in L_{k-1} .

The pruning stage’s correctness is guaranteed by Property 1. From the property, if g_k is frequent, all its $(k - 1)$ subsets must be frequent. In other words, if any one $(k - 1)$ subset of an a-group g_k is not frequent, g_k is not frequent too. We thus prune such g_k s. The correctness of antGrpMining-gen function follows from Lemma 1. Due to space constraint, we move the proofs of all lemmas and theorems to [2].

Lemma 1. For $k \geq 3$, given a set of all size- $(k - 1)$ frequent a-group, i.e., L_{k-1} , every size- k frequent a-group, i.e., L_k , is in the candidate set, i.e., C_k , output by Algorithm 2.

An example to illustrate the process of candidate generation via merging and deletion is given below.

Example 2. Let L_3 be $\{(u_1, u_2u_3), (u_5, u_2u_3), (u_1u_4, u_2), (u_1u_5, u_2), (u_4u_5, u_2)\}$. After the merge step, C_4 will be $\{(u_1u_5, u_2u_3), (u_1u_4u_5, u_2)\}$. The deletion step serving as apriori-based pruning, will delete the a-group (u_1u_5, u_2u_3) because the a-group (u_1u_5, u_3) is not in L_3 . We will then left with only $\{(u_1u_4u_5, u_2)\}$ in C_4 .

Subset Function. Candidate a-groups are stored in a hashtree as mentioned in line 7 of Algorithm 1. Each node of the hashtree contains either a hashtable (interior node), or a list of candidates (leaf). Each node is labeled with a user

identifier representing the user associated with this node. The hashtable at interior nodes contains mappings to nodes at the next level, with each hash key being the corresponding user identifier. Every candidate is sorted according to the user identifier, and is then inserted into the hashtree.

The subset function in line 9 of Algorithm 1 finds all the candidate a-groups among raters of item t . The raters of item t is first sorted by their user identifiers. The raters are then traversed one by one. A pointer list is kept to maintain a list of nodes which are visited, which initially has only the root of the hashtree. For a rater u , we traverse through all the nodes in the pointer list, if a child node of the current node is found with label u , the child node is further checked to see whether it is interior or leaf. If it is an interior node, we add it to the pointer list and if it is a leaf, every a-group stored in the leaf is marked as a subset of raters of t . A node is removed from the pointer list if all of its child nodes are in the list (i.e., are visited). The process is repeated through all the raters of item t . At the end, all the candidates which are subset of raters of t will be marked.

Filtering Non-Closed A-Group. The filtering of non-closed a-groups corresponds to line 15 in Algorithm 1. The function works as follows. For each a-group g_k in L_k , we traverse through every a-group g_{k-1} in L_{k-1} . If g_k subsumes g_{k-1} , and the count and antcount of the two groups are equal, g_{k-1} can be filtered. This step ensures all the a-groups in L_{k-1} are closed. By iterating through k , we can have all the non-closed a-group of any size filtered. Note that a closed a-group could potentially subsumes a combinatorial number of sub-groups. Removal of non-closed a-group potentially reduces the number of reported a-groups significantly.

Correctness of the algorithm. The correctness of the algorithm is guaranteed by Theorems 1 & 2 stated below.

Theorem 1. *Mined a-group set G contains all valid and closed a-groups.*

Theorem 2. *Mined a-group set G contains only valid and closed a-groups.*

Scalability Variant: Divide and Conquer Strategy. At times, the main memory required to generate all the candidates could be prohibitive. If there are too many L_2 patterns, storing all of them in the memory would not be feasible. To address this issue, we perform a divide and conquer strategy by partitioning the database, mining for each partition, and merging the result. We first state some new definitions and describe a property.

Definition 7 (User Containment). *Consider a member $m = it_{id} \mapsto \text{PairSet}$ in a database of ratings DB_R . We say that a user u_i is contained in the entry, denoted by $u_i \in m$, iff $\exists (u_i, rtg)$ where $rtg \in \{hi, lo, mid\}$ and (u_i, rtg) is in PairSet . We also say that a user u_i is in an a-group $a = (S_1, S_2)$ iff $(u_i \in S_1 \vee u_i \in S_2)$*

Example 3. To illustrate, consider the first entry etr in the example rating database shown in Table 1(left). The first entry etr contains users a , b and d : $a \in etr$, $b \in etr$, and $d \in etr$.

Definition 8 (Database Partition). Consider a user u_i and a database of ratings DB_R . The partition of the database with respect to user u_i , denoted as $DB_R[u_i]$ is defined as: $\{etr|u_i \in etr \wedge etr \in DB_R\}$

Example 4. To illustrate, projection of the database shown in Table 1(left) with respect to user d is the database shown in Table 1(right).

Using the two definitions above, Lemma 2 describes the divide and merge mining process.

Lemma 2 (Divide and Merge). Consider a database of ratings DB_R , support threshold λ , and confidence threshold σ . Let U_{set} be the set of users in DB_R and Cm be the shorthand of the Clagmine operation in Algorithm 1. The following is guaranteed:

$$Cm(\lambda, \sigma, DB_R) = \bigcup_{u_i \in U_{set}} \{g|u_i \in g \wedge g \in Cm(\frac{\lambda \times |DB_R|}{|DB_R[u_i]|}, \sigma, DB_R[u_i])\}$$

Based on Lemma 2, our algorithm to perform divide and conquer is shown in Algorithm 3. The algorithm partitions the database one item at a time and subsequently calls the original closed antagonistic group mining algorithm defined in Algorithm 1. Theorem 3 guarantees that the mined result is correct and a complete set of a-groups are mined by Algorithm 3.

Theorem 3. Algorithm 3 would return a complete set of closed and valid a-groups and all returned a-group would be closed and valid.

Note that the divide and conquer algorithm reduces memory costs however it could potentially increase the runtime cost since the database would now need to be scanned more number of times. In Section 5, we show the results of running the two algorithms over a number of datasets.

5 Performance and Case Studies

In this section we describe our performance study using various data generated from our synthetic data generator with various parameter values. We then describe a case study from a real book rating dataset.

Performance Study. As a summary, our synthetic data generator accepts as input I (in '000)(the number of items), U (in '000)(the number of users), P (the expected number of users rating an item), N_G (average size of maximal potential large a-group), and N_L (in '000) (number of maximal potential large a-group). We use the following datasets:

| | |
|--------|-----------------------------------|
| DS_1 | $I=100, U=10, P=20, N_G=6, N_L=2$ |
| DS_2 | $I=100, U=50, P=20, N_G=6, N_L=2$ |

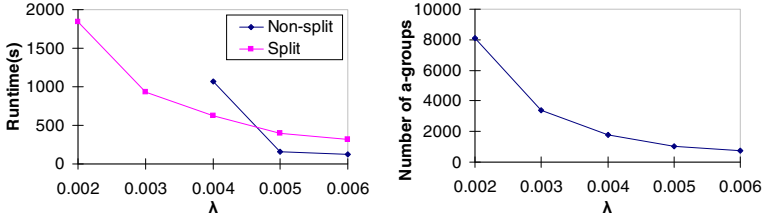


Fig. 1. Runtime and Patterns: DS_1 at various support values

The result for dataset DS_1 when varying the support threshold from 0.002 to 0.006 with $\sigma=0.7$ is shown in Figure 1. The first graph shows the runtime needed to execute the algorithm at various support thresholds. “Non-split” and “Split” correspond to Algorithms 1 & 3 respectively. We only include 3 data points for “Non-split”, as mining at lower thresholds are too long to complete. The second graph shows the numbers of a-groups mined at various support thresholds.

Input: λ ; σ ; rating database

Output: valid and closed a-group of all size

```

1  $U_{Set}$  = Set of all users in  $DB_R$ ;
2  $G = \{\}$ ;
3 foreach  $u_i \in U_{Set}$  do
4    $G = G \cup \{ag | u_i \in ag \wedge ag \in \text{Clagmine}(\frac{\lambda \times |DB_R|}{|DB_R[u_i]|}, \sigma, DB_R[u_i])\}$ ;
5 end
6 Output  $G$ ;

```

Algorithm 3. Clagmine-partitional(λ, σ, DB_R)

The result shows that the time taken grows larger when support threshold is reduced. This growth is accompanied by the growth in the number of a-groups mined. Also, many longer patterns are mined as the support threshold is lowered.

For DS_2 , we consider a larger number of users. The results for various support thresholds with $\sigma=0.7$ are shown in Figure 2. We have also conducted additional performance studies and their results can be found in our technical report [1].

The performance study has shown that the algorithm is able to run well on various settings. The lower the support threshold the more expensive it is to mine. Also, the larger the number of users (or items or expected number of users rating an item – see [1]), the more expensive it is to mine.

Case Study. For the case study, we consider a dataset of book ratings from Amazon. There are a total of 99,255 users ratings 108,142 books in 935,051 reviews. The experiment is run with $\sigma=0.5$. The result is shown in Figure 3.

The number of mined a-groups in the real dataset is small even on much lower support threshold. Interestingly, we find that antagonistic behavior is not so much apparent on item ratings. This might be the case since the objects rated are not “sensitive” items that tend to divide people into opposing groups.

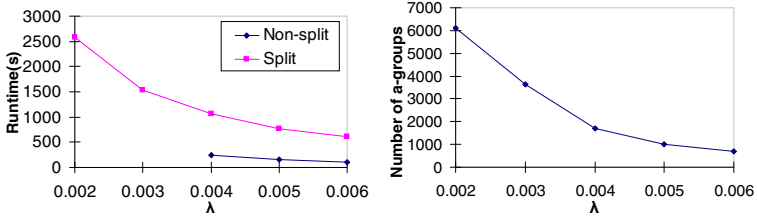


Fig. 2. Runtime and Patterns: DS_2 at various support values

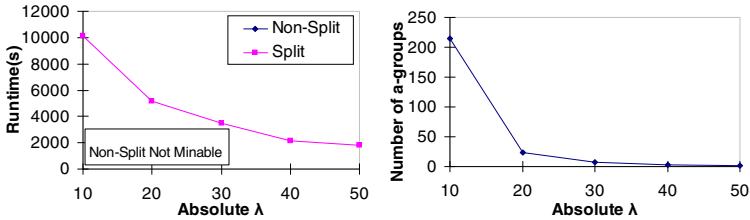


Fig. 3. Runtime and Patterns: Book ratings dataset at various support values

Several interesting a-groups are discovered from the Amazon dataset by running the mining algorithm with absolute support (i.e., $\lambda \times |I|$) of 10 and $\sigma=0.5$. Out of 167 a-groups generated, 147 are of size 2, 18 of them are of size 3, and 2 of them are of size 4. We post-process to retain those with $aconf > 0.7$, and at least one user has (commonly-rated-items/ totally-rated-items) > 0.6 .

Table 2. Interesting Examples from Amazon Book Rating Dataset

| ID | Antagonistic Groups | Commonly Ratings | Ratings by User 1 | Ratings by User 2 | Ratings by User 3 |
|----|-----------------------------------|------------------|-------------------|-------------------|-------------------|
| 1 | ({Johnston}, {Weissgarber}) | 12 | 56 | 13 | - |
| 2 | ({Johnston, Jump}, {Weissgarber}) | 10 | 56 | 61 | 13 |
| 3 | ({Johnston, Hill}, {Weissgarber}) | 10 | 56 | 106 | 13 |
| 4 | ({Leeper}, {Weissgarber}) | 10 | 137 | 13 | - |
| 5 | ({Kern}, {Sklarski}) | 14 | 452 | 22 | - |

After post-processing, we note 5 of the most interesting a-groups. We select those having highest $aconf$ and average (common-item/total item) over all constituent users. They are represented in table 2. We select the first a-group and observe the following:

- *High antagonistic level:* We observe that the two users in the first a-group rated with a high level of antagonism. Among Jason Johnston’s 56 rated books, 12 have ratings opposite to the ratings by Luke Weissgarber. Similarly for Weissgarber, 12 of all his 13 rated books have ratings opposite to those by Johnston, which means more than 92% of Weissgarber’s ratings are opposite to Johnston’s. It is a significantly high figure.

- *Antagonistically rated books*: We found that for books with opposite ratings from Weissgarber and Johnston are some novels with similar story background. These books are clearly liked by Johnston but not by Weissgarber.
- *Antagonistically behaved users*: It is interesting that Weissgarber appears in four a-groups. His ratings are opposite to other 4 users for at least 10 books.

6 Conclusion and Future Work

In this study, we proposed a new pattern mining algorithm to mine for antagonistic communities. Our algorithm traverses the search space of possible antagonistic groups and uses several pruning strategies to remove search space containing no antagonistic pattern. We also propose a new variant of the algorithm that adopts a divide and conquer strategy in mining when the first algorithm becomes prohibitively expensive to run. A performance study is conducted on various synthetic datasets to show the scalability of our approach on various parameter values. We also mine from an Amazon book rating dataset. The result shows that antagonistic communities exists but are not particularly many or large in the Amazon dataset. In the future, we plan to investigate more “sensitive” datasets and further speed up the mining algorithm.

Acknowledgement. This work is supported by National Research Foundation of Singapore under project NRF2008IDM-IDM004-036. We would like to thank Paolo Massa for sharing his Epinions dataset. We would also like to thank Bing Liu for sharing the book ratings dataset.

References

1. <http://www.mysmu.edu/phdis2008/kuan.zhang.2008/agroup.pdf> (2009)
2. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proceedings of International Conference on Very Large Data Bases (1994)
3. Cai, D., Shao, Z., He, X., Yan, X., Han, J.: Community mining from multi-relational networks. In: Jorge, A.M., Torgo, L., Brazdil, P.B., Camacho, R., Gama, J. (eds.) PKDD 2005. LNCS (LNAI), vol. 3721, pp. 445–452. Springer, Heidelberg (2005)
4. Dasgupta, I.: ‘living’ wage, class conflict and ethnic strife. *Journal of Economic Behavior & Organization* (2009) (in press)
5. Dasgupta, I., Kanbur, R.: Community and class antagonism. *Journal of Public Economics* 91(9), 1816–1842 (2007)
6. Denrell, J.: Why most people disapprove of me: Experience sampling in impression formation. *Psychological Review* 112(4), 951–978 (2005)
7. Ding, B., Lo, D., Han, J., Khoo, S.-C.: Efficient mining of closed repetitive gapped subsequences from a sequence database. In: ICDE (2009)
8. Flake, G., Lawrence, S., Giles, C., Coetzee, F.: Self-organization and identification of web communities. *Computer* 35(3), 66–71 (2002)
9. Gibson, D., Kleinberg, J., Raghavan, P.: Inferring web communities from link topology. In: *Hypertext* (1998)
10. Giles, M., Evans, A.: The power approach to intergroup hostility. *The Journal of Conflict Resolution* 30(3), 469–486 (1986)

11. Golbeck, J.: Trust and nuanced profile similarity in online social networks. *ACM TWeb* 3(4), 1–33 (2009)
12. Guha, R., Kumar, R., Raghavan, P., Tomkins, A.: Propagation of trust and distrust. In: *WWW* (2004)
13. Kunegis, J., Lommatzsch, A., Bauckhage, C.: The slashdot zoo: Mining a social network with negative edges. In: *WWW* (2009)
14. Labovitz, S., Hagedorn, R.: A structural-behavioral theory of intergroup antagonism. *Social Forces* 53(3), 444–448 (1975)
15. Liu, H., Lim, E.-P., Lauw, H., Le, M.-T., Sun, A., Srivastava, J., Kim, Y.: Predicting trusts among users of online communities: an epinions case study. In: *EC* (2008)
16. McPerson, M., Smith-Lovin, L., Cook, J.: Birds of a feather: Homophily in social networks. *Annual Review of Sociology* 27(3), 415–444 (2001)
17. Tolsma, J., Graaf, N.D., Quillian, L.: Does intergenerational social mobility affect antagonistic attitudes toward ethnic minorities? *British Journal of Sociology* 60(2), 257–277 (2009)
18. Wang, J., Han, J.: BIDE: Efficient mining of frequent closed sequences. In: *ICDE* (2004)

As Time Goes by: Discovering Eras in Evolving Social Networks

Michele Berlingerio¹, Michele Coscia^{1,2}, Fosca Giannotti^{1,3},
Anna Monreale^{1,2}, and Dino Pedreschi^{2,3}

¹ ISTI - CNR, Area della Ricerca di Pisa, Italy
{name.surname}@isti.cnr.it

² Computer Science Dep., University of Pisa, Italy
{coscia, annam, pedre}@di.unipi.it

³ Center for Complex Networks Research - Northeastern University, Boston, MA

Abstract. Within the large body of research in complex network analysis, an important topic is the temporal evolution of networks. Existing approaches aim at analyzing the evolution on the global and the local scale, extracting properties of either the entire network or local patterns. In this paper, we focus instead on detecting clusters of temporal snapshots of a network, to be interpreted as *eras* of evolution. To this aim, we introduce a novel hierarchical clustering methodology, based on a dissimilarity measure (derived from the Jaccard coefficient) between two temporal snapshots of the network. We devise a framework to discover and browse the eras, either in top-down or a bottom-up fashion, supporting the exploration of the evolution at any level of temporal resolution. We show how our approach applies to real networks, by detecting eras in an evolving co-authorship graph extracted from a bibliographic dataset; we illustrate how the discovered temporal clustering highlights the crucial moments when the network had profound changes in its structure. Our approach is finally boosted by introducing a meaningful labeling of the obtained clusters, such as the characterizing topics of each discovered era, thus adding a semantic dimension to our analysis.

1 Introduction

In the last years, much attention has been devoted to topics related to Social Network Analysis. One research direction that has attracted researchers in various fields, including Data Mining, is analyzing networks that evolve over time. Time in networks can play a double role: the entities involved may perform actions, and the connectivity structure may change. In this last setting, several phenomena can be analyzed, and much effort has been devoted in this direction so far [13, 12, 10, 4, 3].

In this paper, we focus on detecting clusters of temporal snapshots of an evolving network, to be interpreted as *eras* of evolution of the network. By analyzing the similarity of the structures of consecutive temporal snapshots of the same network, we observe that, despite a global increase of similarity, it is possible to detect periods of sudden change of behavior, where people act in a

counter-trend fashion, making this similarity either decrease, or suddenly start increasing very fast, much more than the average.

In real-life social networks, in fact, a common phenomenon is that people tend to both keep being part of the networks, and keep alive all the connections created in the past. On the other hand, new users join the networks as time goes by, and people set new relationships while keeping the previous ones [13]. However, while the number of newly created relationships tends to be almost constant at every snapshot, the number of previous relationships kept alive grows, thus the global effect of newly added nodes or edges loses importance over time [3]. Because of this, the similarity of the structure of two consecutive temporal snapshots increases almost at each step. The increase, however, is not locally uniform: for example, there can be one snapshot where suddenly people change behavior and start giving more importance to creating new connections, in other words, despite a global moderate *conservative* trend, people can suddenly alternate highly more conservative periods, or a highly more *innovative* behavior.

The aim in this paper is to catch these sudden changes by detecting the snapshots in which they start. Intuitively, these are starting points of new eras. In a globally changing world, we then want to detect eras characterized not by changes in structure (that we not only allow within the same cluster of snapshots, but we also expect), but rather characterized by a change in counter-trend with the previous era: either the previous results more conservative, or it is actually more innovative than the era under investigation.

To this aim, we introduce a novel hierarchical clustering methodology, based on a dissimilarity measure derived from the Jaccard coefficient computed between two temporal snapshots of the network. We devise a framework to discover and browse the era hierarchy either in top-down or a bottom-up fashion, from the lowest level of the single temporal snapshots, to the highest level of the complete period of existence of the network.

In order to do so, we find a measure of the dissimilarity of two temporal snapshots, and we show how to use it as a basis for detecting starting points of new eras. In our experimental section, we show how this measure is not affected by classical phenomena detectable in real-life networks, such the presence of highly connected nodes. We apply this methodology to real data, extracted by the well known bibliographic database DBLP. We build a co-authorship network from it, and analyze two different aspects of the network, namely the nodes (authors), and the edges (collaborations).

Our contribution can be then summarized as follows: we define a dissimilarity measure between two temporal snapshots of an evolving network; we describe the clustering process driven by this measure; we show how to apply labels to the obtained clusters, in order to add a semantic dimension to our analysis; we present the results obtained on the DBLP network.

2 Related Work

Interesting properties have been recently studied and discovered on evolving networks, such as shrinking diameters, and densification power law. Specifically, the

authors in [13] discover that in most of these networks the number of edges grows superlinearly in the number of nodes over time and that the average distance between nodes often shrinks over time. In literature, many models capturing these properties have been proposed; an interesting survey is presented in [7].

Three more recent works are [12],[14],[16]. In the first, Leskovec et al. present a detailed study of network evolution. They analyze four networks with temporal information about node and edge arrivals and use a methodology based on the maximum-likelihood principle to show that edge locality plays a critical role in evolution of networks. In the second, McGlohon et al. study the evolution of connected components in a network. In [16], the authors propose a novel model which captures the co-evolution of social and affiliation networks.

The notion of *temporal graph* has been studied in [10]. The main aim of this paper is to study how do the basic properties of graphs change over time. A similar setting is used in [11] where Kossinets et al. study the temporal dynamics of communications. They define a temporal notion of “distance” in the underlying social network measuring the minimum time required for information to spread between two nodes. Other works related to the temporal analysis in a network propose the study of aspects of the temporal evolution of the Web [6,8,9,5].

For our temporal analysis we perform hierarchical clustering: an interesting survey on existing clustering approaches can be found in [2].

3 Problem Definition

We are given an evolving network G , whose evolution is described by a temporally ordered sequence of temporal snapshots $T = \{t_1, t_2, \dots, t_n\}$, where t_i represents the i -th snapshot. T can be either computed on the sets of nodes, i.e. each snapshot t_i is represented by the set of nodes involved, or on the sets of edges, i.e. each snapshot is represented by the set of edges in it.

Based on a dissimilarity measure $d : (t_i, t_{i+1}) \rightarrow]-\infty, +\infty[$, we want to find a hierarchical clustering on T , returning clusters $C_i = \{t_j, \dots, t_{j+k}\}$, with $j \geq 1$, and $0 \leq k \leq n - j$.

Each cluster represents then an era of evolution. Due to the global evolution of real-life networks, we do allow alterations of the structure of the network among snapshots of the same cluster, as long as they follow a constant trend. As soon as this trend changes, we want to set the corresponding snapshot as the first of a new era. The stronger is the change, the higher should be the dissimilarity of that snapshot with the previous one. The definition of the dissimilarity function should reflect this intuition.

We then want to assign to each cluster C_i a label describing the represented era. This step adds a semantic dimension to our framework.

4 Framework for Temporal Analysis

In this section we describe the details of the framework that we propose.

Dissimilarity. In order to perform clustering, the first step is to define a measure of dissimilarity among elements that we want to cluster. In our setting, a simple way to do this is to use the Jaccard coefficient. In a generic network, we can easily apply this coefficient on either two sets of nodes or two sets of edges, where each set corresponds to a temporal snapshot of the network. As we show later in the paper, clustering temporal snapshots actually corresponds to perform a segmentation of the sequence of the snapshots, thus we are interested only in computing this Jaccard coefficient for every pair of consecutive snapshots.

Real-life networks are well known to follow global evolutionary trends, then if we plot the Jaccard coefficient for each snapshot, we shall see a global increase (or decrease), characterized by an almost constant slope of the Jaccard coefficient plot, alternated by (moderate to high) changes of this slope (we prove this intuition in Section 5). An immediate way, to define starting point of new eras is to detect the snapshots corresponding to these changes. This could be done by computing the second derivative of the Jaccard and finding values different from zero. However, the Jaccard is continuous but not derivable exactly in the points we need. To overcome this problem, we consider an approximation of the second derivative defined as follows. We take triples of consecutive years, and we trace the segment that has as endpoints the Jaccard computed for the first and the third snapshot. If the middle point is distant from the segment, the corresponding snapshot should be considered as the start of a new era. The Euclidean distance between the middle point and the segment also gives as a quantitative analysis of how important is the change: the higher the distance, the higher the change.

Definition 1. *Given a temporal snapshot t_j , we define the following measure:*

$$s_N(t_j) = \frac{|c_N(t_j) - (m \times j) - q|}{\sqrt{1 + (m^2)}}$$

where $m = \frac{c_N(t_{j-1}) - c_N(t_{j+1})}{t_{j-1} - t_{j+1}}$, $q = -(j + 1) \times m + c_N(t_{j+1})$, and $c_N(t_k) = \frac{|N_{k-1} \cap N_k|}{|N_{k-1} \cup N_k|}$ is the Jaccard coefficient computed on the node sets.

Defining s_E , which is the counterpart computed on the set of edges, requires to consider c_E instead of c_N , where c_E is the Jaccard computed on the edges.

However, this measure takes, formally, only one snapshot as input, thus it is not intuitive to use as basis for a clustering methodology. In order to tackle this problem, we define a dissimilarity between any two snapshots as follows.

Definition 2. *Given an ordered sequence t_1, t_2, \dots, t_n of temporal snapshots of a network G , the dissimilarity between any two snapshots t_i and t_j computed on their node sets is defined as*

$$d_N(t_i, t_j) = \begin{cases} s_N(t_{\max(i,j)}) & \text{if } |i - j| = 1 \\ \text{undefined} & \text{otherwise} \end{cases}$$

Defining the similarity on the edges d_E requires to consider s_E instead of s_N .

Moreover, this dissimilarity measure allows for a straightforward hierarchical clustering: an higher dissimilarity corresponds to a stronger separation between

two consecutive eras. This means that by setting fixed threshold, we can draw a dendrogram of the hierarchical clustering, driven by this dissimilarity as a criterion for merging two consecutive clusters in a bigger one.

Merging Clusters. In hierarchical clustering, when merging clusters, there are various main approaches followed in the literature to define the distance between two clusters: the maximum distance between any two points belonging to the two clusters (complete linkage), the minimum (single linkage), the average (average linkage), the sum of all the intra-cluster variance, and so on.

Given two clusters $C_i = \{t_1, t_2, \dots, t_k\}$ and $C_j = \{t_{k+1}, t_{k+2}, \dots, t_{k+p}\}$, in order to define the distance between two clusters, we shall first compute all the distances between every pair (t_i, t_j) , with $1 \leq i \leq k$ and $k+1 \leq j \leq k+p$.

However, according to Definition 2, only one pair of snapshots has a dissimilarity defined: (t_k, t_{k+1}) . At this point, we use this dissimilarity as inter-cluster distance. As one can immediately see, taking the only available dissimilarity value as distance between clusters actually corresponds not only to both the complete linkage and the single linkage, but also to the average. In our case, thus, the three of them are identical.

Assigning Labels to Clusters. Once we have computed the cluster hierarchy, we want to add a description of every era. In order to do so, we label each cluster with the node (or edge, or a property of it), that maximizes the ratio between its relative frequency in that cluster, and its relative frequency in the entire network. This strategy may produce several values equal to 1 (identical numerators and denominators). In order to discern among these cases, we weight the numerator by multiplying it again for the relative frequency in the cluster under analysis. In this way, we give more importance to 1s deriving from nodes (or edges) with a higher number of occurrences in the cluster.

With this frequency based strategy, we are assigning labels that truly characterize each cluster, as each label is particularly relevant in that cluster, but less relevant for the entire network.

One important caveat in this methodology is what to take as label for the edges. In fact, while for the nodes it is straightforward to consider the identity of the corresponding entity of the network as candidate label, the edge express a relationship with a semantic meaning, thus each network requires some effort in defining exactly which label could be applied to a cluster computed on edges. For example, in a co-authorship network, where two authors are connected by the papers that they have written together, a possible strategy is to take every keyword in the title of the papers as possible label. In the experimental section we show exactly this kind of labeling.

5 Experiments

From the DBLP¹ database, we created a co-authorship graph for the years 1979-2006, where each node represents an author and each edge a paper written

¹ <http://dblp.uni-trier.de>

together by the two connected authors. We then considered each year as temporal snapshot of DBLP, generating then 28 snapshots. In each snapshot we put only the nodes or the edges appearing in the corresponding year, thus not following a cumulative approach.

Jaccard Coefficient. Figure 1(a) shows the Jaccard on both the nodes and the edges. These plots confirm the general increasing behavior of the Jaccard during time, both on nodes and on edges, broken by short series of years in which people acted in counter-trend. Two questions might be raised on the effectiveness of following a Jaccard-based approach for clustering eras: what would the Jaccard computed on non consecutive snapshot tell us? Is the Jaccard noise free?

We start answering the first question by plotting the coefficient computed for every pair of snapshots: figures 2(a) and 2(b) show that the Jaccard decreases when computed between snapshots more distant in time. This observation justifies a dissimilarity measure that takes into account only consecutive snapshots, as two distant snapshots are not likely to be similar, thus they will belong to different clusters. Temporal segmentation is then a good model for clustering real-life evolving networks. Please note that while in this paper we only show the results obtained on one dataset, these considerations are well accepted in the literature regarding evolving networks [3,13].

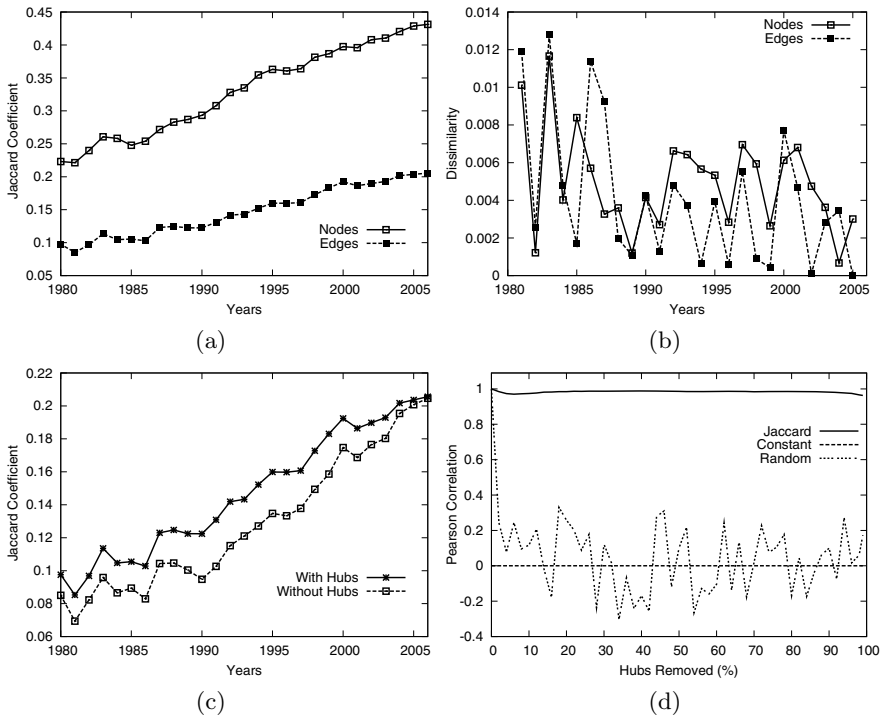


Fig. 1. (a) Evolution of the Jaccard Coefficient in DBLP; (b) Dissimilarity in DBLP; (c) Jaccard Coefficient with and without hubs; (d) Pearson Correlation

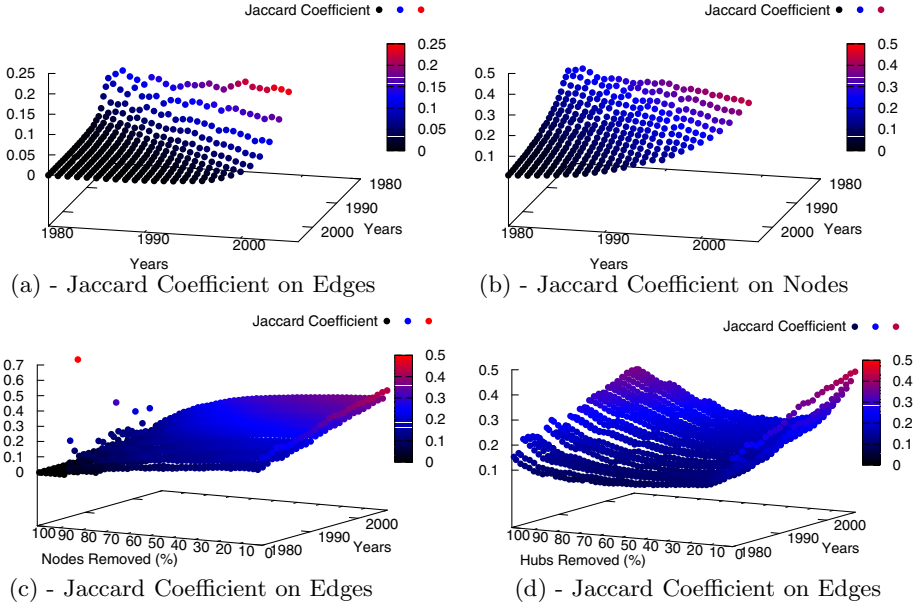


Fig. 2. Jaccard Coefficient in DBLP

Note that by answering the second question we also say something about possible noise on the dissimilarity measure. As the two are closely related, proving that the Jaccard is noise free also proves the same for the dissimilarity. However, using the Jaccard to this aim results much more intuitive. To answer the second question, we analyzed what happens to the Jaccard coefficient when removing a possible cause of noise in the structure: from the entire network, we removed the 1% of top connected hubs, i.e. highly connected nodes, and recomputed the Jaccard. Figure 2(c) shows that, despite a general decrease of the Jaccard values, the global increasing trend, as well as the local sudden changes, are almost unchanged. In order to further support this observation, we plotted the Jaccard coefficient calculated on different versions of the network snapshots after an increasing percentage of hubs removed. Figure 2(d) shows an interesting result: while the global Jaccard dramatically decreases after removing about 10% of the top hubs, always keeping the global evolutionary behavior, it increases again after removing 70-80% of the hubs. This behavior can be explained by considering the intrinsic inter-components function of hub nodes: after removing the majority of the hubs, we have only the small connected components left in the network, and each of them keeps a high Jaccard during its evolution, acting as a separated network. This does not happen when removing an increasing percentage of random nodes (Figure 2(c)), which makes the Jaccard index globally decrease. As last proof of the strength of the Jaccard as good similarity measure, we report in Figure 2(d) the values of the Pearson correlation ρ between the series of Jaccard coefficient computed on the original dataset, and the ones

computed after removing the hubs. The figure shows that removing an arbitrary percentage of hubs does not affect the correlation with the Jaccard computed on the original network. In the figure we have also reported the correlation with a constant series and a random one.

Dissimilarity. The second step of the framework required to compute our dissimilarity on the basis of the Jaccard coefficient computed on the network. Figure 2(b) reports the values of the dissimilarity for both the edge and the node cases. As one can see, the quantitative analysis of our dissimilarity measure is effective: its values have a considerable standard deviation. That is, we can effectively perform hierarchical clustering finding a well distributed strength of starting snapshots for new eras of evolution.

Another observation that can be done is that while the Jaccard values computed on nodes or edges look similar, stronger differences can be found in the dissimilarity plots. That is, we expect the eras computed on nodes to slightly differ from the ones computed on the edges.

As last note, we see that in the first years under investigation there are a few very high peaks of dissimilarity. This is mainly due to the data acquired by DBLP before year 1990. In the first decades, in fact, the set of publications recorded in the database was more restrictive, and sometimes limited only to publications in German. This created a kind of noise in the cluster, and it is the reason why in the final dendrograms we see the years up to 1985 to be among the last ones to be added to the global cluster (Figure 3(a)-(b)), and why also we see labels in German in the final labeling (Figure 3(c)).

Merging Clusters. We then started to compute the clusters on the sequences of temporal snapshots. We started from clusters containing only one year and then, driven by the dissimilarity values computed in the previous step, we merged similar consecutive clusters, with increasing values of dissimilarity. Figure 3(a)-(b) show the dendrograms of clusters obtained using the sets of edges (on the left) or the sets of nodes (on the right). Please note that the dissimilarity is reported in percentage of the highest value found.

As one can see, there are actually a few differences between the two dendrograms, even if at a higher level the two look similar. A deeper view would show a more uniform distribution of the joins between clusters. We recall that in the dendrogram on the left we report the analysis performed on the edges, i.e. the collaborations, while on the right we show the clusters obtained on the nodes, i.e. the authors. This can be read saying that, while there are different, uniformly distributed, ways of changing evolutionary behavior for the collaboration, there are less thresholds for changing eras while looking at the nodes: i.e., the strength of changes of eras can be further clustered in a few similar thresholds.

Assigning Labels. As last step in our framework, we computed the labels for each cluster obtained. Figure 3 reports, in (c) and (d), 5 labels for each cluster with at least two years (due to space constraints, we do not report labels for single years). Please note that the rows in these tables are sorted by order of

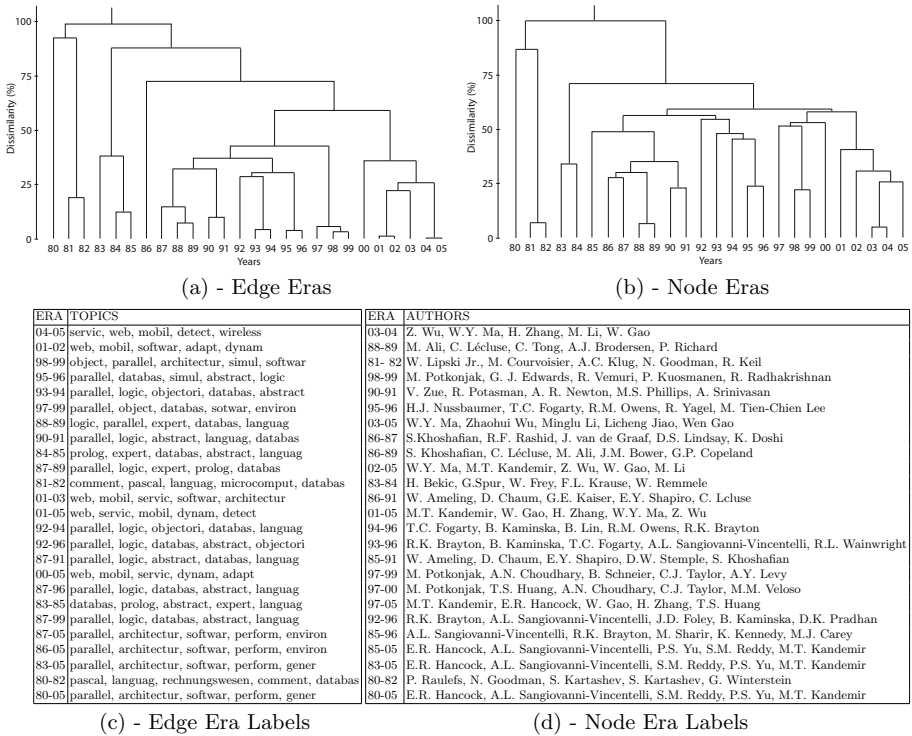


Fig. 3. DBLP Eras discovered on Edges or Nodes

cluster formation. Please also note that the keywords of the publications were pre-processed using the Porter’s stemming algorithm [15].

We recall that for each cluster C_i we assign the set of the k labels maximizing the ratio between their frequency in C_i and their frequency in the entire network. Due to our strategy, it is then not surprising that, if we look at the node cluster labels, in all the clusters except the complete network we do not find the most active authors, but the ones that mostly published only on each specific cluster. That is, we can find as labels authors with a not so strong publication record, but whose publication record was extremely stronger in a specific cluster w.r.t the entire network. This behavior is less evident in the edge era labels, where topics such as “parallel” can be found in different clusters.

In this table, however, another consideration can be done. If we compare the era labels with the dissimilarity plot, we can see which are the labels that correspond to more conservative or more dynamic eras. If we exclude the first noisy years, the highest peak in the dissimilarity plot is around year 2000. This, in fact, corresponds to the creation of the cluster starting at year 2001. We can say that from year 2000, a short era of very conservative collaborations started. One of the most representing label for the collaborations in these cluster is “web”. One can say that this topic is highly representative for highly conservative collaborations, i.e., collaborations that take place among the same (large) group of people.

6 Conclusions and Future Work

We have proposed a framework for the discovery of eras in an evolving social network. Based on a dissimilarity measure derived from the Jaccard coefficient, we have presented a methodology to perform hierarchical clustering of the temporal snapshots of a network. We have applied our methodology to real-life data, showing the effectiveness of our approach.

Future research directions include the application to several different evolving networks, possibly showing different temporal behaviors, different definitions of dissimilarity, and the introduction of a label-driven temporal clustering strategy.

References

1. Thirteen ways to look at the correlation coefficient. *The American Statistician* 42(1), 59–66 (1988)
2. Berkhin, P.: Survey of clustering data mining techniques. Technical report, Accrue Software, San Jose, CA (2002)
3. Berlingerio, M., Bonchi, F., Bringmann, B., Gionis, A.: Mining graph evolution rules. In: *ECML/PKDD*, vol. (1), pp. 115–130 (2009)
4. Berlingerio, M., Coscia, M., Giannotti, F.: Mining the temporal dimension of the information propagation. In: Adams, N.M., Robardet, C., Siebes, A., Boulicaut, J.-F. (eds.) *IDA 2009. LNCS*, vol. 5772, pp. 237–248. Springer, Heidelberg (2009)
5. Bordino, I., Boldi, P., Donato, D., Santini, M., Vigna, S.: Temporal evolution of the uk web. In: *ICDM Workshops*, pp. 909–918 (2008)
6. Brewington, B.E., Cybenko, G.: Keeping up with the changing web. *IEEE Computer* 33(5) (2000)
7. Chakrabarti, D., Faloutsos, C.: Graph mining: Laws, generators, and algorithms. *ACM Comput. Surv.* 38(1) (2006)
8. Cho, J., Garcia-Molina, H.: Estimating frequency of change. *ACM Trans. Internet Techn.* 3(3), 256–290 (2003)
9. Gomes, D., Silva, M.J.: Modelling information persistence on the web. In: *ICWE 2006: Proceedings of the 6th international conference on Web engineering*, pp. 193–200 (2006)
10. Kempe, D., Kleinberg, J.M., Kumar, A.: Connectivity and inference problems for temporal networks. In: *STOC*, pp. 504–513 (2000)
11. Kossinets, G., Kleinberg, J.M., Watts, D.J.: The structure of information pathways in a social communication network. In: *KDD*, pp. 435–443 (2008)
12. Leskovec, J., Backstrom, L., Kumar, R., Tomkins, A.: Microscopic evolution of social networks. In: *KDD*, pp. 462–470 (2008)
13. Leskovec, J., Kleinberg, J.M., Faloutsos, C.: Graphs over time: densification laws, shrinking diameters and possible explanations. In: *KDD*, pp. 177–187 (2005)
14. McGlohon, M., Akoglu, L., Faloutsos, C.: Weighted graphs and disconnected components: patterns and a generator. In: *KDD*, pp. 524–532 (2008)
15. Robertson, S.E., van Rijsbergen, C.J., Porter, M.F.: Probabilistic models of indexing and searching. In: *SIGIR*, pp. 35–56 (1980)
16. Zheleva, E., Sharara, H., Getoor, L.: Co-evolution of social and affiliation networks. In: *KDD*, pp. 1007–1016 (2009)

Online Sampling of High Centrality Individuals in Social Networks

Arun S. Maiya and Tanya Y. Berger-Wolf

Department of Computer Science
University of Illinois at Chicago
851 S. Morgan, Chicago, IL 60607, USA
{amaiya2,tanyabw}@uic.edu

Abstract. In this work, we investigate the use of online or “crawling” algorithms to sample large social networks in order to determine the most influential or important individuals within the network (by varying definitions of network centrality). We describe a novel sampling technique based on concepts from expander graphs. We empirically evaluate this method in addition to other online sampling strategies on several real-world social networks. We find that, by sampling nodes to maximize the expansion of the sample, we are able to approximate the set of most influential individuals across *multiple* measures of centrality.

1 Introduction and Motivation

Given a large or even massive social network, how can one efficiently identify the most important or influential individuals *without* complete access to the entire network at once? This scenario arises when the network is too large for conventional analysis to be computationally feasible. It may also arise in the context of mining data from a network whose complete structure is hidden from public view (such as a friendship network in Web-based social media) or has a highly distributed structure (such as the network of blogs or the Web itself). The efficient identification of influential individuals (by varying definitions of influence) in large social networks has many applications, from the prevention of computer worms to viral marketing. In this work, we investigate the use of online sampling in identifying such critical individuals.

Online Sampling of Centrality. One of the key tasks in social network analysis is determining the relative importance of individuals based on their positions in the structure of the network [1, 2]. This is referred to as the *centrality* of individuals, and there are many notions of what it means to be *central* to a network [2]. In a sampling approach to centrality approximation, a subset of the individuals in the network is sampled, and an induced subgraph consisting only of these individuals and the links among them is produced. The centrality computation, then, is performed on this induced subgraph *instead* of the entire network, with the centrality scores of the sample being used as approximations of the true centrality of sampled individuals. It is clear that, for this approach to

be useful, two criteria must be met. First, the sampling method must produce an induced subgraph that is representative of centrality in the original network (e.g. the centrality ranking in the sample should be consistent with that of the original network). Ideally, the sampling method will quickly find high centrality nodes, and these nodes will also be ranked highly when computing centrality on the sample. A second criterion is that the sampling method must be an *online* algorithm, since the network may be too large for its global structure to be accessed in its entirety or the global view of the entire network may be limited. By *online*, we mean that the sampling is produced in an iterative, sequential manner, without a priori access to the entire input (*i.e.*, sampling via crawling); at each iteration the new addition to the sample is based on the properties of the nodes crawled thus far. The next node selected for inclusion in the sample is always chosen from the set of nodes connected directly to the current sample. This is also referred to as *snowball sampling* or *neighborhood sampling*. In this work, we systematically investigate the task of *online sampling* of centrality in large social networks. We show that, by sampling nodes to maximize the *expansion* of the sample, the set of most influential individuals can be approximated across *three* different centrality measures: betweenness, closeness, and eigenvector centrality. Remarkably, these sets of top ranked individuals can be approximated reasonably well with sample sizes as small as 1%.

2 Background and Related Work

Centrality in Networks. The idea that the structural position of an individual in a social network may be correlated with the relative influence or importance of that individual was first postulated by Bavelas in the 1940s in the context of organizational communication [3,1]. Since then, many notions of what it means to be important or *central* in a network have been proposed and applied with great success in a variety of different contexts (e.g. [4,5]). In this paper we focus on three widely-used measures: *betweenness centrality*, *closeness centrality*, and *eigenvector centrality*. The *betweenness* of a node is defined as the fraction of the overall shortest paths passing through a particular node [6,7,1]. The *closeness* of an individual in a network is a function of the inverse of the average distance to every other individual [1]. Finally, *eigenvector centrality* is a measure of prestige or popularity proposed by Bonacich [8]. When representing the entire network (or graph) as an adjacency matrix \mathbf{A} , the eigenvector centrality of individuals in the network is the eigenvector \mathbf{x} of matrix \mathbf{A} corresponding to the largest eigenvalue λ [8]. The PageRank measure [5] used by Google to rank search results is, in fact, simply a variant of eigenvector centrality and has been shown to be highly effective in approximating the prestige and authority of Web pages.

Related Work. Web crawlers, programs that traverse the Web and index pages for search engines in an automated manner, can be viewed as *online* sampling algorithms. Although the goal of Web crawlers is to collect and store the Web link graph for offline processing, it is highly advantageous for crawling algorithms to

seek out high PageRank pages early in the crawl [9]. As a result, there have been several studies evaluating these algorithms in their ability to sample PageRank (e.g. [9,10,11,12]). In Section 4.2, we evaluate the performance of several of these algorithms in the context of undirected social networks and in their ability to sample alternative notions of network centrality. Finally, also related to this work are the existing studies on representative subgraph sampling such as [13,14,15].

3 Proposed Method

We employ an *online* sampling algorithm to sample individuals in the social network. Let $G = (V, E)$ be a *network* or *graph* where V is set of vertices (or nodes) and $E \subseteq V \times V$ is a set of edges (or links between the nodes). We begin by selecting a single individual $v \in V$ uniformly at random and specifying a desired sample size k where $k \ll V$. The sample $S \subset V$, then, is initialized to $\{v\}$. Each subsequent individual selected for inclusion in the sample is chosen from the current neighborhood $N(S)$, where $N(S) = \{w \in V - S : \exists v \in S \text{ s.t. } (v, w) \in E\}$. Next, upon constructing the sample S , a centrality measure is computed on the induced subgraph of the sample, $G(S)$. The critical question is how to select individuals from $N(S)$ such that:

1. The ranking of individuals by centrality scores in $G(S)$ corresponds to the ranking of individuals in G (most importantly for the highest centrality individuals, as they are generally of greater interest).
2. The highest centrality individuals are quickly included in the sample.

Moreover, as an *online* algorithm, these selection decisions must be made solely on the basis of local information (i.e. information obtained only from those individuals already crawled). In the following sections, we describe several different approaches to online sampling.

Expansion Sampling. We now describe a novel sampling technique based on the concept of expansion in graphs [16]. We refer to this method as *expansion sampling* (XS). The *expansion* of a sample S is defined as $\frac{|N(S)|}{|S|}$. In this approach, we seek out the sample S of size k with the maximal expansion: $\operatorname{argmax}_{S: |S|=k} \frac{|N(S)|}{|S|}$. We propose a simple algorithm that greedily selects nodes in order to maximize the expansion of the current sample. That is, the next node v selected for inclusion in the sample is chosen based on the expression: $\operatorname{argmax}_{v \in N(S)} |N(\{v\}) - (N(S) \cup S)|$.

Web Crawlers. As mentioned previously, the process of web crawling is essentially an online sampling process. There are several crawlers explicitly designed to include high PageRank nodes into the sample more quickly. One such approach is referred to as *Backlink Count* (BLC) in which the next node selected for inclusion into the sample is the node with the most links to nodes already in the sample [11]. A second approach is referred as the *OPIC* algorithm [10]. In this approach, all individuals are assigned a default “cash” value. When a node

is included in the sample, its cash is distributed to its neighbors in equal proportion. The next node selected for inclusion into the sample, then, is a function of the sum of cash a node has received from its neighbors. We evaluate both these methods not only in their ability to sample PageRank (or Eigenvector Centrality), but also other centrality measures.

BFS, DFS, and Random Walks. As a basis for comparison, we evaluate the performance of several basic approaches to online sampling. Specifically, we evaluate the breadth-first search (BFS), the depth-first search (DFS), and sampling based on a random walk (RW) of the social network.

4 Evaluation

4.1 Experimental Setup

Datasets. We evaluate four diverse, real-world social networks¹. These include a co-authorship network (Cond-Mat [17]), an email network (Enron [18]), an online trust network (Epinions [19]), and an online social network (Slashdot [20]).

Sampling Methodology. As described earlier, our aim is to sample a *minute* fraction of the individuals in the original network in a way that the individuals ranked highly in the entire network are both present *and* ranked highly in the induced subgraph of the sample. We execute each sampling algorithm on each dataset and sample up to 5% of the nodes. Ten samples are produced by each algorithm on each dataset. During the sampling process, at one percent intervals (e.g. 1%, 2%, ..., 5%), we compute centrality and evaluation statistics on the induced subgraph of the sample and compare those with the original network (evaluation criteria are described in the next section). When evaluating eigenvector centrality, we employ the PageRank variant (PageRank is a variant of eigenvector centrality, as described in Section 2).

Measuring Sample Quality. We employ two evaluation criteria to measure the quality of samples. First, we perform a rank correlation between the centrality ranking in the sample and the true centrality ranking in the original network using Kendall’s Tau [21]. This measure (which ranges in value from -1 to 1) evaluates the extent to which the relative ordering by centrality of all nodes in the sample is consistent with that of the original network. A value of 1 indicates the rankings are perfectly consistent, and a value of -1 indicates they are inversely consistent. However, in most cases, it is the set of *high* centrality individuals in the network that is of the most interest. A sample exhibiting a high rank correlation, but consisting only of *low* centrality individuals is not of interest in most cases. Therefore, we employ a second, more informative, evaluation criterion based on the Jaccard measure of set similarity [22], which is the size of

¹ For all networks, we extract the giant component as an undirected, unweighted graph.

the intersection of two sets divided by the size of the union. A Jaccard similarity of 1 indicates that all the elements are shared between the sets, and a score of 0 indicates that none of the elements are shared. We take the top k individuals in the sample centrality ranking and the top k individuals in the ranking of the original network and measure the Jaccard set similarity. This measure indicates how well each sampling algorithm is able to determine the *identity* of the top k highest centrality individuals. For our experiments, we use $k = 50$.

4.2 Experimental Results

Identifying High Betweenness Individuals. Figure 1a shows the extent to which each sampling algorithm is able to identify individuals with the highest betweenness scores in the network. On all datasets, the *expansion sampling* algorithm consistently (and significantly) outperforms all other approaches. Recall that, in *expansion sampling*, nodes selected for inclusion in the sample are those that maximize the expansion of the sample at each step: $\operatorname{argmax}_{v \in N(S)} |N(\{v\}) - (N(S) \cup S)|$. By sampling nodes that contribute most to the expansion, the algorithm seeks out individuals with the most dissimilar neighborhood. These nodes, then, should be brokers or bridges *between* different neighborhoods or clusters of individuals, which captures the notion of betweenness.

The random walk sampling, in some cases, also identifies high betweenness fairly well (though, not as well as *expansion sampling*). Intuitively, high betweenness individuals will appear with high frequency on the paths between other individuals. Performing a random walk, then, should tend to traverse through high betweenness nodes, thereby, including them in the sample. Finally, we see that BFS and OPIC sampling perform particularly poorly when being used to find high betweenness individuals.

Identifying High Closeness Individuals. As shown in 1b, *expansion sampling* also outperforms other methods in its ability to sample high closeness individuals. Recall that closeness centrality is a measure of how close an individual is to all other individuals in the network. Why should *expansion sampling* also perform best on this centrality measure? Consider a sample S with high expansion (i.e. $\frac{|N(S)|}{|S|}$ is significantly large). This means that the sample S , as a whole, is one hop away from many other individuals in the network, which, by definition, is closeness. In addition, as with betweenness, BFS sampling again performs poorly in identifying high closeness individuals.

Identifying High PageRank Individuals. Figure 1c shows the performance of each sampling technique in sampling PageRank. Surprisingly, *expansion sampling* again outperforms other methods (although, not as dramatically as with betweenness and closeness). This result is quite unexpected, as the Web crawling algorithms have been specifically designed to sample high PageRank individuals quickly. Moreover, one would expect a random walk sampling strategy to perform much better (if not the best) because, by the very formulation of PageRank centrality, high PageRank individuals will tend to be visited more frequently by

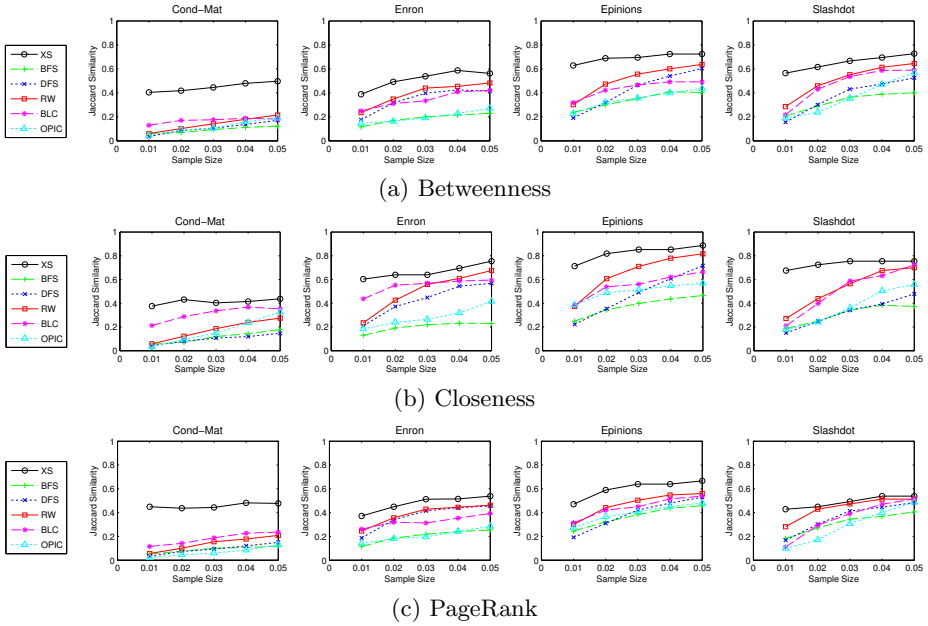


Fig. 1. Jaccard set similarity between Top 50 of original network and Top 50 of sample for each centrality measure on each dataset. For all datapoints, standard error is very low, and, for ease of illustration, the standard error bars are omitted. **Key:** *XS* = *Expansion Sampling*, *BFS* = *Breadth-First Search*, *DFS* = *Depth-First Search*, *RW* = *Random Walk*, *BLC* = *Backlink Count*, *OPIC* = *OPIC algorithm*.

a random walker. But, clearly, this property holds only in the limit and not for a small sample. Overall, once again, it is *expansion sampling* that identifies high PageRank individuals most effectively.

On the Concordance Across Centrality Measures. Thus far, we have shown that *expansion sampling* performs best in identifying influential individuals by *all three* centrality measures evaluated. A single sample produced by the *expansion sampling* algorithm, then, includes individuals ranking highly on multiple centrality measures. This begs the question: to what extent do these different centrality measures coincide or correspond with one another in real-world, social networks? That is, does *expansion sampling* simply find individuals that simultaneously rank highly on all three of the centrality measures? Upon closer inspection, we find this to *not* always be the case. Although we do find some agreement or overlap across the centrality measures (i.e. there exist individuals ranking highly on multiple measures), *expansion sampling* does, in fact, find individuals over and above this overlap. We compute the Jaccard similarity between the sets of the top 50 ranked individuals in the *entire* network by each centrality measure (i.e. the similarity between each of the *true, global* centrality rankings). For instance, in the Enron dataset, we find that the Jaccard similarity of the

betweenness ranking and the closeness ranking (both of the entire Enron network) is 0.37. However, as shown in Figure 1b, the Jaccard similarity between the closeness ranking of the entire network and the closeness ranking of the sample produced by *expansion sampling* ranges from 0.6 to 0.75 (over and above 0.37). This indicates that the *expansion sampling* method does, in fact, find individuals ranked highly only on closeness and not on betweenness. Using similar analysis, *expansion sampling* also finds individuals ranking highly on betweenness and not on closeness. Although *expansion sampling* does indeed identify individuals ranking highly on multiple measures, remarkably, this method also finds individuals that each rank highly on *different* measures of centrality. This is striking, as one would not expect a *single*, biased sampling strategy to be able to find high ranked nodes across *multiple* (and diverse) measures of centrality. The *expansion sampling* method, however, does just this.

Consistency in Relative Ordering of Sample and Original Network. Finally, we show the Kendall’s Tau rank correlation between the centrality ranking of the samples and the true centrality ranking in the original network (over all individuals in the sample). Due to space constraints, we only show the results for the Enron network in Figure 2. Although results vary slightly across measures and datasets, all sampling algorithms seem to exhibit a relatively strong consistency in the relative ordering of sampled individuals by centrality in comparison to the true centrality ranking. The key to effectively identifying the top ranked individuals, then, is finding and including them early in the sampling process. As discussed in Sections 4.2, 4.2, and 4.2, it is *expansion sampling* that is most effective in this regard.

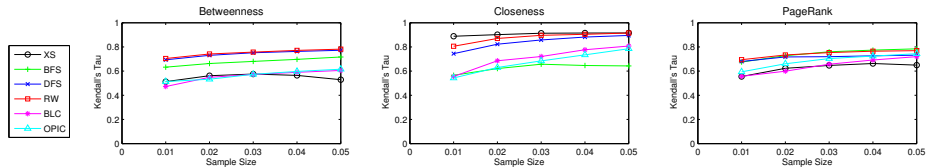


Fig. 2. [Enron Dataset] Kendall’s Tau rank correlation between the sample centrality rankings and the true ranking in the original network for each centrality measure

5 Conclusion

In this work, we have studied the use of *online sampling* to identify the set of individuals exhibiting the highest centrality in large social networks. We showed that, by sampling nodes to maximize the *expansion* of the sample, the set of most influential individuals can be approximated across multiple centrality measures. For future work, we plan to investigate the effect of network and graph-theoretic properties on the performance of these and other sampling strategies.

References

1. Freeman, L.C.: Centrality in social networks. *Social Networks* 1, 215–239 (1979)
2. Wasserman, S., Faust, K., Iacobucci, D.: *Social Network Analysis: Methods and Applications*. Cambridge University Press, Cambridge (November 1994)
3. Bavelas, A.: Communication patterns in task-oriented groups. *J. Acoustical Soc. of Am.* 22(6), 725–730 (1950)
4. Russo, T., Koesten, J.: Prestige, centrality, and learning: A social network analysis of an online class. *Communication Education* 54(3), 254–261 (2005)
5. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab (1998)
6. Anthonisse, J.: The rush in a graph. *Mathematische Centrum*, Amsterdam (1971)
7. Freeman, L.: A set of measures of centrality based on betweenness. *Sociometry* 40, 35–41 (1977)
8. Bonacich, P.: Power and centrality: A family of measures. *American J. Sociology* 92(5), 1170–1182 (1987)
9. Boldi, P., Santini, M., Vigna, S.: Paradoxical effects in pagerank incremental computations. In: *Workshop on Web Graphs* (2004)
10. Abiteboul, S., Preda, M., Cobena, G.: Adaptive on-line page importance computation. In: *WWW* (2003)
11. Cho, J., Molina, H.G., Page, L.: Efficient crawling through url ordering. *Computer Networks and ISDN Systems* 30(1-7), 161–172 (1998)
12. Najork, M.: Breadth-first search crawling yields high-quality pages. In: *WWW 2001* (2001)
13. Leskovec, J., Kleinberg, J., Faloutsos, C.: Graphs over time: densification laws, shrinking diameters and possible explanations. In: *KDD 2005* (2005)
14. Krishnamurthy, V., Faloutsos, M., Chrobak, M., Cui, J., Lao, L., Percus, A.: Sampling large internet topologies for simulation purposes. *Computer Networks* 51(15), 4284–4302 (2007)
15. Hubler, C., Kriegel, H.P., Borgwardt, K., Ghahramani, Z.: Metropolis algorithms for representative subgraph sampling. In: *ICDM 2008* (2008)
16. Hoory, S., Linial, N., Wigderson, A.: Expander graphs and their applications. *Bull. Amer. Math. Soc.* 43, 439–561 (2006)
17. Leskovec, J., Kleinberg, J., Faloutsos, C.: Graph evolution: Densification and shrinking diameters. *ACM TKDD* 1(1), 2 (2007)
18. Shetty, J., Adibi, J.: Enron email dataset. Technical report (2004)
19. Richardson, M., Agrawal, R., Domingos, P.: Trust Management for the Semantic Web. In: Fensel, D., Sycara, K., Mylopoulos, J. (eds.) *ISWC 2003*. LNCS, vol. 2870, pp. 351–368. Springer, Heidelberg (2003)
20. Leskovec, J., Lang, K.J., Dasgupta, A., Mahoney, M.W.: Statistical properties of community structure in large social and information networks. In: *WWW 2008* (2008)
21. Kendall, M., Gibbons, J.D.: *Rank Correlation Methods*, 5th edn. (September 1990)
22. Jaccard, P.: Étude comparative de la distribution florale dans une portion des alpes et des jura. *Bulletin del la Société Vaudoise des Sciences Naturelles* 37, 547–579 (1901)

Estimate on Expectation for Influence Maximization in Social Networks*

Yao Zhang, Qing Gu**, Jun Zheng, and Daoxu Chen

State Key Lab. for Novel Software and Technology,
Nanjing University, Nanjing 210093, China
zhangyao.cs@gmail.com, junzheng@smail.nju.edu.cn,
{guq, cdx}@nju.edu.cn

Abstract. Finding the most influential nodes is an important issue in social network analysis. To tackle this issue, Kempe et al. proposed the natural greedy strategy, which, although provides a good approximation, suffers from high computation cost on estimating the influence function even if adopting an efficient optimization. In this paper, we propose a simple yet effective evaluation, the *expectation*, to estimate the influence function. We formulate the expectation of the influence function and its marginal gain first, then give bounds to the expectation of marginal gains. Based on the approximation to the expectation, we put forward a new greedy algorithm called Greedy Estimate-Expectation (GEE), whose advantage over the previous algorithm is to estimate marginal gains via expectation rather than running Monte-Carlo simulation. Experimental results demonstrate that our algorithm can effectively reduce the running time while maintaining the influence spread.

1 Introduction

Information diffusion is one of the most important issues in social network analysis. A problem in this field is to find a k -nodes subset S that nodes in S can influence the largest number of nodes in the whole network. This problem, referred as *influence maximization problem*, can be applied to many areas such as product marketing and application promotion in online communities.

Domingos and Richardson [1][2] first investigated the influence propagation in the area of viral marketing [3][4]. Then, Kempe et al. [5] formulated the influence maximization problem. They proposed a natural greedy algorithm to solve the influence maximization problem, which provided a $(1 - 1/e)$ -approximation. However, their greedy algorithm was quite time-consuming to evaluate the influence spread, as it needed to run random process for a large amount of times to guarantee an accurate estimate on the influence spread.

* This work is supported in part by the National High-Tech Research and Development Plan of China (863) under Grant No. 2006AA01Z177, the 973 Program of China under Grant No. 2009CB320705, the NSFC Project under Grant No. 60873027 and Jiangsu Provincial NSF Project under Grant No. BK2006115.

** Corresponding author.

Recent researches have focused on solving this drawback, and several improvements have been proposed. Leskovec et al. put forward an optimization called Cost-Effective Lazy Forward (CELf) [6]; Kimura and Saito posed a bond-percolation based improvement [7] and a model called SPM (Shortest Path Model) [8]; Chen et al. [9] studied the influence maximization from two complementary directions: one was to improve the simple greedy algorithm, and the other was to design new efficient heuristics.

In this paper, we propose a novel evaluation, the *expectation*, to the influence spread, whose advantage is that it avoids running Monte-Carlo simulation. We formulate the expectation of influence function and its marginal gain, and give bounds to the expectation of marginal gains in theory. Then, we show that a good estimate on the expectation can be obtained by graph-based algorithms, and furthermore, a pruning technique is proposed for estimating the expectation.

Based on the expectation, a new greedy algorithm, referred as *GEE (Greedy Estimate-Expectation)*, is put forward for the influence maximization problem. Experimental results demonstrate that GEE is well-performed in the influence spread and running time for both independent cascade (IC) model and weighted cascade (WC) model compared to the simple greedy algorithm with CELf optimization (10-140 times faster in running time and only at most 2.4% lower in influence spread). And moreover, the running time would be even faster if we apply CELf optimization to our GEE algorithm.

The main contributions of this paper can be concluded as follows: first, we provide a novel evaluation, the *expectation*, to estimate the influence function, which, to the best of our knowledge, is the first time that using expectation to circumvent a large amount of computation on running random process; second, we give a theoretical explanation to the effectiveness of SPM (Shortest Path Model) [8]; third, we put forward the first expectation-based greedy algorithm and demonstrate its effectiveness on real-life networks.

2 Background

Influence Maximization Problem. We define $\sigma(S)$ as the number of nodes that are influenced by k -nodes set S , then the influence maximization problem is formulated as *finding a subset \hat{S} in V , where $|\hat{S}| = k$, to maximize $\sigma(\hat{S})$* . The computation of $\sigma(S)$ is based on information diffusion models.

Information Diffusion Models. We discuss two information diffusion models: independent cascade model and weighted cascade model [5]. In both of them, node v is influenced by its neighbor u with a probability $p_{u,v}$. In IC model, $p_{u,v}$ is an independent parameter, and in WC model, $p_{u,v}$ is assigned to $1/d_v$.

The information diffusion process for two models is described below [5]. First, the initiate set S is given. We call nodes in S active nodes, while nodes in $V \setminus S$ inactive. Nodes can transform from active state to inactive state, but can not switch verse vice. When node u first becomes active at step t , it provides only a single chance to activate each currently inactive neighbor v with probability $p_{u,v}$. If u succeeds, v will become active at step $t + 1$, and u can not activate

v any more after step t . If v has multiple active neighbors at step t , neighbors' activations are sequenced in an arbitrary order. The diffusion process stops when there are no more activities in the network.

General Greedy Algorithm. Kempe et al. proposed a simple greedy algorithm to approximate the solution [5], which starts with an empty set $S = \emptyset$, and iteratively, selects a node u for set S to maximize the marginal gain $\delta_s(u) = \sigma(S \cup \{u\}) - \sigma(S)$, then the algorithm stops until $|S| = k$.

$\sigma(S)$ is computed by simulating the random process for R times (R could be very large in order to guarantee efficiency). Leskovec et al. [6] proposed a CELF optimization, which can get the same result but is much faster than Kempe et al.'s algorithm, for its great reduction of computing $\delta_s(u)$. But it still costs for hours on large-scale networks.

3 Proposed Method

In this section, we use *expectation* to estimate the influence function $\sigma(S)$ and the marginal gain $\delta_s(u)$, and give an approximation to the expectation of $\delta_s(u)$. Then we propose an algorithm called *Greedy Estimate-Expectation(GEE)* for the influence maximization problem.

3.1 Estimate on Expectation

We denote $p(S, v)$ as the propagation probability that v is influenced by Set S . Suppose the probabilities that other nodes influence node v are independent, according to information diffusion models described above, we have $p(S, v) = 1 - \prod_{u \in S} (1 - p(u, v))$.

The expectation of $\sigma(S)$, formulated as $E(\sigma(S))$, is:

$$E(\sigma(S)) = \sum_{\forall v \in V} p(S, v) * |\{v\}| = \sum_{\forall v \in V} p(S, v) \quad (1)$$

According to $\delta_s(u) = \sigma(S \cup \{u\}) - \sigma(S)$ and the above equations, we have $E(\delta_s(u)) = \sum_{\forall v \in V} (1 - p(S, v)) * p(u, v)$. Suppose $R(u, G)$ is the set of nodes which are reachable from u , so:

$$E(\delta_s(u)) = \sum_{v \in R(u, G)} (1 - p(S, v)) * p(u, v) \quad (2)$$

We denote $p_{path_i}(u, v)$ as the propagation probability from u to v through path i . Let $\hat{p}(u, v) = \max\{p_{path_i}(u, v) | v \in R(u, G)\}$, and $\lambda_u = \max\{\lambda_{u,v} | v \in R(u, G)\}$, where $\lambda_{u,v}$ is the number of paths from u to v .

Theorem 1. For IC model and WC model, if $p(S, v)$ is given, then:

$$\begin{aligned} & \sum_{v \in R(u, G)} (1 - p(S, v)) * \hat{p}(u, v) \leq E(\delta_s(u)) \\ & \leq \min\{1, \lambda_u \sum_{v \in R(u, G)} (1 - p(S, v)) * \hat{p}(u, v)\}. \end{aligned}$$

We define $k_{u,v}$ as the distance(shortest path) from u to v , and then, for IC model with uniform propagation probability p , we have $\hat{p}(u, v) = p^{k_{u,v}}$.

If we denote t_i as the number of path whose length is i from node u to node v , then in IC model, $p(u, v) \leq \sum_{i=k_{u,v}}^l t_i p^i$, where l is the maximum length of paths from u to v .

Theorem 2. For IC model with uniform propagation probability p , if $p(S, v)$ is given, and $t = \max(t_{k_{u,v}}, t_{k_{u,v}+1}, \dots, t_l)$, then:

$$E(\delta_s(u)) \leq t * \frac{p^{n-1}-1}{p-1} \sum_{v \in R(u,G)} (1 - p(S, v)) p^{k_{u,v}}.$$

Theorem 1 gives bounds to the expectation of $\delta_s(u)$ for both IC model and WC model. Theorem 2 provides an upper bound that is closer to the expectation of $\delta_s(u)$ for IC model. In a sparse graph with a small value of p , $t \frac{p^{n-1}-1}{p-1}$ is close to 1, which means $E(\delta_s(u))$ can be estimate by $\sum_{v \in R(u,G)} (1 - p(S, v)) p^{k_{u,v}}$ effectively. It is an amazing result that theoretically interprets why Shortest Path Model (SPM: the model where each node is activated only through the shortest paths) [8] works well.

Theorem 1 and Theorem 2 show that the expectation of $\delta_s(u)$ can be estimate by computing $\hat{p}(u, v)$. Suppose $\hat{E}(\sigma(S))$ is the estimate of $E(\sigma(S))$ through estimating $p(u, v)$ by $\hat{p}(u, v)$, then $\hat{E}(\delta_s(u)) = \sum_{v \in R(u,G)} (1 - p(S, v)) \hat{p}(u, v)$.

According to the equation that $p(S \cup \{u\}) = p(S, v) + (1 - p(S, v)) p(u, v)$, $p(S \cup \{u\}, v)$ can be estimated by previous $p(S, v)$ in a greedy approximate algorithm. The value of $p(u, v)$ can be effectively approximated by $\hat{p}(u, v)$. For IC model, we are able to obtain $\hat{p}(u, v)$ by $k_{u,v}$ through *Breadth-First Search (BFS)*, which takes $O(n(n + m))$ time. For WC model, the algorithm to get $\hat{p}(u, v)$ is resemble to shortest-path algorithm in a weighted graph, such as the *Dijkstra Algorithm*. Using *Fibonacci heap*, the running time is $O(n(n \log n + m))$.

We denote $d_{u,max}$ as the maximum degree of node $v \in R(u, G)$. Let $z = p * (d_{u,max} - 1)$.

Theorem 3. For IC model with uniform propagation p , if $z < 1$, $\forall \varepsilon > 0$, $\exists K = \lceil \log_z (\varepsilon * \frac{1-z}{d_u p} + z^n) \rceil$, for all node $v \in R(u, G)$ and $k_{u,v} \leq K$, then

$$\hat{E}(\delta_s(u)) - \sum_{\substack{v \in R(u,G) \\ k_{u,v} \leq K}} (1 - p(S, v)) p^{k_{u,v}} < \varepsilon.$$

Theorem 3 suggests that under the condition of $z < 1$, $\hat{E}(\delta_s(u))$ in IC model can be approximated effectively in the case that $k_{u,v} \leq K$, which means *BFS* for computing $\hat{p}(u, v)$ can be terminated within K layers! We call this *BFS pruned BFS*. For those networks whose K are not large, *pruned BFS* makes progress in running time compared to original *BFS* that takes $O(n(n + m))$ time.

3.2 GEE Algorithm

We put forward an algorithm called *Greedy Estimate-Expectation(GEE)* for influence maximization problem. There are two phases in *GEE*: calculating $\hat{p}(u, v)$ and greedily obtaining the set S_k .

Algorithm 1. Greedy Estimate-Expectation Algorithm

Phase One

- 1: If $MODEL=IC$, then Get $\hat{p}(u, v)$ for all (u, v) by *full BFS* or *pruned BFS*
- 2: If $MODEL=WC$, then Get $\hat{p}(u, v)$ for all (u, v) by *Fibonacci heap* or *pruned BFS*

Phase Two

- 1: Input $\hat{p}(u, v)$ for all (u, v)
- 2: Initialize $S := \emptyset$, and $p(S, v) := 0$ Foreach $v \in V$
- 3: **for** $i := 1$ to k **do**
- 4: Foreach $u \in V \setminus S$, $\hat{E}(\delta_s(u)) := \sum_{v=0}^{|V|} \hat{p}(u, v)(1 - p(S, v))$, if $\hat{p}(u, v) \neq 0$
- 5: Select a node u with maximum $\hat{E}(\delta_s(u))$
- 6: $S := S \cup \{u\}$
- 7: $p(S, u) := 1$
- 8: Foreach $v \in V \setminus S$, update $p(S, v) := p(S, v) + \hat{p}(u, v)(1 - p(S, v))$
- 9: **end for**
- 10: **return** S

The running time of the first phase depends on its implementation. As mentioned above, $\hat{p}(u, v)$ for all nodes (u, v) can be obtained by running *BFS* or *pruned BFS* with $O(n(n+m))$ time or $O(n^{\frac{\bar{d}^k-1}{d-1}})$ time, and by using *Fibonacci heap* with $O(n(n \log n + m))$ time. We also take *pruned BFS* as a complement for WC model, and it performs well in our experiments.

The second phase of *GEE* is to get a node u for S_k once at a time with the maximum $\hat{E}(\delta_s(u))$. Instead of running random process for sufficient times, *Phrase Two* takes $O(kn^2)$ in running time, and if we only consider K layers in *BFS*, the average running time could be $O(kn^{\frac{\bar{d}^K-1}{d-1}})$. Moreover, we can also use the CELF optimization to accelerate *Phrase Two*.

$\hat{E}(\sigma(S))$, the estimate function of $E(\sigma(S))$, is a submodular function, which means *GEE* algorithm provides a $(1-1/e)$ -approximation according to the property of submodular function [10].

4 Experiments

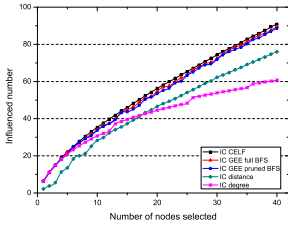
We employ two collaboration networks from paper-lists in sections of the e-print arXiv. The first network is from the "General Relativity and Quantum Cosmology" (*Gr-Qc*) section with 5242 nodes and 28980 edges [4]. The second network is from the "High Energy Physics - Theory" (*Hep*) section with 15233 nodes and 58891 edges [9]. All experiments are implemented on a PC with Intel 2.20GHz Pentium Dual E2200 processor and 4GB memory.

Table 1 lists algorithms used in our experiments. *Degree* and *Distance* are simple heuristics used in [5]. Note that in *GEE with pruned BFS*, if initializing $\varepsilon = 10^{-8}$ and using \bar{d} , then we have $K = 6$ in *Gr-Qc* graph and $K = 5$ in *Hep* graph. To make experiment results convincing, we simulate the random process for $R = 20000$

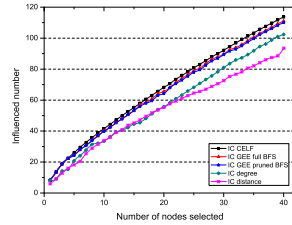
¹ <http://snap.stanford.edu/data/ca-GrQc.html>

Table 1. Algorithms used in experiments

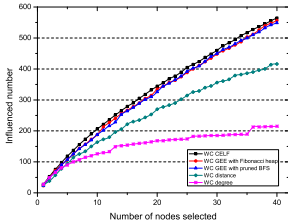
| Algorithm | Applied Model | Description |
|--------------------------------|---------------|--------------------------------------|
| <i>Degree</i> | IC, WC | Degree heuristic |
| <i>Distance</i> | IC, WC | Distance heuristic |
| <i>Greedy with CELF</i> | IC, WC | CELF optimization ($R = 20000$) |
| <i>GEE with pruned BFS</i> | IC, WC | Algorithm 1 ($\epsilon = 10^{-8}$) |
| <i>GEE with full BFS</i> | IC | Algorithm 1 |
| <i>GEE with Fibonacci heap</i> | WC | Algorithm 1 |



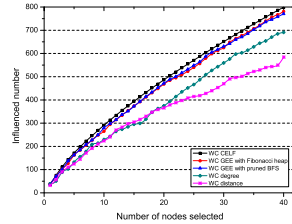
(a) IC Model for Gr-Qc Graph



(b) IC Model for Hep Graph



(c) WC Model for Gr-Qc Graph



(d) WC Model for Hep Graph

Fig. 1. Influence spread. (a) IC model for Gr-Qc graph. (b) IC model for Hep graph. (c) WC model for Gr-Qc graph. (d) WC model for Hep graph.

times (the same as times in [9]), and then, take the average of the influence spread numbers as the influence spread results for each algorithm.

In IC model, we mainly discuss experiments with an uniform $p = 0.01$. We also consider $p = 0.02$. We do not report its result as its trends on influence performance and running time are similar to the situation that $p = 0.01$.

Influence spread. In all figures, we discuss about percentages of influence spread for the case of $k = 40$. Figure 1(a)(b) demonstrate the influence spread results of different algorithms with probability $p = 0.01$ in IC model. Our *GEE with pruned BFS* and *GEE with full BFS*, performs quite well. They are only about 1.5% and 2.4% lower than the *Greedy with CELF* in the *Gr-Qc* graph and *Hep* graph respectively. And the differences between two *GEE* algorithms are less than 1%, which means our pruning technique works well as we expect.

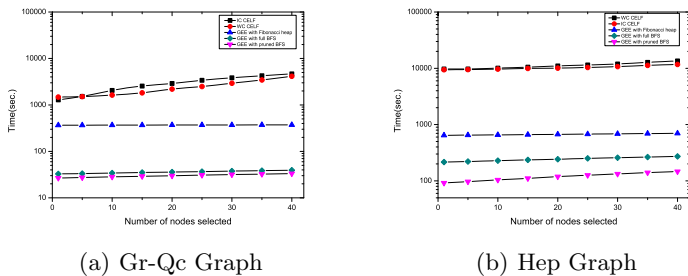


Fig. 2. Running time(sec.). (a) Gr-Qc graph. (b) Hep graph.

Figure 1(c)(d) demonstrate the influence spread results of different algorithms in WC model. The performance of our *GEE with pruned BFS* and *GEE with Fibonacci heap* are quite close to *Greedy with CELF*. *GEE with Fibonacci heap* is only 1.4% and 2.3% lower than the *Greedy with CELF* in the *Gr-Qc* graph and *Hep* graph respectively. It is surprising that *GEE with pruned BFS* with much faster running time, outperforms *GEE with Fibonacci heap* for some value of k when $k < 40$.

Running time. Figure 2 demonstrates the running time of various algorithms. We do not list running times of *Degree* and *Distance* for their poor performances on the influence spread. Our *GEEs* run orders of magnitude faster compared to *Greedy with CELF*. Specifically, when $k = 40$, *GEE with Fibonacci heap* is 11 times and 19 times faster than *Greedy with CELF* in the *Gr-Qc* and *Hep* graph respectively, and *GEE with full BFS* is 119 times and 43 times faster than *Greedy with CELF* in the *Gr-Qc* and *Hep* graph respectively. *GEE with pruned BFS*, for its pruning technique, impressively saves the running time for about 80 to 140 times in the *Gr-Qc* and *Hep* graph!

5 Discussion

Our *GEEs* show very impressive experiment results for both IC model and WC model: they further improve the running time (range from 10 times to 140 times faster when 40 nodes is selected), while almost match the influence spread of *Greedy with CELF* (only 1.4% to 2.4% lower).

There are other well-performed algorithms for the influence maximization problem: shortest path model(SPM) [8], bond-percolation based algorithm(BP) [7], and degree discount heuristic [9]. We do not list these experimental results for the following reasons: degree discount heuristic, although almost matches the influence spread in much faster time compared to the simple greedy algorithm, is only limited to the IC model; BP is similar to the improvement of the greedy algorithm discussed in [9], which costs as much time as the CELF optimization; SPM can only apply to IC model, and moreover, it can get the similar result as our *GEE with full BFS* in slower time due to the implementation of SPM.

6 Conclusion

In this paper, we propose a new evaluation, the *expectation*, to estimate the influence function and its marginal gain for the influence maximization problem. We give bounds to the expectation of $\delta_s(u)$, and further, theoretically interpret the effectiveness of Shortest Path Model (SPM) [8]. Then we put forward an expectation based algorithm called Greedy Estimate-Expectation (GEE). Using two collaboration networks, we experimentally demonstrate that our GEE algorithm impressively shortens the running time while maintaining the influence results that obtained by the simple greedy algorithm with the CELF optimization.

Future research will try to use the expectation to estimate the influence function on other diffusion models. Another direction is to explore the internal structures of networks to improve the influence spread.

References

1. Domingos, P., Richardson, M.: Mining the network value of customers. In: KDD, pp. 57–66 (2001)
2. Richardson, M., Domingos, P.: Mining knowledge-sharing sites for viral marketing. In: KDD, pp. 61–70 (2002)
3. Goldenberg, J., Libai, B., Muller, E.: Talk of the networks: a complex systems look at the underlying process of word-of-mouth. *Marketing Letters* 12 (2001)
4. Leskovec, J., Adamic, L.A., Huberman, B.A.: The dynamics of viral marketing. *TWEB* 1(1) (2007)
5. Kempe, D., Kleinberg, J., Tardos, E.: Maximizing the spread of influence through a social network. In: KDD, pp. 137–146 (2003)
6. Leskovec, J., Krause, A., Guestrin, C., Faloutsos, C., VanBriesen, J., Glance, N.S.: Cost-effective outbreak detection in networks. In: KDD, pp. 420–429 (2007)
7. Kimura, M., Saito, K., Nakano, R.: Extracting influential nodes for information diffusion on a social network. In: AAI, pp. 1371–1376 (2007)
8. Kimura, M., Saito, K.: Tractable models for information diffusion in social networks. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) PKDD 2006. LNCS (LNAI), vol. 4213, pp. 259–271. Springer, Heidelberg (2006)
9. Chen, W., Wang, Y., Yang, S.: Efficient influence maximization in social networks. In: KDD, pp. 199–208 (2009)
10. Nemhauser, G., Wolsey, L., Fisher, M.: An analysis of the approximations for maximizing submodular set functions. *Mathematical Programming* 14, 265–294 (1978)

A Novel Scalable Multi-class ROC for Effective Visualization and Computation

Md. Rafiul Hassan¹, Kotagiri Ramamohanarao¹, Chandan Karmakar²,
M. Maruf Hossain¹, and James Bailey¹

¹ Department of Computer Science and Software Engineering,
The University of Melbourne, VIC 3010, Australia

² Department of Electrical Engineering,
The University of Melbourne, VIC 3010, Australia
{mrhassan,jbailey,rao}@csse.unimelb.edu.au

Abstract. This paper introduces a new cost function for evaluating the multi-class classifier. The new cost function facilitates both a way to visualize the performance (expected cost) of the multi-class classifier and a summary of the misclassification costs. This function overcomes the limitations of ROC in not being able to represent the classifier performance graphically when there are more than two classes. Here we present a new scalable method for producing a scalar measurement that is used to compare the performance of the multi-class classifier. We mathematically demonstrate that our technique can capture small variations in classifier performance.

Keywords: Multi-class, Receiver operating characteristics, classifier evaluation, cost-function.

1 Introduction

Receiver Operating Characteristic (ROC) analysis [1] [2] [3] is a widely used technique for evaluating classifiers. The technique considers the confusion matrix in order to generate a plot and compare the performance of the classifiers. While evaluating the classifier, the ROC plot considers all possible operating (decision) points in the classifier's prediction as a means of identifying the operating point at which the best performance is achieved. A confusion matrix forms at the operating point that consists of "True positive", "True Negative", "False positive" and "False negative" values. The ROC plot is obtained for a classifier by plotting the true-positive rate (TPR) over the y-axis and the false-positive rate (FPR) over the x-axis. The points obtained on the plot are connected to the points where the values of both TPR and FPR are extreme: i.e. the points will be (0,0) and (1,1). This is done to form a complete curve. The area under the curve (AUC) is a scalar measurement used to evaluate the classifier, while the plot provides the visual representation that is used to compare the classifiers' performances.

Challenges: The ROC plot and AUC have been a popular technique for evaluating classifiers, however their application is limited to binary class problems.

Although the ROC and AUC have been developed in theory to deal with multi-dimensional problems, their inherent computational complexity and representational comprehensibility hinders their use in practice. For instance, a confusion matrix obtained from a problem of 'M' classes is an $M \times M$ matrix, and $d = (M.(M - 1))$ dimensions: i.e. to evaluate a classifier, all the possible misclassification combinations are needed [4]. The ROC plot for a binary classifier facilitates a visual comparison of binary classifiers, however, the ROC method is a tedious task if a plot is to be drawn for a multi-class classifier, as the dimensionality of the plot increases dramatically with just a small increase in the number of classes. For example, if we were to generate a plot for a three-class classifier the number of dimensions we have to deal is six while that of binary class classifier is two. The problem is not only limited to the representation or visualization, but also to the computational complexity. This cost is $O(n^d)$ for a convex hull of n points for d dimensions.

Contributions: Our main contributions in this paper are as follows:

- Developing a new method for visualizing the performance of multi-class classifiers on a single plot;
- Developing a new method for representing the performance of multi-class classifiers using a scalar measurement with reduced computational cost; and
- Experimental investigation of the developed methodologies that demonstrates that our methods can compare classifiers both visually and numerically.

Organization: The remainder of the paper is organized as follows. Section 2 lists the studies that developed cost function for multi-class problem using AUC. In Section 3 we briefly describe the receiver operating characteristics curves. Section 4 describes the proposed methodology in details. Section 5 details about the experimental setup and generation of dataset. In Section 6 we analyze the results obtained using our method and compare with one of the existing techniques. Finally, in Section 7 we conclude the paper.

2 Related Work

In recent years, a few approaches have been developed that have extended the AUC measure for multi-class problems. Srinivasan [5] developed a 6-dimensional ROC surface for a three-class problem. These six dimensions represent six misclassification cells. However, the challenge was to compute the volume of the convex hull in multi-dimensional space. There are computational approaches for calculating the volume of the convex hull however, there is the possibility of missing some points in the construction of the convex hull due to the high dimensional surfaces [6] [2]. Amongst the existing approximations for the AUC measure for more than two classes are: Hand and Till [2], Mossman [7] and Ferri et al. [4]. Hand and Till [2] extended the AUC for two classes to multiple classes by averaging the AUCs of all the possible combinations of pairs of classes. Mossman [7] prioritized one class over the others and demonstrated his approximation for only three classes. Ferri et al. [4] used the Monte Carlo method for

their approximation for the AUC. They also introduced the method called the Hyperpolyhedron Search Algorithm (HSA). These existing approximations have the nature of intuitive extensions however, they lack a solid theoretical justification. Other limitations of these approaches include the theoretical limitation of computing the maximum volumes in the high-dimensional problem, and the fact that at least $d(d - 1)$ dimensional variable for d classes is needed for obtaining their volumes.

3 Receiver Operating Characteristics Curves

The fundamental issue for rating a classifier's performance is the confusion matrix where the numbers represent the total number of actual classes and predicted classes. However, the usual applications of the confusion matrix are limited to two-class classifiers because the classifier's performance can be easily represented using a two-dimensional matrix. In this matrix the numbers along the rows are the actual class and the numbers along the columns are the predicted class. The classes are predicted considering an operating point that divides the data into the respective classes. Usually, the numbers in the confusion matrix are known as the "true positive (TP)", "false positive (FP)", "true negative (TN)" and "false negative (FN)". The definitions of these terms are as follows:

- True Positive (TP)= the number of classified positive data that are positive in the actual data set.
- True Negative (TN)= the number of classified negative data that are negative in the actual data set.
- False Positive (FP)= the number of classified positive data that are negative in the actual data set.
- False Negative (FN)= the number of classified negative data that are positive in the actual data set.

The Receiver Operating Characteristics (ROC) plot is obtained for a binary class classifier by plotting the true-positive rate (Sensitivity) over the y-axis and false-positive rate (1- Specificity) over the x-axis. The true-positive and false positive rates are calculated using the following equations:

$$\text{True positive rate (Sensitivity)} = TP/(TP + FN)$$

$$\text{False-positive rate}(1 - \text{Specificity}) = FP/(FP + TN)$$

Figure 1 shows one such ROC plot where they sweep through all the possible operating points, and plot the corresponding false-positive and true-positive rates. There is also a simple three-point ROC curve (for discrete classifiers), where there are three points considered in order to obtain the plot. These three points are: (0,1), (FP, TP) and (1,0).The point (0,1) refers to a classifier where every instance has been classified as correct: i.e. a perfect classifier is represented by the point (0,1). The point (1,0) refers to a classifier where every instance is classified as wrong: i.e. this point refers to the worst classifier. The point (FP,

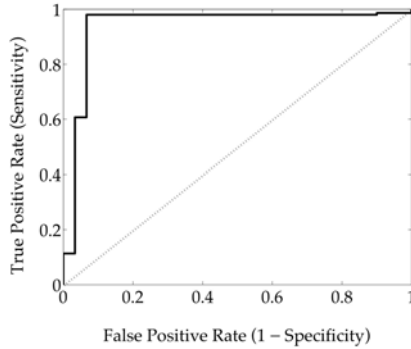


Fig. 1. A typical ROC plot

TP) represents the classifier where the values of FP and TP are obtained from the confusion matrix. The Area Under the Curve (AUC) is a scalar value that can be used to order which ROC curves are better than others.

4 Our Method

To extend the existing ROC plot and AUC for multi-class classifiers, we use the properties of the ROC plot for binary classifiers. In this extension we calculate all the possible pairwise (pair of classes) AUCs of the multiple classes.

Let us consider three classes - 1, 2 and 3 - and assume that for an operating point P_1 (i.e. if the predicted output is $Y \leq P_1$ classify the instance as class 1, and if the predicted output is $Y > P_1$ classify the instance as class 2), between the two classes 1 and 2, the values of TP and FP are 0.9 and 0.2 respectively. For these TP and FP we obtain the AUC of 0.85. Similarly, we obtain the values of TP and FP for an operating point P_2 , while classifying classes 2 and 3, as 0.8 and 0.2. The AUC value for classes 2 and 3 is 0.8. In a similar way we obtain the AUC value for classes 3 and 1 as 0.75 (for the operating point considered as P_3). For this three-class problem we obtain $\binom{3}{2} = 3$ AUC values for the three pair-wise classes.

To visualize this performance using the usual ROC plot we should consider a surface/curve plotted on a six-dimensional surface, as there are six features (i.e. three pairs of true positive and false positive rates) that need to be considered to plot the curve. Even though there are a few studies that have provided methodologies for generating the surface for multi-class problems, these plots are not helpful in comparing the classifiers' performances through visualization. An alternative solution is to represent the pairwise AUCs onto a multidimensional surface. However, the problem still remains as to how to analyze the surface for multiple classes. In our method we consider representing the AUCs using polar co-ordinates. This representation is somewhat similar to the cobweb plot, although there is some notable dissimilarity.

Initially, we consider that the range of angle is $0 \rightarrow 2\pi$. Because there are three AUC values, we divide the maximum angle (2π) by three. For a three-class problem after the division the angle becomes $\frac{2}{3}\pi$ (i.e., 120°). There will be three lines that form a 120° angle with each other. We consider that the maximum length of each of the lines is 1 because the maximum achievable value of an AUC for a binary class problem is 1. In our method we consider the binary AUC values for each of the pair of classes and draw the plot using these AUC values on an equiangular line drawn on a polar co-ordinate. The equiangular lines for a three-class problem are shown in Fig. 2. In this figure the length of each line is 1, assuming that the AUC values for each of the pair of classes are maximum.

Each radius (r) on the plot represents the AUC value for a different pair of classes. If there are m classes, there will be $q = \binom{m}{2}$ AUC values for q number of pairs of classes; hence, to visualize the performance of a classifier for an m -class problem we must have q number of equiangular lines. Let us assume that these AUC values are: $r_1, r_2, r_3, \dots, r_q$. The corresponding angles are: $\theta_1, \theta_2, \theta_3, \dots, \theta_q$, where, $\theta_1 = \theta_2 = \theta_3 = \dots = \theta_q = 2\pi/q$. If we consider all the AUC values are equal, i.e., $\forall_i r_i = r$ where $1 \leq i \leq q$, the area covered by the AUC values is:

$$Area = \frac{qr^2}{2} \sin\left(\frac{2\pi}{q}\right) \quad (1)$$

If the values of pairwise AUCs are not equal to each other as shown in Fig. 3 (i.e., $r_1 \neq r_2 \neq r_3 \neq \dots \neq r_q$), the area covered by the AUC values is:

$$area = \frac{1}{2} \sin\theta \left(\left(\sum_{i=1}^{q-1} r_i \times r_{i+1} \right) + r_q \times r_1 \right) \quad (2)$$

In Eq. 2, AUC values r_1, r_2, \dots, r_q are plotted such that r_1 is neighbor to r_2 , r_2 is neighbor to r_3 , \dots , r_q is neighbor to r_1 .

Before using the area formed through connecting the neighboring points on the plot as a classifier performance measurement, we must address the following issues:

1. What is the range of the minimum-maximum area obtained for a different number of classes; and
2. How the calculated area is affected by the ordering of the AUC values plotted on the graph.

The range of minimum-maximum area for multiple classes: The maximum area for a given number of classes is obtained, if we consider all the AUC for each pairwise classes are 1 (because the maximum AUC for a binary class problem is 1). Thus, for an m -class problem and $q = \binom{m}{2}$ AUC values, the maximum area is:

$$area_{max} = \frac{q}{2} r^2 \sin\left(\frac{2\pi}{q}\right) \quad (3)$$

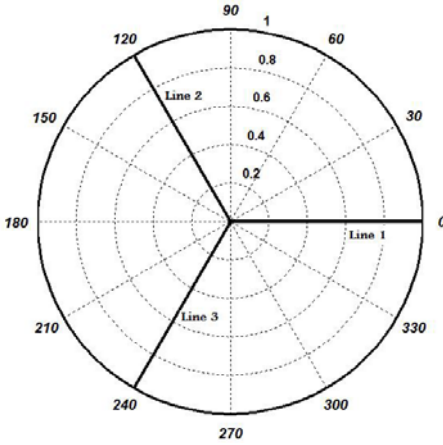


Fig. 2. Visual representation for a three-class classifier using pairwise(class) AUC values where each of the AUC values is 1

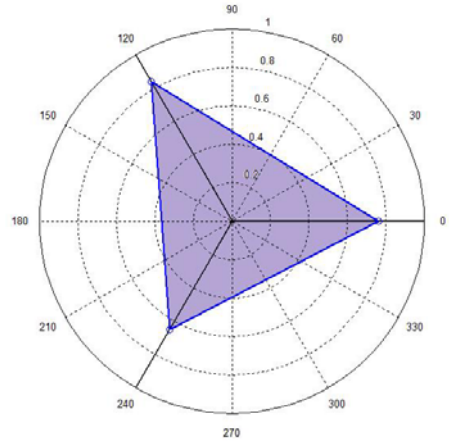


Fig. 3. The area covered by the three points for a three-class classifier where the values of pairwise AUC are: $AUC_{(1,2)} = 0.76, AUC_{(2,3)} = 0.84, AUC_{(3,1)} = 0.65$

Similarly, the minimum area is obtained, if we consider all the AUC for each pairwise classes are 0.5 (because the minimum AUC for a binary class problem is 0.5). We obtain the minimum area for an m-class problem from the following equation:

$$area_{min} = \frac{q}{2} \left(\frac{r}{2}\right)^2 \sin\left(\frac{2\pi}{q}\right) \tag{4}$$

Table 1 lists the minimum and maximum values of areas for a different number of classes.

The ordering of the AUCs: The maximum and minimum area is calculated considering the maximum ($r = 1$) AUC of each pair of classes and minimum ($r = 0.5$) AUC of each pair of classes. As a result the radius value remains the same and the value of total area is not affected by the ordering of the AUC values when plotting them onto the equiangular graph. However, in the real world we obtain varying AUC values for different pairs of classes.

Let us consider the q AUC values are: $r_1, r_2, r_3, \dots, r_{i-1}, r_i, r_j, r_{j+1}, \dots, r_q$, where $r_1 \geq r_2 \geq r_3 \geq \dots \geq r_q$ and $j = i + 1$. The area γ for these values is:

$$\gamma \propto r_1 \times r_2 + r_2 \times r_3 + \dots + r_{i-1} \times r_i + r_i \times r_j + r_j \times r_{j+1} + \dots + r_{q-1} \times r_q + r_q \times r_1$$

Let us interchange the positions of AUC values r_i with r_j in the above ordering. Now, the area γ' is:

$$\gamma \propto r_1 \times r_2 + r_2 \times r_3 + \dots + r_{i-1} \times r_j + r_j \times r_i + r_i \times r_{j+1} + \dots + r_{q-1} \times r_q + r_q \times r_1$$

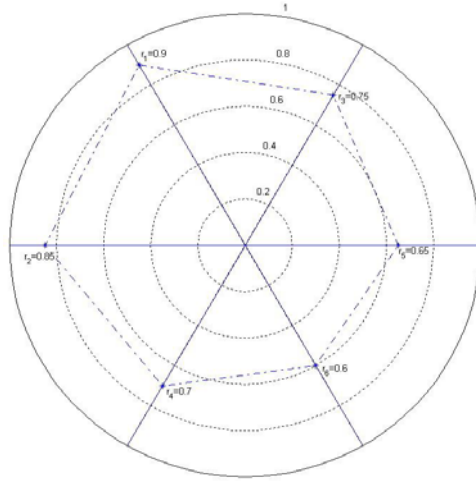


Fig. 4. Arranging AUC values such that the maximum area is found

By subtracting γ' from γ we find that $\gamma \neq \gamma'$.

This example shows that the ordering of the AUC values affects the calculated area that we use to compare the performances of the classifiers. We propose to use the maximum area, achieved through re-ordering the AUC values, to compare the performances of the classifiers. The maximum area is found if we arrange the AUC values following lemma.

Lemma 1. *The maximum area is found if the AUC values are ordered on the plot in a way such that the neighboring value of a given point (or AUC value) is the next available large value. That is, while calculating the area, the large values will be multiplied to each other.*

Proof. Let us consider the AUC values are: $r_1, r_2, r_3, \dots, r_{i-1}, r_i, r_j, r_{j+1}, \dots, r_m$. Here, $r_1 \geq r_2 \geq r_3 \dots r_i \geq r_j \geq r_{j+1} \geq \dots r_{m-1} \geq r_m$ and $j > i$. According to the lemma, a maximum area will be found if the values are arranged as in Fig. 4. In this figure, the neighboring points of r_1 are the next available large values r_2 and r_3 , while the neighboring point of r_2 (the largest of r_2 and r_3) is r_4 , which is the next available large value to r_2 . As there is no other neighboring space of r_2 on which to put any other values we move forward to r_3 and put the next large value r_5 as its neighboring point. The process is continued until all AUC values are allocated.

If the AUC values are arranged as described according to this lemma, the area covered by the AUC values is obtained using the following equation:

$$area = (r_1 \times r_2) + \sum_{i=1}^{i < k} (r_{2i-1} \times r_{2i+1}) + \sum_{i=1}^{i < k} (r_{2i} \times r_{2i+2}) + (r_{q-1} \times r_q) \quad (5)$$

Here, $k = \frac{q}{2} - 1$.

There are the following two scenarios that can affect the area:

- Replace any pair of neighboring points with each other in the above list
- Replace any pair of points with each other, where the points are not neighbor to each other in the above list.

Let us replace any pair of neighboring points r_i with r_j in the above list (where, $j > i$). The area α is calculated using Eq. 2 for this list. The area difference between α and γ is:

case $i = 1$ and $j=2$

$$\begin{aligned}\gamma - \alpha &= r_i \times r_{i+2} - r_j \times r_{i+2} + r_j \times r_{j+2} - r_i \times r_{j+2} \\ &= (r_i - r_j)(r_{i+2} - r_{j+2}) \geq 0\end{aligned}$$

case $i = 1$ and $j= 3$

$$\begin{aligned}\gamma - \alpha &= r_i \times r_{i+1} - r_j \times r_{i+1} + r_j \times r_{j+2} - r_i \times r_{j+2} \\ &= (r_i - r_j)(r_{i+1} - r_{j+2}) \geq 0\end{aligned}$$

case $i = 2$ and $j=4$

$$\begin{aligned}\gamma - \alpha &= r_i \times r_{i-1} - r_j \times r_{i-1} + r_j \times r_{j-2} - r_i \times r_{j-2} \\ &= (r_i - r_j)(r_{i-1} - r_{j-2}) \geq 0\end{aligned}$$

case $i = q-1$ and $j=q$

$$\begin{aligned}\gamma - \alpha &= r_i \times r_{i-2} - r_j \times r_{i-2} + r_j \times r_{j-2} - r_i \times r_{j-2} \\ &= (r_i - r_j)(r_{i-2} - r_{j-2}) \geq 0\end{aligned}$$

case $i = q-3$ and $j=q-1$

$$\begin{aligned}\gamma - \alpha &= r_i \times r_{i-2} - r_j \times r_{i-2} + r_j \times r_{j+1} - r_i \times r_{j+1} \\ &= (r_i - r_j)(r_{i-2} - r_{j+1}) \geq 0\end{aligned}$$

case $i = q-2$ and $j=q$

$$\begin{aligned}\gamma - \alpha &= r_i \times r_{i-2} - r_j \times r_{i-2} + r_j \times r_{j-2} - r_i \times r_{j-2} \\ &= (r_i - r_j)(r_{i-2} - r_{j-2}) \geq 0\end{aligned}$$

All other cases

$$\begin{aligned}\gamma - \alpha &= r_i \times r_{i-2} - r_j \times r_{i-2} + r_j \times r_{j+2} - r_i \times r_{j+2} \\ &= (r_i - r_j)(r_{i-2} - r_{j+2}) \geq 0\end{aligned}$$

Let us replace any pair of points (other than the neighboring points) r_i with r_j in the above list (where, $j > i$). The area α is calculated using Eq. 2 for this list. The area difference between α and γ is:

case $i = 1$ and $j = q$

$$\gamma - \alpha = (r_i - r_j)(r_{i+1} - r_{j-2} + r_{i+2} - r_{j-1}) \geq 0$$

case $i = 1$ and $j = q-1$

$$\gamma - \alpha = (r_i - r_j)(r_{i+1} - r_{j-2} + r_{i+2} - r_{j+1}) \geq 0$$

case $i = 1$ and $j =$ any other values

$$\gamma - \alpha = (r_i - r_j)(r_{i+1} - r_{j-2} + r_{i+2} - r_{j-2}) \geq 0$$

case $i = 2$ and $j = q$

$$\gamma - \alpha = (r_i - r_j)(r_{i-1} - r_{j-2} + r_{i+2} - r_{j-1}) \geq 0$$

case $i = 2$ and $j = q-1$

$$\gamma - \alpha = (r_i - r_j)(r_{i-1} - r_{j-2} + r_{i+2} - r_{j+1}) \geq 0$$

case $i = 2$ and $j =$ any other values

$$\gamma - \alpha = (r_i - r_j)(r_{i-1} - r_{j-2} + r_{i+2} - r_{j+2}) \geq 0$$

case $i =$ any other values and $j = q$

$$\gamma - \alpha = (r_i - r_j)(r_{i-2} - r_{j-2} + r_{i+2} - r_{j-1}) \geq 0$$

case $i =$ any other values and $j = q-1$

$$\gamma - \alpha = (r_i - r_j)(r_{i-2} - r_{j-2} + r_{i+2} - r_{j+1}) \geq 0$$

All other cases

$$\gamma - \alpha = (r_i - r_j)(r_{i-2} - r_{j-2} + r_{i+2} - r_{j+2}) \geq 0$$

Therefore, it is proved that the maximum area is found if the AUC values are ordered on the plot in a way such that the neighboring value of a given point (or AUC value) is the next available large value.

It should be noted that the AUC values needed to be arranged while computing the area to represent the performance using a scalar value. To compare the performances of different classifiers visually, there is no need to rearrange the AUC values.

Advantages of Our Method Compared with the Generalized AUC

In generalized AUC, the AUC that signifies a classifier's performance is obtained through a linear combination of individual AUC values for each pair of classes. Our method, on the other hand, considers a nonlinear combination among the individual AUC values. As a result, our method can find the differences in the performances of two classifiers if, the change in one pairwise AUC value is equal to the change of another pairwise AUC value for these two classifiers.

Let us assume we have n ($n = \binom{m}{2}$ for m -class problem) AUC values of r_1, r_2, \dots, r_n (here the values are ordered randomly). The area γ obtained using our method is: $\gamma \propto r_1 \times r_2 + r_2 \times r_3 + \dots + r_{n-1} \times r_n + r_n \times r_1$. The area $AUC_{generalized}$ is obtained as follows: $AUC_{generalized} \propto r_1 + r_2 + \dots + r_n$.

Let us take a small change δ in r_1 that is compensated by r_2 . The area $AUC'_{generalized}$ is obtained as follows:

$$\begin{aligned} AUC'_{generalized} &\propto (r_1 - \delta) + (r_2 + \delta) + \dots + r_n \\ &\propto r_1 + r_2 + \dots + r_n \quad \propto AUC_{generalized} \end{aligned} \quad (6)$$

The area γ' in this case is

$$\begin{aligned} \gamma' &\propto (r_1 - \delta) \times (r_2 + \delta) + (r_2 + \delta) \times r_3 + \dots + r_{n-1} \times r_n + r_n \times (r_1 - \delta) \quad (7) \\ &\propto r_1 \times r_2 + \delta(r_1 - r_2) - \delta^2 + r_3 \times \delta + r_2 \times r_3 + \dots + r_{n-1} \times r_n \\ &\quad + r_n \times r_1 - \delta \times r_n \end{aligned}$$

We see that the generalized AUC has not been affected through the changes in AUC values as mentioned above. However, in our method these changes have the effect of changing the area (since, $\gamma \neq \gamma'$) which is $-\{\delta \times (r_1 - r_2) + \delta^2 + \delta \times r_3 - \delta r_n\}$. This quality does not have to be zero and does allow us to find differences in the performance of two classifiers unlike the generalized AUC[3].

Complexity of the Proposed Method

The overall complexity of computing the scalar measurement for a multi-class ROC consists of the complexity of computing AUC for all pairs of classes. For a dataset D of n examples (points) the algorithm would need an $O(n \log n)$ computation to compute AUC for a pair of classes. For M classes there will be a total ${}^M C_2$ pairs of classes. Therefore, ${}^M C_2$ pairwise AUC values will require $O({}^M C_2 \log({}^M C_2))$ computation to arrange the AUCs for the calculation of a maximum area. Thus the total complexity is $O({}^M C_2 n \log n + {}^M C_2 \log({}^M C_2))$.

5 Experimental Design and Data Sets

Initially we have generated a synthetic dataset by randomly generating numbers. These numbers are generated following normal distribution where the mean and standard deviations were varied to generate a three class dataset. This dataset

consists of three classes where each of the classes has 100 data instances. We classified the dataset using four classifiers: Random Forest, Support Vector Machine (SVM), Random Tree and Decision Tree (C4.5). We have also classified the Segment dataset [8] using above classifiers. There are 2310 data instances and 7 classes in this dataset.

6 Results and Discussion

Results: Figure 5 shows the area covered by the maximum AUC values obtained for each of the pairs of classes. For this dataset, we obtained the performance using a generalized AUC as 0.92833, while our method produces a scalar value of 1.1188. Table 3 lists the pairwise AUC values for Segment dataset for each of the classifiers and provides the generalized AUC and AUC calculated using our method. Fig. 7 compares the performances of four classifiers (SVM, Random Forest, Random Tree and C4.5) for the segment dataset.

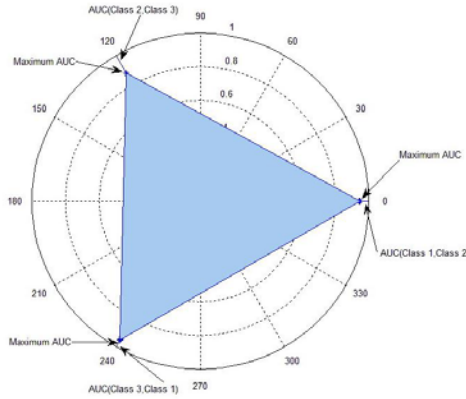


Fig. 5. The total area that signifies the performance of the classifier

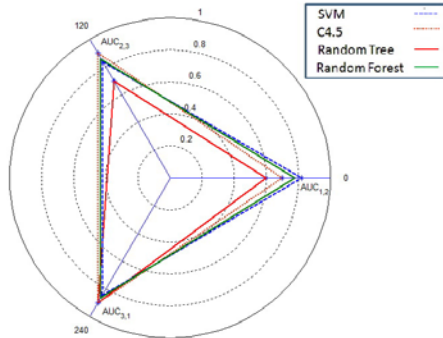


Fig. 6. Comparison of classifiers’ performances using visual representation for randomly generated 3-class dataset

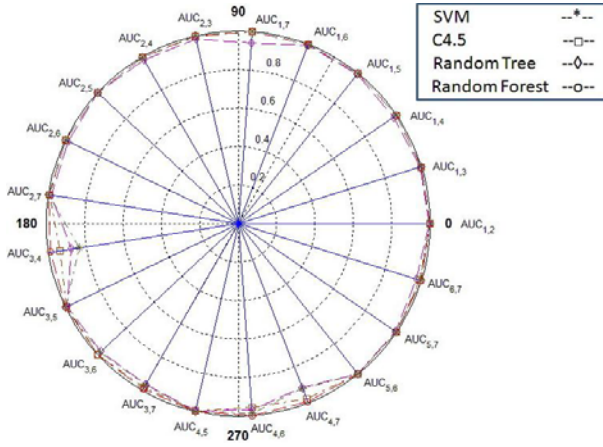


Fig. 7. Comparison of classifiers’ performances using visual representation for Segment dataset

Discussion: The experimental results show that our method can measure the performance of a classifier while classifying more than two classes. As shown in Tab. 2, for the synthetically generated dataset, the performance measurement using a generalized AUC cannot differentiate the performances between SVM and Random Forest. It is interesting that our method can rank these classifiers according to their performances. Fig. 6 also represents the performance variation visually.

Table. 3 presents the pair-wise AUC values for the four classifiers: SVM, Random Forest, Random Tree and C4.5 when classifying the segment dataset. The overall performance using our method for these classifiers is: SVM: 3.00, C4.5: 3.04, Random Tree: 2.95 and Random Forest: 3.06. The performances using generalized AUC are: SVM: 0.98, C4.5: 0.99, Random Tree: 0.98 and Random Forest: 0.99. We see that the generalized AUC [2] cannot differentiate the performances of SVM and Random Tree, while our method can distinguish between the performances of these two classifiers. As indicated by the area, C4.5 performs better than does SVM for the dataset. Similarly, we find that amongst the four classifiers, Random Forest performs the best. According to the area obtained using our method the rank of the classifiers is (from the best to the worst): Random Forest, C4.5, SVM and Random Tree.

Fig. 7 represents the visual comparison of the performances of four classifiers (SVM, Random Forest, Random Tree and C4.5) for the segment dataset that has seven classes. The plot shows that the SVM performs worse than other classifiers when classifying between class 3 and class 4. The classification performances for both Random Tree and SVM are the same for most of the pairs of classes except at a few points (e.g. for the pair class 1 versus class 7, where Random Tree is worse than SVM; and for the pair class 3 versus class 4, where Random Tree performs better than SVM). This plot reveals that the performance of Random Forest is the best among the all classifiers.

Table 1. The minimum and maximum values of areas for a different number of classes

| # of Classes | Minimum Area | Maximum Area |
|--------------|--------------|--------------|
| 3 | 0.3248 | 1.2990 |
| 4 | 0.6495 | 2.5981 |
| 5 | 0.7347 | 2.9389 |
| 6 | 0.7626 | 3.0505 |
| 7 | 0.7737 | 3.0949 |
| 8 | 0.7788 | 3.1153 |
| 9 | 0.7814 | 3.1257 |
| 10 | 0.7828 | 3.1314 |
| 11 | 0.7837 | 3.1348 |
| 12 | 0.7842 | 3.1368 |
| 13 | 0.7845 | 3.1382 |
| 14 | 0.7848 | 3.1391 |
| 15 | 0.7849 | 3.1397 |
| 16 | 0.7850 | 3.1402 |

Table 2. The comparison of performances measurement between our method and Generalized AUC [2] using synthetic dataset

| Class X Vs. Class Y | SVM | C4.5 | Rand Tree | RF |
|---------------------|-------|-------|-----------|-------|
| X=1, Y=2 | 0.82 | 0.70 | 0.60 | 0.78 |
| X=1, Y=3 | 0.84 | 0.90 | 0.7 | 0.86 |
| X=2, Y=3 | 0.85 | 0.90 | 0.90 | 0.87 |
| Generalized AUC [2] | 0.837 | 0.833 | 0.733 | 0.837 |
| Our method | 0.908 | 0.896 | 0.689 | 0.909 |

Table 3. The comparison of performances measurement between our method and Generalized AUC [2] using segment dataset [8]

| Class X vs. Class Y | SVM | C4.5 | Rand Tree | RF |
|---------------------|-------|-------|-----------|-------|
| X=1, Y=2 | 1 | 0.998 | 0.994 | 1 |
| X=1, Y=3 | 1 | 0.998 | 0.993 | 1 |
| X=1, Y=4 | 0.998 | 1 | 0.981 | 1 |
| X=1, Y=5 | 1 | 0.998 | 0.993 | 1 |
| X=1, Y=6 | 1 | 1 | 0.995 | 1 |
| X=1, Y=7 | 1 | 0.999 | 0.942 | 1 |
| X=2, Y=3 | 1 | 0.998 | 0.985 | 1 |
| X=2, Y=4 | 1 | 1 | 0.981 | 1 |
| X=2, Y=5 | 1 | 0.998 | 1 | 1 |
| X=2, Y=6 | 1 | 1 | 0.985 | 1 |
| X=2, Y=7 | 1 | 0.995 | 0.984 | 1 |
| X=3, Y=4 | 0.837 | 0.941 | 0.881 | 0.97 |
| X=3, Y=5 | 1 | 0.995 | 0.991 | 1 |
| X=3, Y=6 | 0.983 | 0.998 | 0.978 | 1 |
| X=3, Y=7 | 0.967 | 0.984 | 0.965 | 0.95 |
| X=4, Y=5 | 1 | 0.998 | 0.998 | 1 |
| X=4, Y=6 | 0.988 | 0.96 | 0.971 | 0.990 |
| X=4, Y=7 | 0.92 | 0.977 | 0.914 | 0.994 |
| X=5, Y=6 | 1 | 0.998 | 1 | 1 |
| X=5, Y=7 | 1 | 0.998 | 0.993 | 1 |
| X=6, Y=7 | 0.995 | 0.99 | 0.974 | 0.990 |
| Generalized AUC [2] | 0.98 | 0.99 | 0.98 | 0.99 |
| Our method | 3.00 | 3.04 | 2.95 | 3.06 |

7 Conclusion

In this paper, we have proposed a new method to represent performance of a classifier both visually and numerically when there are more than two classes in dataset. Using the proposed method, classifiers can be compared and ranked according to their cost performance. The method considers optimal configuration of pairwise AUC values of a classifier providing more accurate performance of the classifier. Furthermore, the computational cost of the method is $O(M C_2 n \log n + M C_2 \log(M C_2))$ that is far less than $O(n^d)$; $d = M.(M - 1)$ (if we consider d -dimensions to plot and compute the volume under ROC surface) for a dataset of M classes and n datapoints. This method is a milestone to fill up the limitation of the existing AUC and ROC plot for binary class problem.

References

1. Flach, P., Blockeel, H., Ferri, C., Hernández-Orallo, J.S.: Decision support for data mining; introduction to roc analysis and its applications. *Data Mining and Decision Support: Integration and Collaboration*, 81–90 (2003)
2. Hand, D.J., Till, R.J.: A simple generalisation of the area under the roc curve for multiple class classification problems. *Machine Learning* 45, 171–186 (2001)
3. Swets, J., Dawes, R., Monahan, J.: Better decisions through science. *Scientific American*, 82–87 (2000)
4. Ferri, C., Hernández-Orallo, J., Salido, M.: Volume under the roc surface for multi-class problems. In: *Proceedings of ECML* (2003)
5. Srinivasan, A.: Note on the location of optimal classifiers in roc space. Technical report, Oxford University Technical Report PRG-TR-2-99 (1999)
6. Li, Y.: A generalization of AUC to an ordered multi-class diagnosis and application to longitudinal data analysis on intellectual outcome in pediatric brain-tumor patient. PhD thesis, Georgia University (2009)
7. Mossman, D.: Three-way roc's. *Medical Decision Making* 19, 78–89 (1999)
8. Asuncion, A., Newman, D.: UCI machine learning repository (2007)

Efficiently Finding the Best Parameter for the Emerging Pattern-Based Classifier PCL^{*}

Thanh-Son Ngo¹, Mengling Feng², Guimei Liu¹, and Limsoon Wong¹

¹ National University of Singapore
{ngothanh, liugm, wongls}@comp.nus.edu.sg

² Institute for Infocomm Research
mfeng@i2r.a-star.edu.sg

Abstract. Emerging patterns are itemsets whose frequencies change sharply from one class to the other. PCL is an example of efficient classification algorithms that leverage the prediction power of emerging patterns. It first selects the top-K emerging patterns of each class that match a testing instance, and then uses these selected patterns to decide the class label of the testing instance. We study the impact of the parameter K on the accuracy of PCL. We have observed that in many cases, the value of K is critical to the performance of PCL. This motivates us to develop an algorithm to find the best value of K for PCL. Our results show that finding the best K can improve the accuracy of PCL greatly, and employing incremental frequent itemset maintenance techniques reduces the running time of our algorithm significantly.

1 Introduction

Classification is the task of learning from one data set and making predictions in another data set. The learning data is called training data which consists of entities with their labels. The other data set is called testing data which consists of entities without labels, and the classifying task is to make prediction of their labels based on what we learn from the training data. Many classifiers have been proposed in the literature. Here, we focus on pattern-based classifiers.

Frequent patterns are patterns that appear frequently in the dataset. Frequent patterns whose supports in the training data set change significantly from one class to the other are used to construct the classifiers. These patterns are called emerging patterns (EP) [9,10,16] and these patterns are applied to testing instances to predict their class memberships. As a classifier may generate many rules for the same class, aggregating their discrimination power gives better prediction results [4,8,10,20]. Using EPs has the advantage that they not only predict the class labels but also provide explanation for the decision.

Jumping emerging patterns (JEP) are a special type of EPs that have non-zero occurrence in one class and zero occurrence in all other classes [10,11]. PCL [9,10] is a classifier based on aggregating the prediction power of frequent JEPs. Given

^{*} Supported by A*STAR grant (SERC 072 101 0016).

a test instance t , PCL selects the top-K JEPs with the highest supports that are contained in t from each class and then computes the score for each class using these selected patterns. PCL assigns the class label with the highest score to the testing instance. The choice of a good value for K is tricky and its optimal value varies on different datasets, as shown in Fig. 1. Remarkably, the problem of choosing the best value of K has not been investigated previously.

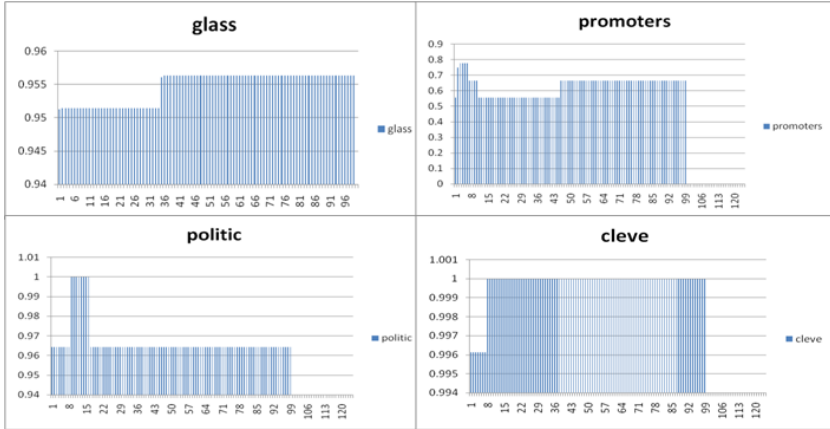


Fig. 1. Accuracy of PCL on four datasets with different values of K. X-axis shows different values of K and Y-axis shows the accuracy. When K increases, the accuracy does not always increase.

Here, to avoid overfitting K to the data, we sample many subsets of the training data to get the value of K that appears the best on average. By this method, we maximize the likelihood that the chosen K will produce the best results in the whole dataset. Our main contributions are summarized as follows: (i) We revisit the PCL algorithm [9,10] and propose a method to find the most appropriate parameters to improve the performance of PCL. (ii) We introduce a method to speed up the proposed algorithm as well as cross-validation methodology for frequent-pattern-based classification algorithms in general.

2 Related Work

Emerging patterns are a special type of frequent patterns. They occur frequently in one class but rarely in other classes. Emerging pattern-based classification benefits enormously from the advancement of frequent pattern mining algorithms and better understanding of the patterns space. Many methods have been proposed to select a good set of patterns for constructing classifiers [15]. Most of these algorithms first generate frequent patterns satisfying a certain minimum support constraint, and then select patterns based on some criteria and make

predictions based on the selected patterns. These algorithms mainly differ in how patterns are selected and how class labels are determined.

The first class of methods pick the best pattern to classify testing instances, like CBA [14] and MMAC [18]. CBA first derives classification rules from frequent patterns, and then ranks the rules in descending order of their confidence. A very small set of rules is then selected to maximize the classification accuracy on the training data. The class label of a new instance is decided by the first rule that matches the instance. MMAC extends CBA to multiple classes and labels. Only a small set of patterns are selected for classification. So it is possible that a new testing instance does not contain any EP that is selected.

The second class of methods treats frequent patterns as additional features and then use classical algorithms to classify the data [2,3]. Cheng et al. [2,3] build a connection between pattern frequency and discriminative measures such as information gain score, and develop a strategy to set minimum support in frequent pattern mining for generating useful patterns for classification. Based on this strategy, coupled with a proposed feature selection algorithm, selected EPs are used as additional features to build high quality classifiers. The results show that using EPs as additional features can improve the accuracy of C4.5 and SVM. The main drawback of this approach is that the intuitiveness of the pattern-based classifiers may be lost.

The last class of methods selects the top-K patterns and aggregates the predictive power of the selected patterns. They include CAEP [4], iCAEP [20], PCL [9,10], CEP [1], CPAR [17], CMAR [8] and HARMONY [19]. CAEP uses EPs with high growth rate to do classification. iCAEP aggregates the prediction power of EPs based on information theory. PCL uses only JEPs. JEPs occur in one and only one class, which may be too restrictive in some cases. CEP relaxes this constraint by putting an upper bound on the number of occurrences of the EPs in other classes. CPAR uses the expected accuracy to select classification rules. It compares the average expected accuracy of the best K rules of each class and chooses the class with the highest expected accuracy as the predicted class. CMAR uses a weighted measure to select rules, and the score of a class is also calculated using this weighted measure. HARMONY directly mines the final set of classification rules by using an instance-centric rule generation approach.

The aggregation-based algorithms generally show better performance than other algorithms. However, the value of K is critical to their performance in many cases. Here, we use PCL as an example to study how to select proper values for parameter K to maximize the accuracy of the aggregation-based algorithms.

3 Preliminaries

Let $I = \{i_1, i_2, \dots, i_n\}$ be a set of distinct literals called items. An itemset or a pattern is a subset of I . A transaction is a non-empty set of items. Let $C = \{C_1, C_2, \dots, C_k\}$ be a set of distinct labels called class labels. A transactional database consists of a set of transactions associated with their labels. A classification task involves two phases: training and testing. In training, the class

labels are revealed and in testing, class labels are hidden. The classifier builds a model based on the training set and uses this model to predict the class labels of transactions in the testing set. The accuracy of a classifier A is the proportion of test-instances which are correctly classified by A .

A pattern P covers a transaction t if $P \subseteq t$. The support of P is the number of transactions that are covered by P . We use $sup(P, D)$ to indicate the support of P in the dataset D . We use $P|D$ to indicate the set of transactions in D that contain P . Given a 2-class dataset, jumping emerging patterns (JEP) are patterns whose frequency in one class is non-zero and in other class is zero. An EP_i is called a JEP from class A to class B if its support in class A is zero. A pattern P is a generator if and only if for every $P' \subset P$, $sup(P', D) > sup(P, D)$. A JEP generator is both a generator and a JEP.

4 Algorithms

4.1 PCL

We present here an overview of PCL [9,10]. The dataset D is divided into positive and negative classes. Given a test-instance t , two sets of JEPs that cover t are used: $EP_{t1}^+, EP_{t2}^+, \dots, EP_{tn}^+$ from negative to positive classes and $EP_{t1}^-, EP_{t2}^-, \dots, EP_{tn}^-$ from positive to negative classes. The K most frequent JEPs that cover t are sorted in descending order of their supports. Suppose the set of JEPs (based on the training set) from negative to positive classes are $EP_1^+, EP_2^+, \dots, EP_n^+$ in descending order of their supports. Similarly, $EP_1^-, EP_2^-, \dots, EP_n^-$ is the set of JEPs from positive to negative. It is hypothesized that if t belongs to one class, it should contain more JEPs of this class than the other class. So Li and Wong [9] formulated the scores of t with respect to the two classes as below.

$$Score(t, +) = \frac{\sum_{i=1}^k sup(EP_{ti}^+, D)}{\sum_{i=1}^k sup(EP_i^+, D)} \quad Score(t, -) = \frac{\sum_{i=1}^k sup(EP_{ti}^-, D)}{\sum_{i=1}^k sup(EP_i^-, D)}$$

4.2 PSM

Maintenance of EPs was first introduced in [11], though a complete solution for insertion and deletion was not discussed. Here, we describe a more efficient method called PSM for complete maintenance introduced recently in [5].

PSM is a maintenance algorithm for frequent pattern space. It is based on the GE-tree, an effective data structure described in [6,7] for enumerating generators. Frequent generators are stored in the GE-tree and the tree is incrementally adjusted when new transactions are added or existing transactions are removed. Each node in the tree represents a generator. To find EPs, we modify the GE-tree so that each node stores both positive and negative supports of the corresponding generator. In addition to frequent generators, GE-tree maintains a negative border which comprises infrequent generators whose immediate subsets are all

frequent [5,6]. The negative border helps generate new frequent generators and equivalence classes efficiently when transactions are added or removed.

Computating frequent generators is expensive. The benefit of PSM is that a new set of frequent generators does not need to be computed from scratch when the data is modified. As a small change in dataset seldom causes a big change in the set of frequent patterns, PSM is effective for pattern space maintenance.

4.3 rPCL and ePCL

A good choice of K has a big impact on prediction results in PCL. We propose a method to tackle this problem as follows. According to the Central Limit Theorem, the distribution of accuracy will behave like normal distribution. Indeed, Fig. 2 suggests the convergence of average classification accuracy in training data to the real value of accuracy in the whole dataset. So we simulate the actual process of classification in training set and choose the value of K that maximizes the mean accuracy. The simulation is run repeatedly to determine which value of K appears as the best on average. By the Central Limit Theorem, the average accuracy of each K approaches the true accuracy of that value of K , given sufficient number of simulation runs. Thus, the value of K that has the best mean accuracy in the simulation runs will also perform well in the whole set of data.

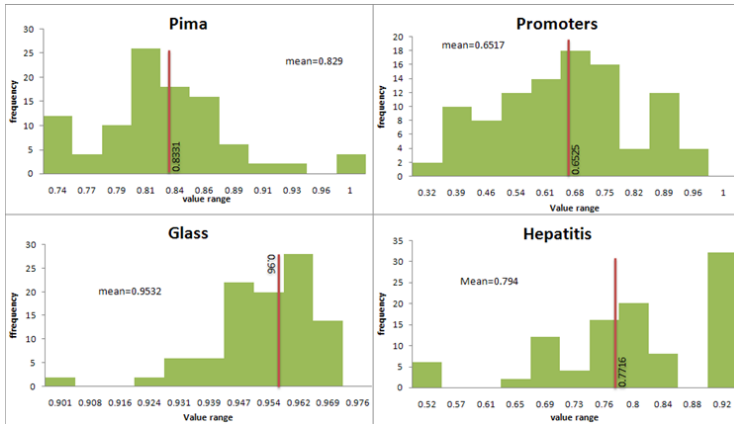


Fig. 2. Distribution of accuracies across 100 runs given a fixed $K=10$ over training set. The vertical red line indicates the actual value of accuracy when we evaluate the same K in the whole dataset. The means are shown to be close to the actual accuracy.

We describe two algorithms rPCL and ePCL. Algorithm rPCL is a direct but naive solution for the method above. We use it as a benchmark to understand where inefficiencies might be and to assess the improvement of a more efficient method using maintenance. ePCL is the fast version where PSM is used. It produces the same results as rPCL but runs many times faster.

rPCL

Input: training set D , maxtime , maxK .
Output: best K

```

1. Initialize the array  $a[i]=0$  for  $i=1..\text{maxK}$ 
2. For ( $j=0$  to  $\text{maxtime}$ ) do
3.   Randomly select 1/10 of database as testing ( $D_t$ ), the remaining is used as training ( $D_n$ )
4.   Construct a set of frequent patterns from  $D_n$ 
5.   Build a classifier from  $D_n$ 
6.   For ( $K=1$  to  $\text{maxK}$ ) do
7.     #success=0
8.     For (each instance  $t$  in  $D_t$ ) do
9.       compute  $\text{Score}_K[t, +]$ ,  $\text{Score}_K[t, -]$  wrt  $K$ 
10.      Decision= $\text{makeDecision}(\text{Score}_K[t, +], \text{Score}_K[t, -])$ 
11.      If (decision is correct) then #success++
12.    endfor
13.    accuracy= #success / | $D_t$ |
14.     $a[K] = a[K] + \text{accuracy}$ 
15.  endfor
16. endFor
17.  $K = \text{argmax}(a[i]), i=1..\text{maxK}$ 
18. Return  $K$ 

```

Fig. 3. The rPCL algorithm

In rPCL (Fig. 3) we use a technique called repeated random sub-sampling validation as a framework to assess the best parameter for PCL. It involves several rounds, in each round the training set is randomly divided into new complementary testing and training set. For each K , we perform this procedure to determine the expected accuracy of the classifier wrt this K . The chosen K is the one which returns the best average accuracy over these runs. 10-fold cross validation is popular for accessing classifier performance. So, we subsample 10% of training set as testing subsample and 90% as training subsample.

In rPCL, makeDecision simply decides if the classifier with $\text{Score}_K[t, +]$ and $\text{Score}_K[t, -]$ correctly predicts the class of t , wrt to the current K . At step 9, we can compute $\text{Score}_K[t, +]$ from $\text{Score}_{(K-1)}[t, +]$ in constant time. To achieve this, we use a vector to store $\text{Score}_K[t, +]$ and $\text{Score}_K[t, -]$ for all $t \in D_t$. To determine a good value of K , a significantly large number of iterations maxtime is performed and all values of K in the range $1..\text{maxK}$ are considered.

Constructing a set of frequent EPs from the training set (step 4) is computationally very expensive. It involves frequent pattern mining. And this step is repeated many times with different D_t and D_n . Because the testing-training separations are largely similar among all the runs, we should not need to construct the set of EPs from scratch in such a naive fashion. PSM [5,6] is used to achieve this purpose. PSM allows us to adjust the set of frequent patterns when the training fold is changed. We only need to construct the frequent patterns space at the beginning and repeatedly use PSM to maintain the set of rules when the sub-sampling folds are changed. That is, we only need to mine frequent patterns once at the beginning. With this innovation, we present an improved algorithm called ePCL (Fig. 4) which stands for enhanced PCL.

ePCL

Input: training set D , maxtime , maxK
 Output: best K

1. **Construct a frequent patterns maintenance FG which consists of frequent generators from D**
2. Initialize the array $a[i]=0$ for $i=1..\text{maxK}$
3. For ($i=0$ to maxtime) do
4. Randomly select 1/10 of database as testing (D_t), the remaining is used as training (D_n)
5. **PSM.delete(D_t)**
6. Build a classifier from D_n
7. For ($K=1$ to maxK) do
8. #success=0
9. For (each instance t in D_t) do
10. compute $\text{Score}_K[t, +]$, $\text{Score}_K[t, -]$ wrt K
11. Decision=**makeDecision**($\text{Score}_K[t, +]$, $\text{Score}_K[t, -]$)
12. If (decision is correct) then #success++
13. **endfor**
14. accuracy= #success / $|D_t|$
15. $a[K] = a[K] + \text{accuracy}$
16. **endfor**
17. **PSM.add(D_t)**
18. **endfor**
19. $K = \text{argmax}(a[i], i=1..\text{maxK})$
20. Return K

Fig. 4. ePCL algorithms. The highlighted codes indicate the involvement of PSM.

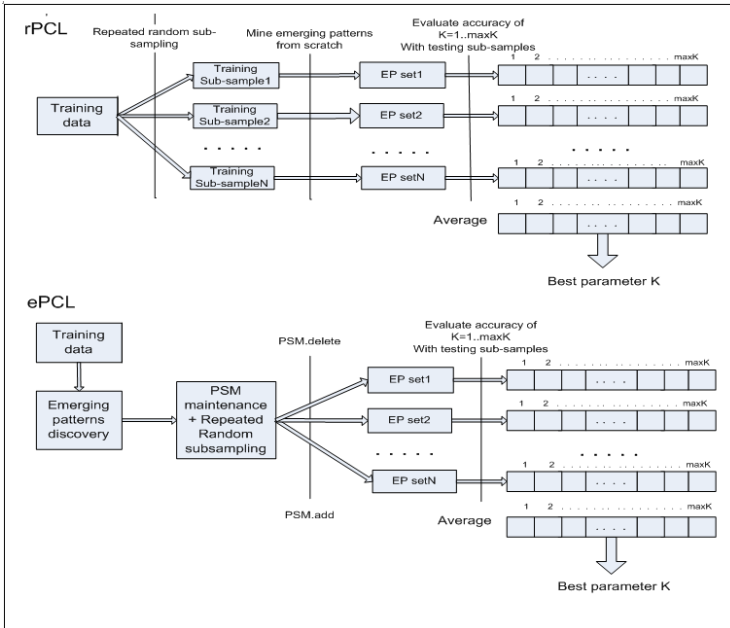


Fig. 5. Workflow for rPCL and ePCL. EP set indicates the set of EPs used to construct PCL.

The set of frequent generators are incrementally maintained. Line 5 and 17 show the execution of PSM: PSM.delete is invoked when a fold is removed from the training set and PSM.add is invoked to add this fold into original set. That eliminates building a new set of frequent patterns when we run the simulation with a new testing-training separation. Fig. 5 is the workflow of rPCL and ePCL.

4.4 Complexity Analysis

We compare the theoretical improvement of ePCL from rPCL. In rPCL, the outer loop repeats maxtime, which is the number of runs needed to find the best K . In each loop, we need to run PCL classifier which involves a frequent pattern mining step (FPM) and evaluation for all K from 1 to maxK. The total time is $maxtime * (FPM + PCL(maxK))$. $PCL(maxK)$ is the time to compute scores for $K = 1..maxK$. This step involves visiting entire frequent pattern set to get the top K patterns and compute scores accordingly for each test instance. In ePCL, we only need to build frequent patterns once and incrementally adjust the pattern space. The total time is $FPM + maxtime * (|D|/10 * maintenance + PCL(maxK))$, where maintenance is the running time for maintaining one transaction, $|D|$ is the size of the dataset. According to [5], PSM is much more efficient than mining-from-scratch algorithms. At 10% of the data size, the speed-up is at least 3 times faster for incremental and 5 times faster for decremental maintenance than mining from scratch.

5 Experimental Studies

5.1 Experiment Setup

Experiments are conducted using 18 data sets of various sizes from UCI repository. Continuous attributes are discretized by the entropy method. Table 1 gives details of these data sets. We evaluate the performance on two-class datasets. For multi-class datasets, we select one class as positive and the remaining as negative. The process is done for all classes. PCL can be extended to handle multiple-class datasets; however, the method of choosing parameter is still applicable. 10-fold cross validation is used to assess the efficiency and average accuracies are reported. The datasets are separated in such a way that the proportion of each class is consistent in both testing and training.

For the original PCL, we use the frequent pattern mining algorithm provided by [13]. The experiments are done in Windows machine with 2.6 GHz processor and 1G memory. We assess the accuracy improvement of rPCL over PCL and running time comparison of ePCL and rPCL. Time is measured in seconds.

When we compute the score of an instance in two classes, it is possible that the scores are both zero. Such a test-instance is not covered by any pattern and therefore unclassifiable by both PCL and ePCL. The average percentage of unclassified data is 11% and there is no dataset with more than 20%. We do not include these cases in our accuracy computation. The purpose is to evaluate the improvement of rPCL over the original PCL only.

Table 1. Datasets information

| Dataset | # attributes | # instances | # continuous attributes | # nomial attributes |
|-------------|--------------|-------------|-------------------------|---------------------|
| mushroom | 4 | 8124 | 4 | 0 |
| iris | 8 | 150 | 8 | 0 |
| Glass | 22 | 214 | 0 | 22 |
| zoo | 16 | 101 | 0 | 16 |
| Promoter | 18 | 106 | 0 | 18 |
| Vote | 10 | 435 | 10 | 0 |
| Splice | 19 | 3190 | 0 | 19 |
| Hepatitis | 59 | 155 | 0 | 59 |
| Pima | 61 | 768 | 0 | 61 |
| Hypothyroid | 25 | 3163 | 7 | 18 |
| Breast | 16 | 3190 | 0 | 16 |
| Cleve | 10 | 2727 | 0 | 10 |
| German | 12 | 2700 | 0 | 12 |
| Lymph | 19 | 1332 | 0 | 19 |
| Vehicle | 19 | 3809 | 0 | 19 |
| Waveform | 20 | 9000 | 0 | 20 |
| Wine | 14 | 178 | 0 | 14 |
| Tic-tac-toe | 10 | 4312 | 0 | 10 |

5.2 Parameters Setting

All the patterns used in PCL and its improved versions are JEP generators [10]. The exception is an early paper [8]; it uses the “boundary” JEPs, which are defined in [9] as those JEPs where none of their proper subsets are JEPs. Here, we use JEP generators as discriminative patterns because:

- It is known that the JEP space is partitioned into disjoint equivalence classes [9,10]. So the set of JEP generators (which correspond to the most general patterns in each equivalence class) adequately characterizes the data. In contrast, boundary JEPs are insufficient to capture all equivalence classes.
- The minimum description length principle is a well-accepted general solution for model selection problems. The choice of JEP generators is in accord with this principle [12]. Also, our experiments show that JEP generators are less noise-sensitive than boundary JEPs.
- JEP generators are efficient to compute. We can make use of many efficient frequent patterns mining algorithms [12].

For rPCL and ePCL, we set maxtime to 50 and we run K in a range from 1 to 50. We limit the range to 50 because small-frequency patterns have minimal prediction power over the top ones. Patterns ranked higher than 50 generally have low support. As shown in Fig. 1, the predictions stabilize after $K = 50$, so no need to consider $K > 50$. For original PCL, we use $K = 10$ as suggested in [9]. Minimum support is set to make sure enough EPs are found.

5.3 Efficiency

PCL with generators and boundary EPs. For PCL, we have choices over which type of frequent patterns are used to make predictions. We have done experiments to justify our choice of JEP generators, as suggested in [9], rather than boundary JEPs. We want to test the robustness of PCL in the case of noise with these two types of patterns. When a part of the original dataset is missing, the noisy dataset reduces the accuracy of the classifier. Fig. 6 shows accuracy of PCL with generators (gen PCL) and boundary JEPs (boundary PCL) for different levels of missing rate (We assume items in the dataset are missing with certain rate): 0%, 10%, 20%, 30%, 40% and 50%. The accuracies are average over 18 datasets. Gen PCL is less noise-sensitive than boundary PCL.

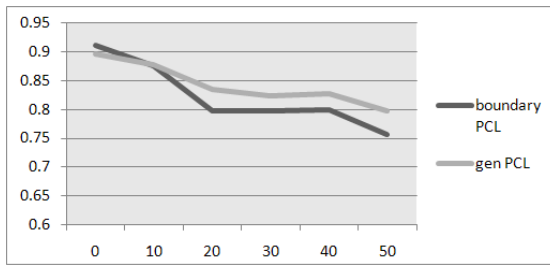


Fig. 6. Accuracy of gen PCL and boundary PCL in the presence of missing information

PCL and ePCL. We compare the accuracy of the original PCL and ePCL. Since rPCL and ePCL give the same results, we do not show the accuracy of rPCL here. Table 2 shows accuracy of ePCL and the original PCL in 18 datasets. Overall, the improvement is 3.19%. In some datasets like Promoters, Wine and Hepatitis, we get improvement of 26%, 11% and 15% respectively.

Table 2. Accuracy comparison of ePCL and PCL

| Datasets | ePCL | PCL | Improve (%) | Datasets | ePCL | PCL | Improve (%) |
|----------|---------|--------|-------------|-------------|--------|--------|-------------|
| Mushroom | 1 | 1 | 0 | Iris | 0.9999 | 0.9658 | 3.534591 |
| Glass | 0.9714 | 0.9675 | 0.404967 | Zoo | 0.9997 | 0.9528 | 4.925911 |
| Promoter | 0.8216 | 0.6525 | 25.92549 | Vote | 0.9439 | 0.9492 | -0.55892 |
| Splice | 0.6667 | 0.6667 | 0 | Hepatitis | 0.8872 | 0.7716 | 14.98068 |
| Pima | 0.8192 | 0.8331 | -1.67043 | Hypothyroid | 0.9962 | 0.9861 | 1.017527 |
| Breast | 0.9912 | 0.9907 | 0.050572 | Cleve | 0.9644 | 0.9959 | -3.16567 |
| German | 0.9644 | 0.994 | -2.97462 | Lymph | 1 | 1 | 0 |
| Vehicle | 0.9569 | 0.9437 | 1.390222 | Waveform | 0.8347 | 0.8053 | 3.643507 |
| Wine | 0.97892 | 0.8789 | 11.3941 | Tic-tac-toe | 0.9512 | 0.965 | -1.42591 |

Table 3. Accuracy comparison of ePCL and PCL in difficult cases

| Datasets | Difficult cases (%) | PCL | ePCL | Datasets | Difficult cases (%) | PCL | ePCL |
|----------|---------------------|--------|--------|-------------|---------------------|--------|--------|
| mushroom | 0 | - | - | Iris | 0 | - | - |
| Glass | 33.94 | 0.9071 | 0.9183 | zoo | 26.53 | 0.8274 | 0.9989 |
| Promoter | 49.02 | 0.3123 | 0.647 | Vote | 16.31 | 0.6978 | 0.6663 |
| Splice | 100 | 0.6667 | 0.6667 | Hepatitis | 100 | 0.7716 | 0.8872 |
| Pima | 12.10 | 0 | 0 | Hypothyroid | 20 | 0.9327 | 0.9813 |
| Breast | 10.90 | 0.917 | 0.9214 | Cleve | j1 | - | - |
| German | 10.60 | 0.9449 | 0.6745 | Lymph | 0 | - | - |
| Vehicle | 3.7453 | 0 | 0 | Waveform | j2 | - | - |
| Wine | 31 | 0.8788 | 0.9789 | Tic-tac-toe | 30.2 | 0.553 | 0.578 |

Recall in PCL, to classify one test instance, one score is computed for each class. Some instances received zero score in one class and non-zero score in the other class; thus the association rules vote unanimously to one class. We call these the easy cases and the rest are difficult cases. The value of K has a much greater impact on the difficult cases. Table 3 shows the accuracy of original PCL and ePCL in difficult cases. Although we do not show statistics for easy cases here, the accuracy is close to 100% for most easy cases. We do not show datasets with too few difficult cases since there are not enough samples to evaluate accurately.

Performance. We compare the running time of rPCL, ePCL and original PCL. The performance of ePCL compared to rPCL is good, demonstrating the superiority of using pattern maintenance. Table 4 shows that ePCL is an order of magnitude faster in many cases. In one case (vehicle.dat), rPCL could not complete after more than one day, but ePCL completed within 4 hours. Nevertheless, ePCL takes more time than PCL due to the repeating part; fortunately, this is compensated by the much better accuracy of ePCL.

Table 4. Performance comparison

| Datasets | PCL | rPCL | ePCL | Speed up (rPCL/ePCL) | Datasets | PCL | rPCL | ePCL | Speed up (rPCL/ePCL) |
|-----------|------|----------|------|----------------------|-------------|------|------|------|----------------------|
| mushroom | 2.75 | 84 | 35 | 2.4 | Iris | 2 | 99 | 3 | 33 |
| glass | 8.5 | 120 | 2 | 60 | zoo | 5 | 291 | 7 | 41.5 |
| promoters | 1 | 51 | 7 | 7.2 | Vote | 0.75 | 47 | 11 | 4.2 |
| splice | 2.5 | 129 | 4 | 32.2 | hepatitis | 0.5 | 38 | 3 | 12.6 |
| pima | 0.75 | 42 | 9 | 4.6 | hypothyroid | 2 | 105 | 43 | 2.4 |
| breast | 2.5 | 109 | 60 | 1.8 | cleve | 4.25 | 181 | 123 | 1.4 |
| german | 11 | 513 | 611 | 0.8 | lymph | 27.5 | 1224 | 512 | 2.3 |
| vehicle | 2518 | too long | 6966 | A lot | waveform | 117 | 5738 | 2507 | 2.3 |
| wine | 3.25 | 188 | 17 | 11.1 | tictactoe | 13 | 88 | 43 | 2 |

6 Discussion and Conclusion

Cross-validation is a technique to assess the performance of classification algorithms. Typically the testing set is much smaller than the training set and a classifier is built from the training set which is largely similar between runs. Our framework of maintenance can be applied to efficiently perform cross-validation for pattern-based classification. Recall PSM allows us to maintain the set of frequent generators while adding or deleting a relatively small portion of the dataset. We only need to maintain a single frequent pattern set and incrementally adjust using PSM. In addition, for more efficient computation, PSM can be extended to handle multiple additions/deletions at the same time by organizing transactions in a prefix tree.

We showed the importance of finding the top K patterns for PCL. We introduced a method to perform the task efficiently, making use of the PSM maintenance algorithm [5]. The maintenance framework can be applied to general cross-validation tasks which involve frequent update of the pattern space. Our experiments showed improvement in accuracy. However, we compromised on complexity. We hope to get a better implementation for ePCL, thus minimizing the running time to close to original PCL and implement the mentioned extensions.

References

1. Bailey, J., Manoukian, T., Ramamohanarao, K.: Classification using constrained emerging patterns. In: Dong, G., Tang, C., Wang, W. (eds.) WAIM 2003. LNCS, vol. 2762, pp. 226–237. Springer, Heidelberg (2003)
2. Cheng, H., et al.: Discriminative frequent pattern analysis for effective classification. In: Proc 23th ICDE, pp. 716–725 (2007)
3. Cheng, H., et al.: Direct discriminative pattern mining for effective classification. In: Proc 24th ICDE, pp. 169–178 (2008)
4. Dong, G., et al.: CAEP: Classification by Aggregating Emerging Patterns. In: Proc. 2nd Intl. Conf. on Discovery Science, pp. 30–42 (1999)
5. Feng, M.: Frequent Pattern Maintenance: Theories and Algorithms. PhD thesis, Nanyang Technological University (2009)
6. Feng, M., et al.: Negative generator border for effective pattern maintenance. In: Proc 4th Intl. Conf. on Advanced Data Mining and Applications, pp. 217–228 (2008)
7. Feng, M., et al.: Evolution and maintenance of frequent pattern space when transactions are removed. In: Zhou, Z.-H., Li, H., Yang, Q. (eds.) PAKDD 2007. LNCS (LNAI), vol. 4426, pp. 489–497. Springer, Heidelberg (2007)
8. Li, W., Han, J., Pei, J.: CMAR: Accurate and efficient classification based on multiple class-association rules. In: Proc 1st ICDM, pp. 369–376 (2001)
9. Li, J., Wong, L.: Solving the fragmentation problem of decision trees by discovering boundary emerging patterns. In: Proc. 2nd ICDM, pp. 653–656 (2002)
10. Li, J., Wong, L.: Structural geography of the space of emerging patterns. *Intelligent Data Analysis* 9(6), 567–588 (2005)


11. Li, J., Ramamohanarao, K., Dong, G.: The space of jumping emerging patterns and its incremental maintenance algorithms. In: Proc. of 17th ICML, pp. 551–558 (2000)
12. Li, J., et al.: Minimum description length principle: Generators are preferable to closed patterns. In: Proc. 21st Natl. Conf. on Artificial Intelligence, pp. 409–415 (2006)
13. Li, H., et al.: Relative risk and odds ratio: A data mining perspective. In: Proc 24th PODS, pp. 368–377 (2005)
14. Liu, B., Hsu, W., Ma, Y.: Integrating classification and association rule mining. In: Proc. 4th KDD, pp. 80–86 (1998)
15. Ramamohanarao, K., Fan, H.: Patterns based classifiers. In: Proc. 16th WWW, pp. 71–83 (2007)
16. Ramamohanarao, K., Bailey, J.: Discovery of emerging patterns and their use in classification. In: Gedeon, T(T.) D., Fung, L.C.C. (eds.) AI 2003. LNCS (LNAI), vol. 2903, pp. 1–12. Springer, Heidelberg (2003)
17. Thabtah, F.A., Cowling, P., Peng, Y.: MMAC: A new Multi-class Multi-label Associative Classification approach. In: Proc. 4th ICDM, pp. 217–224 (2004)
18. Yin, X., Han, J.: CPAR: Classification based on Predictive Association Rules. In: Proc. 3rd SDM, pp. 331–335 (2003)
19. Wang, J., Karypis, G.: HARMONY: Efficiently mining the best rules for classification. In: Proc. 5th SDM, pp. 205–216 (2005)
20. Zhang, X., Dong, G., Ramamohanarao, K.: Information-based classification by aggregating emerging patterns. In: Proc. 2nd Intl. Conf. on Intelligent Data Engineering and Automated Learning, pp. 48–53 (2000)

Rough Margin Based Core Vector Machine

Gang Niu¹, Bo Dai², Lin Shang¹, and Yangsheng Ji¹

¹ State Key Laboratory for Novel Software Technology,
Nanjing University, Nanjing 210093, P.R.China
{niugang, jiyangsheng, lshang}@ai.nju.edu.cn

² NLPR/LIAMA, Institute of Automation, Chinese Academy of Science,
Beijing 100190, P.R.China
bdai@nlpr.ia.ac.cn

Abstract.  The recently proposed rough margin based support vector machine (RMSVM) could tackle the overfitting problem due to outliers effectively with the help of rough margins. However, the standard solvers for them are time consuming and not feasible for large datasets. On the other hand, the core vector machine (CVM) is an optimization technique based on the minimum enclosing ball that can scale up an SVM to handle very large datasets. While the 2-norm error used in the CVM might make it theoretically less robust against outliers, the rough margin could make up this deficiency. Therefore we propose our rough margin based core vector machine algorithms. Experimental results show that our algorithms hold the generalization performance almost as good as the RMSVM on large scale datasets and improve the accuracy of the CVM significantly on extremely noisy datasets, whilst cost much less computational resources and are often faster than the CVM.

1 Introduction

People in computer science societies have been questing for faster algorithms since long before. When come to mind the solving techniques of SVMs, there are several approaches ranged from the chunking method [\[1\]](#) to the sequential minimal optimization [\[2\]](#), as well as scaling down the training data and low-rank kernel matrix approximations [\[3\]](#). Eventually, the *Core Vector Machine* (CVM) algorithms [\[4, 5, 6, 7\]](#) have gone to an extreme that they have linear asymptotic time complexity and constant asymptotic space complexity, since they transform the *quadratic programming* (QP) involved in SVMs to the *minimum enclosing ball* (MEB) problems. In order to perform this transformation the CVM takes the 2-norm error, which may cause it less robust and thus hurt the accuracy. Fortunately the notion of the rough margin in the *Rough Margin based Support Vector Machine* (RMSVM) [\[8\]](#) could make SVMs less sensitive to noises and outliers, and consequently reduce the negative effects of outliers. For this reason we propose our *Rough Margin based Core Vector Machine* (RMCVM), which unites the merits of the two aforementioned methods.

¹ Supported by National Natural Science Foundation of China (No. 60775046) and Natural Science Foundation of Jiangsu Province (No. BK2009233).

After brief introductions to the CVM and the RMSVM in Section 2 and 3, we will first of all define the primal problem of the RMCVM. Next we shall elaborate how to solve an RMCVM through the approximate MEB finding algorithm. We will also investigate the loss functions used by the RMSVM and RMCVM. In the end experimental results are shown in Section 5.

2 Core Vector Machine with Minimum Enclosing Ball

Given a set of points $S = \{\mathbf{x}_1, \dots, \mathbf{x}_m\}$, where $\mathbf{x}_i \in \mathbb{R}^d$ for some integer d , the minimum enclosing ball of S (denoted $\text{MEB}(S)$) is the smallest ball which contains all the points in S [5]. Formally, let φ be a kernel induced feature map,

$$\min_{R, \mathbf{c}} R^2 \quad \text{s.t.} \quad \|\mathbf{c} - \varphi(\mathbf{x}_i)\|^2 \leq R^2, \quad i = 1, \dots, m. \quad (1)$$

Let $B(\mathbf{c}^*, R^*)$ be the exact $\text{MEB}(S)$. Given an $\epsilon > 0$, a $(1 + \epsilon)$ -approximation of $B(\mathbf{c}^*, R^*)$ is a ball $B(\mathbf{c}, (1 + \epsilon)R)$ such that $R \leq R^*$ and $S \subset B(\mathbf{c}, (1 + \epsilon)R)$. A subset Q of S is called a core-set, if $S \subset B(\mathbf{c}, (1 + \epsilon)R)$ where $B(\mathbf{c}, R)$ is $\text{MEB}(Q)$. The approximate MEB finding algorithm [9] uses a simple iterative scheme: at the t -th iteration, Q_t is expanded by including the farthest point of S from \mathbf{c}_t , then optimize (1) to get $B(\mathbf{c}_{t+1}, R_{t+1})$; this is repeated until $S \subset B(\mathbf{c}_t, (1 + \epsilon)R_t)$. A surprising property is that the number of iterations, and thus the size of the final core-set, depend only on ϵ but not on d or m [9, 5].

The dual of (1) is $\max_{\alpha} \alpha^\top \text{diag}(\mathbf{K}) - \alpha^\top \mathbf{K} \alpha$ s.t. $\alpha \geq \mathbf{0}, \alpha^\top \mathbf{1} = 1$, where α is the Lagrange multiplier and \mathbf{K} is the kernel matrix. Conversely, any QP of this form can be regarded as an MEB problem [4]. In particular when

$$k(\mathbf{x}, \mathbf{x}) = \kappa, \quad (2)$$

where κ is a constant (this is true for many popular kernels), we can drop the linear term $\alpha^\top \text{diag}(\mathbf{K})$ and obtain a simpler QP,

$$\max_{\alpha} -\alpha^\top \mathbf{K} \alpha \quad \text{s.t.} \quad \alpha \geq \mathbf{0}, \alpha^\top \mathbf{1} = 1. \quad (3)$$

Definition 1 (CVM [4])

$$\min_{\mathbf{w}, b, \xi, \rho} \|\mathbf{w}\|^2 + b^2 - 2\rho + C \sum_{i=1}^m \xi_i^2 \quad \text{s.t.} \quad y_i(\mathbf{w}^\top \varphi(\mathbf{x}_i) + b) \geq \rho - \xi_i, \quad i = 1, \dots, m, \quad (4)$$

where C is a regularization parameter and ξ_i are slack variables.

The dual of (4) is analogous to the dual (3), in which \mathbf{K} is replaced with $\tilde{\mathbf{K}}$ that $\tilde{\mathbf{K}}_{ij} = \tilde{\varphi}(\mathbf{x}_i, y_i)^\top \tilde{\varphi}(\mathbf{x}_j, y_j)$ where $\tilde{\varphi}(\mathbf{x}_i, y_i) = [y_i \varphi(\mathbf{x}_i)^\top, y_i, \frac{1}{\sqrt{C}} \mathbf{e}_i^\top]^\top$ (\mathbf{e}_i is all 0 except that the i -th component is 1). Hence the CVM is an MEB if (2) is true. To deal with the situation that (2) is not satisfied, Tsang et al. extend the MEB to the center-constrained MEB [10], and propose the generalized core vector machine [11] which is applicable for any kernel and can also be applied to kernel methods such as SVR and ranking SVM.

Note that the 2-norm error is used here. It could be less robust in the presence of outliers in theory to some extent [5].

3 Rough Margin Based Support Vector Machine

The rough set theory, which is based on the concept of the lower and upper approximation of a set, is a mathematical tool to cope with uncertainty and incompleteness [12]. The rough margins [8] are expressed as a lower margin $\frac{2\rho_l}{\|\mathbf{w}\|}$ and an upper margin $\frac{2\rho_u}{\|\mathbf{w}\|}$ where $0 \leq \rho_l \leq \rho_u$. They are corresponding with the lower and upper approximations of the outlier set, such that the samples in the lower margin are considered as outliers, the samples outside the upper margin are not outliers, and the samples between two rough margins are possibly outliers.

When the training procedure takes place, the RMSVM tries to give major penalty to samples lying within the lower margin, and give minor penalty to other samples [8]. Notice that the *Universum SVM* [13] uses a similar strategy. In practice, the RMSVM introduces slack variables ζ_i and penalize them τ times larger than ξ_i , since $\zeta_i > 0$ means that $\varphi(\mathbf{x}_i)$ is in the lower margin. Formally,

Definition 2 (RMSVM [8])

$$\begin{aligned} \min_{\mathbf{w}, b, \xi, \zeta, \rho_l, \rho_u} \quad & \frac{1}{2} \|\mathbf{w}\|^2 - \nu\rho_l - \nu\rho_u + \frac{1}{m} \sum_{i=1}^m \xi_i + \frac{\tau}{m} \sum_{i=1}^m \zeta_i \\ \text{s.t.} \quad & y_i(\mathbf{w}^\top \varphi(\mathbf{x}_i) + b) \geq \rho_u - \xi_i - \zeta_i, \\ & 0 \leq \xi_i \leq \rho_u - \rho_l, \zeta_i \geq 0, \rho_l \geq 0, \rho_u \geq 0, i = 1, \dots, m, \end{aligned} \tag{5}$$

where $\nu \in (0, 1), \tau > 1$ are regularization parameters and ξ_i, ζ_i are slack variables.

3.1 Justification of the Rough Margin

Apparently the RMSVM should encounter severer overfitting problem since it emphasizes more on outliers than the ν -SVM. However, the dual of (5) is

$$\min_{\alpha} \frac{1}{2} \alpha^\top \mathbf{Q} \alpha \quad \text{s.t. } \mathbf{y}^\top \alpha = 0, \alpha^\top \mathbf{1} \geq 2\nu, 0 \leq \alpha_i \leq \frac{\tau}{m}, i = 1, \dots, m, \tag{6}$$

where $\mathbf{Q}_{ij} = y_i y_j \varphi(\mathbf{x}_i)^\top \varphi(\mathbf{x}_j)$. Hence $2\nu/\tau$ is the fraction of samples permitted to lie within the lower margin. For fixed ν , the larger the τ is, the less overfitting the RMSVM is, and ultimately the RMSVM would become an underfitting classifier.

Likewise the loss function, which was an absent topic in [8], could justify the rough margin well. Let $f(\mathbf{x}_i) = \mathbf{w}^\top \varphi(\mathbf{x}_i) + b$,

Proposition 1. *The loss function of the RMSVM is*

$$L_1^{\rho_l, \rho_u}(\mathbf{x}_i, y_i, f) = \begin{cases} \rho_u + (\tau - 1)\rho_l - \tau y_i f(\mathbf{x}_i) & , y_i f(\mathbf{x}_i) \leq \rho_l \\ \rho_u - y_i f(\mathbf{x}_i) & , \rho_l < y_i f(\mathbf{x}_i) \leq \rho_u \\ 0 & , y_i f(\mathbf{x}_i) > \rho_u \end{cases}$$

Proof. (Sketch) In fact ξ_i increases before ζ_i , while ζ_i has to stay at zero until ξ_i arrives at $\rho_u - \rho_l$, since ξ_i suffers less penalty in (5). \square

In other words, though $\frac{1}{m} < \alpha_i \leq \frac{\tau}{m}$ when $y_i f(\mathbf{x}_i) \leq \rho_l$, ρ_l is smaller than ρ , the loss $L_1^{\rho_l, \rho_u}(\mathbf{x}_i, y_i, f)$ may still not be large, and the number of samples satisfying $y_i f(\mathbf{x}_i) \leq \rho_l$ is usually smaller than the number of samples satisfying $y_i f(\mathbf{x}_i) \leq \rho$ in the ν -SVM. Therefore the notion of the rough margin is a tool and technique to avoid the overfitting problem.

There is a side effect that ρ_u is usually larger than ρ , which makes the RMSVM generate much more support vectors satisfying $\rho_l < y_i f(\mathbf{x}_i) \leq \rho_u$ such that $\varphi(\mathbf{x}_i)$ lies between two rough margins with tiny α_i . This phenomenon slows down the speed and increases the storage of the RMSVM.

4 Rough Margin Based Core Vector Machine

For the sake of using the approximate MEB finding algorithm to solve the rough margin based SVM, we use 2-norm error because it allows a soft-margin L2-SVM to be transformed to a hard-margin one. Subsequently we have

Definition 3 (RMCVM)

$$\begin{aligned} \min_{\mathbf{w}, b, \xi, \zeta, \rho_l, \rho_u} \quad & \|\mathbf{w}\|^2 + b^2 - 2\rho_l - 2\rho_u + C \sum_{i=1}^m \xi_i^2 + \tau C \sum_{i=1}^m \zeta_i^2 \\ \text{s.t.} \quad & y_i(\mathbf{w}^\top \varphi(\mathbf{x}_i) + b) \geq \rho_u - \xi_i - \zeta_i, \quad \xi_i \leq \rho_u - \rho_l, \quad i = 1, \dots, m, \end{aligned} \tag{7}$$

where $C > 0, \tau > 1$ are regularization parameters and ξ_i, ζ_i are slack variables. The dual of (7) is

$$\begin{aligned} \max_{\alpha, \beta} \quad & -\alpha^\top \left(\mathbf{K} \circ \mathbf{y}\mathbf{y}^\top + \mathbf{y}\mathbf{y}^\top + \frac{1}{\tau C} \mathbf{I} \right) \alpha - \frac{1}{C} \|\alpha - \beta\|^2 \\ \text{s.t.} \quad & \alpha^\top \mathbf{1} = 2, \quad \beta^\top \mathbf{1} = 1, \quad \alpha \geq \mathbf{0}, \quad \beta \geq \mathbf{0}, \end{aligned} \tag{8}$$

where $\mathbf{y} = [y_1, \dots, y_m]^\top$ and the operator \circ denotes the Hadamard product.

Remark 1. We omit the constraint $\zeta_i \geq 0$ since it is dispensable for L2-SVMs. We omit the constraints $\rho_l, \rho_u \geq 0$ based on the fact that certain inequality constraints in the dual problem can be replaced by the corresponding equality constraints [14, 15]. Finally we omit $\xi_i \geq 0$, otherwise there will be $\frac{1}{C} \|\alpha - \beta + \gamma\|^2$ in the objective and additional constraint $\gamma \geq \mathbf{0}$, and the optimal $\gamma^* = \mathbf{0}$ obviously. The constraint $\rho_u \geq \rho_l$ is indeed implicated by (7) already.

Remark 2. Note that the regularization parameter of the original SVM [16] and the ν -SVM [17] is C and ν respectively. In the CVM it is C through which we control the trading off between the flatness and training errors. Since the order of $\|\mathbf{w}\|$ and ξ_i are equal in (4), their coefficients would change simultaneously under scaling, which means that the coefficient of ρ does not influence very much.

Remark 3. It is obvious that there is only ν -RMSVM insofar as it applies 1-norm error. Similarly the RMSVM using 2-norm error would be C -RMSVM inherently.

Therefore we demonstrate that *Definition 3* is proper. Furthermore,

Proposition 2. *The loss function of the RMCVM is*

$$L_2^{\rho_l, \rho_u}(\mathbf{x}_i, y_i, f) = \begin{cases} (\rho_u - \rho_l)^2 + \tau(\rho_l - y_i f(\mathbf{x}_i))^2 & , y_i f(\mathbf{x}_i) \leq \frac{\tau+1}{\tau}\rho_l - \frac{1}{\tau}\rho_u \\ \frac{\tau}{\tau+1}(\rho_u - y_i f(\mathbf{x}_i))^2 & , \frac{\tau+1}{\tau}\rho_l - \frac{1}{\tau}\rho_u < y_i f(\mathbf{x}_i) \leq \rho_u \\ 0 & , y_i f(\mathbf{x}_i) > \rho_u \end{cases}$$

Proof. (Sketch) We have $\xi_i = \tau\zeta_i$ when $0 < \xi_i < \rho_u - \rho_l$, by equalling the partial derivatives of the objective function of (7) w.r.t. ξ_i and ζ_i respectively. \square

4.1 Solving Rough Margin Based CVM

From now on we will proof that (7) can be solved approximately by the CVM.

Proposition 3. *The RMSVM cannot be transformed to an MEB problem unless we drop the group of constraints $\alpha_i \leq \frac{\tau}{m}$ in (6).*

Lemma 1. *Given a non negative vector $\alpha \in \mathbb{R}^m$, the optimal value of*

$$\min_{\beta} \|\alpha - \beta\|^2 \quad \text{s.t. } \beta^\top \mathbf{1} = \frac{1}{2}\alpha^\top \mathbf{1}, \beta \geq \mathbf{0} \tag{9}$$

is between $\frac{1}{4m}(\alpha^\top \mathbf{1})^2$ and $\frac{1}{4}\|\alpha\|^2$.

Proof. (Sketch) The upper bound is quite straightforward by setting $\beta = \frac{1}{2}\alpha$. Let $V = \{\beta : \beta^\top \mathbf{1} = \frac{1}{2}\alpha^\top \mathbf{1}\}$, then V consists of a hyperplane in \mathbb{R}^m . The lower bound is given by $\min_{\beta \in V} \|\alpha - \beta\| = \frac{|\alpha^\top \mathbf{1} - \frac{1}{2}\alpha^\top \mathbf{1}|}{\sqrt{\mathbf{1}^\top \mathbf{1}}} = \frac{|\alpha^\top \mathbf{1}|}{2\sqrt{m}}$. \square

Actually $\beta_i = 0$ iff $\cot(\alpha, \mathbf{e}_i) \geq \sqrt{2} + 1$. In other words, β is even sparser than α , which is consistent with that there are less outliers than support vectors.

Theorem 1. *The optimum of (8) is bounded by*

$$\max_{\alpha} -\alpha^\top \left(\mathbf{K} \circ \mathbf{y}\mathbf{y}^\top + \mathbf{y}\mathbf{y}^\top + \frac{\tau+4}{4\tau C} \mathbf{I} \right) \alpha \quad \text{s.t. } \alpha^\top \mathbf{1} = 2, \alpha \geq \mathbf{0}, \tag{10}$$

$$\max_{\alpha} -\alpha^\top \left(\mathbf{K} \circ \mathbf{y}\mathbf{y}^\top + \mathbf{y}\mathbf{y}^\top + \frac{\mathbf{1}\mathbf{1}^\top}{4\mathbf{1}^\top \mathbf{1} C} + \frac{1}{\tau C} \mathbf{I} \right) \alpha \quad \text{s.t. } \alpha^\top \mathbf{1} = 2, \alpha \geq \mathbf{0}. \tag{11}$$

Proof. (Sketch) Denote the objective function of (8) as $\max_{\alpha, \beta} -h(\alpha, \beta) = -\min_{\alpha, \beta} h_1(\alpha) + h_2(\alpha, \beta)$ where $h_1(\alpha) = \alpha^\top (\mathbf{K} \circ \mathbf{y}\mathbf{y}^\top + \mathbf{y}\mathbf{y}^\top + \frac{1}{\tau C} \mathbf{I}) \alpha$ and $h_2(\alpha, \beta) = \frac{1}{C} \|\alpha - \beta\|^2$. Then

$$\min_{\alpha, \beta} h(\alpha, \beta) \iff \min_{\alpha} \left(h_1(\alpha) + \min_{\beta} h_2(\alpha, \beta) \right)$$

since $h(\alpha, \beta)$ is convex. According to Lemma 1, $\min_{\beta} h_2(\alpha, \beta)$ for given α is bounded by $\frac{1}{4mC}(\alpha^\top \mathbf{1})^2$ and $\frac{1}{4C}\alpha^\top \alpha$. \square

Corollary 1. *The RMCVM can be solved approximately using the approximate MEB finding algorithm.*

Proof. Let $\mathbf{Q}^{(1)}, \varphi_1, \mathbf{Q}^{(2)}, \varphi_2$ be

$$\mathbf{Q}_{ij}^{(1)} = \varphi_1(\mathbf{x}_i, y_i)^\top \varphi_1(\mathbf{x}_j, y_j), \quad \varphi_1(\mathbf{x}_i, y_i) = \left[y_i \varphi(\mathbf{x}_i)^\top, y_i, \sqrt{\frac{\tau + 4}{4\tau C}} \mathbf{e}_i^\top \right]^\top,$$

$$\mathbf{Q}_{ij}^{(2)} = \varphi_2(\mathbf{x}_i, y_i)^\top \varphi_2(\mathbf{x}_j, y_j), \quad \varphi_2(\mathbf{x}_i, y_i) = \left[y_i \varphi(\mathbf{x}_i)^\top, y_i, \frac{1}{2\sqrt{mC}}, \frac{1}{\sqrt{\tau C}} \mathbf{e}_i^\top \right]^\top.$$

Then (10) and (11) can be regarded as MEB problems with parameters

$$R_t = \sqrt{\frac{1}{2} \boldsymbol{\alpha}^\top \text{diag}(\mathbf{Q}^{(t)}) - \frac{1}{4} \boldsymbol{\alpha}^\top \mathbf{Q}^{(t)} \boldsymbol{\alpha}}, \quad \mathbf{c}_t = \frac{1}{2} \sum_{i=1}^m \alpha_i \varphi_t(\mathbf{x}_i, y_i), \quad t = 1, 2. \quad \square$$

The convergence of (10) and (11) are as same as the CVM but we omit the proof here. Hence the approximate RMCVM algorithms based on (10) and (11) have linear asymptotic time complexity and constant asymptotic space complexity. Recall that the RMSVM is slower than the ν -SVM since it generates more support vectors. Surprisingly the RMCVM most often generates less core vectors and are faster than the CVM, even though we solve two MEB problems.

5 Experiments

We implement the RMSVM using LibSVM [18] and the RMCVM using LibCVM. The parameters are fixed to $\tau = 5$ as [8] suggested, and $\epsilon = 10^{-6}$ as [5] recommended. We tune $\nu \in \{0.1, 0.2, \dots, 0.9\}$ providing it is feasible. The kernel is Gaussian and its width is the better one computed by the default methods of LibSVM and LibCVM. The computation method of kernel width is kept unchanged over one dataset.

The notation RC_l stands for the solution $f_l(\mathbf{x}) = \sum \alpha_i y_i k(\mathbf{x}, \mathbf{x}_i) + b$ from (10) and RC_u for $f_u(\mathbf{x})$ from (11), where $b = \mathbf{y}^\top \boldsymbol{\alpha}$ respectively. We denote RC_{avg} as the average solution of them. For multi-class tasks the default one-versus-one strategy is used. The datasets² are listed in Table 1.

To begin with, we conduct experiments on a small dataset shown in Table 2. Our accuracy is almost always higher than the CVM. We find that RC_{avg} is not always the best and RC_u is usually better than RC_l . The next are results on large datasets in the top of Table 3, where the time for reading input and writing output files is excluded from the training time. Note that the CVM and the RMCVM are very fast on `extended-usps`, on which all the RMSVM fail to give a solution in 24 hours. Moreover, the RMCVM is usually better than the CVM and even beat the RMSVM once on `web`. At last we display results

² Download from <http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/> and <http://www.cse.ust.hk/~ivor/cvm.html>

Table 1. Data sets

| name | #class | #training | #testing | #feature |
|--------------------------|--------|-----------|----------|----------|
| satimage | 6 | 4,435 | 2,000 | 36 |
| web | 2 | 49,749 | 14,951 | 300 |
| ex-usps | 2 | 266,079 | 75,383 | 676 |
| intrusion | 2 | 4,898,431 | 311,029 | 127 |
| SensIT Vehicle: acoustic | 3 | 78,823 | 19,705 | 50 |
| SensIT Vehicle: seismic | 3 | 78,823 | 19,705 | 50 |

Table 2. Experimental results on satimage (accuracy (%))

| C | $\epsilon = 10^{-5}$ | | | | $\epsilon = 10^{-6}$ | | | | ν | ν -SVM | RMSVM |
|-----|----------------------|-----------------|-----------------|-------------------|----------------------|-----------------|-----------------|-------------------|-------|------------|-------|
| | CVM | RC _l | RC _u | RC _{avg} | CVM | RC _l | RC _u | RC _{avg} | | | |
| 0.1 | 82.5 | 83.5 | 84.8 | 84.1 | 82.5 | 83.5 | 84.8 | 84.1 | 0.5 | 81.9 | 88.8 |
| 1 | 85.7 | 86.9 | 87.7 | 86.8 | 85.6 | 86.9 | 87.8 | 86.9 | 0.4 | 84.8 | 89.5 |
| 10 | 89.0 | 88.3 | 88.8 | 88.3 | 88.6 | 89.2 | 89.4 | 89.3 | 0.3 | 87.2 | 89.8 |
| 100 | 89.3 | 88.3 | 84.9 | 88.4 | 89.6 | 89.8 | 90.0 | 90.1 | 0.2 | 88.9 | 89.6 |

Table 3. Experimental results on large datasets

| C | accuracy (%) | | | | | training time (seconds) | | | | |
|---------------------------------|--------------|-----------------|-----------------|-------------------|-------|-------------------------|-----------------|-----------------|-------------------|-------------------|
| | CVM | RC _l | RC _u | RC _{avg} | RMSVM | CVM | RC _l | RC _u | RC _{avg} | RMSVM |
| web | | | | | | | | | | |
| 10 | 98.6 | 98.7 | 98.8 | 98.7 | 98.9 | 423 | 218 | 79 | 295 | 134 |
| 100 | 98.9 | 99.0 | 98.9 | 99.1 | | 39 | 29 | 23 | 53 | |
| 1,000 | 96.2 | 97.1 | 97.3 | 98.2 | | 20.7 | 11.6 | 12.9 | 24.6 | |
| 10,000 | 95.9 | 96.1 | 93.0 | 97.4 | | 8.88 | 7.75 | 4.53 | 12.22 | |
| extended-usps | | | | | | | | | | |
| 100 | 99.46 | 99.50 | 99.49 | 99.49 | - | 166 | 121 | 109 | 229 | > 1 day |
| 1,000 | 99.45 | 99.47 | 99.46 | 99.47 | | 81 | 92 | 108 | 201 | |
| intrusion | | | | | | | | | | |
| 100 | 91.8 | 92.0 | 92.1 | 92.1 | - | 209 | 36 | 13 | 49 | not enough memory |
| 10,000 | 91.8 | 91.7 | 91.8 | 92.1 | | 3.61 | 3.45 | 3.58 | 6.77 | |
| 1,000,000 | 91.8 | 92.0 | 88.3 | 92.3 | | 2.41 | 1.65 | 1.61 | 2.84 | |
| SensIT Vehicle: acoustic | | | | | | | | | | |
| 100 | 50.0 | 26.7 | 31.9 | 41.9 | 66.1 | 3923 | 706 | 138 | 832 | 79434 |
| 1,000 | 39.4 | 37.1 | 52.4 | 58.4 | | 68.5 | 23.6 | 15.1 | 38.2 | |
| 10,000 | 40.9 | 42.1 | 41.0 | 46.6 | | 9.47 | 7.46 | 6.98 | 14.1 | |
| 1,000,000 | 40.6 | 50.6 | 42.4 | 47.7 | | 4.90 | 5.45 | 4.37 | 9.64 | |
| SensIT Vehicle: seismic | | | | | | | | | | |
| 100 | 50.1 | 23.2 | 23.5 | 26.2 | 64.6 | 3882 | 796 | 164 | 951 | 35194 |
| 1,000 | 28.1 | 46.5 | 59.1 | 57.3 | | 55.0 | 21.5 | 11.6 | 33.3 | |
| 10,000 | 51.2 | 41.6 | 50.3 | 55.0 | | 8.50 | 7.17 | 6.61 | 13.7 | |
| 1,000,000 | 34.6 | 46.7 | 41.1 | 45.9 | | 4.60 | 4.25 | 4.67 | 8.70 | |

on the extremely noisy **SensIT Vehicle** in the bottom of *Table 3*. Perhaps the RMCVM could always choose the right core vector and have less iteration before convergence as a result. When C is too small the RMCVM is inferior to the CVM since it is underfitting.

6 Conclusion

Motivated by the rough margin, we propose the rough margin based core vector machine and demonstrate that it can be solved efficiently. Experimental results show that the derived algorithms can handle very large datasets and the accuracy is almost comparable to the rough margin based SVM.

References

- [1] Vapnik, V.N.: Statistical Learning Theory. John Wiley & Sons, New York (1998)
- [2] Platt, J.C.: Fast training of support vector machines using sequential minimal optimization. In: *Advances in Kernel Methods*, pp. 185–208. MIT Press, Cambridge (1999)
- [3] Smola, A.J., Schölkopf, B.: Sparse greedy matrix approximation for machine learning. In: *17th ICML*, pp. 911–918 (2000)
- [4] Tsang, I.W., Kwok, J.T., Cheung, P.M.: Very large svm training using core vector machines. In: *20th AISTATS* (2005)
- [5] Tsang, I.W., Kwok, J.T., Cheung, P.M.: Core vector machines: Fast svm training on very large data sets. *JMLR* 6, 363–392 (2005)
- [6] Tsang, I.W., Kocsor, A., Kwok, J.T.: Simpler core vector machines with enclosing balls. In: *24th ICML*, pp. 911–918 (2007)
- [7] Asharaf, S., Murty, M.N., Shevade, S.K.: Multiclass core vector machine. In: *24th ICML*, pp. 41–48 (2007)
- [8] Zhang, J., Wang, Y.: A rough margin based support vector machine. *Information Sciences* 178, 2204–2214 (2008)
- [9] Bădoiu, M., Clarkson, K.L.: Optimal core-sets for balls. In: *DIMACS Workshop on Computational Geometry* (2002)
- [10] Tsang, I.W., Kwok, J.T., Lai, K.T.: Core vector regression for very large regression problems. In: *22nd ICML*, pp. 912–919 (2005)
- [11] Tsang, I.W., Kwok, J.T., Zurada, J.M.: Generalized core vector machines. *IEEE Transactions on Neural Networks* 17, 1126–1140 (2006)
- [12] Pawlak, Z.: Rough sets. *International Journal of Computer and Information Sciences* 11, 341–356 (1982)
- [13] Weston, J., Collobert, R., Sinz, F., Bottou, L., Vapnik, V.: Inference with the universum. In: *23rd ICML*, pp. 1009–1016 (2006)
- [14] Crisp, D.J., Burges, C.J.C.: A geometric interpretation of ν -svm classifiers. In: *NIPS*, vol. 12, pp. 244–250 (1999)
- [15] Chang, C.C., Lin, C.J.: Training ν -support vector classifiers: Theory and algorithms. *Neural Computation* 13, 2119–2147 (2001)
- [16] Cortes, C., Vapnik, V.: Support-vector networks. *Machine Learning* 20, 273–297 (1995)
- [17] Schölkopf, B., Smola, A.J., Williamson, R.C., Bartlett, P.L.: New support vector algorithms. *Neural Computation* 12, 1207–1245 (2000)
- [18] Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines (2001), <http://www.csie.ntu.edu.tw/~cjlin/libsvm>

BoostML: An Adaptive Metric Learning for Nearest Neighbor Classification

Nayyar Abbas Zaidi¹, David McG. Squire¹, and David Suter²

¹ Clayton School of IT Monash University, Clayton VIC 3800, Australia

² School of Computer Science University of Adelaide,
North Terrace SA 5005, Australia

{nayyar.zaidi,david.squire}@infotech.monash.edu.au,
david.suter@adelaide.edu.au

Abstract. A Nearest Neighbor (NN) classifier assumes class conditional probabilities to be locally smooth. This assumption is often invalid in high dimensions and significant bias can be introduced when using the nearest neighbor rule. This effect can be mitigated to some extent by using a locally adaptive metric. In this work we propose an adaptive metric learning algorithm that learns an optimal metric at the query point. We learn a distance metric using a feature relevance measure inspired by boosting. The modified metric results in a smooth neighborhood that leads to better classification results. We tested our technique on major UCI machine learning databases and compared the results to state of the art techniques. Our method resulted in significant improvements in the performance of the K-NN classifier and also performed better than other techniques on major databases.

Keywords: Adaptive Metric Learning, Nearest Neighbor, Bias-Variance analysis, Curse-of-Dimensionality, Feature Relevance Index.

1 Introduction

Nearest Neighbor (NN) methods for pattern recognition are widely applicable and have proven to be very useful in machine learning. Despite their simplicity, their performance is comparable to other, more sophisticated, classification and regression techniques. A nearest neighbor classifier works by assigning to a query point the label of the majority class in its neighborhood. Their only assumption is smoothness of the target function to be estimated. Therefore each point in the neighborhood votes for the prediction based on its distance from the query point.

The performance of a nearest neighbor classifier depends critically on two major factors: (a) the distance metric used and (b) K , size of the neighborhood. K denotes the number of nearest neighbors. The size of the neighborhood controls the smoothness of the predicted function and is usually tuned through cross-validation. A small K implies small bias but high variance, and vice-versa. Typical ‘Metric Learning’ algorithms aim at finding a metric that results in small intra-class and large inter-class distances [1,2]. ‘Metric Learning’ has also been

introduced as a bias reduction strategy in high dimensions [3]. In this paper we focus on the latter version, that is optimizing a distance metric to reduce bias. Our goal is an optimal metric that depends on the problem at hand, as characterized by the respective class distribution and, within a given problem, on the location of the query point in that space.

Bias can be introduced due to a variety of reasons. The primary reason for the introduction of bias is the ‘curse-of-dimensionality’ (COD) effect. Let us consider training data of size N drawn from a uniform distribution in a p -dimensional unit hypercube. The expected diameter of a $K = 1$ neighborhood using Euclidean distance is $d_1(p, N) = 2 \left(\frac{p\Gamma(p/2)}{2\pi^{p/2}N} \right)^{1/p}$. It can be seen that even for a moderate number of dimensions, a very large amount of training data is required to make even a $K = 1$ nearest neighborhood relatively small. This has the consequence that the bias can be large even for the smallest possible value of K . Bias can be reduced by learning a metric that gives no influence to the irrelevant features (feature selection). This removes irrelevant features, thereby reducing the dimensionality. This in turn reduces the diameter of the K -NN neighborhood hence lowering the bias.

In order to understand why metric learning can improve classification performance, we need to analyze the classification performance itself. In a classification scenario, all that is required to obtain an optimal decision is that largest estimated class probability correspond to the class with the largest true probability, irrespective of its actual value or the values (or the order) of the estimated probabilities for the other classes, as long as the following equation holds: $\forall j : \max \hat{f}_j(x) = \max f_j(x)$ where j is class index, \hat{f} is the estimated probability, and f is the unknown true probability. It can often be the case that bias, though very large, affects each of $\hat{f}_j(x)$ in same way so that their order relation is similar enough to that of true underlying probabilities $\{f_j(x)\}_1^J$ to realize an optimal class assignment [3]. But the requirement of a large neighborhood for high dimensions and the presence of irrelevant features can affect bias differentially for the respective class probability estimates enough to cause non-optimal assignment, therefore decreasing classification performance. This differential bias can be reduced by taking advantage of the fact that the class probability functions may not vary with equal strength or in the same manner in all directions in the measurement space. Elongating a neighborhood in the directions in which class probabilities do not change and constricting along those dimensions where class probabilities do change—by choosing an appropriate metric—not only reduces bias, but will also result in smoother class conditional probabilities in the modified neighborhood, resulting in better classification performance (refer to figure [1]).

In this paper we propose a technique for local adaptive metric learning to reduce bias in high dimensions. As will be discussed in section [2], current work in adaptive metric learning determines feature relevance at a query point using some numerical index. This index gauges the relevance of a feature and controls the form of the metric around the query point. Our proposed index is inspired by work in the field of boosting [4], where at each iteration data is partitioned across

the most discriminative dimension. The index is based on the logit-transform of the class probability estimate. In our work using this index, we pick the dimension that is most discriminative. This is similar to ‘boosting classifiers’ where at each iteration a feature is selected on which the data can be classified most accurately based on the weight distribution.

2 Related Work

Friedman in [3] proposed a technique for reducing bias in high dimensional machine learning problems. Our work is inspired by this paper. The main difference of our work from [3] is feature relevance determination at each step. We have used a measure inspired by the boosting literature, whereas in [3] a GINI-like (entropy-based) index is used for feature relevance. In our work a feature is deemed more relevant if it is more discriminatory, but in [3] a feature is considered relevant if the class label varies the most.

In [5], Hastie and et al. proposed an adaptive metric learning algorithm (DANN) based on linear discriminant analysis. A distance metric is computed as a product of properly weighted within- and between-class sum-of-squares matrices. The major limitation of their method is that, in high dimensions there may not be sufficient data to fill in $p \times p$ within-class sum-of-square matrices (due to sparsity). In our work we estimate only the diagonal terms of the matrix.

Some other notable techniques for adaptive metric learning are proposed in [6,7,8,9]. In [6] an algorithm is proposed for adaptive metric learning based on analysis of the chi-squared distance. Similarly, an algorithm for metric learning has been proposed in [7] that uses SVM-based analysis for feature relevance. A similar, but slightly modified, method for metric learning based on SVMs is proposed in [9]. As will be discussed in section 3 our method differs from these methods in the sense that it is recursive. We recursively home in around the query point and the estimated metric is modified iteratively. In the above mentioned methods, however, a metric is estimated in a single cycle.

Other research on query-sensitive metric methods includes Zhou et al. [10], who investigated a query-sensitive metric for content-based image retrieval.

3 Approach

In this section we describe our two algorithms, BoostML1 and BoostML2, for adaptive metric learning. BoostML2 is a variant of BoostML1. In the following discussion we will denote the query point by x_0 and training points by x_n , where $n = [1, \dots, N]$, and N is the number of training data. P denotes the number of features, and x_{0p} and x_{np} denote the value at the p th feature of the x_0 and x_n data points respectively.

3.1 Feature Relevance

We start by describing our local feature relevance measure technique. The feature used for splitting is the one that maximizes the estimated relevance score

$p(x_0)$ as evaluated at query point x_0 . The estimate of relevance is: $p^*(x_0) = \operatorname{argmax}_{1 \leq p \leq P} c_p(x_0)$ where $c_p(x_0)$ is defined as:

$$c_p(x_0) = I_p(x_0) / \sum_{p=1}^P I_p(x_0) \quad (1)$$

and $I_p(x_0)$ is defined in the following equation:

$$I_p(x_0) = \sum_{c=1}^C \operatorname{abs} \left(\frac{1}{2} \ln \left(\frac{\Pr(c|x_{np} = x_{0p}) + \epsilon}{\Pr(c|x_{np} \neq x_{0p}) + \epsilon} \right) \right) \quad (2)$$

The ϵ in equation 2 is introduced for numerical tractability. Small I_p (close to zero) implies that there is an equal split of positive and negative training data points in the neighborhood of x_0 , whereas large I_p implies that one class dominates the other class. The computation of $\Pr(c|x_{np} \neq x_{0p})$ in equation 2 is not trivial, as we may not have sufficient data in the neighborhood of the query point to accurately define the probability. The probabilities in equation 2 are computed as in equation 3.

$$\Pr(c|x_{np} = x_{0p}) = \frac{\sum_{x_n \in N(x_0)} 1(|x_{np} - x_{0p}| \leq \delta_p) 1(y_n = c)}{\sum_{x_n \in N(x_0)} 1(|x_{np} - x_{0p}| \leq \delta_p)} \quad (3)$$

A small neighborhood around query point x_0 denoted by $N(x_0)$ is defined and a value of δ_p is chosen to make sure that the neighborhood contains L points:

$$\sum_{n=1}^N 1(|x_{np} - x_{0p}| \leq \delta_p) = L \quad (4)$$

In other words, we look for L points that are close to the query point on feature p and compute the probabilities in equation 2 using these points. The output of feature relevance analysis is a $p \times p$ diagonal matrix, the diagonal terms of which are the estimated relevances of the features. Based on equation 1 we can write the distance metric as a matrix A for local relevance (equation 5). This local metric is used to measure distances.

$$A(x_0) = \begin{pmatrix} c_1 & 0 & \cdots & 0 \\ 0 & c_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & c_p \end{pmatrix} \quad (5)$$

3.2 Details of the Algorithm

Given a query point x_0 and training data $\{(x_n, y_n)\}_{n=1}^N$, the goal is to estimate the label of the query point. Our method starts by initializing the neighborhood of the query point to be the entire measurement space (R_0). The neighborhood

is split in two based on the feature that maximizes the relevance score. Thus for the same training data, different features can be selected for this split for different query points x_0 based on the relevance of that feature at that location in the input measurement space. The split divides the input measurement space into two regions: $R_1(x_0)$, that contains the query point and the M_1 training points that are closest to it on the chosen feature, and other (complement) region $R_2(x_0)$, that contains the $N - M_1$ points that are farthest from x_0 on that feature. $R_2(x_0)$ is removed from further consideration. Thus the result of the split is just one region, $R_1(x_0)$. The above procedure is then applied again on region $R_1(x_0)$. We have named this method BoostML1 and its outline is given in algorithm 1. Refer to figure 1 for an illustration of BoostML1 algorithm.

Algorithm 1. BoostML1: Local Adaptive Metric Learning Algorithm

Require: Testing data: x_0 , Training data: $\{x_n, y_n\}_{n=1}^N$, k : Number of elements in final neighborhood, α : Stepping size. Initialize $K = N$ and A as a p dimensional unit matrix. $N_K(x_0)$ denotes neighborhood of x_0 consisting of K points.

while flag **do**

- Find all x in $N_K(x_0)$.
- Get Feature Relevance index $c_p(x_0)$ at x_0 (equation 1), update A (equation 5).
- Choose feature $r = \operatorname{argmax}_p c$.
- Modify neighborhood by setting $K = \alpha K$.
- Find all x in $N_K(x_0)$ using feature r . (Note: In case of BoostML2, metric A is used to find K neighbors)

if $N_K(x_0) < k$ **then**

flag = false.

end if

end while

- Do K-NN classification in the final neighborhood of k points using metric A .
-

As can be seen in algorithm 1, the splitting procedure is recursively applied until there are only k training observations left in the final neighborhood. The metric A (equation 5) obtained at the final step is used to measure the distance to the k nearest neighbors that predict the label of query point. At each step, a region is split on the feature that is estimated to be most relevant in terms of capturing the variation of the target functions within that region. All diagonal terms of the A matrix in equation 5 are ignored except the one with the maximum value, which is retained to split the region at each step. This is a greedy approach which is not necessarily effective all the time. BoostML2 is a variant of the above method, but at every iteration it splits the region based on the metric defined by matrix A in equation 5, as computed in the current iteration. As will be shown in section 4, BoostML2 is an improvement on algorithm 1 and results in an improvement in classification performance.

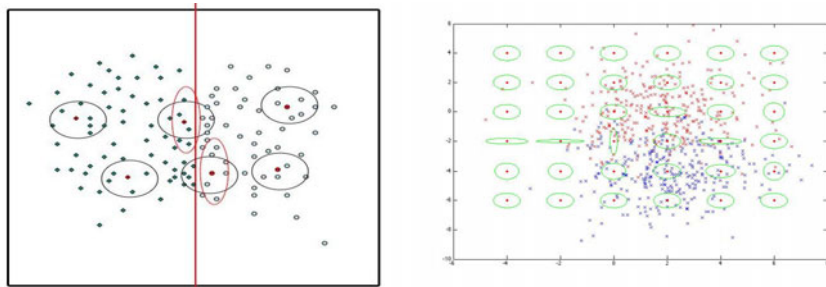


Fig. 1. Left: Illustration of Adaptive Metric Learning on synthetic two class 2-D data. Round curves depicts the Euclidean metric. The assumption of isotropy is not valid near class boundaries and a modified metric depicted as elliptical curves seems more accurate, Right: Demonstration of our algorithm BoostML on synthetic two class 2-D data. Green ellipses shows the learnt metric at different points in the measurement space.

4 Experimental Results

In this section we show the results of our adaptive metric learning algorithm on some well known databases from UCI Machine Learning Repository [11]. Databases were selected such that the competing techniques perform best on at least one of the databases. The other competing local adaptive metric learning techniques against which we tested our algorithms are as follows:

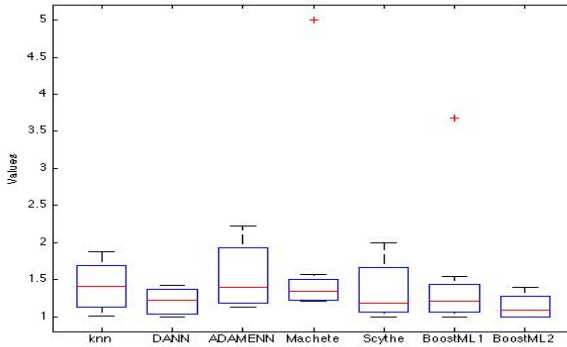
- **k-NN**: k nearest neighbor classifier with Euclidean distance.
- **DANN**: Discriminative Adaptive Nearest Neighbor classifier based on [5] as described in section 2.
- **ADAMENN**: Adaptive metric nearest neighbor classification technique based on chi-squared analysis as implemented in [6].
- **Machette**: Recursive partitioning algorithm as described in [3].
- **Scythe**: This is a generalization of the Machette algorithm in which features influence each split in proportion to their estimated local relevance, in contrast to the ‘winner-takes-all’ strategy of Machette.
- **BoostML1**: Algorithm 1. The implementation details regarding tuning of input parameters are described in following discussion.
- **BoostML2**: Variant of BoostML1 as described in section 3.2.

To obtain error rates, we used leave-one-out cross-validation for the Iris, Ionosphere, Dermatology, Echocardiogram and Heart data sets. 10 rounds of two-fold cross-validation were used for the Credit and Diabetes data sets.

Our metric learning algorithm results in an improvement of k-NN classification. This improvement, however, does come at an extra cost. BoostML1 has introduced two new tuning parameters. The value of L in equation 4 determines bias-variance trade-off but does not effect the performance provided it is neither too small nor too large. A value of 20 for L was used in our experiments. The

Table 1. Average classification error rates for different techniques across various databases, $k = 10$

| | Iris 150,4,3 | Ionosphere 351,34,2 | Dermatology 358,34,6 | Credit 653,15,2 | Echocardiogram 61,12,2 | Heart 270,13,2 | Sonar 208,60,2 | Diabetes 768,8,2 |
|----------|-----------------|------------------------|-------------------------|--------------------|---------------------------|-------------------|-------------------|---------------------|
| K-NN | 4.66 | 16.52 | 3.35 | 13.47 | 8.19 | 20 | 23.07 | 25.91 |
| DANN | 4.66 | 12.53 | 3.35 | 14.09 | 6.55 | 17.4 | 13.46 | 26.17 |
| ADAMENN | 4.66 | 15.95 | 5.58 | 15.15 | 10.11 | 21.48 | 18.75 | 26.56 |
| Machete | 4 | 12.53 | 3.07 | 16.23 | 24.59 | 24.81 | 21.15 | 29.55 |
| Scythe | 4 | 16.8 | 2.51 | 15.62 | 9.83 | 19.62 | 19.23 | 23.43 |
| BoostML1 | 3.333 | 8.83 | 2.79 | 15.62 | 18.03 | 23.33 | 20.67 | 29.16 |
| BoostML2 | 3.333 | 11.68 | 3.07 | 13.32 | 4.91 | 19.25 | 18.75 | 25.13 |

**Fig. 2.** Box plots for various techniques

α parameter which controls the size of the neighborhood at each step is critical to the performance. A large value of α results in a better performance at an increased computational cost. A small value of α results in poorer performance, but will be faster. A tradeoff has to be achieved between computational cost and performance. A value of 0.8 is used in all our experiments.

Table 1 shows the average classification error rates for different techniques. Each database's name is followed by number of data, features and classes. It can be seen that BoostML1 and BoostML2 perform well on the majority of data sets. It results in significant improvement over the performance of the basic k-NN classifier, and also performs better than the competing algorithms in some cases.

To compare the robustness of our algorithm with other algorithms we used the technique described in [3]. This test measures how well a particular method m performs on average in situations that are most favorable to other procedures. Robustness can be measured by computing the ratio b_m of its error rate e_m and the smallest error rate over all other methods that are compared in that example. That is:

$$b_m = \frac{e_m}{\min_{1 \leq k \leq 7} e_k} \quad (6)$$

The best method m^* will have $b_m^* = 1$ and all other methods will have values larger than 1. The larger the value of b_m the worse the performance is of the m^{th} method in relation to the best one for that data set. Figure 2 shows the distribution of b_m for each method over all data sets considered. BoostML2 turned out to be most robust among all the methods, with DANN coming second.

5 Conclusion

In this work we introduced an adaptive metric learning algorithm based on an index inspired by work on boosting for reducing bias in high dimensional spaces. We tested our algorithm on a variety of well-known machine learning databases and found that our system performs better than several well known techniques for adaptive metric learning. This improvement, however, comes at an extra cost. Our algorithm is computationally expensive compared to simple k-NN. We had to introduce two new parameters, the values of which should be optimized. Though this complicates matters, other competing algorithms also have one or more tuning parameters, so it should not be taken as a major drawback of our algorithm.

References

1. Davis, J., Dhillon, I.: Structured metric learning for high dimensional problems. In: KDD (2008)
2. Weinberger, K., Blitzer, J., Saul, L.: Distance metric learning for large margin nearest neighbor classification. In: NIPS (2005)
3. Friedman, J.: Flexible metric nearest neighbor classification. Tech Report, Dept. of Statistics, Stanford University, Tech. Rep. (1994)
4. Shapire, R., Singer, Y.: Improved boosting algorithms using confidence rated predictions. In: Conf. on Computational Learning Theory (1998)
5. Hastie, T., Tibshirani, R.: Discriminative adaptive nearest neighbor classification. IEEE transactions on Pattern Analysis and Machine Intelligence (1996)
6. Domenciconi, C., Peng, J., Gunopulos, D.: An adaptive metric machine for pattern classification. In: NIPS (2000)
7. Peng, J., Heisterkamp, D., Dai, H.: Lda/svm driven nearest neighbor classification. In: CVPR (2001)
8. Janusz, K. (ed.): Support Vector Machines: Theory and Applications. Springer, Heidelberg (2005); ch. Adaptive Discriminant and Quasiconformal Kernel Nearest Neighbor Classification
9. Domenciconi, C., Peng, J., Gunopulos, D.: Large margin nearest neighbor classifiers. IEEE transactions on Pattern Analysis and Machine Intelligence (2005)
10. Zhou, Z., Dai, H.: Query-sensitive similarity measure for content-based image retrieval. In: ICDM (2006)
11. Mertz, C., Murphy, P.: Machine learning repository (2005), <http://archive.ics.uci.edu/ml/>

A New Emerging Pattern Mining Algorithm and Its Application in Supervised Classification

Milton García-Borroto^{1,2}, José Francisco Martínez-Trinidad²,
and Jesús Ariel Carrasco-Ochoa²

¹ Centro de Bioplantas. Carretera a Moron km 9, Ciego de Avila, Cuba
mil@bioplantas.cu

² Instituto Nacional de Astrofísica, Óptica y Electrónica. Luis Enrique Erro No. 1,
Sta. María Tonanzintla, Puebla, México, C.P. 72840
{ariel, fmartine}@ccc.inaoep.mx

Abstract. Obtaining an accurate class prediction of a query object is an important component of supervised classification. However, it could be important to understand the classification in terms of the application domain, mostly if the prediction disagrees with the expected results. Many accurate classifiers are unable to explain their classification results in terms understandable by an application expert. Emerging Pattern classifiers, on the other hand, are accurate and easy to understand. However, they have two characteristics that could degrade their accuracy: global discretization of numerical attributes and high sensitivity to the support threshold value. In this paper, we introduce a novel algorithm to find emerging patterns without global discretization, which uses an accurate estimation of the support threshold. Experimental results show that our classifier attains higher accuracy than other understandable classifiers, while being competitive with Nearest Neighbors and Support Vector Machines classifiers.

Keywords: Emerging pattern mining, Understandable classifiers, Emerging pattern classifiers.

1 Introduction

The main goal of a supervised classification algorithm is to build a model based on a representative sample of the problem classes [1]. This model can be used to predict the class of new objects or to gain understanding of the problem domain. In many cases, the result of the classification is not enough; the user could need to understand the classification model and the classification results, mostly if the classification disagrees with the expected results.

Many accurate classifiers, like Neural Networks [2] or Support Vector Machines [3], are unable to explain their classification results in terms understandable by an application expert. Emerging Pattern classifiers, on the other hand, build accurate and easy to understand models. An emerging pattern is a combination of feature values that appears mostly in a single class. This way, an

emerging pattern can capture useful contrasts among the problem classes [4], which can be used to predict the class of unseen objects.

Emerging pattern classifiers are very valuable tools to solve real problems in fields like Bioinformatics [5], streaming data analysis [6], and intruder detection [7].

Current methods for finding Emerging Patterns in a database have two main drawbacks:

- Global discretization of numerical attributes, which could seriously degrade the classification accuracy
- High sensitivity to the support threshold value, which makes very hard for the user to select a good value

In this paper, we introduce the Crisp Emerging Pattern Mining (CEPM), a novel algorithm to find emerging patterns, which does not apply global discretization of numerical attributes. CEPM extracts patterns using a special procedure, from a collection of C4.5 decision trees. To find a representative collection of patterns, our algorithm uses a novel object weighting scheme. CEPM applies local discretization, using only such attribute values appearing in the objects on each tree node. Additionally, CEPM finds an accurate estimation of the minimal support threshold, testing different values decrementally. It starts from a high enough value and ends when the threshold attains the expected abstention ratio. CEPM returns a set of emerging patterns with the highest support value associated with the lowest expected abstention ratio.

The rest of the paper is organized as follows: Section 2 presents a brief revision about classification using emerging patterns, Section 3 introduces the new algorithm for mining emerging patterns without global discretization, Section 4 presents the algorithm to estimate the minimal support threshold, Section 5 shows the experimental results, and Section 6 presents our conclusions.

2 Classification Using Emerging Patterns

A *pattern* is an expression, defined in a language, which describes a collection of objects; the amount of objects described by a pattern is the pattern *support*. In a supervised classification problem, we say that a pattern is *emerging* if its support increases significantly from one class to the others [4]. Emerging patterns (EPs) are usually expressed as combinations of feature values, like ($Color = green, Sex = male, Age = 23$) or as logical properties, like $[Color = green] \wedge [Sex = male] \wedge [Age > 23]$.

Most algorithms for emerging pattern mining have the goal of finding the patterns that satisfy a desired property: being supported by a single class, minimality over subset inclusion, or tolerance to noisy objects. These algorithms have the following general steps:

1. Selection of the minimal support threshold μ
2. Global discretization of numerical attributes

3. Representation of the transformed objects using a particular structure
4. Traversing the structure to find emerging patterns
5. Pattern filtering

Using this traditional algorithm has two important drawbacks:

1. Global discretization of numerical attributes could drastically degrade the classifier accuracy, since an emerging pattern relates a combination of feature values with a class. Therefore, discretizing a numerical attribute without considering the values of other features could hide important relations. In Table 1, we can see that SJEP [8], one of the most accurate emerging pattern classifiers, obtains very poor accuracies in databases like *Iris*, while all other classifiers attain accuracies above 93%. In some other databases, SJEP is unable to extract even a single pattern, because most numerical features are discretized into a single categorical value. This behavior is mainly due to using the Entropy discretization method [9], but other discretization methods obtain similar results, maybe in different databases.
2. High sensitivity to the support threshold value. The accuracy of the classifier could have a serious degradation on small variations of the minimal support value. For example, in *chess* and *census* databases, the accuracy drops 3% with a variation of 2 in the threshold value [10].

3 Crisp Emerging Pattern Mining (CEPM)

In this section, we introduce CEPM, a new emerging pattern mining algorithm with local discretization of numerical features. CEPM extracts patterns from a collection of C4.5 decision trees [11], using a special pattern mining procedure during the tree induction. To guarantee that CEPM finds a representative collection of patterns, it uses a novel object weighting scheme.

The tree induction procedure has the following characteristics:

- Candidate splits are binary. Nominal attributes use properties like [*Feature* = *a*] and [*Feature* ≠ *a*] for each one of their values; numerical attributes use properties like [*Feature* > *n*] and [*Feature* ≤ *n*] for all candidate cut points
- If a node has less than μ objects, it is not further split because it cannot generate emerging patterns
- To select the best split, our algorithm evaluates the *weighted information gain*. The weighted information gain is similar to the classical information gain but there is a weight associated to each object. This way, P_{Class} and P_{child} are calculated using (1). Note that objects with weight close to 0 have low influence in the determination of the best split.

$$P_{Class} = \frac{\sum_{o \in Class} w_o}{\sum w_o}, \quad P_{child} = \frac{\sum_{o \in child} w_o}{\sum w_o}. \quad (1)$$

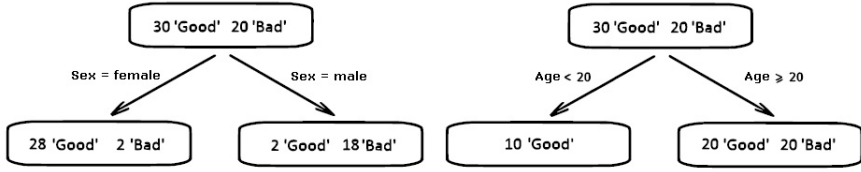


Fig. 1. Example of an emerging pattern appearing in a non-optimal candidate split

During the tree induction, every tree node that (A) has at least μ objects in a class, and (B) has at most one object in the complement of that class, generates a new emerging pattern. Each pattern consists in the conjunction of the properties from the node to the root.

Additionally, CEPM extracts patterns while evaluating the splits, even if a split does not have the highest gain; any tree node that fulfills (A) and (B) generates an emerging pattern. For example, Fig. 1 shows two candidate splits, using different properties. Although the first one has the highest information gain, the second contains the emerging pattern ($Age < 20$). So, this pattern is extracted although the split is discarded.

CEPM iteratively induces several decision trees, updating the object weights after each induction. The algorithm updates the weights using (2).

$$w_o = \frac{\operatorname{arccot} \left(5 \cdot \frac{\operatorname{Support}_o}{\operatorname{averageSupport}} \right)}{\pi} . \tag{2}$$

where

- $\operatorname{Support}_o$ is the sum of the support of such patterns contained in o . If a pattern belongs to a different class than o , its support is multiplied by -1
- $\operatorname{averageSupport}$ is the average support of the patterns in the database, which is estimated based on the patterns found in the first built tree.

We can describe CEPM using the following pseudocode:

1. Initialize object weights to 1
2. Induce the first decision tree with the initial weights and extract the first emerging patterns
3. Calculate the average support, used in weight recalculation
4. Repeat while a new pattern is added in the last iteration
 - (a) Recalculate object weights
 - (b) Induce the new decision tree with current weights and extract the emerging patterns
 - (c) Add the new patterns to the pattern collection
5. Return mined patterns

It is worth to mention that CEPM returns a set of the most general emerging patterns with support greater or equal to μ . A pattern P is more general than

a pattern Q if the set of objects described by Q is strictly contained in those described by P , considering all the objects in the universe. Additionally, CEPM returns the abstention ratio, which is the ratio of objects that are not covered by the resultant patterns.

The CEPM based classifier, named CEPMC, uses the following decision rule: to assign an object to the class with maximum value of the total votes given by the patterns contained in the object. Every pattern contained in the object votes for its own class with its total support. If no pattern supports the object or there is a tie in the votes, the classifier refuses to classify the object.

4 Estimating the Minimal Support Threshold for CEPM

Selecting the minimal support threshold for an emerging pattern classifier is a difficult task; a classifier using patterns with higher μ values, is a more accurate classifier, but could reject to classify more objects. On the contrary, a classifier using patterns with lower μ values might contain many useless patterns, which could degrade the classification accuracy.

The algorithm proposed for calculating the minimal support value infers the initial support (*MaxSupport*) and the minimal expected abstention rate (*MinAbstRate*). Then, it tests support values, starting from $\mu = \text{MaxSupport}$, until a μ value produces an abstention rate lower than *MinAbstRate*. The value of μ is decremented using a calculated $\text{Step} = \text{MaxSupport}/10$, because if *MaxSupport* is high, decrementing μ by 1 could be too costly.

Some important remarks:

1. *MaxSupport* is inferred based on two criteria. If it is higher than the optimum, the algorithm makes unnecessary iterations; if it is lower, better models (with higher μ) are disregarded.
2. *MinAbstRate* is inferred using $\mu = 2$, so it measures the minimal expected abstention ratio of a pattern based classifier using CEPM. The algorithm searches for more accurate classifiers (having patterns with higher μ values) with the same abstention level.

5 Experimental Results

To compare the performance of the CEPMC classifier, we carried out some experiments over 22 databases from the UCI Repository of Machine Learning [12]. We selected six state-of-the-art classifiers: Nearest Neighbors [13], Bagging and Boosting [14], Random Forest [15], C4.5 [11] and Support Vector Machines [3]. For each classifier, we used the Weka 3.6.1 implementation [16] with its default parameters. We also tested SJEP [8], which is one of the most accurate emerging pattern based classifiers, using the minimal support threshold suggested by their authors.

We performed 10-fold cross validation, averaging the results. In both SJEP and CEPMC we reported abstentions as errors. In these objects, the classifier

Table 1. Accuracy results of the classifiers in the selected databases. The highest accuracy per database is bolded.

| DBName | 3NN | AdaBoost | Bagging | C4.5 | RandFor | SVM | SJEP | CEPMC |
|-----------------|-------------|-------------|-------------|-------------|-------------|-------------|------|--------------|
| balance-scale | 85.4 | 71.7 | 82.6 | 77.6 | 79.4 | 87.5 | 16.0 | 79.5 |
| breast-cancer | 70.3 | 72.4 | 71.0 | 73.4 | 65.8 | 70.7 | 44.5 | 72.0 |
| breast-w | 96.7 | 94.9 | 96.0 | 94.9 | 95.1 | 96.7 | 96.1 | 96.0 |
| cleveland | 82.5 | 84.2 | 79.9 | 78.2 | 78.6 | 84.5 | 77.9 | 81.2 |
| haberman | 70.6 | 70.9 | 72.5 | 68.0 | 67.6 | 72.5 | 0.0 | 68.6 |
| hayes-roth | 71.4 | 53.6 | 75.0 | 89.3 | 85.7 | 53.6 | 0.0 | 78.6 |
| heart-c | 81.2 | 83.2 | 81.9 | 76.2 | 80.9 | 82.8 | 78.6 | 81.2 |
| heart-h | 83.6 | 81.6 | 79.9 | 79.6 | 79.3 | 82.6 | 46.3 | 81.0 |
| heart-statlog | 79.3 | 80.7 | 79.3 | 79.3 | 79.3 | 83.0 | 64.8 | 80.0 |
| hepatitis | 82.0 | 81.2 | 82.0 | 78.8 | 81.3 | 86.5 | 77.5 | 82.5 |
| iris | 96.0 | 96.7 | 93.3 | 94.0 | 94.7 | 96.0 | 66.7 | 95.3 |
| labor | 90.7 | 87.0 | 84.0 | 80.0 | 86.7 | 90.7 | 82.0 | 89.0 |
| liver-disorders | 65.5 | 66.1 | 68.7 | 68.7 | 70.7 | 57.7 | 0.0 | 69.3 |
| lymph | 85.9 | 75.7 | 77.7 | 78.5 | 79.9 | 87.9 | 51.5 | 83.7 |
| mp1 | 50.0 | 50.0 | 50.0 | 50.0 | 50.0 | 50.0 | 57.9 | 100.0 |
| mp2 | 51.4 | 50.0 | 55.1 | 59.7 | 58.6 | 50.5 | 34.0 | 83.8 |
| mp3 | 50.0 | 50.0 | 50.0 | 50.0 | 50.0 | 50.0 | 63.7 | 97.5 |
| shuttle | 60.0 | 65.0 | 60.0 | 60.0 | 55.0 | 45.0 | 0.0 | 50.0 |
| spect | 64.7 | 66.8 | 61.5 | 66.8 | 62.0 | 67.9 | 0.0 | 78.1 |
| tic-tac-toe | 98.5 | 73.5 | 91.0 | 83.8 | 91.9 | 98.3 | 91.3 | 96.5 |
| vote | 92.0 | 94.7 | 95.2 | 96.1 | 96.1 | 95.9 | 91.1 | 94.0 |
| wine | 96.1 | 87.5 | 94.3 | 92.7 | 97.2 | 98.9 | 55.1 | 93.3 |
| Average | 77.4 | 74.4 | 76.4 | 76.2 | 76.6 | 76.8 | 49.8 | 83.2 |

is unable to assign a class; returning the majority or a random class could hide these undesirable cases. In Table 1, we can find the accuracy results, in percent.

Experimental results show that SJEP has low accuracy values in some databases, compared to other classifiers. In those databases, most numerical attributes were discretized into a categorical attribute with a single value, so they were useless for mining patterns. CEPMC has higher accuracies than SJEP in most databases. It also has the highest average accuracy from all tested classifiers.

In order to determine if the differences in accuracy are statistically significant, we performed a pairwise comparison between our classifier and the others. Each cell in Table 2 contains the number of databases where our classifier Win/Lose/Tie to each other classifier. We detect ties using a two-tailed T-Test [17] with significance of 0.05. The pairwise comparison shows that, in the tested databases, CEPMC is more accurate than other understandable classifiers, while being competitive with Nearest Neighbors and Support Vector Machines classifiers.

The model built by CEPMC is very easy to understand in terms of the problem domain, unlike Nearest Neighbors and Support Vector Machines models. Each

Table 2. Pairwise comparison between our classifier and the others. Each cell shows the number of Win/Loss/Tie of CEPMC with respect to the corresponding classifier over the selected 22 databases.

| | 3NN | 7NN | AdaBoost | Bagging | C4.5 | RandFor | SVM | SJEP |
|-------|--------|--------|----------|---------|--------|---------|-------|--------|
| CEPMC | 6/5/11 | 6/6/10 | 10/3/9 | 8/3/11 | 10/3/9 | 9/4/9 | 7/7/8 | 21/0/1 |

Table 3. Classifier model built by CEPMC for one of the folds in database Iris

| |
|---|
| iris-setosa [<i>PetalLength</i> ≤ 1.90] [<i>PetalWidth</i> ≤ 0.60] |
| iris-versicolor [<i>PetalLength</i> > 1.90] ∧ [<i>PetalLength</i> ≤ 4.90] ∧ [<i>PetalWidth</i> ≤ 1.60] |
| iris-virginica [<i>PetalLength</i> > 1.90] ∧ [<i>PetalWidth</i> > 1.60] [<i>PetalLength</i> > 4.90] |

class is described as a collection of discriminative properties, as you can see in the example appearing in Table 3.

6 Conclusions

In this paper, we introduced CEPM, a new algorithm for mining Emerging Patterns. It uses local discretization of numerical values for solving the global discretization drawback of previous emerging pattern classifiers. CEPM extracts patterns from a collection of decision trees, using a special extraction procedure during the tree induction. To obtain a collection of representative patterns, CEPM uses a novel object weighting scheme. Furthermore, this paper proposes an algorithm for accurately estimate the minimal support threshold.

Experimental results show that CEPMC, a classifier based on CEPM, is more accurate than one of the most accurate emerging pattern classifiers in the majority of tested databases. A pairwise comparison reveals that CEPMC is more accurate than other understandable classifiers, and as accurate as Nearest Neighbors and Support Vector Machines, while the model built by CEPMC for classification is easy to understand in terms of the problem domain.

In the future, we will work on speeding up the algorithm to estimate the minimal support threshold, which is the slowest component of the current algorithm.

Acknowledgments

This work is partly supported by the National Council of Science and Technology of México under the project CB-2008-01-106443 and grant 25275.

References

1. Berzal, F., Cubero, J.C., Sánchez, D., Serrano, J.M.: Art: A hybrid classification model. *Machine Learning* 54, 67–92 (2004)
2. Haykin, S.: *Neural Networks: A Comprehensive Foundation*. Prentice Hall PTR, Englewood Cliffs (1998)
3. Cortes, C., Vapnik, V.: Support-vector networks. *Machine Learning* 20(3), 273–297 (1995)
4. Dong, G., Li, J.: Efficient mining of emerging patterns: Discovering trends and differences. In: *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Diego, California, United States, pp. 43–52. ACM, New York (1999)
5. Quackenbush, J.: Computational approaches to analysis of dna microarray data. *Methods Inf. Med.* 45(1), 91–103 (2006)
6. Alhammady, H.: Mining streaming emerging patterns from streaming data. In: *IEEE/ACS International Conference on Computer Systems and Applications*, Amman, pp. 432–436 (2007)
7. Chen, L., Dong, G.: Masquerader detection using oclep: One-class classification using length statistics of emerging patterns. In: *WAIMW 2006: Proceedings of the Seventh International Conference on Web-Age Information Management Workshops*, Washington, DC, USA, vol. 5, IEEE Computer Society, Los Alamitos (2006)
8. Fan, H., Ramamohanarao, K.: Fast discovery and the generalization of strong jumping emerging patterns for building compact and accurate classifiers. *IEEE Trans. on Knowl. and Data Eng.* 18(6), 721–737 (2006)
9. Fayyad, U., Irani, K.: Multi-interval discretization of continuous-valued attributes for classification learning. In: *13th Int'l Joint Conf. Artificial Intelligence (IJCAI)*, pp. 1022–1029 (1993)
10. Bailey, J., Manoukian, T., Ramamohanarao, K.: Fast algorithms for mining emerging patterns. In: Elomaa, T., Mannila, H., Toivonen, H. (eds.) *PKDD 2002. LNCS (LNAI)*, vol. 2431, pp. 39–208. Springer, Heidelberg (2002)
11. Quinlan, J.R.: *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., San Francisco (1993)
12. Merz, C., Murphy, P.: *Uci repository of machine learning databases*. Technical report, University of California at Irvine, Department of Information and Computer Science (1998)
13. Dasarathy, B.D.: *Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques*. IEEE Computer Society Press, Los Alamitos (1991)
14. Kuncheva, L.I.: *Combining Pattern Classifiers*. In: *Methods and Algorithms*. Wiley-Interscience, Hoboken (2004)
15. Ho, T.K.: The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20(8), 832–844 (1998)
16. Frank, E., Hall, M.A., Holmes, G., Kirkby, R., Pfahringer, B., Witten, I.H.: *Weka: A machine learning workbench for data mining*. In: Maimon, O., Rokach, L. (eds.) *Data Mining and Knowledge Discovery Handbook: A Complete Guide for Practitioners and Researchers*, pp. 1305–1314. Springer, Berlin (2005)
17. Dietterich, T.G.: Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation* 10(7), 1895–1923 (1998)

Hiding Emerging Patterns with Local Recoding Generalization

Michael W.K. Cheng, Byron Koon Kau Choi, and William Kwok Wai Cheung

Department of Computer Science
Hong Kong Baptist University, Kowloon Tong, Hong Kong
{michael, choi, william}@comp.hkbu.edu.hk

Abstract. Establishing strategic partnership often requires organizations to publish and share meaningful data to support collaborative business activities. An equally important concern for them is to protect sensitive patterns like unique emerging sales opportunities embedded in their data. In this paper, we contribute to the area of data sanitization by introducing an optimization-based local recoding methodology to hide emerging patterns from a dataset but with the underlying frequent itemsets preserved as far as possible. We propose a novel heuristic solution that captures the unique properties of hiding EPs to carry out iterative local recoding generalization. Also, we propose a metric which measures (i) frequent-itemset distortion that quantifies the quality of published data and (ii) the degree of reduction in emerging patterns, to guide a bottom-up recoding process. We have implemented our proposed solution and experimentally verified its effectiveness with a benchmark dataset.

Keywords: Emerging patterns, pattern hiding, data sanitization, frequent itemsets.

1 Introduction

Organizations often publish and share their data to support business collaboration. In the context of marketing and sales, companies can leverage on the customer pools of each other for cross-selling so that the involved parties can gain sales volume increase. Due to the equally important need of privacy protection, customers often expect their data to be anonymized before sharing [27] and studies on privacy-preserving data publishing have bloomed [12]. Furthermore, trade secrets embedded in data are valuable to organizations [6] and needed to be properly protected. For instance, patterns like recent increase in the sales volume of a product line for a certain customer group (emerging marketing trends) can be an example. Leaking of related intelligence could cause company loss in gaining the first-mover advantage. Even though companies understand that data sharing is unavoidable to support collaborative activities like cross-selling, they may face a great hindrance to data sharing if the emerging sales opportunities of their own business cannot be hidden.

Among others, *emerging patterns* [19] embedded in data carry sensitive information, that data owners may prefer to hide. In fact, previous studies have revealed that emerging patterns are highly discriminative when used as features for classification [31][11][8],

and thus carry salient features of the data. The hiding, however, is technically challenging as collaborative data analysis is still often expected to facilitate collaboration. That is, some statistical properties of the data to-be-shared are preserved as far as possible. In particular, frequent itemset mining has already been well-supported in most commercial data-mining packages. Therefore, in this paper, we study how to hide emerging patterns while preserving frequent itemsets.

To hide emerging patterns, we adopt recoding generalization methods. In particular, we adopt local recoding which is (intuitively) a value-grouping generalization process, given an attribute generalization hierarchy. To ensure that the generalized data neither (i) reveal sensitive information nor (ii) produce a highly distorted mining result, we propose metrics for quantifying the two competing objectives. With the metrics, we present an iterative, bottom-up optimization framework. Compared with hiding frequent itemsets [25], hiding emerging patterns is more technically challenging. In particular, the *a priori anti-monotone* property does not hold in emerging patterns. Thus, the search space of emerging patterns is huge. Worst still, a local recoding may hide an emerging pattern while generating new emerging patterns. To the best of our knowledge, there has not been work on hiding emerging patterns.

2 Related Work

Studies on data sanitization can be dated back to the earlier work on statistical disclosure control [1]. Recent development in privacy preserving data mining [26] has contributed to some advances in privacy measure and data sanitization method. For example, to avoid personal identity to be recovered from an anonymized demographic dataset, a number of privacy measures were proposed in the literature, e.g., k -anonymity [27] and ℓ -diversity [22]. Other privacy measures include k^m -anonymity [28] and (h, k, p) -coherence [33]. Given a particular measure, recoding generalization [19, 14, 18, 26, 9, 32, 20] and perturbation [2, 10, 17] are two commonly adopted data sanitization approaches. Recoding generalization is often preferred over the perturbation approach as the dataset sanitized by recoding generalization is still semantically consistent with the original one, even though it is “blurred”. While this study aims at hiding emerging patterns instead of personal identities, the concepts like equivalence classes and recoding generalization are adopted in the proposed methodology.

To control the distortion of the data caused by the sanitization, attempts have been made to preserve as much information of the original dataset as possible to, say, preserve the subsequent classification accuracy [15] and clustering structure [13]. In addition, there has been some recent work studying the tradeoff between privacy and utility [21] in the context of privacy-preserving data publishing. In this work, we try to preserve the frequent itemsets of the data as far as possible.

Recently, there has been work [25, 24] on hiding patterns like frequent itemsets where users specify a subset of frequent itemsets, namely sensitive frequent itemsets, that are not supposed to be disclosed to the public. In our study, we focus on hiding emerging patterns, which makes a unique contribution to the area of pattern hiding. Emerging patterns (EP) are features that are distinctive from one class to another and has been found to be effective for building highly accurate classifiers [16, 31, 11, 8]. Mining EPs

from large databases is technically intriguing as the total number of EPS, in the worst case, is exponential to the total number of attributes in transactions, and there has not been a corresponding notion of the apriori anti-monotone property of frequent itemsets in EPS so that the search space can be pruned. Previous work on EPS mainly focuses only on the mining efficiency, e.g., using a border-based approach [4], a constraint-based approach [34], or focusing only on jumping EPS [3]. So far, there exists no related work on emerging pattern hiding.

3 Background and Problem Statement

In the following, we present the definitions, notations used and the problem statement.

A *transactional dataset* is a set of transactions. Let $I = \{i_1, i_2, \dots, i_n\}$ be a finite set of distinct items in D . A *transaction* t has a set of nominal attributes $A = \{a_1, a_2, \dots, a_m\}$ and each attribute a_i takes values from a set $V_i \subseteq I$. We make two simple remarks about these notations. (i) While we assume transactional data with nominal attributes, data of a continuous domain can be cast into nominal data, for example by defining ranges. (ii) One may consider a relation as a set of transactions of a fixed arity.

An *itemset* X is a (proper) subset of I . $Supp_D(X)$ denotes the support of an itemset X in a dataset D , which can be computed as $\frac{|\{t \mid X \subseteq t \wedge t \in D\}|}{|D|}$. Given a support threshold σ , X is said to be a σ -frequent itemset if $Supp_D(X) \geq \sigma$. The growth rate of an itemset is the ratio of its support in one dataset to that in the other.

Definition 1. [7] Given two datasets, namely D_1 and D_2 , the growth rate of an itemset X , denoted as $GR(X, D_1, D_2)$, from D_1 to D_2 is defined as $GR(X, D_1, D_2) =$

$$\begin{cases} 0 & , \text{ if } Supp_{D_1} = 0 \text{ and } Supp_{D_2} = 0 \\ \infty & , \text{ if } Supp_{D_1} = 0 \text{ and } Supp_{D_2} > 0 \\ \frac{Supp_{D_2}(X)}{Supp_{D_1}(X)} & , \text{ otherwise.} \end{cases}$$

Definition 2. [7] Given a growth rate threshold ρ and two datasets D_1 and D_2 , an itemset X is a ρ -emerging pattern (ρ -EP) from D_1 to D_2 if $GR(X, D_1, D_2) \geq \rho$. Intuitively, given two datasets, emerging patterns (EPS) [7] are the itemsets whose support increases significantly from one dataset to another. The formal definition of EPS is presented in Definition 2. An emerging pattern with a growth rate ∞ (i.e., itemset that appears in one dataset but not the other) is called a *jumping emerging pattern*.

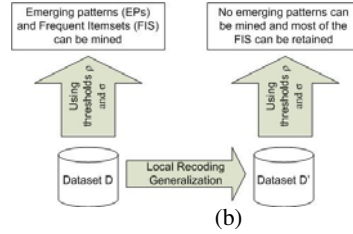
For ease of presentation, we may skip σ , ρ , D_1 and D_2 of EPS when they are not essential to our discussions.

Example 1. Figure 1(a) shows a simplified hypothetical dataset D of the `Adult` dataset [30]. It contains some census information of the United States. More description of the dataset can be found in Section 7. We opt to present some nominal attributes of `Adult` for discussions. Each record (or transaction) represents a person. Consider two subsets D_1 containing married people and D_2 containing those who do not. From Figure 1(a), we find the following emerging patterns, among many others.

- The pattern $(MSE, manager)$ has a support of 75% in D_1 and 20% in D_2 . Therefore, the growth rate of $(MSE, manager)$ from D_1 to D_2 is 3.75. When we set ρ to 3, $(MSE, manager)$ is a ρ -emerging pattern in D_2 .

| ID | Edu. | Marital | Occup. | Rel. | Race | Sex |
|----|------|---------|-----------|---------|-------|-----|
| 1 | BA | married | executive | wife | black | F |
| 2 | MSE | married | manager | husband | black | M |
| 3 | MSE | married | manager | wife | white | F |
| 4 | MSE | married | manager | husband | black | M |
| 5 | BA | never | manager | NA | white | M |
| 6 | MSE | never | manager | NA | white | F |
| 7 | HS | never | repair | NA | black | M |
| 8 | BA | never | manager | NA | white | M |
| 9 | BA | never | manager | NA | black | F |

(a)



(b)

Fig. 1. (a) A hypothetical subset of `Adult` and (b) The problem statement illustration

- High-school graduate (`HS`) has a support of 0% in D_1 but 20% in D_2 . Hence, its growth rate from D_2 to D_1 is infinite. (`HS`) is a jumping emerging pattern in D_1 .

Next, we state the formal problem statement of this paper below (Figure 1(b)).

Problem statement. Given two datasets (D_1, D_2) , σ and ρ , we want to sanitize (D_1, D_2) to (D'_1, D'_2) such that no ρ -EPs from D'_1 to D'_2 can be mined while the distortion between σ -frequent itemsets of (D_1, D_2) and those of (D'_1, D'_2) is minimized. \square

4 Multidimensional Local Recoding

Our algorithm for hiding emerging pattern is based on local recoding generalization (`multi-local-recode` in Figure 3). In this section, we give an overview of recoding generalization, or *recoding* for simplicity.

Recoding. As discussed in Section 1, recoding has been proposed for anonymization. The idea of recoding is to modify values into more general ones such that more tuples will share the same values and cannot be distinguished individually. Thus, anonymization can be achieved. Here, we recode values in emerging patterns with some non-emerging values. Thus, the recoded patterns become less emerging.

Multidimensional local recoding. In this work, we adopt the notion of multidimensional local recoding [9, 32, 20], from the context of k -anonymity. It recodes values at “cell level”. It relies on equivalence classes. An *equivalence class* of attributes A is a set of tuples T , where $\pi_A(T)$ is a singleton. That is, the tuples in T have the same value in attributes A . In a recoding, the tuples in an equivalence class (of a set of attributes) and those in another equivalence class are recoded into the lowest common ancestors along the hierarchies. One subtle point is that this recoding does not require the entire equivalence class to be recoded, as long as anonymity can be achieved. Hence, both original and generalized values may co-exist in the recoded dataset.

Example 2. Let us revisit the dataset in Figure 1(a) and the emerging pattern (`MSE`, `manager`). The emerging pattern is related to the attributes of education background (`Edu.`) and occupation (`Occup.`). Regarding (`Edu.`, `Occup.`), the equivalence classes in D_2 are $\{\{5, 8, 9\}, \{6\}, \{7\}\}$, where the numbers are the IDs. In multidimensional local recoding, we may recode the `Edu.` attribute of the subset of $\{2, 3, 4\}$ and $\{5, 8,$

9}. For example, we may recode $\{2, 3, 4\}$ with $\{8, 9\}$. and we may recode BA and MSE into degree holder Deg. The growth rate of $(\text{Deg.}, \text{manager})$ in the recoded dataset is $75\%/40\% = 1.875$. Hence, $(\text{Deg.}, \text{manager})$ is not ρ -emerging when $\rho = 3$. In addition, after such a recoding, all BA, MSE and Deg. appear in the recoded dataset.

Other notions of recoding, including single-dimensional global recoding [5,14,18,26] and multidimensional global recoding [19], generalize values in a relatively coarse granularity and very often result in over-generalization.

5 Metric for Multidimensional Local Recoding

In this section, we define an utility gain (`util_gain`) to quantify the effectiveness of a local recoding. `util_gain` will guide the process for hiding emerging patterns `local-recoding`, in Figure 3. A recoding is effective if (i) the distortion of frequent itemsets is small and (ii) the reduction in the growth rate of emerging patterns is large.

Metric for the distortion of frequent itemsets. For presentation clarity, we will present our proposed metric for global recoding followed by its adaption for local recoding.

(A) *Distortion metric for single-dimensional global recoding.* Single-dimensional global recoding performs recoding on the domain of an attribute in a dataset. It recodes a value of the domain to another (generalized) value. That is, if a particular value is recoded, the attribute of all the tuples containing that particular value will be recoded. No frequent itemsets disappear but may appear in a generalized form after a recoding (Figure 2(a)).

Inspired by the distortion metric proposed in [20], we propose a metric for measuring the *recoding distance* ($RDist$) between the original and generalized form of a tuple. Then, we define a metric called *value distance* (VD) which measures the distance between the original and generalized form of a single attribute value. We will use VD as a building block for the definition of distortion (RD). Since a recoding always assumes an attribute hierarchy, we may skip the hierarchy H when it is clear from the context.

Definition 3. Recoding Distance ($RDist$): Consider a recoding G which generalizes a set of non-generalized values V to a single generalized value v_g , where V is the set of values under v_g in an attribute hierarchy. The recoding distance of G $RDist(G)$ is $|V|$.

Definition 4. Value Distance (VD): Let h be the height of an attribute hierarchy H , where level h and 0 is the most generalized and specific level, respectively. Consider a value v at level p which is generalized to a value v' at level q . Let G_i denotes the recoding that generalizes an attribute from level $i - 1$ to i , where $0 < i \leq h$. The value distance between v and v' is: $VD(v, v') = \sum_{i=p}^q \frac{i \cdot RDist(G_i)}{h}$.

Value distance is unfavorable to recoding (i) many values into one single generalized value; and (ii) a value into a generalized value that is close to the top of the hierarchy. This gives a measure for the distortion of a value due to a recoding. Next, we extend VD to measure the distortion of a tuple and frequent itemsets due to recoding.

Definition 5. Tuple Distance (TD): Suppose a tuple $f = (v_1, v_2, \dots, v_n)$ is generalized to $f' = (v'_1, v'_2, \dots, v'_n)$. The tuple distance between f and f' is defined as: $TD(f, f') = \sum_{i=1}^n VD(v_i, v'_i)$.



Fig. 2. (a) An attribute hierarchy of Edu. ; and (b) the relationship between F and F' in (i) global recoding and (ii) local recoding

Definition 6. Recoding Distortion (RD): Let $F = \{f_1, f_2 \dots f_n\}$ be a set of σ -frequent itemsets in D and $F' = \{f'_1, f'_2 \dots f'_m\}$ be the set of σ -frequent itemsets in D' , where $m \leq n$. The corresponding frequent itemset of f_i due to global recoding is denoted as $f'_j = G(f_i)$. The recoding distance between F and F' is defined as: $RD(F, F') = \sum_{i=1}^n TD(f_i, G(f_i))$.

Example 3. Following up Example 2, we compute the (global) recoding distortion of generalizing (MSE, manager) to (Deg., manager). Figure 2 shows the attribute hierarchy of Edu. The recoding distortion $RD(\{(MSE, manager)\}, \{(Deg., manager)\})$, RD , can be computed as follows: $RD = TD((MSE, manager), (Deg., manager)) = VD(MSE, Deg.) + VD(manager, manager) = \sum_{i=0}^2 \frac{i \cdot RDist(G_i)}{h} = \frac{1 \times 3}{2} + \frac{2 \times 0}{2} = 1.5$

(B) *Distortion metric for local recoding.* Since single-dimensional global recoding may often lead to over-generalization, we adopted local recoding. We remark that there are two unique challenges in computing recoding distance for local recoding (Figure 2(b)).

(B.i) *An itemset in F having no correspondence in F' .* Local recoding allows part of the tuples that share the same attribute values to be generalized. Such recoding may generalize some supporting tuples of a frequent itemset which makes the itemset (in the original or generalized form) not frequent anymore. To address this, we measure the distortion of the disappeared frequent itemset to the most general form. The reason is that the frequent itemset can be trivially recovered when the entire dataset is generalized to the most general form.

Specifically, given a f in F , if we cannot find a corresponding frequent itemset in F' , we first create an itemset, f_{max} , which contains the most generalized value of each value in f . Then, RD of f is the recoding distance between f and f_{max} .

Example 4. Reconsider the dataset in Figure 1(a). Suppose we recode the Edu. attribute of Records 1 and 2 to Deg. When σ is 40%, $\{BA\}$ and $\{MSE\}$ were frequent itemsets (not minimal for illustration purposes) before recoding and there is no frequent itemset after recoding.

(B.ii) *An itemset in F having more than one corresponding itemset in F' .* As discussed, local recoding may generalize a frequent itemset f in F into more than one correspondence in F' , denoted as F_f . In this case, we calculate the tuple distance of each of the corresponding itemsets in F_f and take the *minimum* tuple distance as the tuple distance of f . This is because the itemset with the minimum tuple distortion has been revealed in F' , even when there may be more distorted itemsets.

With the above, we have the following recoding distance for local recoding:

Definition 7. Recoding Distance for Local Recoding (RD_{local}): Let $F = \{f_1, f_2 \dots f_n\}$ be a set of σ -frequent itemsets in D and $F' = \{f'_1, f'_2 \dots f'_m\}$ be the set of σ -frequent itemsets in D' . The corresponding frequent itemset(s) of f_i due to local recoding is denoted as $F_f = G(f_i)$. The recoding distance between F and F' is: $RD_{local}(F, F') = \frac{1}{n} \sum_{i=1}^n \frac{TD_{local}(f_i, G(f_i))}{TD_{local}(f_i, f_{max})}$, where $TD_{local}(f_i, G(f_i)) =$

$$\begin{cases} \theta_q \times TD(f_i, G(f_i)), & \text{if } f \text{ has 1 correspondent in } F' \\ (1 - \theta_q) \times TD(f_i, f_{max}), & \text{if } f \text{ has no correspondent in } F' \\ \theta_q \times \min(TD(f_i, f_j)), & \text{where } f_j \in G(f_i), \text{ otherwise,} \end{cases}$$

θ_q is a parameter that specifies the relative importance of the itemset distortion and missing itemsets, due to G , and $TD_{local}(f_i, f_{max})$ is for normalizing RD_{local} .

Example 5. Following up Example 4, when σ is 30%, the frequent itemset $\{(BA)\}$ corresponds to the frequent itemsets $\{(BA), (Deg)\}$ in the recoded datasets.

Metric for the change in growth rate. The second component of our heuristics concerns the growth rate of the emerging patterns. Intuitively, we aim at a recoding that significantly reduces the growth rate of the emerging patterns in order to hide them. Given an emerging pattern e and the result of a local recoding e' , the reduction in growth rate due to the recoding can be easily defined as the growth rate of e minus the growth rate of e' . Then, the growth rate reduction of E due to a local recoding G , denoted as $RG_{local}(G, E)$, can be defined as the total reduction in the growth rate of e in E divided by the total growth rate of e in E .

Putting all these together. Based on the derivations above, the utility gain due to a local recoding G for a set of emerging patterns E is defined as:

$$\text{util_gain}(G, E) = \theta_p RG_{local}(G, E) - (1 - \theta_p) RD_{local}(F, F').$$

The two parameters θ_p and θ_q , where $\theta_p, \theta_q \in [0, 1]$, are specified by users.

6 Algorithm for Hiding Emerging Patterns

In this section, we present the overall algorithm `hide-eps` (shown in Figure 3) for hiding emerging patterns with a minimal distortion in frequent itemsets.

Overview of `hide-eps`. The main ideas of `hide-eps` can be described as follows. First, we determine the emerging patterns to be hidden (Line 02) and the frequent itemsets (incrementally) to be preserved (Line 04). We refer the details of Lines 02 and 04 to previous works [29, 34], since our focus is on *hiding* emerging patterns. For each selected emerging pattern (Line 05), we carry out a local recoding `local-recoding-sa` (Line 06, more details soon). This process (Lines 03-08) is repeated until there is no more emerging pattern to hide (Line 03). To avoid sub-optima, we present `hide-eps` in the style of simulated annealing search (Lines 01, 07 and 18).

Next, we discuss the details of the major steps of the algorithm.

Mining emerging patterns (`mine-eps`, Lines 02 and 08). During recoding, we invoke `mine-eps` [34] to determine if all the emerging patterns have been hidden (Line

```

Procedure hide-eps
Input: two datasets,  $D_i$  and  $D_j$ , the threshold of growth rate and frequent itemsets  $\rho$  and  $\sigma$ ,
         the heuristic parameters  $p$  and  $q$ , an initial temperature  $t_0$  and the cooling parameter  $\alpha$ 
Output: transformed datasets  $(D_i, D_j)$ 
01  $t = t_0$  H.init() // initialization
02  $E := \text{mine-eps}(D_i, D_j, \rho)$  // [34]
03 while  $E \neq \emptyset$ 
04    $F := \text{incr-mine-fis}(D_i \cup D_j, \sigma)$  // [29]
05    $e := \text{next-overlapping-ep}(E)$ 
06   if  $e$  is not null
07      $(D_i, D_j) := \text{local-recoding-sa}(D_i, D_j, e, F, t, \alpha, H)$ 
08    $E := \text{mine-eps}(D_i, D_j, \rho)$ 
09 return  $(D_i, D_j)$ 

Procedure local-recoding-sa
Input: two datasets,  $D_i$  and  $D_j$ , an emerging pattern  $e$ , a frequent itemset  $F$ , a temperature  $t$ , a hashtable  $H$  for caching
         the utility gain of local recodings
Output: transformed datasets  $(D_i, D_j)$ 
10 let  $D_i$  be the dataset where  $e$  has a higher support
11 denote  $c_e$  be the equiv. class of  $e$  in  $D_i$ 
12 compute equiv. classes  $C$  of  $D_j$  of the attributes of  $e$ 
   // compute the utility gain of the local recoding of each equiv. class  $c_k$  in  $C$  with  $c_e$ 
13 for each  $c_k$  in  $C$ 
14   if  $\text{determine-missing-FIS}(G_{(c_e, c_k)}, F) = \emptyset$  then
15     if  $\text{determine-new-singleton-eps}(G_{(c_e, c_k)}, E) = \emptyset$  then
16       if  $H[c_e][c_k]$  is null then
17          $H[c_e][c_k] := \text{util\_gain}(G_{(c_e, c_k)}, E)$  // Section 5
18  $c_k := \text{get-next-step-sa}(c_e, H, t)$ 
19  $D_i := \text{multi-local-recode}(D_i, c_e, c_k)$  // Section 4
20  $D_j := \text{multi-local-recode}(D_j, c_e, c_k)$ 
21 return  $(D_i, D_j)$ 

```

Fig. 3. The overall algorithm

08). To the best of our knowledge, there does not exist any incremental algorithm for mining emerging patterns. As verified by our experiments, `mine-eps` is a bottleneck of runtime of `hide-eps`. However, it should be remarked that the emerging patterns may often be altered slightly by most local recodings, in practice. To address this performance issue, in Section 7 we tested another version of `hide-eps`, where `mine-eps` is invoked *only* when all previously mined emerging patterns have been hidden.

Incremental mining of frequent itemsets (`incr-mine-fis`, **Line 04**). A local recoding may alter the existing frequent itemsets. Figure 2 (b) (ii) shows an example. Since a local recoding changes only part of D_1 and D_2 , we need not mine the dataset from scratch but do it incrementally using algorithms like [29].

Selecting emerging patterns for recoding (`next-overlapping-ep`, **Line 05**). Given a set of emerging patterns E , `next-overlapping-ep` determines the emerging pattern e in E such that it overlaps with the remaining emerging patterns the most. The intuition is that reducing the growth rate of e may indirectly reduce the growth rate of the overlapping emerging patterns as well. We verify with some experiments that this approach consistently outperforms a number of other strategies (see Section 7).

Determining the next local recoding (`local-recoding-sa`, **Lines 06, 10-21**). Assume that c_e is the equivalence class of the emerging pattern e . We first compute the

equivalence classes of the attributes of e to generalize with c_k (Line 12). We apply the utility gain defined in Section 5 to determine the goodness of local recodings (Line 16). Since there can be many equivalence classes, this is another bottleneck of runtime. We speed up that step using (i) a hashtable (Lines 01 and 16-17) to cache the utility gain values computed, and (ii) two filters on equivalence classes (Lines 14 and 15). The first filter is that we ignore the equivalence classes that would result in missing frequent itemsets, which is obviously undesirable. This can be computed by the change in support of itemsets in F due to a local recoding. Second, we discard a local recoding that would yield new single-attribute emerging patterns. This can be computed by determining the growth rate of the equivalence classes with the attributes of e . We did not compute possible new multi-attribute emerging patterns because of its daunting complexity.

With the utility gain of equivalence classes, we use a simulated annealing search (`get-next-step-sa`, Line 18), as a black box, to get the next local recoding.

Analysis. Given that A_E is the set of attributes of the emerging pattern E , and \mathcal{D} and \mathcal{H} are the overall domain size and the maximum height of the hierarchy of all possible A_E 's, respectively. In the worst case, there can be $O(\mathcal{D} \times \mathcal{H})$ possible recodings. Also, local recoding allows tuple-wise recoding and thus in the worst case, $|D_1| + |D_2|$ recoding operations can be carried out. Thus, the search space of finding the optimal recoding is $O((|D_1| + |D_2|) \times \mathcal{D} \times \mathcal{H})$. This work proposes a heuristic search for this problem. While the loop (Lines 03-08) may repeat many times in the worst case, the number of iterations needed was found small in practice. As discussed, `mine-eps` and the computation of `util_gain` are the bottlenecks of runtime. The runtime of the former is experimentally evaluated in [34]. The time complexity for the latter is $O(|A_e| \times \mathcal{D} \times \mathcal{H} \times |F|)$, where $e \in E$, $|A_e| \times \mathcal{D} \times \mathcal{H}$ is the number of possible equivalence classes and for each class, $O(|E|)$ and $O(|F|)$ are used to compute RG_{local} and RD_{local} , respectively.

7 Experimental Evaluation

To verify the effectiveness and efficiency of our proposed algorithms, we conducted several experiments on `Adult` dataset [30] using the attribute hierarchies from [14].

We implemented our algorithm in `JAVA SDK 1.6`¹. We have run our experiments on a PC with a Quad CPU at 2.4GHz and 3.25GB RAM. The PC is running Windows XP operating system. We have used system calls to invoke the implementations from [34] and [23] to determine emerging patterns and frequent itemsets, respectively.

The simplified `Adult` dataset contains 8 attributes. We removed the records with missing values. The records in the dataset were divided into two classes - people who have more than \$50k/year (7508 records) and people who do not (22654 records).

The effect of the parameters θ_p and θ_q . The first experiment is to verify the effects on the parameters θ_p and θ_q on the heuristic algorithm. In this experiment, we do not apply any filter (i.e., Lines 14-15 in `hide-eps`) and SA search (Line 18) in order to observe the effects on the parameters clearly. Instead, we used a `Greedy` search. The *performance* was presented in “*distortion on the frequent itemsets / the number of missing frequent*

¹ The implementation is available at <http://www.comp.hkbu.edu.hk/~michael/source.rar>

itemsets”, unless otherwise specified. When σ and ρ were set to 40% and 5, respectively, the frequent itemsets obtained are: {(Husband, Married-civ-spouse, Male), (Married-civ-spouse, White), (Married-civ-spouse, United-States), (Male, Private, White), (Male, Private, United-States), (Male, White, United-States), (Private, White, United-States)}.

When we recode all attributes to All in the frequent itemsets, we obtain the maximum distortion of the frequent itemsets of Adult 623.1.

To illustrate the possible effect of recordings on the resulting frequent itemsets, we list the frequent itemsets after we applied Greedy, where θ_p and θ_q were both set to 0.8: {(Relationship, United-States), (Married, White, United-States), (Male, Private, White), (Male, Private, United-States), (Male, White, United-States), Private, White, United-States}. The distortion obtained is 21.5 (out of 623.1).

Next, we varied θ_p and θ_q and measured the performance of Greedy. The performance is shown in Table 1 (LHS). The average runtime is 58 mins and 16 out of 58 mins is spent on mining EPS. The average number of local recordings is 14.

Table 1. The effect of the parameters in util_gain on Greedy’s performance (LHS) and the performance of the determine-new-singleton-eps filter (RHS)

| $\theta_p \setminus \theta_q$ | 0.0 | 0.2 | 0.4 | 0.6 | 0.8 | 1.0 |
|-------------------------------|--------|--------|--------|--------|--------|--------|
| 0 | 73.3/1 | 50.0/1 | 50.0/1 | 50.0/1 | 50.0/3 | 50.0/1 |
| 0.2 | 73.3/1 | 59.7/1 | 38.2/1 | 38.2/1 | 46.5/1 | 11.5/4 |
| 0.4 | 73.3/1 | 59.7/1 | 38.2/1 | 21.5/1 | 46.5/1 | 11.5/4 |
| 0.6 | 73.3/1 | 59.7/1 | 21.5/1 | 38.2/1 | 46.5/1 | 11.5/4 |
| 0.8 | 73.3/1 | 59.7/1 | 21.5/1 | 21.5/1 | 38.2/1 | 0/5 |
| 1.0 | 11.5/3 | 11.5/3 | 11.5/3 | 11.5/3 | 11.5/3 | 11.5/3 |

| $\theta_p \setminus \theta_q$ | 0.0 | 0.2 | 0.4 | 0.6 | 0.8 | 1.0 |
|-------------------------------|--------|--------|--------|--------|--------|--------|
| 0 | 62.7/0 | 43.7/1 | 43.7/1 | 35.8/3 | 35.8/3 | 11.1/4 |
| 0.2 | 62.7/0 | 32.0/1 | 32.0/1 | 15.7/2 | 9.7/3 | 7.5/4 |
| 0.4 | 62.7/0 | 32.0/1 | 16.8/1 | 21.9/2 | 9.7/3 | 7.5/4 |
| 0.6 | 71.5/0 | 32.0/1 | 16.8/1 | 21.9/2 | 7.2/3 | 0/5 |
| 0.8 | 71.5/0 | 32.0/1 | 16.8/1 | 15.7/2 | 7.2/3 | 0/5 |
| 1.0 | 73.1/1 | 73.1/1 | 73.1/1 | 73.1/1 | 73.1/1 | 73.1/1 |

We make four observations from Table 1 (LHS). Firstly, when θ_p is set to 0, the algorithm concerns only distortion (regardless the corresponding reduction in growth rate) during local recordings. In such a case, the distortion on frequent itemsets of various θ_q ’s is in general large. The reason is that when θ_p is 0, the heuristics does not effectively reduce the growth rate and the search takes more recordings that are not relevant to hiding emerging patterns. Secondly, when θ_p is 1, the algorithm concerns only the reduction in growth rate. Note that 3 out of 7 frequent itemsets have disappeared. Thirdly, when we set θ_q to 1, we do not concern the missing frequent itemsets. Hence, more frequent itemsets were lost. Similarly, when we set θ_q to 0, we concern only the frequent itemsets that do not disappear. Since there is one missing frequent itemset during recordings, overlooking this led to more distortion. Fourthly, we found that a significant runtime (42 mins) was spent on calculating the utility gain of equivalence classes. The reason is that no filters had been applied yet.

In all, we found that on Adult, Greedy yields frequent itemsets with a distortion 21.5 (out of 623.1) and 1 missing frequent itemset when both θ_p and θ_q are moderate.

The effect of the determine-new-singleton-eps filter. This filter is used to avoid recoding equivalence classes that would yield new single-attribute EPS (Line 15 of hide-eps). The performance of Greedy with this filter is shown in Table 1 (RHS).

We observe from Table 1 (RHS) that there are similar trends on the performance with various θ_p and θ_q . The distortion is sometimes smaller but the missing frequent itemsets may sometimes be more. However, the average runtime of this experiment is 26 mins (compared to 58 previously). Specifically, the time for computing utility gain has been reduced from 42 to 15 mins. The number of recordings reduces from 14 to 9. The runtime improvement is due to (i) the smaller number of equivalence classes for computing the utility gain and (ii) fewer (if any) new EPS generated during `hide-eps`.

The effect of the `determine-missing-FIS` filter. From the previous experiments, we note that there are missing frequent itemsets in most cases. Here, we test the effectiveness of `determine-missing-FIS` filter (Line 14). The performance is shown in Table 2 (LHS). The average runtime for computing the utility gain increased from 15 to 21 mins and the number of recordings increased from 9 to 14. At first glance, the distortion might have increased. However, there is no missing frequent itemset for all θ_p 's and θ_q 's. This improvement comes at the expense of a slight increase in runtime.

Table 2. The performance of the `determine-missing-FIS` filter (LHS) and the performance of invoking `mine-eps` only when E is empty (RHS)

| $\theta_p \setminus \theta_q$ | 0.0 | 0.2 | 0.4 | 0.6 | 0.8 | 1.0 | $\theta_p \setminus \theta_q$ | 0.0 | 0.2 | 0.4 | 0.6 | 0.8 | 1.0 |
|-------------------------------|---------|---------|---------|---------|---------|---------|-------------------------------|---------|---------|---------|---------|---------|---------|
| 0 | NA | 89.2/0 | 89.2/0 | 89.2/0 | 71.5/0 | 71.5/0 | 0 | NA | 97.2/0 | 97.2/0 | 81.3/0 | 70.1/0 | 81.3/0 |
| 0.2 | 105.3/0 | 89.2/0 | 78.6/0 | 78.6/0 | 41.5/0 | 41.5/0 | 0.2 | 105.3/0 | 97.2/0 | 97.6/0 | 81.3/0 | 59.3/0 | 59.3/0 |
| 0.4 | 105.3/0 | 78.6/0 | 50/0 | 50/0 | 41.5/0 | 41.5/0 | 0.4 | 105.3/0 | 97.6/0 | 64.9/0 | 64.9/0 | 59.3/0 | 59.3/0 |
| 0.6 | 105.3/0 | 78.6/0 | 50/0 | 50/0 | 41.5/0 | 41.5/0 | 0.6 | 105.3/0 | 78.6/0 | 64.9/0 | 64.9/0 | 59.3/0 | 59.3/0 |
| 0.8 | 105.3/0 | 78.6/0 | 61.3/0 | 61.3/0 | 41.5/0 | 41.5/0 | 0.8 | 105.3/0 | 78.6/0 | 64.9/0 | 70.1/0 | 59.3/0 | 59.3/0 |
| 1.0 | 105.3/0 | 105.3/0 | 105.3/0 | 105.3/0 | 105.3/0 | 105.3/0 | 1.0 | 105.3/0 | 105.3/0 | 105.3/0 | 105.3/0 | 105.3/0 | 105.3/0 |

The effect of calling `mine-eps` when E is empty. In the last experiment, 15 out of 36 mins was spent on mining EPS. In this experiment, we attempt to improve the runtime by hiding all existing EPS first before calling `mine-eps`, as opposed to calling `mine-eps` after each recoding. The performance is shown in Table 2 (RHS). From the result, we found that there is a slight increase in distortion. However, the time for mining EPS is reduced from 15 to 8 mins. The average runtime for computing the utility gain increased from 21 to 23 mins and the number of iterations remains unchanged.

The effect of hiding the EP with the minimum overlapping. To justify the decision of hiding the maximum overlapping EP in Line 05 of `hide-eps`, we conducted an experiment which first hides the EP with the minimum overlapping. The result is shown in Table 3 (LHS). We observed that the distortion is slightly larger than that of the maximum overlapping. However, the average runtime for computing the utility gain increased from 23 to 39 mins and the time for mining EP increased from 8 to 23 mins.

Simulated annealing search. After demonstrating the effects of various settings with Greedy, we applied SA on the algorithm (Line 18 of `hide-eps`). We set a low temperature ($T=10$) of SA with a high cooling rate ($\alpha=0.4$). Hence, SA initially has some chances to avoid local sub-optima and then converges to Greedy quickly. To explore the search space more, each SA was allowed to restart fifty times. The results are shown in Table 3 (RHS). SA introduces some randomness in the performance. Compared to the best versions (Table 2), SA often produces better results, at the expense of runtime.

Table 3. The performance of hiding the EP with the minimum overlapping (LHS) and the performance of simulated annealing search (RHS)

| $\theta_p \setminus \theta_g$ | 0.0 | 0.2 | 0.4 | 0.6 | 0.8 | 1.0 |
|-------------------------------|---------|---------|---------|---------|---------|---------|
| 0 | NA | 101.8/0 | 101.8/0 | 95.6/0 | 95.6/0 | 95.6/0 |
| 0.2 | 127.1/0 | 98.3/0 | 75.8/0 | 75.8/0 | 53.7/0 | 53.7/0 |
| 0.4 | 127.1/0 | 98.3/0 | 50.0/0 | 50.0/0 | 53.7/0 | 53.7/0 |
| 0.6 | 127.1/0 | 78.6/0 | 67.2/0 | 58.5/0 | 53.7/0 | 53.7/0 |
| 0.8 | 127.1/0 | 80.2/0 | 61.3/0 | 65.3/0 | 48.1/0 | 48.1/0 |
| 1.0 | 127.1/0 | 127.1/0 | 127.1/0 | 127.1/0 | 127.1/0 | 127.1/0 |

| $\theta_p \setminus \theta_g$ | 0.0 | 0.2 | 0.4 | 0.6 | 0.8 | 1.0 |
|-------------------------------|--------|--------|--------|--------|--------|--------|
| 0 | 67.4/0 | 27.8/0 | 58.5/0 | 50.0/0 | 44.3/0 | 23.6/0 |
| 0.2 | 80.1/0 | 62.5/0 | 22.4/0 | 53.8/0 | 47.8/0 | 52.1/0 |
| 0.4 | 84.1/0 | 81.4/0 | 55.7/0 | 29.3/0 | 53.7/0 | 37.4/0 |
| 0.6 | 85.0/0 | 50.0/0 | 97.0/0 | 31.8/0 | 40.5/0 | 22.8/0 |
| 0.8 | 64.9/0 | 45.2/0 | 30.0/0 | 59.6/0 | 39.5/0 | 32.3/0 |
| 1.0 | 84.1/0 | 31.7/0 | 47.9/0 | 44.1/0 | 28.9/0 | 51.6/0 |

8 Conclusions

We presented a heuristic local-recoding algorithm for hiding emerging patterns of a dataset while preserving its frequent itemsets as far as possible. We tested our algorithm with a benchmark dataset and showed its effectiveness.

References

1. Adam, N.R., Worthmann, J.C.: Security-control methods for statistical databases: A comparative study. *ACM Computing Surveys* 21(4), 515–556 (1989)
2. Agrawal, D., Aggarwal, C.: On the design and quantification of privacy preserving data mining algorithms. In: *PODS* (2001)
3. Bailey, J., Manoukian, T., Ramamohanarao, K.: Fast algorithms for mining emerging patterns. In: *ECML/PKDD* (2002)
4. Bayardo, J.R.: Efficiently mining long patterns from databases. In: *SIGMOD* (1998)
5. Bayardo, R., Agrawal, R.: Data privacy through optimal k-anonymization. In: *ICDE* (2005)
6. Davenport, T.H., Harris, J.G.: *Competing on Analytics: The New Science of Winning*, 1st edn. Harvard Business School Press (2007)
7. Dong, G., Li, J.: Efficient mining of emerging patterns: Discovering trends and differences. In: *SIGKDD* (1999)
8. Dong, G., Zhang, X., Wong, L.: CAEP: Classification by aggregating emerging patterns. In: Arikawa, S., Furukawa, K. (eds.) *DS 1999. LNCS (LNAI)*, vol. 1721, p. 30. Springer, Heidelberg (1999)
9. Du, Y., Xia, T., Tao, Y., Zhang, D., Zhu, F.: On multidimensional k-anonymity with local recoding generalization. In: *ICDE*, pp. 1422–1424 (2007)
10. Evfimievski, A., Strikant, R., Agrawal, R., Gehrke, J.: Privacy preserving mining of association rules. In: *SIGKDD* (2002)
11. Fan, H., Ramamohanarao, K.: A Bayesian approach to use emerging patterns for classification. In: *ADC* (2003)
12. Fung, B., Wang, K., Fu, A., Yu, P.: *Privacy-Preserving Data Publishing: Concepts and Techniques*. Chapman & Hall/CRC (2010)
13. Fung, B., Wang, K., Wang, L., Debbabi, M.: A framework for privacy-preserving cluster analysis. In: *ISI* (2008)
14. Fung, B., Wang, K., Yu, P.: Top-down specialization for information and privacy preservation. In: *ICDE* (2005)
15. Fung, B., Wang, K., Yu, P.: Anonymizing classification data for privacy preservation. *TKDE* 10(5), 711–725 (2007)
16. Ramamohanarao, H.F.K.: Pattern based classifiers. In: *WWW* (2007)

17. Kargupta, H., Datta, S., Wang, Q., Sivakumar, K.: Random-data perturbation techniques and privacy-preserving data mining. *KAIS* 7(4), 387–414 (2005)
18. LeFevre, K., Dewitt, D., Ramakrishnan, R.: Incognito: Efficient full-domain k-anonymity. In: *SIGMOD* (2005)
19. LeFevre, K., Dewitt, D., Ramakrishnan, R.: Mondrian multidimensional k-anonymity. In: *ICDE* (2006)
20. Li, J., Wong, R., Fu, A., Pei, J.: Anonymization by local recoding in data with attribute hierarchical taxonomies. *TKDE* 20(9), 1181–1194 (2008)
21. Li, T., Li, N.: On the tradeoff between privacy and utility in data publishing. In: *SIGKDD* (2009)
22. Machanavajjhala, A., Kifer, D., Gehrke, J., Vénkitasubramaniam, M.: L-diversity: Privacy beyond k-anonymity. *TKDD* 1(1), 3 (2007)
23. MAFIA. Mining Maximal Frequent Itemsets, <http://himalaya-tools.sourceforge.net/Mafia/>
24. Moustakides, G., Verykios, V.: A maxmin approach for hiding frequent itemsets. *DKE* 65(1), 75–79 (2008)
25. Oliveira, S., Zaiane, O.R.: Privacy preserving frequent itemset mining. In: *ICDM Workshop on Privacy, Security and Data Mining*, vol. 14, pp. 43–54 (2002)
26. Sweeney, L.: Achieving k-anonymity privacy protection using generalization and suppression. *IJUFKS* 10(5), 571–588 (2002)
27. Sweeney, L.: k-anonymity: A model for protecting privacy. In: *IJUFKS*, pp. 557–570 (2002)
28. Terrovitis, M., Mamoulis, N., Kalnis, P.: Privacy-preserving anonymization of set-valued data. *PVLDB* 1(1), 115–125 (2008)
29. Tobji, M.A.B., Abrougui, A., Yaghlane, B.B.: Gufi: A new algorithm for general updating of frequent itemsets. In: *CSEWORKSHOPS* (2008)
30. UCI Machine Learning Repository. Adult Datas, <http://archive.ics.uci.edu/ml/datasets>
31. Wang, Z., Fan, H., Ramamohanarao, K.: Exploiting maximal emerging patterns for classification. In: *AUS-AI* (2004)
32. Xu, J., Wang, W., Pei, J., Wang, X., Shi, B., Fu, A.: Utility-based anonymization using local recoding. In: *SIGKDD* (2006)
33. Xu, Y., Wang, K., Fu, A.W.-C., Yu, P.S.: Anonymizing transaction databases for publication. In: *SIGKDD* (2008)
34. Zhang, X., Dong, G., Ramamohanarao, K.: Exploring constraints to efficiently mine emerging patterns from large high-dimensional datasets. In: *SIGKDD* (2000)

Anonymizing Transaction Data by Integrating Suppression and Generalization

Junqiang Liu^{1,2} and Ke Wang¹

¹ Simon Fraser University, Burnaby, B.C. V5A 1S6, Canada

² Zhejiang Gongshang University, Hangzhou, 310018, China
{jjliu,wangk}@cs.sfu.ca

Abstract. Privacy protection in publishing transaction data is an important problem. A key feature of transaction data is the extreme sparsity, which renders any single technique ineffective in anonymizing such data. Among recent works, some incur high information loss, some result in data hard to interpret, and some suffer from performance drawbacks. This paper proposes to integrate generalization and suppression to reduce information loss. However, the integration is non-trivial. We propose novel techniques to address the efficiency and scalability challenges. Extensive experiments on real world databases show that this approach outperforms the state-of-the-art methods, including global generalization, local generalization, and total suppression. In addition, transaction data anonymized by this approach can be analyzed by standard data mining tools, a property that local generalization fails to provide.

Keywords: Anonymity, privacy, information security, transaction data.

1 Introduction

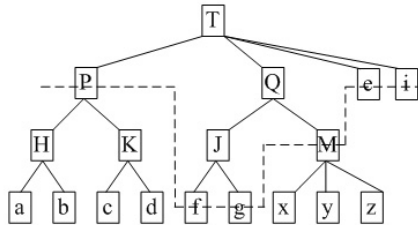
Transaction data, such as shopping transactions [1], web query logs [11], and movie ratings [10], are important sources for knowledge discovery. People are increasingly releasing transaction data to the data mining research community for discovering knowledge that helps improve services. However, transaction data contains significant amount of personal and sensitive information. The release of such data to the public or a third party could breach privacy, as highlighted by recent incidents [2][10]. Transaction data must be anonymized before release.

Recently, several works started to address the transaction data anonymization problem [4][5][13][14]. However, these works suffer from a few limitations, namely, incurring high information loss, failing to enable standard data mining tools, and introducing invalid analysis results. Let us examine those prior works using the transaction data in Fig. 1 (a) and the taxonomy in Fig. 1 (b).

Global generalization [13]. The k^m -anonymity in [13] requires that every subset of no more than m items is contained in at least k transactions. The global generalization technique (a.k.a. *full subtree generalization* [6]) is employed in [13], which is vulnerable to excessive distortion in the presence of outliers. Let k^∞ -anonymity denote k^m -anonymity with m being the longest transaction length.

| TID | transaction database D | Transactional l -anonymity, local generalization | 2^∞ -anonymity, global generalization | 2^∞ -anonymity, suppression | 2^∞ -anonymity, gen. with suppr. [ours] |
|-----|--------------------------|--|--|------------------------------------|--|
| 1 | b, c, d | T | T | *, *, d | P |
| 2 | a, f, g | P, f, g | T | a, f, g | P, f, g |
| 3 | d, f, y, z | K, f, M | T | d, f, *, * | P, f, M |
| 4 | c, d, f, x | K, f, M | T | *, d, f, * | P, f, M |
| 5 | a, b, c, f, g | P, f, g | T | a, *, *, f, g | P, f, g |
| 6 | e, i | T | T | e, * | e, * |
| 7 | e | T | T | e | e |
| 8 | i | T | T | * | * |

(a) Transactional database D and a variety of anonymization solutions



(b) Domain generalization hierarchy H_P

Fig. 1. Transactional database D , anonymizations of D , and taxonomy tree H_P

For example, to achieve 2^∞ -anonymity, as in the 4th column in Fig. 1 (a), all items are generalized to the top level because of the outlier, {e, i}.

Suppression [14]. The (h, k, p) -coherence in [14] demands that every subset of no more than p public items must be contained in at least k transactions and no more than h percent of these transactions contain a common private item. k^m -anonymity is its special case with $h = 100\%$ and $p = m$. [14] employs the total item suppression technique to enforce (h, k, p) -coherence, which incurs high information loss when the data is sparse. E.g., in the 5th column in Fig. 1 (a), all occurrences of b, c, i, x, y, and z, are removed as indicated by *.

Local generalization [5]. The *transactional k-anonymity* in [5] requires that each transaction has at least k duplicates. Such a requirement is stronger than k^∞ -anonymity and introduces much more distortion than necessary. The *multi-dimensional generalization* technique [7] is employed in [5]. But, it destroys the *domain exclusiveness* property, e.g., the 3rd column in Fig. 1 (a) shows the data anonymized by [5] where items T, P, and K coexist in the anonymized data, but their domains are not exclusive of each other. The analysis result based on such data are hard to interpret, e.g., according to such data, whenever a transaction contains K, it also contains f. But it is not true with the original data, e.g., the first transaction contains c and d (and hence K), but it does not contain f.

Band matrix method [4]. A method for grouping transactions and permuting the private items in each group to enforce l -diversity [9] is presented in [4]. However, invalid analysis results could be derived from the anonymized data. The example in [4] explained this: in the original data, all customers who bought

cream but not *meat* have also bought a *pregnancy test*; while in the data anonymized by [4], only a half of such customers have bought a *pregnancy test*.

This paper, motivated by the limitations of the prior works, proposes to integrate the global generalization technique with the total item suppression technique for enforcing k^m -anonymity. Our observation is that suppression can remove outlier items that otherwise will cause substantially generalization of many other items, and generalization can slightly generalize items that otherwise must be suppressed. While a single technique could not perform well, the integration can greatly reduce the overall information loss. Our approach has two strong properties: the anonymized data can be analyzed by standard data mining tools, and results derived from it are true in the original data. This is because both techniques preserve the domain exclusiveness property. For example, the last column in Fig. 1 (a) shows the data anonymized by our approach which suppresses item *i* and generalizes some other items.

Integrating generalization and suppression is non-trivial because the search space is much larger than only employing one of them. We propose a multi-round, top-down greedy search strategy to address the challenge. Extensive comparative experiments showed that our approach yields better data utility than the prior works and is efficient and scalable in anonymizing real world databases.

The rest of the paper is organized as follows. Section 2 describes the privacy requirement and the anonymization model, Section 3 presents the basic approach that integrates generalization with suppression, Section 4 proposes the key techniques that make our approach efficient and scalable, Section 5 evaluates the applicability of our approach, and Section 6 concludes the paper.

2 Privacy and Anonymization Model

A publisher wants to release a transaction database $D = \{t_1, t_2, \dots, t_n\}$, where each transaction t_i corresponds to an individual and contains items from an item universe $I = \{i_1, i_2, \dots, i_q\}$. An adversary tries to link a target individual to his/her transaction with a high probability. To do so, the adversary acquired knowledge from external sources. That is, the adversary knows that the transaction is in the released data and knows some items of the target individual. The publisher wants to prevent such a linking attack.

Definition 1 (Privacy threats and k^m -anonymity): A subset of items is called an *itemset*. An itemset X with $|X| \leq m$ is called a *privacy threat* if the number of transactions in D that support X , denoted by $sup(X)$, is less than a user specified anonymity threshold k , i.e., $sup(X) < k$. A transaction t *supports* X if X is a subset of t ; D *observes k^m -anonymity* [13] if there is no privacy threat supported by D . ■

Enforcing the privacy notion in Definition 1 assures that the adversary's certainty in making any linking attack is no more than $1/k$.

Anonymization solutions: To enforce the privacy notion, the *full subtree generalization* technique [6] [13] and the *total item suppression* technique [14] are

integrated to anonymize D . We assume that a taxonomy tree H_P for generalizing items is available. With the *full subtree generalization* technique, a generalization solution is defined by a cut on H_P . A cut contains *exactly one* item on every root-to-leaf path on H_P , and is denoted by *the set of such items*. E.g., $\{P, f, g, M, e, i\}$ denotes the cut depicted by a dash line on H_P in Fig. 1(b). With the *total item suppression* technique, to eliminate privacy threats in the generalized data, some constituent items of *Cut* are totally removed from all transactions. The set of items to be removed is called a *suppression scenario of Cut*.

In other words, an anonymization is defined by *Cut* and *SS*, a generalization cut and the suppression scenario associated with the cut. The anonymized data D'' is derived in two steps: first the original items in D are generalized to their taxonomic ancestors in *Cut* to get $D' = g(D, \text{Cut})$, and then items in *SS* are suppressed from D' to eliminate threats, which results in $D'' = s(D', \text{SS})$.

Running Example: Consider the transaction database D in the 2nd column in Fig. 1(a) and the taxonomy H_P in Fig. 1(b). Suppose that we enforce 2[∞]-anonymity. By generalizing D to the cut $\{P, f, g, M, e, i\}$, only one privacy threat, $\{e, i\}$, exists in the generalized data $D' = g(D, \{P, f, g, M, e, i\})$. If we suppress item i from D' , we get the anonymized data $D'' = s(D', \{i\})$ where no privacy threat exists, as shown in the last column in Fig. 1(a). ■

Anonymization causes information loss. Given *Cut* and *SS*, a generalization cut and its associated suppression scenario, $\text{cost}_G(\text{Cut})$ denotes the information loss incurred by generalizing D to get $D' = g(D, \text{Cut})$, and $\text{cost}_S(\text{SS})$ denotes that incurred by suppressing items in *SS* from D' to get $D'' = s(D', \text{SS})$. The total cost is $\text{cost}(\text{Cut}, \text{SS}) = \text{cost}_G(\text{Cut}) + \text{cost}_S(\text{SS})$.

Anonymization can be measured by a variety of metrics. As most cost metrics are additive, we can write $\text{cost}_G(\text{Cut}) = \sum_{x^* \in \text{Cut}} O(x^*) \cdot \text{IL}_G(x^*)$, and $\text{cost}_S(\text{SS}) = \sum_{x^* \in \text{SS}} O(x^*) \cdot \text{IL}_S(x^*)$, where $O(x^*)$ is the total number of occurrences in D of all leaf items that are descendants of x^* , $\text{IL}_G(x^*)$ is the generalization cost *per occurrence* of x^* , and $\text{IL}_S(x^*)$ is an extra suppression cost in addition to $\text{IL}_G(x^*)$ if x^* is suppressed.

We use *LM* [6] in our discussion, and assume that D only contains leaf items on H_P . With *LM*, $\text{IL}_G(x^*) = (\#\text{leaves}(x^*) - 1) / (\#\text{leaves}(H_P) - 1)$, where $\#\text{leaves}(x^*)$ and $\#\text{leaves}(H_P)$ denotes the number of leaves in the subtree rooted at x^* and that in the taxonomy H_P respectively. If x^* is suppressed, it is deemed that all descendants of x^* are generalized to the top level of H_P , the overall information loss per occurrence of x^* is 1, so the extra suppression cost is $\text{IL}_S(x^*) = 1 - \text{IL}_G(x^*)$. For example, for D'' in the last column in Fig. 1(a), $\text{Cut} = \{P, f, g, M, e, i\}$, $\text{SS} = \{i\}$. With *LM*, $\text{cost}_G(\text{Cut}) = 3.6$, and $\text{cost}_S(\text{SS}) = 2$. The total cost is $\text{cost}(\text{Cut}, \text{SS}) = 5.6$.

3 Integrating Generalization and Suppression

An anonymization is defined by (Cut, SS) , a generalization cut with its associated suppression scenario, and can be found by two nested loops.

As the number of cuts is exponential in the number of items and so is the number of suppression scenarios for a cut, a complete enumeration for either loop is intractable. Therefore, we present a basic approach, *heuristic generalization* with *heuristic suppression*, namely HgHs.

3.1 Top-Down Greedy Search of the Lattice of Cuts

The outer loop of HgHs enumerates generalizations (cuts) by a top-down greedy search of a lattice of all possible cuts [8], where a specific cut (child) is derived from a general cut (parent) by replacing one constituent item of the parent cut by its child items on the taxonomy tree.

Starting from the top-most cut which consists of only the root (item) of the taxonomy tree, the outer loop continues with the most promising child cut of the current cut, which is achieved by evaluating the suppression scenario for each child of the current cut by running the inner loop (*detailed in the next subsection*), and computing the anonymization cost. The outer loop stops when no child cut reduces the anonymization cost.

For example, Fig. 2 (a) describes the searching process. The outer loop starts from $cut_1 = \{T\}$ with $cost(cut_1, SS_1) = 23$ where the suppression scenario SS_1 for $cut_1 = \{T\}$. The only child of cut_1 is $cut_2 = \{P, Q, e, i\}$. There is one privacy threat, $\{e, i\}$, in $D' = g(D, cut_2)$. The suppression scenario SS_2 for cut_2 (*computed by the inner loop described in the next subsection*) is $\{i\}$. So $cost(cut_2, SS_2) = cost_G(cut_2) + cost_S(SS_2) = 6.6 + 2 = 8.6$. The search continues to evaluate the children of cut_2 . The best child is cut_4 since $cost(cut_4, SS_4) = 6.2$ while $cost(cut_3, SS_3) = 10.2$. The search stopped at $cut_6 = \{P, f, g, M, e, i\}$ with $SS_6 = \{i\}$ as no child of cut_6 reduces the cost. So, (cut_6, SS_6) defines the anonymized data D'' as shown in the last column in Fig. 2 (a).

3.2 Finding a Good Suppression Scenario for a Cut

The inner loop of HgHs is responsible for finding an item suppression scenario SS to eliminate privacy threats from $D' = g(D, Cut)$ where Cut is the cut currently being enumerated by the outer loop. SS is a subset of Cut , all occurrences of items in SS will be suppressed from D' .

To determine SS , the inner loop greedily searches the so called suppression scenario enumeration tree, *which is built per cut*. Each node on the suppression scenario enumeration tree is denoted by a *headlist* and a *taillist*. The items in *headlist* are to be *kept*, and the items not in *headlist* are to be suppressed. We also use the set notation to represent *headlist* and *taillist*. Thus, the suppression scenario represented by a node N is $Cut - N.headlist$. And its suppression cost is $\sum_{x^* \in Cut - N.headlist} O(x^*) \cdot IL_S(x^*)$. For the root node, $headlist = \{\}$ and $taillist = Cut$, i.e., all items are suppressed. The j^{th} child node C of a parent node P is derived based on the j^{th} item, i_j , in P.taillist such that $C.headlist = P.headlist \cup \{i_j\}$ and $C.taillist =$ the suffix of P.taillist after i_j .

For example, Fig. 2 (b) is the suppression scenario enumeration for cut_3 in Fig. 2 (a). N_1 is the root with $N_1.headlist = \{\}$ and $N_1.taillist = cut_3$ where

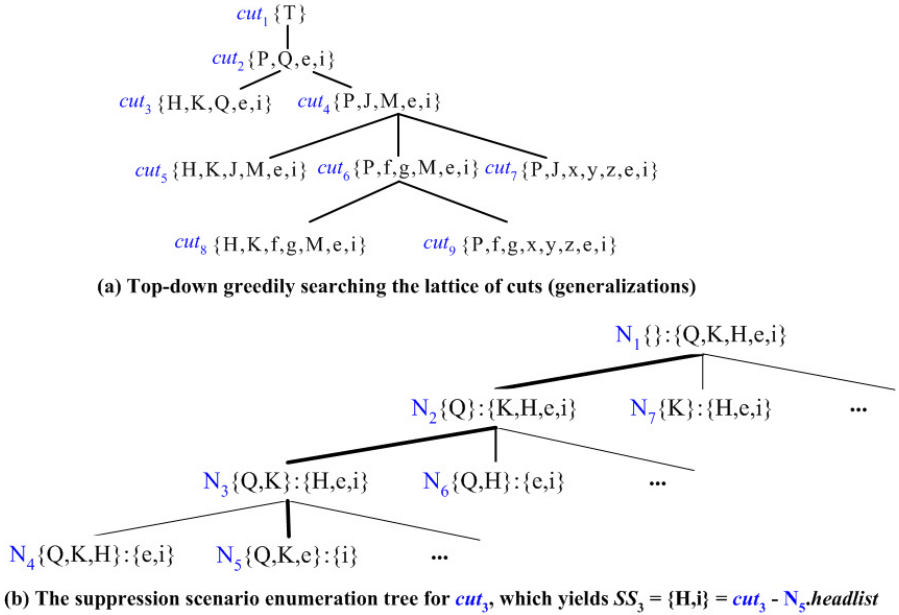


Fig. 2. Searching the cut lattice and finding the suppression scenario for each cut

items are listed in the descending order of suppression costs. N_2 is derived from N_1 , by moving the first item Q from *tailist* to *headlist*, which means that all items except Q are suppressed.

Clearly, a suppression scenario represented by a node N is *valid* if and only if no threat in D' is contained by $N.headlist$. Moreover, if a threat X is contained by $N.headlist$, then X is also contained by the *headlist* of any descendant of N . Therefore, if N is invalid, all its descendants are invalid, we can stop searching the subtree rooted at N . If items in *headlist* and *tailist* are in the descending order of suppression costs, the first valid child of any node is the most promising child of the node, as it is valid and reduces the suppression cost most.

For example, Fig. 2(b) shows how the suppression scenario SS_3 for eliminating the threats, $\{H,K,Q\}$ and $\{e,i\}$, from $D' = g(D, cut_3)$ is found. The inner loop of HgHs starts with N_1 which is valid. N_2 is the first child of its parent and is valid, and so is N_3 . And N_4 is the first child of N_3 but it is invalid. The inner loop stopped at N_5 with $N_5.headlist = \{Q, K, e\}$. So, the final suppression scenario $SS_3 = cut_3 - N_5.headlist = \{H, i\}$.

4 Addressing Efficiency and Scalability Issues

Although our basic approach HgHs presented in Section 3 enumerates a limited number of anonymizations, the work for examining each enumerated anonymization is still non-trivial. In this section, we propose the key techniques to address the efficiency and scalability issues in this regard.

4.1 Minimal Privacy Threats

The outer loop of HgHs needs to know the set of privacy threats in $D' = g(D, Cut)$ for the current Cut . If such a set is empty, all threats are already eliminated by generalizing D . If it is not, we have to suppress some generalized items to eliminate all privacy threats from D' . First, we claim that it suffices to generate the set of *minimal privacy threats*.

Definition 2 (Minimal privacy threats): A privacy threat X is a minimal threat if there is no privacy threat that is a subset of X . ■

Since every privacy threat contains some minimal privacy threat, if we eliminate all minimal privacy threats, we also eliminate all privacy threats. However, finding the set of minimal privacy threats on-the-fly is inefficient, since every threat occurs in multiple versions of the generalized data derived by different cuts and hence will be repeatedly generated while the number of cuts to be enumerated is still quite large.

Our approach is to generate the set of the minimal privacy threats supported by all cuts on the taxonomy H_P in an initialization step. For Cut being enumerated by the outer loop, we can retrieve privacy threats relevant to Cut from that set instead of generating $D' = g(D, Cut)$ and mining D' on-the-fly. Given a suppression scenario SS for Cut , to see if all threats are removed from $D'' = s(D', SS)$, we check if no relevant threat is contained in $Cut - SS$.

For the running example, there are 25 threats in the set of the minimal privacy threats, from which we can retrieve the threats, $\{H, K, Q\}$ and $\{e, i\}$, relevant to cut_3 in Fig. 2 (a), for searching suppression scenarios in Fig. 2 (b).

4.2 A Multi-round Approach

The set of the minimal privacy threats supported by all cuts on the taxonomy H_P could be huge when H_P is of a large scale and the maximum size m of privacy threats is large, which makes HgHs not scalable. We propose a multi-round approach, mHgHs, to address the scalability issue.

To find a solution, mHgHs runs HgHs in m rounds. The 1st round finds $(Cut_{best}^1, SS_{best}^1)$ on the original taxonomy H_P , which defines an anonymization observing k^1 -anonymity. The i -th round finds $(Cut_{best}^i, SS_{best}^i)$, which defines an anonymization observing k^i -anonymity, on the reduced taxonomy H_P^{i-1} that is derived by removing nodes under Cut_{best}^{i-1} . Clearly, Cut_{best}^i is above Cut_{best}^{i-1} . In other words, mHgHs performs anonymization progressively. Each round works on a reduced taxonomy based on the precedent round, so the set of the minimal privacy threats supported by all cuts for each round is under control.

For the running example, mHgHs first finds $Cut_{best}^1 = \{a, b, c, d, f, g, M, e, i\}$ with $SS_{best}^1 = \{\}$, and gets H_P^1 by removing nodes under Cut_{best}^1 . Then, mHgHs works on H_P^1 , and so on. After five rounds, mHgHs finds $Cut_{best}^5 = \{P, f, g, M, e, i\}$ with $SS_{best}^5 = \{i\}$, which conforms 2⁵-anonymity.

5 Experimental Evaluation

Our major goal is to investigate if our approach preserves more data utility than others approaches, and if our algorithm is scalable and efficient. We evaluate our algorithm mHgGs by comparing it with several state-of-the-art algorithms, the local generalization algorithm LG [5], the global generalization algorithm AA [13], and the suppression algorithm MM [14]. The executables of AA and MM were provided by the authors. We implemented LG as it is not available.

The POS dataset [15] and the AOL web query log dataset [11] are used in the experiments. The taxonomy tree for the POS dataset was created by [13]. We preprocessed the AOL dataset using WordNet [3] in creating the taxonomy tree. The AOL dataset is divided into 10 subsets. We use the first subset to evaluate the basic features of all algorithms, and use all subsets to evaluate scalabilities. We measure information loss by *NCP*, a variant of *LM* [6], as it was used by AA and LG. Experiments were performed on a PC with a 3.0 GHz CPU and 3.2 GB RAM. In the experiments, the default setting is $k = 5$, $m = 7$.

5.1 Information Loss Evaluation

Fig. 3(a)-(b) show the information loss on the POS dataset. Fig. 3(c)-(d) show that on the AOL dataset. Among all the 4 algorithms, the information loss by MM is the highest for all cases with $m \geq 2$, which is between 7.5% and 70% on POS and between 46% and 96% on AOL. This is consistent with the finding in [14] that MM is not good for sparse datasets as the POS dataset is quite sparse while the AOL dataset is even sparser.

The information loss by AA is the second highest in general. In some cases on POS (with a small m and a large k), LG incurs a little bit more. The information loss by AA on AOL is strikingly high, all around 39% even for $m=1$. Because the AOL dataset is extremely sparse, a lot of very infrequent items spread over the taxonomy. They have to be generalized to high levels, which brings their siblings to the same ancestors by AA. This situation is similar to our motivation example where as items e and i are infrequent, their siblings, P and Q, although quite frequent, have to be generalized to the top level together with e and i by AA. In such cases, suppressing a few outlier items will reduce information loss. This motivates our approach.

The information loss by LG is the third highest. As we pointed out in Section 1, LG exerts excessive distortion as it enforces the *transactional k-anonymity* principle which is too strong to be necessary. LG does not make use of m . So, the curves of LG with a varying m are all horizontal lines.

Our algorithm, mHgGs, incurs the least information loss which is the advantage of integrating suppression and generalization. The data utility gain of mHgGs over LG is moderate on POS, but it is significant on AOL. All the information loss with mHgGs is under 10% on AOL, while the worst case with LG is 27%, e.g., that by mHgGs is around 7.9% with $k = 5$ and $m \geq 5$ on AOL as in Fig. 3(d), while that with LG is 12%. The gap increases when enforcing a more restrictive privacy requirement, e.g., that by mHgGs with $k = 50$ and $m \geq 5$ on

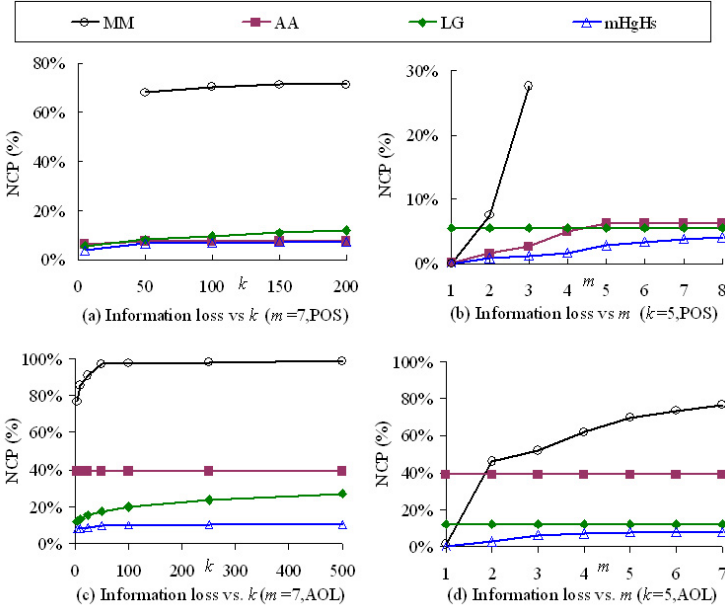


Fig. 3. Information loss of algorithms MM, AA, LG, and mHgHs on POS and AOL

AOL is 9% as in Fig. 3(c), while that by LG is 17%. Notice that the information loss reported for mHgHs is also computed on the *original* taxonomy.

5.2 Efficiency and Scalability Evaluation

We evaluated the efficiencies of all the algorithms. LG is the most efficient because it employs a divide-and-conquer approach. But, it comes with an expense, i.e., the anonymized data does not observe the domain exclusiveness as discussed in Section 1. Our algorithm, mHgHs, is the second most efficient. Although mHgHs is less efficient than LG, the significant gain in data utility by mHgHs over LG is worth the longer runtime. AA and MM are less efficient. This is because the breadth-first search approaches they employ are not efficient in dealing with privacy threats with a large size. One exception is that AA is as efficient as LG on AOL because the search space is greatly pruned by AA, which unfortunately results in the second highest information loss on AOL.

We also evaluated the scalabilities of algorithms on all the 10 subsets of the AOL query logs. The result showed that our algorithm mHgHs is quite scalable.

6 Conclusion

This paper proposed to integrate generalization and suppression to enhance data utility in anonymizing transaction data. We presented a multi-round, top-down greedy search algorithm to address the performance issues. Extensive experiments show that our approach outperforms the state-of-the-art approaches.

Acknowledgements. The research is supported in part by the Natural Sciences and Engineering Research Council of Canada, in part by the Science and Technology Development Plan of Zhejiang Province, China (2006C21034), and in part by the Natural Science Foundation of Zhejiang Province, China (Y105700).

We thank Harshwardhan Agarwal for his help in experimental evaluation.

References

1. Agrawal, R., Imielinski, T., Swami, A.: Mining Association Rules between Sets of Items in Large Databases. In: SIGMOD 1993 (1993)
2. Barbaro, M., Zeller, T.: A Face Is Exposed for AOL Searcher No. 4417749. New York Times (August 9, 2006)
3. Fellbaum, C.: WordNet, An Electronic Lexical Database. MIT Press, Cambridge (1998)
4. Ghinita, G., Tao, Y., Kalnis, P.: On the Anonymization of Sparse High-Dimensional Data. In: ICDE 2008 (2008)
5. He, Y., Naughton, J.: Anonymization of Set-valued Data via Top-down Local Generalization. In: VLDB 2009 (2009)
6. Iyengar, V.: Transforming Data to Satisfy Privacy Constraints. In: KDD 2002 (2002)
7. LeFevre, K., DeWitt, D., Ramakrishnan, R.: Mondrian Multidimensional k -Anonymity. In: ICDE 2006 (2006)
8. Liu, J., Wang, K.: On Optimal Anonymization for l^+ -Diversity. In: ICDE 2010 (2010)
9. Machanavajjhala, A., Gehrke, J., Kifer, D., Venkitasubramanian, M.: l -Diversity: Privacy beyond k -Anonymity. In: ICDE 2006 (2006)
10. Narayanan, A., Shmatikov, V.: How to Break Anonymity of the Netflix Prize Dataset. ArXiv Computer Science e-prints (October 2006)
11. Pass, G., Chowdhury, A., Torgeson, C.: A Picture of Search. In: The 1st Intl. Conf. on Scalable Information Systems, Hong Kong (June 2006)
12. Samarati, P., Sweeney, L.: Generalizing Data to Provide Anonymity When Disclosing Information. In: PODS 1998 (1998)
13. Terrovitis, M., Mamoulis, N., Kalnis, P.: Privacy Preserving Anonymization of Set-valued Data. In: VLDB 2008 (2008)
14. Xu, Y., Wang, K., Fu, A., Yu, P.S.: Anonymizing Transaction Databases for Publication. In: KDD 2008 (2008)
15. Zheng, Z., Kohavi, R., Mason, L.: Real World Performance of Association Rule Algorithms. In: KDD 2001 (2001)

Satisfying Privacy Requirements: One Step before Anonymization

Xiaoxun Sun¹, Hua Wang¹, and Jiuyong Li²

¹ Department of Mathematics & Computing
University of Southern Queensland, Australia
{sunx,wang}@usq.edu.au

² School of Computer and Information Science
University of South Australia, Australia
jiuyong.li@unisa.edu.au

Abstract. In this paper, we study a problem of privacy protection in large survey rating data. The rating data usually contains both ratings of sensitive and non-sensitive issues, and the ratings of sensitive issues include personal information. Even when survey participants do not reveal any of their ratings, their survey records are potentially identifiable by using information from other public sources. We propose a new (k, ϵ, l) -anonymity model, in which each record is required to be similar with at least $k - 1$ others based on the non-sensitive ratings, where the similarity is controlled by ϵ , and the standard deviation of sensitive ratings is at least l . We study an interesting yet nontrivial satisfaction problem of the (k, ϵ, l) -anonymity, which is to decide whether a survey rating data set satisfies the privacy requirements given by users. We develop a slice technique for the satisfaction problem and the experimental results show that the slicing technique is fast, scalable and much more efficient in terms of execution time than the heuristic pairwise method.

1 Introduction

The problem of privacy-preserving data publishing has received a lot of attention in recent years. Privacy preservation on tabular data has been studied extensively. A major category of privacy attacks on relational data is to re-identify individuals by joining a published table containing sensitive information with some external tables. Most of existing work can be formulated in the following context: several organizations publish detailed data about individuals (e.g. medical records) for research or statistical purposes [14,10,9,13].

Privacy risks of publishing microdata are well-known. Famous attacks include de-anonymisation of the Massachusetts hospital discharge database by joining it with a public voter database [14] and privacy breaches caused by AOL search data [5]. Even if identifiers such as names and social security numbers have been removed, the adversary can use linking [12], homogeneity and background attacks [10] to re-identify individual data records or sensitive information of individuals. To overcome the re-identification attacks, k -anonymity was proposed

[12,14]. Specifically, a data set is said to be k -anonymous ($k \geq 1$) if, on the quasi-identifier attributes, each record is identical with at least $k - 1$ other records. The larger the value of k , the better the privacy is protected. Several algorithms are proposed to enforce this principle [2,6,7,8]. Machanavajjhala et al. [10] showed that a k -anonymous table may lack of diversity in the sensitive attributes. To overcome this weakness, they propose the l -diversity [10]. However, even l -diversity is insufficient to prevent attribute disclosure due to the skewness and the similarity attack. To amend this problem, t -closeness [9] was proposed to solve the attribute disclosure vulnerabilities inherent to previous models.

Recently, a new privacy concern has emerged in privacy preservation research: how to protect individuals' privacy in large survey rating data. Though several models and many algorithms have been proposed to preserve privacy in relational data (e.g., k -anonymity [12,14], l -diversity [10], t -closeness [9], etc.), most of the existing studies are incapable of handling rating data, since the survey rating data normally does not have a fixed set of personal identifiable attributes as relational data, and it is characterized by high dimensionality and sparseness. The survey rating data shares the similar format with transactional data. The privacy preserving research of transactional data has recently been acknowledged as an important problem in the data mining literature [3,15,16]. To our best knowledge, there is no current research addressing the issue of how to efficiently determine whether the survey rating data satisfies the privacy requirement.

2 Motivation

On October 2, 2006, Netflix, the world's largest online DVD rental service, announced the \$1-million Netflix Prize to improve their movie recommendation service [4]. To aid contestants, Netflix publicly released a data set containing 100,480,507 movie ratings, created by 480,189 Netflix subscribers between December 1999 and December 2005. Narayanan and Shmatikov shown in their recent work [11] that an attacker only needs a little information to identify the anonymized movie rating transaction of the individual. They re-identified Netflix movie ratings using the Internet Movie Database (IMDb) as a source of auxiliary information and successfully identified the Netflix records of known users, uncovering their political preferences and other potentially sensitive information.

We consider the privacy risk in publishing survey rating data. For example, in a life style survey, ratings to some issues are non-sensitive, such as the likeness of a book. Ratings to some issues are sensitive, such as the income level. Assume that each survey participant is cautious about his/her privacy and does not reveal his/her ratings. However, it is easy to find his/her preferences on non-sensitive issues from publicly available information sources, such as personal weblog. An attacker can use these preferences to re-identify an individual and consequently find sensitive ratings of a victim. An example is given in the Table 1. In a social network, people make comments on various issues, which are not considered sensitive. Some comments can be summarized as in Table 1(b). We assume that people are aware of their privacy and do not reveal their ratings,

Table 1. (a) A published survey rating data (b) Public comments on some non-sensitive issues of some survey participants

| | non-sensitive | | | sensitive |
|-------|---------------|-------------|-------------|-----------|
| ID | issue 1 | issue 2 | issue 3 | issue 4 |
| t_1 | 6 | 1 | <i>null</i> | 6 |
| t_2 | 1 | 6 | <i>null</i> | 1 |
| t_3 | 2 | 5 | <i>null</i> | 1 |
| t_4 | 1 | <i>null</i> | 5 | 1 |
| t_5 | 2 | <i>null</i> | 6 | 5 |

(a)

| | non-sensitive issues | | |
|-------|----------------------|---------|---------|
| name | issue 1 | issue 2 | issue 3 |
| Alice | excellent | so bad | - |
| Bob | awful | top | - |
| Jack | bad | - | good |

(b)

either non-sensitive or sensitive ones. However, individuals in the supposedly anonymized survey rating data are potentially identifiable based on their public comments from other sources. For example, Alice is at risk of being identified, since the attacker knows Alice’s preference on issue 1 is ‘excellent’, by cross-checking Table 1(a) and (b), s/he will deduce that t_1 in Table 1(a) is linked to Alice, the sensitive rating on issue 4 of Alice will be disclosed. This example motivates us the following research question: Given a survey rating data set T with the privacy requirements, how to efficiently determine whether T satisfies the given privacy requirements?

3 (k, ϵ, l)-Anonymity

We assume that a survey rating data set publishes people’s ratings on a range of issues. Each survey participant is cautious about his/her privacy and does not reveal his/her ratings. However, an attacker may find a victim’s preference (not exact rating scores) by personal familiarity or by reading the victim’s comments on some issues from personal weblog. We consider that attackers know preferences of non-sensitive issues of a victim but do not know exact ratings and want to find out the victim’s ratings on some sensitive issues.

The auxiliary information of an attacker includes: (i) the knowledge of a victim being in the survey rating data; (ii) preferences of the victims on some non-sensitive issues. The attacker wants to find ratings on sensitive issues of the victim. In practice, knowledge of Types (i) and (ii) can be gleaned from an external database [11]. For example, in the context of Table 1(b), an external database may be the IMDb. By examining the anonymized data in Table 1(a), the adversary can identify a small number of candidate groups that contain the record of the victim. It will be the unfortunate scenario where there is only one record in the candidate group. For example, since t_1 is unique in Table 1(a), Alice is at risk of being identified. If the candidate group contains not only the victim but other records, an adversary may use this group to infer the sensitive value of the victim. For example, although it is difficult to identify whether t_2 or t_3 in Table 1(a) belongs to Bob, since both records have the same sensitive value, Bob’s private information is identified.

In order to avoid such attack, we propose a (k, ϵ, l) -anonymity model. The first step is to require that in the released data, every transaction should be *similar* with at least $k - 1$ other records based on the non-sensitive ratings so that no survey participants are identifiable. For example, t_1 in Table II(a) is unique, and based on the preference of Alice in Table II(b), her sensitive issues can be re-identified. Jack’s sensitive issues, on the other hand, is much safer. Since t_4 and t_5 in Table II(a) form a similar group based on their non-sensitive rating. Second is to prevent the sensitive rating from being inferred by requiring the sensitive ratings in a similar group be diverse. For instance, although t_2 and t_3 in Table II(a) form a similar group, their sensitive ratings are identical. Therefore, an attacker can immediately infer Bob’s preference on the sensitive issue without identifying which transaction belongs to Bob. In contrast, Jack’s preference on the sensitive issue is much safer than both Alice and Bob.

Given a rating data set T , each transaction contains a set of numbers indicate the ratings on some issues. Let $(o_1, \dots, o_p, s_1, \dots, s_q)$ be a transaction, $o_i \in \{1 : r, null\}$, $i = 1, 2, \dots, p$ and $s_j \in \{1 : r, null\}$, $j = 1, 2, \dots, q$, where r is the maximum rating and *null* indicates that a survey participant did not rate. o_1, \dots, o_p stand for non-sensitive ratings and s_1, \dots, s_q denote sensitive ratings. Let $T_A = \{o_{A_1}, \dots, o_{A_p}, s_{A_1}, \dots, s_{A_q}\}$, $T_B = \{o_{B_1}, \dots, o_{B_p}, s_{B_1}, \dots, s_{B_q}\}$ be the ratings for participants A and B , then dissimilarity of non-sensitive ratings ($Dis(o_{A_i}, o_{B_i})$) and sensitive ratings ($Dis(s_{A_i}, s_{B_i})$) between T_A and T_B is defined as follows:

$$Dis(o_{A_i}, o_{B_i}) = \begin{cases} |o_{A_i} - o_{B_i}| & o_{A_i}, o_{B_i} \in \{1 : r\} \\ 0 & o_{A_i} = o_{B_i} = null \\ r & \text{otherwise} \end{cases} \quad (1)$$

$$Dis(s_{A_i}, s_{B_i}) = \begin{cases} |s_{A_i} - s_{B_i}| & s_{A_i}, s_{B_i} \in \{1 : r\} \\ r & s_{A_i} = s_{B_i} = null \\ r & \text{otherwise} \end{cases} \quad (2)$$

Definition 1 (ϵ -proximate). Given a survey rating data set T with a small positive number ϵ , two transactions $T_A = \{o_{A_1}, \dots, o_{A_p}, s_{A_1}, \dots, s_{A_q}\} \in T$ and $T_B = \{o_{B_1}, \dots, o_{B_p}, s_{B_1}, \dots, s_{B_q}\} \in T$. T_A and T_B are ϵ -proximate, if $\forall 1 \leq i \leq p, Dis(o_{A_i}, o_{B_i}) \leq \epsilon$. The set of transactions T is ϵ -proximate, if every two transactions in T are ϵ -proximate.

If two transactions are ϵ -proximate, the dissimilarity between their non-sensitive ratings is bound by ϵ .

Definition 2 ((k, ϵ) -anonymity). A survey rating data set T is said to be (k, ϵ) -anonymous if and only if every transaction is ϵ -proximate with at least $k - 1$ other transactions. The transaction t with all the other transactions that are ϵ -proximate with t in T form a (k, ϵ) -anonymous group.

Although (k, ϵ) -anonymity can protect identity, it fails to protect sensitive information. For example, in Table II(a), t_2 and t_3 are in a $(2, 1)$ -anonymous group,

but they have the same rating on the sensitive issue, thus Bob’s private information is breaching. This example reflects the shortcoming of the (k, ϵ) -anonymity. To mitigate this limitation, sufficient diversity of the sensitive values in each (k, ϵ) -anonymous group should be allowed.

For a sensitive issue s , let the vector of ratings of the group be (s_1, \dots, s_g) , where $s_i \in \{1 : r, null\}$. The mean of the ratings is $\bar{s} = \frac{1}{Q} \sum_{i=1}^g s_i$, where Q is the number of non-*null* values, and $s_i \pm null = s_i$. The standard deviation of the rating is then defined as $SD(s) = \sqrt{\frac{1}{g} \sum_{i=1}^g (s_i - \bar{s})^2}$.

Definition 3 ((k, ϵ, l)-anonymity). A survey rating data set is said to be (k, ϵ, l) -anonymous if and only if the standard deviation of sensitive ratings is at least l in each (k, ϵ) -anonymous group.

4 The Algorithm

In this section, we formulate the satisfaction problem and develop a slicing technique to determine the following *Satisfaction Problem*.

Satisfaction problem: Given a survey rating data T and k, ϵ, l , the satisfaction problem of (k, ϵ, l) -anonymity is to decide whether T satisfies k, ϵ, l requirements.

The satisfaction problem is to determine whether the user’s given privacy requirement is satisfied by the given data set. If the data set has already met the requirements, it is not necessary to make any modifications before publishing. As follows, we propose a novel slice technique to solve the satisfaction problem.

```

Algorithm 1: Slicing( $\epsilon, T, t_0$ )( $C$ )
1   $Can \leftarrow \{t_0\}; S \leftarrow \emptyset$ 
2  /* To slice out the  $\epsilon$ -proximate of  $t_0$  */
3  for  $j \leftarrow 1$  to  $n$ 
4  do if  $|t_j - t_0| \leq \epsilon$ 
5      then  $Cand \leftarrow Cand \cup \{t_j\}$ 
6           $S \leftarrow S \cup \{j\}$ 
7  /* To trim the  $\epsilon$ -proximate of  $t_0$  */
8   $PCand \leftarrow Cand$ 
9  for  $i \leftarrow 1$  to  $|S|$ 
10 do for  $j \leftarrow 1$  to  $|S|$ 
11     do if  $|t_{S(i)} - t_{S(j)}| > \epsilon$ 
12         then  $PCand \leftarrow PCand \setminus \{t_{S(i)}\}$ 
13 return  $PCand$ 
    
```

We illustrate the slicing technique using an example in 3-D space. Given $t = (t_1, t_2, t_3) \in T$, our goal is to slice out a set of transactions T ($t \in T$) that are ϵ -proximate. Our approach is first to find the ϵ -proximate of t , which is the set of transactions that lie inside a cube C_t of side 2ϵ centered at t . Since ϵ is typically

small, the number of points inside the cube is also small. The ϵ -proximate of C'_t can be found by an exhaustive comparison within the ϵ -proximate of t . If there are no transactions inside the cube C_t , we know the ϵ -proximate of t is empty, so as the ϵ -proximate of the set C'_t . The transactions within the cube can be found as follows. First we find the transactions that are sandwiched between a pair of parallel planes X_1, X_2 and add them to a *candidate set*. The planes are perpendicular to the first axis of coordinate frame and are located on either side of the transaction t at a distance of ϵ . Next, we trim the candidate set by disregarding transactions that are not also sandwiched between the parallel pair of Y_1 and Y_2 , that are perpendicular to X_1 and X_2 . This procedure is repeated for Z_1 and Z_2 at the end of which, the candidate set contains only transactions within the cube of size 2ϵ centered at t . *Slicing*(ϵ, T, t_0) (Algorithm 1) describes how to find the ϵ -proximate of the set C_{t_0} with $t_0 \in C_{t_0}$.

5 Experiments

Our experimentation deploys the MovieLens data downloadable at <http://www.grouplens.org/taxonomy/term/14>, which was made available by the GroupLens Research Project at the University of Minnesota. In the data set, a user is considered as an object while a movie is regarded as an attribute and many entries are empty since a user only rated a small number of movies.

Data used for Fig. 1(a) is generated by re-sampling the MovieLens data set while varying the percentage of data from 10% to 100%. We evaluate the running time for the (k, ϵ, l) -anonymity model with default setting $k = 20, \epsilon = 1, l = 2$. The execution time for (k, ϵ, l) -anonymity is increasing with the increased data percentage. This is because as the percentage of data increases, the computation cost increases too, since the overhead is increased with the more dimensions. Next, we evaluate how the parameters affect the cost of computing. In these experiments, we use the whole MovieLens data and evaluate by varying ϵ . With $k = 20, l = 2$, Fig. 1(b) shows the computational cost as a function of ϵ , in determining (k, ϵ, l) -anonymity. At the initial stage, when ϵ is small, more computation efforts are put into finding ϵ -proximate of the transactions, but less used in exhaustive search for ϵ -proximate of the set, and this explains the initial decent of overall cost. As ϵ grows, the searching time for ϵ -proximate is reduced, but the number of transactions in the ϵ -proximate is increased, which results in huge exhaustive search effort and this causes the eventual cost increase.

In addition to the scalability, we experimented the comparison between the slicing algorithm (Slicing) and the heuristic pairwise algorithm (Pairwise), which works by computing all the pairwise distances to construct the dissimilarity matrix and identify the violation of privacy requirements. We implemented both algorithms and studied the impact of the execution time on the data percentage and ϵ . Fig. 2(a) describe the trend of the algorithms by varying the percentage of the data set. From the graph, the slicing algorithm is far more efficient than the heuristic pairwise algorithm especially when the volume of the data becomes larger. This is because, when the dimension of the data increases, the disadvantage of the heuristic pairwise algorithm, which is to compute all the dissimilarity

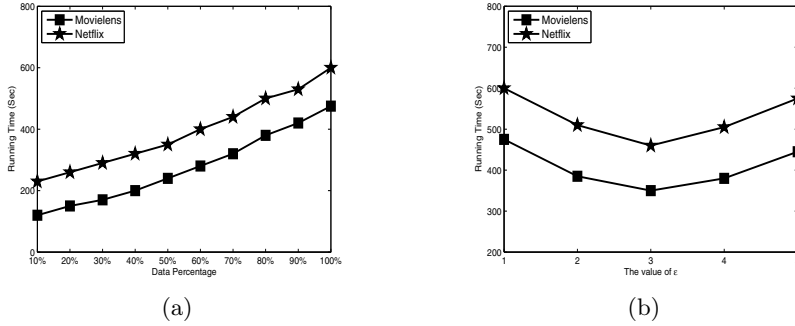


Fig. 1. Running time comparison on Movielens data set vs. (a) data percentage varies; (b) ϵ varies

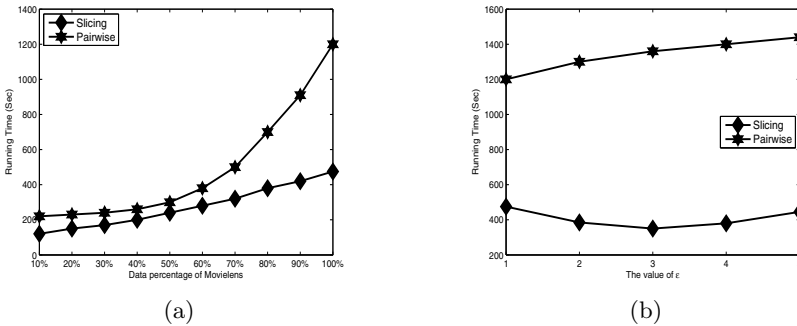


Fig. 2. Running time comparison of Slicing and Pairwise methods (a) data percentage varies; (b) ϵ varies

distance, dominates the execution time. On the other hand, the smarter grouping technique used in the slicing process makes less computation cost for the slicing algorithm. The similar trend is shown in Fig. 2(b) by varying ϵ .

6 Conclusion

We have studied the problem of protecting individuals' sensitive ratings in the large survey rating data. We proposed a novel (k, ϵ, l) -anonymity model and studied the satisfaction problem. A novel slicing technique was proposed to solve the satisfaction problem by searching closest neighbors in large, sparse and high dimensional survey rating data. The experimental results confirm the slicing technique is fast and scalable in practical.

Acknowledgement

Thanks for the reviewers' valuable comments. The research is supported by Australian Research Council (ARC) discovery grants DP0774450 and DP0663414.

References

1. Backstrom, L., Dwork, C., Kleinberg, J.: Wherefore Art Thou R3579x?: Anonymized Social Networks, Hidden Patterns, and Structural Steganography. In: WWW 2007, pp. 181–190 (2007)
2. Fung, B., Wang, K., Yu, P.: Top-down specialization for information and privacy preservation. In: ICDE 2005, pp. 205–216 (2005)
3. Ghinita, G., Tao, Y., Kalnis, P.: On the Anonymisation of Sparse High-Dimensional Data. In: ICDE 2008, pp. 715–724 (2008)
4. Hafner, K.: If you liked the movie, a Netflix contest may reward you handsomely. New York Times (2006)
5. Hansell, S.: AOL removes search data on vast group of web users. New York Times (2006)
6. Le Fevre, K., De Witt, D., Ramakrishnan, R.: Incognito: efficient full-domain k -anonymity. In: SIGMOD 2005, pp. 49–60 (2005)
7. Le Fevre, K., De Witt, D., Ramakrishnan, R.: Mondrian multidimensional k -anonymity. In: ICDE 2006, pp. 25–26 (2006)
8. Li, J., Tao, Y., Xiao, X.: Preservation of Proximity Privacy in Publishing Numerical Sensitive Data. In: SIGMOD 2008, pp. 473–486 (2008)
9. Li, N., Li, T., Venkatasubramanian, S.: t -Closeness: Privacy Beyond k -anonymity and l -diversity. In: ICDE 2007, pp. 106–115 (2007)
10. Machanavajjhala, A., Gehrke, J., Kifer, D., Venkatasubramanian, M.: l -Diversity: Privacy beyond k -anonymity. In: ICDE 2006, pp. 24–25 (2006)
11. Narayanan, A., Shmatikov, V.: Robust De-anonymisation of Large Sparse Datasets. In: IEEE Security & Privacy, 111–125 (2008)
12. Samarati, P.: Protecting respondents' identities in microdata release. IEEE Transactions on Knowledge and Data Engineering 13(6), 1010–1027 (2001)
13. Sun, X., Wang, H., Li, J.: Injecting purposes and trust into data anonymization. In: CIKM 2009, pp. 1541–1544 (2009)
14. Sweeney, L.: k -Anonymity: A Model for Protecting Privacy. International Journal on Uncertainty Fuzziness Knowledge-based Systems 10(5), 557–570 (2002)
15. Xu, Y., Wang, K., Fu, A., Yu, P.S.: Anonymizing Transaction Databases for Publication. In: KDD 2008, pp. 767–775 (2008)
16. Xu, Y., Fung, B., Wang, K., Fu, A., Pei, J.: Publishing Sensitive Transactions for Itemset Utility. In: ICDM 2008, pp. 1109–1114 (2008)

Computation of Ratios of Secure Summations in Multi-party Privacy-Preserving Latent Dirichlet Allocation

Bin Yang¹ and Hiroshi Nakagawa²

¹ Graduate School of Information Science and Technology, The University of Tokyo, Japan
yangbin@r.dl.itc.u-tokyo.ac.jp

² Information Technology Center, The University of Tokyo, Japan
nakagawa@dl.itc.u-tokyo.ac.jp

Abstract. In this paper, we focus our attention on the problem of Gibbs sampling for privacy-preserving Latent Dirichlet Allocation, which is equal to a problem of computing the ratio of two numbers, both of which are the summations of the private numbers distributed in different parties. Such a problem has been studied in the case that each party is semi-honest. Here we propose a new solution based on a weakened assumption that some of the parties may collaborate together to extract information of other parties.

Keywords: Privacy, Data Mining, Latent Dirichlet Allocation, Collusion.

1 Introduction

In recent years, with the increased concerns on privacy protection of personal information, a number of techniques such as randomization and homomorphic encryption have been suggested in order to perform data mining with the private information preserved. In this paper, we propose a new privacy preserving data mining method with m parties from a general problem in data mining: Latent Dirichlet Allocation model (LDA, Blei et al. [1]). LDA is a generative probabilistic model for collections of discrete data such as text corpora, which can be seen as a special mixture model, in which each document is generated by choosing a distribution over topic and then choosing each word in the document from a distribution according to the topic selected. We can employ a Markov chain Monte Carlo algorithm (Griffiths et al. [2]) for inference in this model.

This paper is organized as follows. We introduce the terminology of privacy-preserving data mining and Latent Dirichlet Allocation in section 2. In section 3, we formulate the main problem. And in section 4, we describe our main protocol: RSS protocol, and evaluate the security and efficiency of this protocol. Section 5 illustrates how to assemble our protocol to solve the problem of privacy-preserving LDA described in section 2. In section 6, we present experimental evaluations of these techniques. Finally, we conclude with a summary in section 7.

2 Preliminary

Large numbers of conclusions in privacy-preserving data mining have been obtained by researchers, and almost all of them are under the assumption of semi-honest. In particular, one is a semi-honest party means that the party follows the protocol properly with the exception that it keeps a record of all its intermediate computation results and tries to deduce additional information from them other than the protocol result. In this paper, we also consider a problem called collusion that a subset of the participants, a coalition, might get together after the execution of the protocol in an attempt to deduce additional information of non-coalition parties, although every party also follows the protocol properly, as same as in semi-honest assumption.

A protocol is called t -private if no coalition containing at most t parties can get any additional information from its execution.

We will utilize some standard cryptographic tools and secure protocols to preserve privacy in our protocols. For convenience, we name the two parties "Alice" and "Bob" in the case of 2-party system.

Homomorphic encryption. Public key encryption is a fundamental and widely used technology. The asymmetric key algorithm generates a pair of encryption keys – a public key and a private key. The public key used to encrypt a message is different from the private key used to decrypt it. The private key is kept secret, while the public key may be widely distributed. Given a key pair (sk, pk) and a message msg , $c = E_{pk}(msg)$ denotes an encryption of msg , and $d = D_{sk}(c)$ denotes the decryption of c . In particular, we call such an encryption system a homomorphic encryption, if there is an operation $*$, satisfying the following condition for any msg_1 and msg_2 :

$$E_{pk}(msg_1 + msg_2) = E_{pk}(msg_1) * E_{pk}(msg_2).$$

Secure linear function evaluation (SLFE). Alice has b , while Bob has c and d . After running the secure linear function evaluation protocol, called $SLFE(b, c, d)$, Alice learns the value of $d - bc$, and Bob learns *nothing*.

Protocol SLFE:

1. Alice generates pk and sk , computes $E = E_{pk}(-b)$, then sends E and pk to Bob;
 2. Bob computes $E' = E_{pk}(d) * E^c$, then sends E' back to Alice;
 3. Alice decrypts E' by sk , $D = D_{pk}(E')$.
-

Here E_{pk} and D_{sk} denote the encrypt function and the decrypt function of a homomorphic encryption system respectively. From the homomorphism, we can get $E^c = E_{pk}(-b)^c = E_{pk}(-b) * E_{pk}(-b) * \dots * E_{pk}(-b) = E_{pk}(-bc)$, hence $D = d - bc$.

2.1 Latent Dirichlet Allocation

Typically, LDA model is a generative probabilistic model proposed to research text corpora. A document is seen as a sequence of words. A corpus is a collection of M documents, then we can also see a corpus as a sequence of N words, denoted by $\mathbf{w} = (w_1, w_2, \dots, w_N)$. A document can deal with multiple topics, and the words that appear in that document reflect the particular set of topics it addresses. Each topic is treated as a probability distribution over words in the vocabulary. Thus we can view a

document as a probabilistic mixture of these topics. If there are T topics in total, we can write the probability of the i^{th} word in a document as

$$P(w_i) = \sum_{j=1}^T P(w_i | z_i = j)P(z_i = j) \tag{1}$$

where z_i is a latent variable indicating the topic from which the i^{th} word w_i was drawn. $P(w_i|z_i=j)$ is the probability of the word w_i under the j^{th} topic, whereas $P(z_i=j)$ is the probability of that the topic of i^{th} word, z_i , is equal to j . In other words, each word is generated by drawing a topic z_i from $P(z)$ first, and then drawing a word from $P(w_i|z_i)$.

2.1.1 Gibbs Sampling for LDA

To infer z_i for each word w_i using Gibbs sampling, we need only compute the posterior probabilities that w_i is assigned to topic j , for j from 1 to T , which were derived by Griffiths et al. [2]:

$$p(i, j) = P(z_i = j | \mathbf{z}_{-i}, \mathbf{w}) \propto \frac{n_{-i,j}^{(w_i)} + \beta}{n_{-i,j}^{(\cdot)} + W\beta} \cdot \frac{n_{-i,j}^{(d_i)} + \alpha}{n_{-i,j}^{(d_i)} + T\alpha} \tag{2}$$

$\mathbf{z}_{-i}=(z_1, z_2, \dots, z_{i-1}, z_{i+1}, \dots, z_N)$ is a sequence of $N-1$ topics corresponding to the sequence of words \mathbf{w} except w_i ; $n_{-i,j}^{(w_i)}$ is the number of times that word w_i has been assigned to topic j except the current assignment of z_i ; $n_{-i,j}^{(d_i)}$ is the number of times that a word from document d_i has been assigned to topic j except the current assignment of z_i ; $n_{-i,j}^{(\cdot)} = \sum_{w=1}^W n_{-i,j}^{(w)}$ is the number of times that a word from the corpus has been assigned to topic j except the current assignment of z_i ; and $n_{-i,j}^{(d_i)} = \sum_{j=1}^T n_{-i,j}^{(d_i)}$ is the number of words in the current document d_i except the current assignment of z_i .

To draw a sampling for a latent variable of a word w_i , we only need to compute T probabilities: $P(z_i=1|\mathbf{z}_{-i}, \mathbf{w})$, ..., $P(z_i=T|\mathbf{z}_{-i}, \mathbf{w})$, each of which means the probability that z_i is equal to j under the condition of \mathbf{z}_{-i} and \mathbf{w} .

2.1.2 Distributed LDA

Now we consider a scenario in which more than one parties hold their own data, respectively. They attempt to run Gibbs sampling to infer all z_i with respect to the whole corpus spreading around them without revealing anything. In this case, because any single document is located in just one party, this party can compute $n_{-i,j}^{(d_i)}$ and $n_{-i,j}^{(d_i)}$, thus compute the second item of (2) locally. Then the only remaining problem is computing the first item of (2) in m parties setting. Because the first item of (2) is only related to i and j , we can express it as a function of i and j :

$$q(i, j) = \frac{n_{-i,j}^{(w_i)} + \beta}{n_{-i,j}^{(\cdot)} + W\beta}. \tag{3}$$

Let $n_j^{(w_i)}$ denote the number of times that word w_i has been assigned to topic j including the current assignment of z_i , and $n_j^{(\cdot)} = \sum_{w=1}^W n_j^{(w)}$. Then we have:

$$\begin{cases} n_j^{(w_i)} = n_{-i,j}^{(w_i)} + 1 & \text{if } j = z_i \\ n_j^{(w_i)} = n_{-i,j}^{(w_i)} & \text{if } j \neq z_i \end{cases} \tag{4}$$

$$\begin{cases} n_j^{(\cdot)} = n_{-i,j}^{(\cdot)} + 1 & \text{if } j = z_i \\ n_j^{(\cdot)} = n_{-i,j}^{(\cdot)} & \text{if } j \neq z_i \end{cases} \quad (5)$$

When the data size is quite large, we propose the following approximation:

$$q(i, j) \approx q'(w_i, j) = \frac{n_j^{(w_i)} + \beta}{n_j^{(\cdot)} + W\beta}. \quad (6)$$

Here we notice that $q'(w_i, j)$ is dependent only on w_i and j but not on i . And because the number of words in vocabulary is less than the number of words in corpus, we attempt to approximate $q(i, j)$ by $q'(w_i, j)$, then draw a sampling with privacy being protected. From (4) and (5),

$$q'(w_i, j) = \begin{cases} \frac{n_{-i,j}^{(w_i)} + 1 + \beta}{n_{-i,j}^{(\cdot)} + 1 + W\beta} & \text{if } j = z_i \\ q(i, j) & \text{if } j \neq z_i \end{cases} \quad (7)$$

which means that this kind of approximation always make $q'(w_i, j)$ keep the value of $q(i, j)$ except that $q'(w_i, j)$ is a little bigger than $q(i, j)$ when j is equal to current topic. This means that the value of z_i is a little harder to be changed when sampling. Hence, it will somewhat cause a slow convergence. However, this kind of approximation always does keep the convergence in Gibbs sampling because of the following reasons: 1) It is so near to the real value if data size is big enough; 2) The second term of right hand side of formula (2) is still intact. 3) When converged, this approximation holds because all the values of probabilities are proportional to the real values except what corresponds to current topic. This will be verified by experiments in section 6.

In the rest of the paper, the index attached to the left-top of a variable denotes the party which the variable belongs to. For instance, $n_j^{(k, w)}$ denotes the number of times that word w has been assigned to topic j in all documents in party k . So we can get:

$$n_j^{(w_i)} = \sum_{k=1}^m n_j^{(k, w_i)} \quad \text{and} \quad n_j^{(\cdot)} = \sum_{k=1}^m n_j^{(k)} \quad (8)$$

The variables with nothing being attached to the left-top denote the variables corresponding to the whole m parties, as shown in (8).

To draw a sampling for a word w , we need to compute $\mathbf{q}'_w = (q'(w, 1), \dots, q'(w, T))^T$. Hence, it is sufficient for sampling all the words in the corpus that we only compute \mathbf{q}'_w for w from 1 to W . In other words, we need not compute anything for all words in the entire corpus. Therefore, our whole strategy of sampling is a 2-step iteration: one step is securely computing \mathbf{q}'_w for w from 1 to W , and the other step is locally sampling each word in each party using \mathbf{q}'_w . Since sampling step is similar to original LDA, we concentrate our attention on the computation of $\mathbf{q}'_1, \dots, \mathbf{q}'_W$, expressed as:

$$\mathbf{Q} = [\mathbf{q}'_1 \quad \mathbf{q}'_2 \quad \dots \quad \mathbf{q}'_W] = \begin{bmatrix} q'(1,1) & q'(2,1) & \dots & q'(W,1) \\ q'(1,2) & q'(2,2) & \dots & q'(W,2) \\ \vdots & \vdots & \ddots & \vdots \\ q'(1,T) & q'(2,T) & \dots & q'(W,T) \end{bmatrix} \quad (9)$$

3 Problem Formulation

The problem of privacy-preserving LDA encountered the problem that how to securely compute the ratio of two summations of the data spreading around m parties with nothing of each party being revealed. We formulate our problem as below.

Problem of RSS (Ratio of Secure Summations):

Input:

Party 1 has a pair of data $(^1x, ^1y)$;

Party 2 has a pair of data $(^2x, ^2y)$;

.....

Party m has a pair of data $(^mx, ^my)$.

Output:

All the m parties collaborate to compute:

$$r = \frac{\sum_{i=1}^m ^ix}{\sum_{i=1}^m ^iy} \tag{10}$$

Condition:

Neither ix nor iy of party i is revealed to any party other than i .

Moreover, $\sum_{i=1}^m ^ix$ and $\sum_{i=1}^m ^iy$ are revealed to nobody.

Here it is worthy emphasizing that although the ratio of summations r is known to each party after this computation, ix and iy and any other function of them could not be evaluated by any party other than i .

4 Protocol

We propose a symmetric protocol solving the RSS problem.

Stage 1. Each party generates $2m+1$ random numbers in this stage.

Protocol 2-1:

1. Each party defines $2m+1$ variables of :

Party 1: $^1b_1, ^1b_2, \dots, ^1b_m, ^1c_1, ^1c_2, \dots, ^1c_m, ^1d$;

Party 2: $^2b_1, ^2b_2, \dots, ^2b_m, ^2c_1, ^2c_2, \dots, ^2c_m, ^2d$;

.....

Party m : $^mb_1, ^mb_2, \dots, ^mb_m, ^mc_1, ^mc_2, \dots, ^mc_m, ^md$.

And then generates a random value uniformly to each variable, except $^1b_1, ^2b_2, \dots, ^mb_m$ and $^1c_1, ^2c_2, \dots, ^mc_m$ are reserved to be decided;

2. For each party i , where i from 1 to m :

2-1. Party i runs $SLFE(^ib_j, ^jc_i, ^jd)$ with the collaboration of party j where $j=1, \dots, i-1, i+1, \dots, m$, hence computes the values: $^ie_j = ^jd - ^ib_j ^jc_i$, where $j=1, \dots, i-1, i+1, \dots, m$;

2-2. Party i views $^ib_i ^ic_i$ as a single variable, and decides the value of $^ib_i ^ic_i$ from the following equation, where only $^ib_i ^ic_i$ is unknown:

$$^ie_1 + ^ie_2 + \dots + ^ie_{i-1} + (^id - ^ib_i ^ic_i) + ^ie_{i+1} + \dots + ^ie_m = 0.$$

Therefore, each party generated $2m+1$ numbers satisfying the following condition:

$$\sum_{i=1}^m {}^j b_i^i c_j = d \quad (j=1,2,\dots,m), \tag{11}$$

where

$$d = {}^1 d + {}^2 d + \dots + {}^m d. \tag{12}$$

In next stage, We only use the value of ${}^i b_i^i c_i$ but not ${}^i b_i$ and ${}^i c_i$, so we do not compute the values of ${}^i b_i$ and ${}^i c_i$, even though they are two variables actually.

Stage 2. In this stage, each party collaborate to compute the value of $\sum_{j=1}^m {}^j x \cdot d$.

Protocol 2-2:

1. For i from 1 to m , party i computes ${}^i b_1^i x, {}^i b_2^i x, \dots, {}^i b_m^i x$, and sends ${}^i b_1^i x$ to party 1, ${}^i b_2^i x$ to party 2, ..., ${}^i b_m^i x$ to party m , respectively;
2. For i from 1 to m , party i receives ${}^1 b_i^1 x$ from party 1, ${}^2 b_i^2 x$ from party 2, ..., ${}^m b_i^m x$ from party m , respectively, and then computes ${}^1 b_i^1 c_1^1 x, {}^2 b_i^2 c_2^2 x, \dots, {}^m b_i^m c_m^m x$ locally, by multiplying ${}^i c_1, {}^i c_2, \dots, {}^i c_m$ to each number respectively. Then summates these m numbers to ${}^i f$ and publish it.

$${}^i f = {}^1 b_i^1 c_1^1 x + {}^2 b_i^2 c_2^2 x + \dots + {}^m b_i^m c_m^m x$$
3. All parties compute the summation f ,
i.e. $f = {}^1 f + {}^2 f + \dots + {}^m f$.

In the case that $i=j$, since ${}^i b_i^i c_i$ is known to party i , ${}^i b_i^i c_i x$ can be computed locally. Actually, f is the summation of ${}^j b_i^j c_j^j x$, for i and j from 1 to m . In addition, from (11), we can get that f is just what we want:

$$\begin{aligned} f &= \sum_{i=1}^m \sum_{j=1}^m {}^j b_i^i c_j^j x = \sum_{j=1}^m \sum_{i=1}^m {}^j b_i^i c_j^j x \\ &= \sum_{j=1}^m {}^j x \sum_{i=1}^m {}^j b_i^i c_j^j = \sum_{j=1}^m {}^j x \cdot d \end{aligned} \tag{13}$$

Stage 3. In terms of ${}^j y$, we achieve the operators in stage 1 to regenerate another series of bs , and cs , with the values of ${}^1 d, {}^2 d, \dots, {}^m d$ not being changed, such that

$$\sum_{i=1}^m {}^j b_i^i c_j^j = d \quad (j=1,2,\dots,m),$$

where $d = {}^1 d + {}^2 d + \dots + {}^m d$. Hence we can compute the following value by protocol 2-2:

$$f' = \sum_{j=1}^m {}^j y \cdot d \tag{14}$$

Therefore, the value of f/f' is just the ratio r in (10), which we want to evaluate:

$$\frac{f}{f'} = \frac{\sum_{j=1}^m {}^j x \cdot d}{\sum_{j=1}^m {}^j y \cdot d} = \frac{\sum_{j=1}^m {}^j x}{\sum_{j=1}^m {}^j y} \tag{15}$$

The value $d = {}^1 d + {}^2 d + \dots + {}^m d$ is known to nobody unless all the m parties publish their own ${}^i d$. Thus all parties can deduce nothing. In addition, we achieve stage 1 for only two times, and achieve stage 2 repeatedly.

Security. Following theorem ensures the security of the RSS protocol.

Theorem. The RSS protocol is $(m-1)$ -private.

Performance. For convenience, we denote t_a, t_c, t_r, t_e and t_d the time of arithmetic, communication, random number generation, encryption and decryption, respectively.

Because of the symmetry of RSS protocol, all parties can execute the protocol in parallel. Thus the running time of one party is equal to the global running time. We summarize the running time of RSS protocol in Table 1.

Table 1. The running time of the RSS protocol

| | Stage1 | Stage2 | Stage3 | Total |
|-------|--------|--------|--------|---------|
| t_a | $2m$ | $5m$ | 1 | $14m+1$ |
| t_r | $2m$ | m | - | $6m$ |
| t_e | $2m$ | - | - | $4m$ |
| t_d | m | - | - | $2m$ |
| t_c | $2m$ | $3m$ | - | $10m$ |

In summary, The running time with respect to each operation is $O(m)$.

To compute R ratios, RSS protocol needs just two times stage 1, $2R$ times stage 2 and R time stage 3. Thus we summarize the result in Table 2.

Table 2. Running time of the RSS protocol corresponding to the R times of computations

| | | | | |
|---------|---------|--------|--------|---------|
| t_a | t_r | t_e | t_d | t_c |
| $O(Rm)$ | $O(Rm)$ | $O(m)$ | $O(m)$ | $O(Rm)$ |

Compared with t_c, t_e and t_d , the values of t_a and t_r are so small that we can ignore it in our protocols. So we can say that t_e and t_d are $O(m)$ in RSS protocol.

5 Implementations and Performance

Our main task of distributed LDA is to securely compute a matrix \mathbf{Q} defined by (9) which contains WT ratios of secure summations defined by (6). We can utilize our protocol to compute one ratio in terms of w by setting the variables as follows:

$$\begin{aligned}
 & {}^1x=^1n_j^{(w)} + \frac{\beta}{m}, {}^2x=^2n_j^{(w)} + \frac{\beta}{m}, \dots, {}^mx=^mn_j^{(w)} + \frac{\beta}{m}, \\
 & {}^1y=^1n_j^{(\zeta)} + \frac{W\beta}{m}, {}^2y=^2n_j^{(\zeta)} + \frac{W\beta}{m}, \dots, {}^my=^mn_j^{(\zeta)} + \frac{W\beta}{m}.
 \end{aligned}
 \tag{16}$$

Hence we can securely compute one ratio as

$$\frac{{}^1x+{}^2x+\dots+{}^mx}{{}^1y+{}^2y+\dots+{}^my} = \frac{n_j^{(w)} + \beta}{n_j^{(\zeta)} + W\beta}
 \tag{17}$$

Sampling can be drawn, after all the WT values in \mathbf{Q} are computed.

6 Experiments

In our 2-step iteration of distributed LDA, sampling step is as same as original LDA. As we have mentioned, the approximation in our method makes convergence slower than original LDA. To evaluate it, we compare the results of these two methods in the same machine. The experiment data is a corpus with 300 documents. All documents contain about 200,000 words. The number of words in vocabulary is $W=20$, where the words occurring in almost every document are omitted. The number of topics is $T=5$. Here we just evaluate and compare generated assignments of both methods. As shown in Figure 1, more than 167,000 data are changed in first round of both methods. Original LDA converges after about 10th round, while approximate LDA converges after about 18th round. The final results of sampling of both methods are also the same as each other in terms of generated assignments corresponding to each of them.

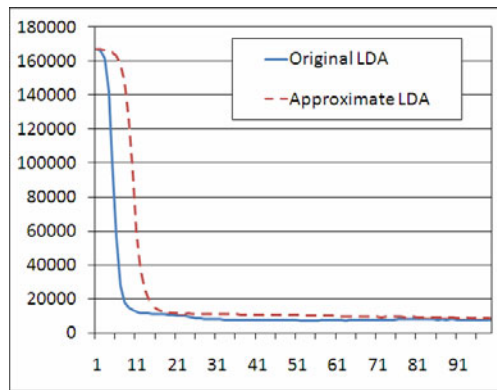


Fig. 1. Speeds of convergence of original LDA and approximate LDA. Horizontal axis denotes the times of sampling, while vertical axis denotes the number of data being changed each time.

In multi-computer environment, sampling step is achieved by m parties in parallel, and without updating parameters, so the 1-round cost of sampling step in distributed LDA is less than $1/m$ of original LDA.

Now we discuss the computing step. Since original LDA does not contain this step, the efficiency of this step is with strong relation to the whole efficiency of distributed LDA. By the limitation of number of computers, we just simulate such a system in a single computer by running all the necessary steps in sequence, recording the running time of each step, and then overlapping the time of parallel steps to compute the whole running time. Splitting all experiment data into m parties, we implement just one round of secure computation in computing step. To compare the cost in different cases, we simulate the cases that m is 5, 10 and 30, and T is 5, 10 and 50, where the data are different from experiment of text analysis above. The result is shown in Table 3. Here, we just fixed $W=10$ in each case. We notice from Table 3 that the cost of RSS protocol is proportional to m , because the total cost is $O(m)$ when T is fixed. In addition, the increment of the cost of RSS protocol is not so fast when T increases, since t_e and t_d is independent on T .

Table 3. Running time (sec.) of one round of RSS protocol

| | $m=5$ | $m=10$ | $m=30$ |
|--------|-------|--------|--------|
| $T=5$ | 89 | 178 | 548 |
| $T=10$ | 165 | 329 | 989 |
| $T=50$ | 774 | 1547 | 4644 |

7 Conclusions

In this paper, we have introduced a protocol of computation of ratio of secure summations. Our RSS protocol has higher security in the case of coalition. Furthermore, we also discussed the performance of the RSS protocols in this paper.

We have also introduced the Privacy-Preserving distributed LDA model which can be applied by the RSS protocol. Moreover, the RSS protocol can be utilized in large numbers of secure computation problems in privacy-preserving data mining, which require computing ratios of secure summations.

References

1. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet Allocation. *Journal of Machine Learning Research* 3, 993–1022 (2003)
2. Griffiths, T.L., Steyvers, M.: Finding scientific topics. In: *PNAS*, pp. 5228–5235 (2004)
3. Jha, S., Kruger, L., McDaniel, P.: Privacy Preserving Clustering. In: *10th European Symposium On Research In Computer Security*, Milan, Italy (2005)
4. Bishop, C.M.: *Pattern Recognition and Machine Learning*. Springer, Heidelberg (2006)
5. Goldreich, O.: *Foundations of Cryptography, Basic Tools*, vol. 1. Cambridge University Press, Cambridge (2001)
6. Goldreich, O.: *Foundations of Cryptography, Basic Applications*, vol. 2. Cambridge University Press, Cambridge (2004)
7. Chor, B., Gereb-Graus, M., Kushilevitz, E.: On the Structure of the Privacy Hierarchy. *Journal of Cryptology* 7, 53–60 (1994)
8. Chor, B., Ishai, Y.: On Privacy and Partition Arguments. *Information and Computation* 167, 2–9 (2001)
9. Aggarwal, C.C., Yu, P.S.: *Privacy-Preserving Data Mining: Models and Algorithms*. Springer, Heidelberg (2008)
10. Paillier, P.: Public-Key Cryptosystems based on Composite Degree Residue Classes. In: Stern, J. (ed.) *EUROCRYPT 1999*. LNCS, vol. 1592, pp. 223–238. Springer, Heidelberg (1999)
11. Damgard, I., Jurik, M.: A Generalisation, a Simplification and some Applications of Paillier's Probabilistic Public-Key System. In: Kim, K.-c. (ed.) *PKC 2001*. LNCS, vol. 1992, pp. 119–136. Springer, Heidelberg (2001)
12. Benaloh, J.C.: Secret sharing homomorphisms: keeping shares of a secret secret. In: Odlyzko, A.M. (ed.) *CRYPTO 1986*. LNCS, vol. 263, pp. 251–260. Springer, Heidelberg (1987)
13. Vaidya, J., Kantarcioglu, M., Clifton, C.: Privacy-preserving Naïve Bayes classification. *The VLDB Journal* 17, 879–898 (2008)

Privacy-Preserving Network Aggregation

Troy Raeder^{1,2}, Marina Blanton², Nitesh V. Chawla^{1,2}, and Keith Frikken³

¹ Interdisciplinary Center for Network Science and Applications
University of Notre Dame, Notre Dame IN 46556 USA
{traeder, nchawla}@cse.nd.edu

² Department of Computer Science and Engineering
University of Notre Dame, Notre Dame IN 46556 USA
mblanton@cse.nd.edu

³ Miami University, Oxford OH 45056 USA
frikkekb@muohio.edu

Abstract. Consider the scenario where information about a large network is distributed across several different parties or commercial entities. Intuitively, we would expect that the *aggregate* network formed by combining the individual private networks would be a more faithful representation of the network phenomenon as a whole. However, privacy preservation of the individual networks becomes a mandate. Thus, it would be useful, given several portions of an underlying network $p_1 \dots p_n$, to securely compute the aggregate of all the networks p_i in a manner such that no party learns information about any other party's network. In this work, we propose a novel privacy preservation protocol for the non-trivial case of weighted networks. The protocol is secure against malicious adversaries.

1 Introduction

As the collection of social network data by enterprises increases, so too does the incidence of proprietary network data. One simple example arises from so-called “viral marketing” campaigns, which exploit the social networks of individuals for targeted marketing. In such a program, a new product is distributed to a few (hopefully influential) individuals with the goal that friends will then buy the product on their recommendation. If the success of recommendations can be tracked, future campaigns can take advantage of the previously-collected network information by targeting “influential” viral marketers, as identified by network structure.

Another scenario comes from the analysis of transactional data as a network of products, where the edge between two products p_i and p_j has a weight w_{ij} equal to the number of times p_i and p_j are purchased together. It has been shown that the structure of such a network can be exploited in order to discover meaningful complex relationships between products and propose potentially profitable promotions [13].

In both cases outlined above, any one view of the network may be *biased* [17], meaning that it is an incomplete or unfaithful representation of the underlying “true” relationships. The simplest way to overcome such a bias is to combine networks from multiple sources on the principle that two networks are unlikely to suffer the same bias, or that the combination of networks will offer a more complete view of the true network.

When networks are highly proprietary or private, participants may not be comfortable disclosing their networks to each other. For instance, an aggregate social network may be useful in collecting product recommendations (if you trust your friends and their friends more than the general population), but the structure of individual social networks is often considered private. In the product networks scenario, while different stores may benefit from sharing, they would still like to preserve their proprietary information such as exact sales and product information. It is with this motivation that we discuss secure methods for network aggregation. Specifically, given a series of networks $\mathcal{G}_1 = (V_1, E_1), \dots, \mathcal{G}_n = (V_n, E_n)$ from participants $\mathcal{P}_1, \dots, \mathcal{P}_n$, we show how to produce an aggregate network \mathcal{G} in a manner such that no participant learns anything about another participant's network. **We then develop a novel protocol for the case of weighted networks that is secure in the presence of malicious adversaries.**

In this paper, we use the product networks scenario of [13] as the driving scenario. Specifically, we develop a protocol with which a set of n stores, selling ℓ products between them, participate in joint computation to securely determine c_{jk} , the number of times product j and product k have sold together in all stores combined (without revealing any information about the products that any one store sells). Each store chooses the products about which it would like to learn, and the stores jointly compute the c_{jk} without leaking unintended information to any of the participants. That is, each store only learns the total counts for products of its interest and other stores do not learn any of the inputs. In addition to the full protocol, we present strategies for efficient implementation and timing results based on real-world data.

It is worth noting that the protocol developed is general and can be applied to any application requiring a secure aggregation of networks. Our focus on product networks is driven by our ability to conduct empirical experiments, since we have data available.

The remainder of the paper is organized as follows: Section 2 describes the problem of secure network aggregation in general and outlines the difficulties associated with weighted networks. Section 3 describes our protocol for secure aggregation of weighted networks, makes suggestions for efficient implementation, and provides timing results on real-world data. Finally, Section 4 offers concluding remarks.

2 Secure Network Aggregation

Assume a series of participants $\mathcal{P}_1 \dots \mathcal{P}_n$ wish to combine their private networks $\mathcal{G}_1 = (V_1, E_1) \dots \mathcal{G}_n = (V_n, E_n)$. These networks can be either weighted or unweighted, but we assume that if any network has edge weights, all do.

In the case of unweighted networks, we can define an aggregate network as the “union” of individual constituent networks: a vertex or edge appears in the aggregate network if it appears in any individual network. In this case, aggregation is simply a union of the sets of vertices and edges, where edges are ordered pairs of vertices. Protocols exist [6] to compute this union efficiently and securely even in the presence of malicious adversaries. Alternatively, one could specify that a vertex or edge appears in the aggregate network only if it appears in at least some number k of individual networks. This is simply a formulation of the *secure over-threshold set union* problem, which has been solved in [9] for the case of semi-honest adversaries.

The case of weighted networks is more difficult. There is no obviously correct way to aggregate edge weights (one could take the minimum, maximum, sum, average, or any other quantity) and no existing secure protocols permit the combination of information such as edge weights during the computation of a set union. If we assume that the desired aggregate is “sum”, as in the case of the product networks mentioned above, one could adapt a secure association-rules protocol (i.e. [8,15]) to our problem in the following manner: For each edge (a, b) of weight w , produce w copies of the transaction $\{a, b\}$. Then run a secure association rules protocol with support s and zero confidence. The two-item association rules in the result will be the edges in the aggregate network whose total weight is at least s .

There are several problems with this approach. First, the size of the transaction database (and therefore the complexity of the algorithm) will depend on the edge weights, which may be arbitrarily large. Second, in these protocols, one participant learns the answer and must forward it to all others rather than informing all participants simultaneously. Third, these protocols do not preserve the actual aggregate (sum), but merely whether the sum exceeds a threshold. Our approach, which we present in the next section, addresses all these concerns.

Furthermore, we provide a mechanism whereby participants commit upfront to the vertices (i.e. products or individuals) about which they would like to learn. This commitment, which can be omitted for efficiency if deemed unnecessary, provides added privacy in that participants can only learn about what they already know. In a viral marketing scenario, for example, participant \mathcal{P}_i could obtain more complete neighborhood information about individuals already in their network but would learn nothing about individuals in \mathcal{P}_j 's network that \mathcal{P}_i had never encountered. Thus \mathcal{P}_j 's competitive advantage is sustained.

3 Privacy-Preserving Solution

In this section, after providing background information, we present our solution, which we call Private Product Correlation (PPC) protocol. We analyze its complexity, propose several efficiency improvements and demonstrate performance on real data. Due to space constraints, the formal security proof has been deferred to [12].

3.1 Preliminaries

Homomorphic encryption. Our solution utilizes semantically secure homomorphic encryption that allows computation on encrypted data without knowledge of the corresponding plaintext. In particular, we use public-key additively homomorphic encryption such as Paillier [11]. Suppose there is a public-private key pair (pk, sk) ; we denote encryption of message m as $E_{pk}(m)$ and decryption of ciphertext c as $D_{sk}(c)$. The additive property gives us $D_{sk}(E_{pk}(m_1) \cdot E_{pk}(m_2)) = m_1 + m_2$ and $E_{pk}(m)^a = E_{pk}(m \cdot a)$. It is also possible, given a ciphertext $c = E_{pk}(m)$, to compute a re-encryption of m such that it is not feasible to tell whether the two ciphertexts correspond to the same message or not; this is done by multiplying the ciphertext by an encryption of 0.

Our protocols use a threshold version of homomorphic encryption. In an (n, k) -threshold encryption scheme, the decryption key is distributed among n parties and the participation of k of them ($k \leq n$) is required to decrypt a ciphertext.

Zero-knowledge proofs. Our protocols rely on zero-knowledge proofs of knowledge from prior literature. In particular, we use a proof of plaintext multiplication defined as follows: given ciphertexts $c_1 = E_{pk}(a)$, $c_2 = E_{pk}(b)$, and $c_3 = E_{pk}(c)$, the prover proves that c corresponds to multiplication of a and b , i.e., $c = a \cdot b$. Cramer et al. [2] give a zero-knowledge protocol for this proof using Paillier homomorphic encryption, which is the type of encryption used in this work.

Privacy-preserving set operations. Prior literature [5][6][9] contains results that permit privacy-preserving operations on sets (or multi-sets). A set $S = \{s_1, s_2, \dots, s_\ell\}$ is represented as the polynomial $f_S(x) = (x - s_1)(x - s_2) \cdots (x - s_\ell)$. This representation has the property that $f_S(s) = 0$ if and only if $s \in S$.

Privacy-preserving operations on sets use encrypted representations of sets. Given a polynomial $f(x) = a_\ell x^\ell + a_{\ell-1} x^{\ell-1} + \dots + a_1 x + a_0$, its encryption is formed as encryption of each coefficient a_i : $E_{pk}(f) = (E_{pk}(a_\ell), \dots, E_{pk}(a_0))$. This representation can be used to perform set operations in privacy-preserving manner. One such an operation used in our solution is *polynomial evaluation*, which given $E_{pk}(f)$, y , and public parameters allows one to compute $E_{pk}(f(y))$. This is done by computing the product $\prod_{i=0}^{\ell} E_{pk}(a_i)^{y^i}$. We also utilize the *set union* protocol of [6], which is the fastest protocol for computing the union of two sets.

3.2 Private Product Correlation Protocol

In our solution we assume that there are n participants (i.e., stores) $\mathcal{P}_1, \dots, \mathcal{P}_n$. Each participant \mathcal{P}_i sells a number of products to which we refer as L_i . We assume that a unique naming convention is used, and different participants will use the same name for a particular product.

Overview of the solution. A natural solution to the product correlation problem in a non-private setting proceeds as follows: each participant counts the number of instances two products were sold together at its store, across all pairs of products the participant offers. Given these counts, the aggregate counts are computed for each pair of products the participants collectively carry. Each participant then saves the aggregate counts corresponding to the products it is interested in. The same logic could be used in constructing a privacy-preserving protocol for product correlation: each participant computes the counts privately, all of them then engage in a variant of a set union protocol preserving (and summing) the counts during the protocol, and finally each participant performs a set intersection on the result to recover the counts for the products of interest.

The existing techniques do not allow this functionality to be implemented in the above form. That is, while privacy-preserving protocols for both set union and set intersection exist, they are not composable, i.e., they cannot be used as sub-protocols in a larger solution which is required to be secure. Furthermore, the way sets are represented in these protocols does not permit additional information (such as a count) to be stored with an element of the set. These limitations led us to design alternative mechanisms for achieving the above task.

Our protocol first requires that the participants agree on an (n, n) -threshold homomorphic encryption scheme and a naming convention for vertices. The simplest choice would be some sort of hashed unique identifier, such as UPC codes for product networks, or e-mail addresses, social security numbers, or ID codes in the more general case. Then, every participant commits to the set of products about which it would like to learn without revealing this set to others. Next, we employ a secure set-union protocol to determine the set of products on which we need to compute. Each participant prepares counts for all pairs of products in the set union and broadcasts encrypted counts to others. The participants jointly add the counts using the homomorphic properties of the encryption scheme. To allow a participant to learn information about the products to which he or she committed (without others learning anything), all parties will aid with the decryption of necessary (unknown to others) counts.

It is conceivably possible to construct a simpler protocol to achieve similar aims. One could repeatedly apply a secure sum protocol to compute the counts we desire, or could omit the commitment step and simply have participants learn about all pairs of products. However, we would argue that these solutions lack potentially desirable security properties. Our full protocol provides added security by preventing participants from learning an arbitrary amount of information regarding products other stores stock (which could result in stores refusing to participate). That is, by limiting the number of products about which a participant learns, we provide the stores with a useful utility without giving anyone the ability to abuse the knowledge they gain.

Protocol description. The participants agree on a (n, n) -threshold homomorphic encryption scheme and generate a public-private key pair (pk, sk) for it.

PPC Protocol:

1. Each participant \mathcal{P}_i creates a list of products $D_i = \{d_1, \dots, d_{m_i}\}$ about which it would like to learn. \mathcal{P}_i commits to this set by committing to the polynomial $Q_i(x) = (x - d_1) \cdots (x - d_{m_i}) = q_{m_i}x^{m_i} + q_{m_i-1}x^{m_i-1} + \cdots + q_1x + q_0$ for constants q_{m_i}, \dots, q_0 . (Q_i may be padded with fake items to hide the size of the set D_i). More specifically, \mathcal{P}_i posts $E_{pk}(q_{m_i}), \dots, E_{pk}(q_0)$ along with a zero-knowledge proof that q_{m_i} is non-zero as described in sub-protocol NZProof below. The proof is necessary to provide a bound on the size of m_i .
2. Each participant \mathcal{P}_i prepares a list of its products L_i . The participants engage in a privacy-preserving set union protocol to determine the set of products all of them sell. Let $L = \{p_1, \dots, p_\ell\}$ denote the outcome of the protocol.
3. For each pair of products $p_j, p_k \in L$, \mathcal{P}_i computes $E_{pk}(c_{jk}^i)$, where c_{jk}^i is its count for the number of times products p_j and p_k were sold together for \mathcal{P}_i . Note that if at least one of p_j and p_k is not in L_i , c_{jk}^i will be 0. \mathcal{P}_i broadcasts the values $E_{pk}(c_{jk}^i)$ for each $0 \leq j, k \leq \ell$ ($j \neq k$).
4. Each \mathcal{P}_i locally computes the encryption of the sum of all counts for each product pair p_i, p_k as $E_{pk}(c_{jk}) = \prod_{i=1}^n E_{pk}(c_{jk}^i)$. \mathcal{P}_i then rearranges the values to form tuples $(p_j, E_{pk}(c_{j1}), \dots, E_{pk}(c_{j\ell}))$ for each $p_j \in L$.
5. Now each \mathcal{P}_i obtains the decryption of counts of the products to which \mathcal{P}_i committed in step 1 (i.e., all counts c_{jk} such that $p_j \in D_i$, $1 \leq k \leq \ell$, and $k \neq j$). To accomplish this, perform the following in parallel for each \mathcal{P}_i :

- (a) One party posts $E_{pk}(Q_i(p_1)), E_{pk}(Q_i(p_2)), \dots, E_{pk}(Q_i(p_\ell))$. Note that $E_{pk}(Q_i(p_j)) = E_{pk}(q_{m_i})^{p_j^{m_i}} \cdot E_{pk}(q_{m_i-1})^{p_j^{m_i-1}} \cdots E_{pk}(q_0)$, and since this is a deterministic process, everyone can verify the result of the computation. Also, note that $Q_i(p_j)$ will be 0 iff p_j was in D_i .
- (b) For each value $E_{pk}(Q_i(p_j)), j = 1, \dots, \ell$, the participants randomize the underlying plaintext by engaging in a NZRM (non-zero random multiplication) protocol described below (each participant executes the NZRM protocol in order). In this protocol each participant multiplies each plaintext by a random non-zero value and proves correctness of the computation. We denote the result of the computation by $E_{pk}(b_j^i)$. Note that b_j^i is 0 if $p_j \in D_i$ and a random value otherwise.
- (c) For each $0 \leq j, k \leq \ell$ ($j \neq k$), one party computes the values $E_{pk}(b_j^i) \cdot E_{pk}(c_{jk}) = E_{pk}(b_j^i + c_{jk})$. Note that the encrypted value is c_{jk} if $p_j \in D_i$ and is a random value otherwise.
- (d) The parties engage in a joint decryption protocol to reveal the values $b_j^i + c_{jk}$ to the \mathcal{P}_i for all $0 \leq j, k \leq \ell$.

Sub-protocols. NZProof Protocol: A user has a ciphertext c and would like to prove that the corresponding plaintext a (where $c = E_{pk}(a)$) is non-zero.

1. The prover chooses a random value b and posts $E_{pk}(b)$ and $E_{pk}(ab)$.
2. The prover proves in zero knowledge that the decryption of $E_{pk}(ab)$ corresponds to the multiplication of the decryptions of $E_{pk}(a)$ and $E_{pk}(b)$.
3. The prover decrypts $E_{pk}(ab)$ and posts ab (this value is jointly decrypted in case of threshold encryption). If ab is non-zero, so must a .

NZRM Protocol: Given a ciphertext $c = E_{pk}(a)$, a participant multiplies the underlying plaintext by a random non-zero value b , outputs $c' = E_{pk}(ab)$ and proves correctness of the computation.

1. The participant chooses a random value b and posts $E_{pk}(b)$ and $c' = E_{pk}(ab)$.
2. The participant proves in zero knowledge that the decryption of $E_{pk}(ab)$ corresponds to the multiplication of the decryptions of $E_{pk}(a)$ and $E_{pk}(b)$.
3. The participant also proves that $E_{pk}(b)$ encrypts a non-zero value using the NZProof protocol.

Complexity Analysis. To simplify the analysis, let m be the upper bound on the number of items to which a participant commits (i.e., $m = \max_i \{m_i\}$). The total work and communication for participant \mathcal{P}_i in step 1 of the protocol is $O(m+n)$, which amounts to $O(mn+n^2)$ communication across all parties. The computation and communication associated with the proof of nonzero q_{m_i} is constant with respect to m and n and does not affect the complexity. In step 2, the overhead associated with the semi-honest (malicious) version of the set union protocol is bounded by $O(n\ell)$ ($O(n\ell^2 + n^2\ell)$, resp.) computation and communication per person and therefore $O(n^2\ell)$ ($O(n^2\ell^2 + n^3\ell)$, resp.) overall communication. In step 3, each participant's work and communication is $O(\ell^2)$ resulting in $O(n\ell^2)$ overall communication. Step 4 involves $O(\ell^2)$ cheaper operations (modular multiplications) per participant and no communication.

To determine the output for a single participant \mathcal{P}_i in step 5, the overhead is as follows: step 5a requires $O(m\ell)$ operations and the same amount of overall communication. Step 5b requires $O(\ell)$ work and communication per participant, resulting in the total of $O(n\ell)$ communication. Here the work and communication added by the NZProof protocol is constant per participant and per product, so the additional complexity introduced is $O(n\ell)$, which does not affect the asymptotic running time. Step 5c involves $O(\ell^2)$ computation and overall communication. Finally, step 5d involves $O(\ell^2)$ threshold decryptions, which amounts to $O(\ell^2)$ work and communication per participant resulting in $O(n\ell^2)$ total communication. Since this part is executed for each participant, the total communication of step 5 over all participants is $O(n^2\ell^2)$.

Thus, if the protocol is built to resist malicious adversaries, the work per participant is $O(n\ell^2 + n^2\ell)$ and the overall communication is $O(n^2\ell^2 + n^3\ell)$.

3.3 Efficiency Improvements

Polynomial multiplication and evaluation. As described in [5], polynomial evaluation can be performed more efficiently by applying Horner’s rule. Recall from section 3.1 that computing $E_{pk}(f(y))$ amounts to calculating $\prod_{i=0}^{\ell} E_{pk}(a_i)y^i$. More efficient computation can be performed by evaluating it from “the inside out” as $E_{pk}(f(y)) = ((\dots(E_{pk}(a_\ell)y E_{pk}(a_{\ell-1}))^y \dots)^y E_{pk}(a_1))^y E_{pk}(a_0)$. Considering that the polynomial is always evaluated on small values (compared to the size of the encryption modulus), this results in a significant performance improvement.

The set union protocol we utilize [6] also uses polynomial multiplication, which for large polynomials becomes inefficient. Given an encrypted polynomial $E_{pk}(f_1) = (E_{pk}(a_{\ell_1}), \dots, E_{pk}(a_0))$ and another polynomial $f_2(x) = b_{\ell_2}x^{\ell_2} + \dots + b_0$, the multiplication consists of computing (encrypted) coefficients c_i of their product: $E_{pk}(c_i) = \prod_{j=\max\{0, i-\ell_2\}}^{\min\{i, \ell_1\}} E_{pk}(a_j)^{b_{i-j}}$ for $i = 0, \dots, \ell_1 + \ell_2$. This can be performed faster by using multi-base exponentiation (see, e.g., [10]), where instead of computing exponentiations $g_1^{x_1}, \dots, g_k^{x_k}$ separately, exponentiation is performed simultaneously as $g_1^{x_1} \dots g_k^{x_k}$ for a fixed (small) value of k . This can speed up computation by several times.

Packing. Assume that the (total) c_{jk} values are at most 2^M . It is likely that $M \ll \rho$, where ρ is the number of bits in the modulus of the encryption scheme. In Step 3 of the PPC protocol, a participant posts ℓ encryptions. We can reduce this number by storing $s = \lfloor \frac{\rho}{M} \rfloor$ values in a single encryption, which would require only $\lceil \frac{\ell}{s} \rceil$ encryptions.

To do the compression, suppose we want to place M -bit values x_1, \dots, x_s into a single encryption. We then compute $E_{pk}(\sum_{i=1}^s 2^{M(i-1)}x_i)$. Note that as long as individual results do not get larger than M bits, we can add to such compressed encryptions (to obtain the value representing the pairwise values) and we can multiply them by constants. In the protocol, addition is the only operation performed on the counts. Such compressed encryptions of counts can also easily be used in Step 5 of the protocol if only the counts that correspond to the same product are combined together (e.g., c_{jk} and $c_{jk'}$ can be included in the same ciphertext for any k, k'). Doing this compression does not reduce the asymptotic communication of the protocol (as this has no effect on the set union), but it does improve the performance of Steps 3–5 in the protocol.

Leaking products with zero sales. In our data, there are many products that never sell together, either because they are relatively unpopular or because they are not available

at the same time. If this information is not valuable to an adversary, pairs of products with zero sales can be excluded from the protocol, meaning that their counts do not need to be encrypted, combined, or decrypted.

3.4 Performance

One persistent concern with privacy-preserving data mining protocols is that they tend to be unacceptably computation-intensive. We now present a discussion of the performance of our protocol and the optimizations that were required to make it tractable in practice. Such a discussion serves both to demonstrate the feasibility of our algorithm and to serve as a baseline for evaluating privacy-preserving data mining techniques.

We implemented the protocol in C using (n, n) -threshold Paillier encryption [234] as the encryption scheme and the protocol of [6] for set union. For efficiency, we extend the set union protocol as follows: the participants decide on a number, B , of buckets, and each participant divides its products among the B buckets according to a hash function. The participants run the set union protocol B times and combine the results of the runs. This can be accomplished without leaking additional information since the combination of several unions is the same as the final union. If the participants split the products uniformly among $B = \frac{\ell}{\log(\ell)}$ buckets, the buckets will contain $\log(\ell)$ products on average, reducing the time complexity of the set union from $O(n\ell^2)$ to $O(n\ell \log(\ell))$. In practice, the participants do not know the size of the union beforehand, so we approximated ℓ with $n \cdot \ell_i$ where ℓ_i is the number of products in each store. We find that this significantly reduces the running time of the set union.

In conducting the experiments, we accounted for the parallelism afforded by the protocol (each participant doing computation in parallel). However, in situations where the computation must be serialized we time the computation of all participants. Specifically we do not include key generation, the construction of unencrypted polynomials for the set union, or the construction of the commitment polynomials. These are pre-processing steps outside the context of PPC. For steps that can be done by all participants simultaneously (steps 2, 3, 4, 5d), we only time the effort of one participant. In steps 4 and 5d, the effort to combine results is also timed.

All experiments were done with a 1024-bit key in keeping with modern standards for public-key encryption, and are run on real-world transaction data from a convenience store at the University of Notre Dame. For the purposes of the experiment, we constructed several stores from a single dataset by randomly assigning products to each store from a pre-determined list of top-selling products. The size of the set union depends on how much overlap there was between the selections of the different stores and is, therefore, random.

Table 1 shows performance as a function of the number of participants n , the size of the set union ℓ , and the size of each store. We present timing results for (i) the full protocol carried out on all pairs of products and (ii) when pairs of products that were never sold together were excluded, as mentioned in Section 3.3. We also report the speedup obtained from the second scenario over the first. The time taken to complete the protocol in both cases scales most significantly with the size of the set union. While increases in the number of participants has some effect, it is partially offset by the increase in the amount of available parallelism afforded by the participation of multiple entities.

Table 1. Timing results for protocol execution

| Participants | Products Per Participant | Products in Union | Time (all pairs) | Time (no zeros) | Speedup |
|--------------|--------------------------|-------------------|------------------|-----------------|---------|
| 3 | 100 | 239 | 105:25 | 14:21 | 7.35 |
| 3 | 200 | 461 | 386:14 | 45:40 | 8.46 |
| 3 | 400 | 942 | 1,637:09 | 98:42 | 16.69 |
| 5 | 100 | 312 | 293:33 | 61:57 | 4.74 |
| 5 | 200 | 601 | 1,097:21 | 167:49 | 6.54 |
| 5 | 400 | 1231 | 4,794:07 | 338:49 | 14.14 |
| 10 | 100 | 375 | 939:17 | 336:15 | 2.79 |
| 10 | 200 | 744 | 1,883:04 | 844:42 | 2.22 |
| 10 | 400 | 1502 | 7,191:15 | 1506:02 | 4.77 |

Improvement as a result of leaking zero counts is substantial, achieving at least 2.22 times speedup. In general, the benefit to leaking zeros increases with the size of the union as zeros become more likely and decreases with the number of participants, as the overhead of each computation is larger. The only exception was the case with 10 stores and 200 products, in which we observe a decrease in speedup from 2.79 to 2.22. Based on other results, this seems to be an aberration, perhaps caused by unusual activity on the machine running the experiment. In all, we observe that leaking zero counts makes the computation substantially more tractable. Whereas computing all $O(\ell^2)$ counts takes almost five days with ten participants and 1500 products in the union, leaking zero counts reduces computation time to just over one day.

4 Conclusion

The goal of this work was to design solutions that utilize the power of networks while ensuring that privacy of the affected parties is preserved and no unintended information leakage takes place. We considered a family of product networks where the stores want to share information about their product/item networks to form a global network, such that they can query this global network for efficient marketing and pricing. The collection of large volumes of customer transaction data is commonplace among both large and small retailers of consumer products. The data collected by any one retailer may potentially be *biased*, i.e., it does not accurately reflect the level of demand for products in the store, for any number of reasons. Moreover, competitive enterprises can mutually improve their standing in a market by sharing information with rivals [14].

To address these concerns, we developed the Private Product Correlation (PPC) protocol for the secure exchange of aggregate network information. The protocol provides more information to participants (namely, a sum of counts) and is secure against malicious as well as semi-honest adversaries. While our work targeted the product networks, it is general enough for applicability to any scenario that requires an aggregation of networks such that no participant learns anything about another participant’s network.

We empirically demonstrate the efficacy of the developed protocol by presenting timing results, which show that our framework is tractable for participants with

reasonable computing power. Our results suggest that execution of the protocol would take on the order of days for participants with a modest number of products using widely-available off-the-shelf hardware. Furthermore, if participants are comfortable with leaking pairs of products that they do not sell together at all, they will see significant gains in performance.

We hope that the above-described analytical techniques and privacy-preserving protocol lead to the increased availability of transactional datasets for scientific study. Product networks contain significantly less information than transaction databases, because some information is lost in the aggregation process. The combination of several product networks, then, should reveal almost nothing about one store's marketing model. As such, retailers could consider an aggregated product network to be devoid of proprietary information and may allow this information to be released and studied.

Acknowledgments. This work was supported in part by the NET Institute, the Air Force Office of Scientific Research under grant AFOSR-FA9550-09-1-0223 and the National Science Foundation under grants CNS-0915843 and BCS-0826958.

References

1. Chawla, N.V., Karakoulas, G.: Learning from labeled and unlabeled data: An empirical study across techniques and domains. *JAIR* 23, 331–366 (2005)
2. Cramer, R., Damgard, I., Nielsen, J.: Multiparty computation from threshold homomorphic encryption. In: Pfitzmann, B. (ed.) *EUROCRYPT 2001*. LNCS, vol. 2045, pp. 280–299. Springer, Heidelberg (2001)
3. Damgard, I., Jurik, M.: A generalisation, a simplification and some applications of paillier's probabilistic public-key system. In: *PKC*, pp. 90–104 (2001)
4. Fouque, P., Poupard, G., Stern, J.: Sharing decryption in the context of voting or lotteries. In: Frankel, Y. (ed.) *FC 2000*. LNCS, vol. 1962, pp. 90–104. Springer, Heidelberg (2001)
5. Freedman, M., Nissim, K., Pinkas, B.: Efficient private matching and set intersection. In: Cachin, C., Camenisch, J.L. (eds.) *EUROCRYPT 2004*. LNCS, vol. 3027, pp. 1–19. Springer, Heidelberg (2004)
6. Frikken, K.: Privacy-preserving set union. In: Katz, J., Yung, M. (eds.) *ACNS 2007*. LNCS, vol. 4521, pp. 237–252. Springer, Heidelberg (2007)
7. Heckman, J.: Sample selection bias as a specification error. *Econometrica: Journal of the econometric society*, 153–161 (1979)
8. Kantarcioglu, M., Clifton, C.: Privacy-preserving distributed mining of association rules on horizontally partitioned data. *IEEE TKDE* 16(9), 1026–1037 (2004)
9. Kissner, L., Song, D.: Privacy-preserving set operations. In: Shoup, V. (ed.) *CRYPTO 2005*. LNCS, vol. 3621, pp. 241–257. Springer, Heidelberg (2005)
10. Menezes, A., van Oorschot, P., Vanstone, S.: *Handbook of Applied Cryptography* (1996)
11. Paillier, P.: Public key cryptosystem based on composite degree residue classes. In: Stern, J. (ed.) *EUROCRYPT 1999*. LNCS, vol. 1592, pp. 223–238. Springer, Heidelberg (1999)
12. Raeder, T., Blanton, M., Chawla, N., Frikken, K.: Privacy-preserving network aggregation. Technical Report TR-2010-02, Notre Dame Computer Science and Engineering (2010)
13. Raeder, T., Chawla, N.: Modeling a store's product space as a social network. In: *Proceedings of ASONAM*, Athens, Greece (2009)
14. Raith, M.: A general model of information sharing in oligopoly. *Journal of Economic Theory* 71(1), 260–288 (1996)
15. Vaidya, J., Clifton, C.: Privacy preserving association rule mining in vertically partitioned data. In: *Proceedings of KDD 2002*, pp. 639–644. ACM, New York (2002)

Multivariate Equi-width Data Swapping for Private Data Publication

Yidong Li and Hong Shen

School of Computer Science, University of Adelaide, SA 5005, Australia
{yidong,hong}@cs.adelaide.edu.au

Abstract. In many privacy preserving applications, specific variables are required to be disturbed simultaneously in order to guarantee correlations among them. Multivariate Equi-Depth Swapping (MEDS) is a natural solution in such cases, since it provides uniform privacy protection for each data tuple. However, this approach performs ineffectively not only in computational complexity (basically $O(n^3)$ for n data tuples), but in data utility for distance-based data analysis. This paper discusses the utilisation of Multivariate Equi-Width Swapping (MEWS) to enhance the utility preservation for such cases. With extensive theoretical analysis and experimental results, we show that, MEWS can achieve a similar performance in privacy preservation to that of MEDS and has only $O(n)$ computational complexity.

Keywords: Private data publication, data swapping, equi-width partitioning, multivariate data perturbation.

1 Introduction

Private data publication has been widely studied in statistical disclosure control and privacy preserving data mining areas. The core task for a qualified publication is to disturb data in a way that does not lead to disclosure of sensitive information, but maintains data utility as much as possible. Many existing studies [10,7,11] are focused on developing algorithms performing univariate data perturbation, which operates each variable in a dataset individually. However, many real-world applications with multivariate data are required to guarantee data utility of several variables (or attributes), which have closer correlations than with others. This paper discusses the utilisation of *data swapping* for multivariate data perturbation.

As the name implies, data swapping is to disturb a dataset by exchanging values of sensitive variables among data tuples. This method is a natural solution to protect confidential information from identity disclosure [10,7], while maintains lower-order statistics of a dataset with its value-invariant property. Equi-depth swapping is a widely used solution to guarantee each bin containing roughly the same number or frequency of data tuples, so as to provide the same level of privacy protection for all entities. However, this approach introduces the computational complexity as $O(n^3)$, where n is the size of dataset in

multivariate scenarios. In addition, it performs ineffectively on preserving data utility for distance-based applications, such as data mining on interval data and multivariate density estimation with histogram, where bins are determined by their relative distances rather than relative orders.

This paper considers the use of data swapping with *equi-width* partitioning, which ensures the width of each bin approximately the same, to overcome the drawbacks above. However, Equi-Width Swapping (EWS) can preserve data privacy at the similar level as EDS for large datasets. It is motivated by the idea that, the value-invariant method hides the detailed partitioning information such as swapping distance. Our analysis shows that, this approach can achieve good tradeoff between data utility and privacy.

The remainder of this paper is organized as follows. Section 2 presents a brief overview of the related literature. In Section 3, we provide algorithms for both MEDS and MEWS. Section 4 presents an exclusive analysis on privacy for the methods above. Section 5 discusses experimental results to justify the effectiveness of MEWS. Finally, we conclude this paper in Section 6.

2 Related Work

Data swapping was first introduced in [4] as an efficient value-invariant approach for statistical disclosure control, and the following work [9] extends the idea to preserve numerical data. Moore et al. [8] described a popular local swapping method based on equi-depth partitioning with univariate ranking. In this method, a term called *swapping distance* defines the depth of each swapping domain, i.e., the number of tuples in each interval for swapping. Then it ranks and localizes data tuples in the light of a specified variable in each iteration, and then swaps candidates in each interval randomly. However, this study is limited with its assumption that the data is uniformly distributed.

The work in [3] follows the idea of ranking but performs random sampling as localization. Although it provides good maintenance of correlations among variables, this technique has been proved highly inefficient [6] and provides practically no protection from attacks. A recent work [11] proposes a swapping-like method named data shuffling, which is based on joint and/or conditional distribution of variables in the original dataset, in order to minimize disclosure risk of sensitive data. A comprehensive study of data swapping and its applications can be found in [5].

3 Multivariate Data Swapping

In this section we first introduce some basic notation that will be used in the remainder of this paper. Let $X = \{x_1, x_2, \dots, x_n\}$ be a dataset. Let A denote the set of all attributes $\{A_1, A_2, \dots, A_m\}$ and $x[A_i]$ denote the value of attribute A_i for a tuple x . We define *swapping set* as a set of attributes $\{A_i, \dots, A_j\} \subset A$ ($i < j$) for simultaneous perturbation, denoted by S , and let \bar{S} represent the set of all other attributes. Then we use $x[S]$ to denote a

Algorithm 1. Multivariate EDS

Input: The input dataset X , the swapping set S , and swapping distance k **Output:** A perturbed dataset Y 1: $x_r = x_s = \mathbf{0}$, $Y = \phi$;2: while $|X| \geq 2k$ do3: select x_r, x_s , where $\|x_r[S] - x_s[S]\|$ is the largest;4: form two bins b_r and b_s containing x_r and x_s with their $k-1$ nearest neighbours;

5: for each group do

6: select pairs of elements randomly;

7: swap swapping set values for each pair;

8: $Y \leftarrow b_r \cup b_s$ and remove them from X ;

partial tuple $(x[S_1 = A_i], \dots, x[S_p = A_j])$ ($p < m$), which is the projection of x onto the attributes in S . Without loss of generality, we assume there is only one swapping set containing the first p attributes A_1, A_2, \dots, A_p , and it is not a unique set to identify individuals.

3.1 Multivariate EDS

This section presents a heuristic method for multivariate equi-depth swapping. The idea is to cluster tuples based on Euclidean distance between each other and then swap attribute values in the swapping set simultaneously. We define $\mathbf{k} = (k[A_1], k[A_2], \dots, k[A_m])$ as a vector of swapping distance for each attribute. As the assumption that the swapping set consists of the first p attributes, the swapping distances for these attributes are equal, i.e., $k[A_1] = k[A_2] = \dots = k[A_p] = k$. Algorithm 1 describes the process of perturbing a dataset according to its swapping set. The perturbation for attributes in \bar{S} is omitted here since it is exactly the same as in univariate EDS [7].

In Algorithm 1, the computational complexity of computing the most distant tuples x_r and x_s is $O(n^2)$. The swapping process only costs $O(\frac{k}{2})$. There are $\lfloor \frac{n}{2k} \rfloor$ iterations. Therefore, the total computational complexity of Multivariate EDS is $O(\frac{n^3}{k})$, where k is the swapping distance. In real-world cases, $k \ll n$, thus the complexity is $O(n^3)$ finally. We can improve this algorithm by calculating a symmetrical distance matrix in advance. This will reduce the time complexity to $O(n^2)$. However, such matrix introduces the storage complexity as $O(\frac{n(n-1)}{2})$, while the original method only has $O(n)$. As the size of data set growth, the distance matrix becomes impractical to be stored in memory.

Since MEDS only considers the relative ordering among tuples for clustering, it inherently leads to errors in many distance-based applications, such as data mining on interval data and density estimation with histogram. These require a perturbation method to maintain not only the ordering but also quantitative properties of the bins including bin width, distance between bins and relative distance between tuples. Moreover, this method only guarantees data utility in bins formed with the p variables in swapping set. That is, if a data user is willing to explore the published data in lower-dimension (e.g., a subset of swapping set),

Algorithm 2. Multivariate EWS

Input: The input dataset X , the swapping set S , and swapping distance t

Output: A perturbed dataset Y

- 1: partition domain of X according to S and t ;
 - 2: allocate each tuple into its corresponding bin;
 - 3: for each non-empty bin do
 - 4: select pairs of elements randomly;
 - 5: swap swapping set values for each pair;
 - 6: end for
 - 7: $Y \leftarrow X$.
-

it may cause unacceptable errors. In the following section, we shall discuss a multivariate swapping technique to improve data utility in the cases above while preserving data privacy.

3.2 Multivariate EWS

To get around of the deficiencies of MEDS, we propose Algorithm 2 by redefining swapping distance as bin width, denoted by $\mathbf{t} = (t[A_1], t[A_2], \dots, t[A_m])$, where $t[A_i]$ is the width on A_i . It implies that the value change for each swapping candidate is at most t . Moreover, to split a dataset with continuous attributes, every bin resulted by univariate partitioning will not be empty if partition degree is larger than a threshold. However, for a multivariate partitioning, many bins may not hold any tuple even with a very small threshold. These empty bins have no use for analysis of utility and privacy. Therefore, we modify the definition of *partition degree* [7] for MEWS as follows:

Definition 1 *The partition degree is the number of non-empty bins formed by splitting the original dataset, denoted by $d \in \mathbb{N}$.*

Let d_{total} denote the total number of bins and for a partitioning PT_i , we have $d^{(i)} \leq d_{total}^{(i)}$. Then, the complexity of partitioning dataset is $O(d_{total})$, and that of allocating process is $O(n)$. The worst case of the iteration step is when $d - 1$ bins contain one tuple each, and the other bin contains the rest $n - d + 1$ tuples. Thus, the computational complexity of Algorithm 2 is $O(dn)$. While in real-world applications, $d \ll n$, we have the complexity as $O(n)$.

The Pruning Scheme. Although the idea of Algorithm 2 is straightforward, it is even challenged by selecting swapping distance t . Theoretically, there exist infinite possibilities for t within the range $(0, \frac{\text{domain}(x)}{2}]$ due to $t \in \mathbb{R}^+$. However, many partitionings with small differences are duplicate and meaningless. We introduce the Pruning scheme that determines a proper domain of swapping distance efficiently.

Step 1: Calculate or assign a upper bound t_{ub} for t ;

Step 2: Compute a lower bound t_{lb} for t ;

Step 3: Reduce the number of partitionings to a finite number.

The first step guarantees data utility of a published dataset, since a too-large t can make perturbed data useless due to its unacceptable bias. Step 2 considers data privacy of a published dataset. The idea behind is to ensure that each non-empty bin must hold at least two tuples for data exchanging. The final step is to further remove the redundant candidates. we define *Minimal Depth Bin (MDB)* to represent a bin that contains the least number of tuples corresponding to t and let N^* represent the number of an MDB. Then, we say two partitionings with $t \in [t_r, t_s]$ are similar if $N_{r-1}^* < N_r^* = \dots = N_s^* < N_{s+1}^*$. Since the similar partitionings lead to the same swapping result in MDB and the number of sets of similar partitionings are limited in a finite dataset, the Pruning scheme finally reduces the number of partitionings to a finite number. (The details of selections can be found in the extended version due to space limits.)

4 Privacy Analysis

MEWS theoretically guarantees zero-loss of utility among variables in swapping set, which makes this approach applicable to a large category of applications of multivariate data analysis. We dedicate this section to discuss how good MEWS is in terms of preserving privacy.

As the concept of privacy may have various concerns, we describe the privacy, P , for the entire dataset as *the probability of revealing a swapping pair who has the highest disclosure risk in the database*, which is $P = \max(Pr(X = x_i)) \quad (1 \leq i \leq n)$. The disclosure risk for a tuple can be computed as,

$$Pr(X = x_j[S]) = \sum_{l=1}^q Pr(d_l)Pr(X = x_j[S] | d_l) \tag{1}$$

where $(x_i[S], x_j[S])$ is a swapping pair, q indicates the number of valid partitionings according to the Pruning scheme. Then, we can compute the privacy for MEDS and MEWS with the following lemmas.

Lemma 1. *Given n data tuples and swapping distance k , the privacy for MEDS is*

$$P_{ed} = \frac{1}{q_{ed}} \times \left(\sum_{i=1}^{q_{ed}/2} \frac{1}{i} + \frac{q_{ed}}{2(q_{ed} + 2)} \right) \simeq \frac{\ln(q_{ed}) + c_1}{q_{ed}} \tag{2}$$

where q_{ed} is the number of possible partitionings and c_1 is a small constant in the range of $[\gamma + 0.2, 1.2]$.

The proof of Lemma 1 is omitted here and a similar process can be found in [7] if interest. The above Lemma shows that, if a dataset is large, the privacy provided by MEDS is only relative to the number of possible partitionings rather than the number of tuples in a bin.

Lemma 2. *Given a dataset X and swapping distance t , the privacy for MEWS is,*

$$P_{ew} = \frac{1}{q_{ew}} \times \sum_{i=1}^{q_{ew}} \frac{1}{N_i^* - 1} \simeq \frac{\ln(q_{ew}) + c_2}{q_{ew}}, \tag{3}$$

where q_{ew} is the number of possible partitionings and c_2 is a small constant within the range $[\gamma, 1]$.

Although the proof of the lemma is omitted due to space constraints, one can see the privacy provided by MEWS is only relative to the number of possible partitionings which is the same as MEDS.

Based on Lemma 1 and 2, let us consider perturbing a large database, which is quite common in data mining. Generally speaking, MEDS provides more possible partitionings than MEWS does, i.e., $q_{ed} \geq q_{ew}$. But for a dataset with large size of tuples, the domain of swapping distance will be large even with the Pruning scheme. That is, q_{ew} can still be large enough for protecting data. Moreover, with the same size of MDB N^* , the two approaches result in different partition degree d_{ed} and d_{ew} , where $d_{ed} \geq d_{ew}$ in usual. However, degree is not a deterministic parameter for privacy measuring. It also implies that MEWS can provide good performance on preserving privacy as MEDS. We will show further analysis on privacy based on specific distributed datasets in the experimental part.

5 Experiments

5.1 Privacy on Binormal Distribution

As the privacy for MEWS is sensitive to data distribution, it is deserved to explore relations between privacy and distribution parameters by generating a dataset with normalized bivariate normal distribution $f_{XY}(x, y; 1, 1, 0, 0, \rho_{XY})$, where ρ is the correlation coefficient of attributes X and Y . In addition, a data suppression is adopted to restrict domains of attributes within the range $[-2, 2]$, which is reasonable because most of sparse data will be cleaned before data analysis.

The results in Table 1 show that the privacy P for MEWS has direct relationships with three factors: ρ , t_{ub} and n . For a large dataset (e.g., $n = 100,000$), the disclosure probability is very small ($P \leq 7\%$) even the attributes are not highly related, and turns to large ($P \geq 30\%$) while the correlations of attributes are small and the size of data is not large (e.g., the up-left corner of the table).

Table 1. The impact of distribution on privacy P for MEWS

| | $n = 10,000$ | | | $n = 100,000$ | | |
|----------|--------------|--------|--------|---------------|--------|--------|
| t_{ub} | $\rho = 0.1$ | 0.5 | 0.9 | 0.1 | 0.5 | 0.9 |
| 0.2 | 0.3466 | 0.2986 | 0.1450 | 0.1324 | 0.0635 | 0.0250 |
| 0.4 | 0.1973 | 0.1060 | 0.0461 | 0.0359 | 0.0172 | 0.0069 |
| 0.8 | 0.0443 | 0.0254 | 0.0125 | 0.0066 | 0.0036 | 0.0017 |

Since most datasets for data mining are very large, and attributes in a swapping set generally have very close relations (otherwise the swapping set will make no sense for data analysis), we can observe that the MEWS method can provide good data protection. This is not intuitive but reasonable because constraints on equi-width partition is much more relaxed than that on equi-depth partition. This property is held only in the context of local data swapping rather than other local perturbation approaches.

5.2 Comparison of Data Utility

Since swapping distance has various definitions for MEDS and MEWS, we consider the impact of partition degree on the correlation matrix here. Given a multivariate dataset X , let ρ_{ij} be correlation coefficient of X_i and X_j and ρ'_{ij} be the corresponding coefficient once data have been swapped using a specified method. Then, the bias of the correlation is $\Delta\rho_{ij} = |\rho_{ij} - \rho'_{ij}|$. We compute the average $\overline{\Delta\rho}$ and the standard deviation $\sigma_{\Delta\rho}$ as utility metrics. The experiments are run over one synthetic and two real datasets: 1) *Syn100k*, contains 100,000 data tuples and four variables, which the first two variables are generated with the standard binormal distribution with $\rho = 0.5$ and others are generated individually with the standard normal distribution; and 2) *Abalone* and *MagicTele*, are both formed by their continuous variables in [2].

Table 2 shows that, the $\overline{\Delta\rho}$ decreases as the partition degree increases, which is intuitive and reasonable. In most cases, MEWS performs better than MEDS, especially when the partition degree is not large. For example, the $\overline{\Delta\rho}$ resulted by MEDS is twice more than that by MEWS in both sythetic datasets and much higher in the real datasets. In addition, during the experiment, we find that MEWS provides more steady performance than MEDS. It implies that the variance of $\sigma_{\Delta\rho}$ for MEDS is larger than that for MEWS.

It should notice that, even the experimental results show that MEWS can achieve a better parametric utility in our tested data, this can not be guaranteed in other cases. We can easily construct some datasets that fit better to MEDS. However, the MEWS method is still a good alternative for commonly used data distributions.

Table 2. Impact of Partition Degree on Correlations

| | | $d = 100$ | | $d = 500$ | | $d = 1000$ | |
|-----------|--------|-------------------------|-----------------------|-------------------------|-----------------------|-------------------------|-----------------------|
| Dataset | Method | $\overline{\Delta\rho}$ | $\sigma_{\Delta\rho}$ | $\overline{\Delta\rho}$ | $\sigma_{\Delta\rho}$ | $\overline{\Delta\rho}$ | $\sigma_{\Delta\rho}$ |
| Syn100k | MEWS | 0.40 | 0.16 | 0.20 | 0.08 | 0.11 | 0.08 |
| | MEDS | 0.61 | 0.25 | 0.37 | 0.18 | 0.18 | 0.11 |
| Abalone | MEWS | 0.20 | 0.06 | 0.15 | 0.05 | 0.10 | 0.05 |
| | MEDS | 0.51 | 0.10 | 0.30 | 0.08 | 0.18 | 0.07 |
| MagicTele | MEWS | 0.30 | 0.09 | 0.21 | 0.10 | 0.12 | 0.08 |
| | MEDS | 0.62 | 0.20 | 0.41 | 0.18 | 0.20 | 0.10 |

6 Conclusion

This paper explores the use of multivariate equi-width swapping as a tool for private data publication. It makes primary benefits on two different grounds. First, the time complexity of the algorithm is linearly proportional to the size of a dataset, which makes this approach quite efficient especially for applying on very large datasets. Then, it provides excellent preservation on distance-based data utilities, which has great potential for use in the real-world data analysis. Compared to the multivariate equi-depth swapping which provides uniform privacy protection for each tuple, the proposed method still performs quite reasonably on privacy protection. In our future work, the basic MEWS method can be optimized to achieve more efficient variants for specified applications. Applying this approach in distributed private data publication offers another interesting direction.

References

1. Aggarwal, C.C., Yu, P.S.: A condensation approach to privacy preserving data mining. In: EDBT, pp. 183–199 (2004)
2. Asuncion, A., Newman, D.: Uci machine learning repository. University of California, Irvine (2007)
3. Carlson, M., Salabasis, M.: A data-swapping technique for generating synthetic samples; a method for disclosure control. *Research in Official Statistics* 5, 35–64 (2002)
4. Dalenius, T., Reiss, S.P.: Data-swapping: A technique for disclosure control (extended abstract). In: *The Section on Survey Research Methods*, Washington, DC, pp. 191–194 (1978)
5. Fienberg, S.E., McIntyre, J.: Data swapping: Variations on a theme by ddalenius and reiss. *J. Official Statist.* 21, 209–323 (2005)
6. Fienberg, S.E.: Comment on a paper by m. carlson and m. salabasis: a data-swapping technique using ranks - a method for disclosure control. *Research in Official Statistics* 5, 65–70 (2002)
7. Li, Y., Shen, H.: Equi-width data swapping for private data publication. In: *PD-CAT 2009: The Tenth International Conference on Parallel and Distributed Computing, Applications and Technologies*, Hiroshima, Japan (2009)
8. Moore, R.A.: Controlled data-swapping techniques for masking public use micro-data sets. Research report RR96/04, U.S. Bureau of the Census, Statistical Research Division, Washington D.C (1996)
9. Reiss, S.P., Post, M.J., Dalenius, T.: Non-reversible privacy transformations. In: *PODS 1982: Proceedings of the 1st ACM SIGACT-SIGMOD symposium on Principles of database systems*, pp. 139–146. ACM, New York (1982)
10. Mukherjee, S., Chen, Z., Gangopadhyay, A.: A privacy-preserving technique for euclidean distance-based mining algorithms using fourier-related transforms. *The VLDB Journal* 15, 293–315 (2006)
11. Muralidhar, K., Sarathy, R.: Data shuffling - a new masking approach for numerical data. *Management Science* 52(5), 658–670 (2006)

Correspondence Clustering: An Approach to Cluster Multiple Related Spatial Datasets

Vadeerat Rinsurongkawong and Christoph F. Eick

Department of Computer Science, University of Houston,
Houston, TX 77204-3010
{vadeerat, ceick}@cs.uh.edu

Abstract. Domain experts are frequently interested to analyze multiple related spatial datasets. This capability is important for change analysis and contrast mining. In this paper, a novel clustering approach called correspondence clustering is introduced that clusters two or more spatial datasets by maximizing cluster interestingness and correspondence between clusters derived from different datasets. A representative-based correspondence clustering framework and clustering algorithms are introduced. In addition, the paper proposes a novel cluster similarity assessment measure that relies on re-clustering techniques and co-occurrence matrices. We conducted experiments in which two earthquake datasets had to be clustered by maximizing cluster interestingness and agreement between the spatial clusters obtained. The results show that correspondence clustering can reduce the variance inherent to representative-based clustering algorithms, which is important for reducing the likelihood of false positives in change analysis. Moreover, high agreements could be obtained by only slightly lowering cluster quality.

Keywords: Spatial Data Mining, Mining Related Spatial Datasets, Variance in Clustering, Representative-based Clustering Algorithms, Change Analysis.

1 Introduction

Domain experts are frequently interested to compare clustering results of two or more related datasets. For example, meteorologists may want to understand the change in this year's sea water temperature patterns with respect to those observed in previous years. Zoologists may attempt to relate animals' habitats and their source of foods. We can use traditional clustering algorithms to cluster each dataset separately and compare the results, but this approach frequently will not lead to good results due to the following reasons:

1. The clustering algorithms do not take into consideration the correspondences between the datasets.
2. The randomness inherent in most clustering algorithms further complicates the correspondence analysis of clusterings. For example, K-means is sensitive to initialization, noise, and outliers.

In this paper, we introduce a novel spatial clustering approach called *correspondence clustering*. Correspondence clustering clusters two or more spatial datasets by taking the correspondence between the different clustering into consideration. Therefore, the obtained clusterings relate to one another; that is, the clustering of one dataset depends on the clusterings of the other datasets. Consequently, variances in clusterings produced by traditional clustering algorithms are reduced. Moreover, the hidden relationships between related clusterings can be discovered.

Applications for correspondence clustering include:

1. Change analysis [7] where changes between two datasets are compared; correspondence clustering reduces the likelihood of identifying false change patterns by enhancing agreement between the clustering results for different datasets.
2. Regional co-location mining [2] that seeks for regions in which two types of events are co-located; for example, correspondence clustering can find regions where deep and severe earthquakes co-locate.
3. Correspondence clustering can help dealing with missing values. For example, when identifying clusters with high ozone concentration, failures of ozone measurement equipments result in missing values. Correspondence clustering can use past clusterings to guide clustering when missing values are present.

Challenges to develop a good correspondence clustering framework include:

1. Techniques have to be developed to deal with the variance inherent to most clustering algorithms. If it is not dealt properly, false changes, and false novelties will be detected.
2. Methods have to be developed that measure the correspondence between two clusterings which is a non-trivial problem if object identity is not known.
3. Clustering algorithms have to be extended to cluster multiple datasets jointly considering cluster agreement or other relationships between the clustered datasets.
4. Measuring cluster correspondence is usually quite expensive, and efficient techniques have to be developed to keep its overhead in check.

The main contributions of the presented paper include:

1. A unique framework for correspondence clustering is introduced.
2. Representative-based correspondence clustering algorithms that allow for plug-in fitness functions are introduced.
3. Novel techniques that measure the agreement between two clusterings are proposed.

2 Correspondence Analysis Framework

In this section, we propose a correspondence analysis framework. Basically, our framework is designed for spatial data, especially for geo-referenced datasets. The challenges of discovering interesting patterns in spatial data include the complexity of spatial data types, the presence of hidden spatial relationships, and spatial

autocorrelation. Moreover, spatial space is continuous and contains many patterns at different levels of granularities.

Let us assume that a set of related spatial datasets $O=\{O_1,\dots,O_n\}$ are given. We are interested in finding interesting relationship among these datasets. In general, our framework seeks for clustering results that maximize two objectives: (1) the interestingness in each clustering, (2) the correspondence between the set of obtained clusterings. Correspondence clustering is defined as follows.

Definition 1. A correspondence clustering algorithm clusters data in two or more datasets $O=\{O_1,\dots,O_n\}$ and generates clustering results $X=\{X_1,\dots,X_n\}$ such that for $1\leq i\leq n$, X_i is created from O_i and the correspondence clustering algorithm seeks for X_i 's such that each X_i maximizes interestingness $i(X_i)$ with respect to O_i as well as maximizes the correspondence $Corr(X_1,\dots,X_n)$ between itself and the other clusterings X_j for $1\leq j\leq n, j\neq i$.

In summary, correspondence clustering can be viewed as a multi-objective optimization problem in which the interestingness of clustering and their correspondence are maximized. Moreover, different interestingness functions i and correspondence functions $Corr$ can be used for different correspondence clustering tasks. In the next section, a representative-based correspondence clustering approach is introduced. The approach allows for plug-in fitness functions that are capable to capture different interestingness functions i and correspondence functions $Corr$.

3 Representative-Based Correspondence Clustering Algorithms

Since our representative-based correspondence clustering approach employs a region discovery framework, we first introduce the region discovery framework.

3.1 Region Discovery Framework

The region discovery framework [1] gears towards finding scientifically interesting places in spatial datasets. The framework adapts clustering algorithms for the task of region discovery by allowing plug-in fitness functions to support variety of region discovery applications corresponding to different domain interests. The goal of region discovery is to find a set of regions X that maximize an externally given fitness function $q(X)$; q is assumed to have the following structure:

$$q(X) = \sum_{c\in X} reward(c) = \sum_{c\in X} i(c) \times |c|^\beta \quad (1)$$

where $i(c)$ is the interestingness of a region c and $|c|$ is the number of objects belonging to a region c is denoted by $|c|$. The reward associated with region sizes is controlled by parameter β ($\beta>1$).

3.2 Representative-Based Correspondence Clustering Algorithms

In general, representative-based clustering algorithms seek for a subset of the objects in the dataset—called the “representatives”—and form clusters by assigning the

remaining objects to the closest representative. In this section, representative-based correspondence clustering algorithms are introduced. The proposed algorithms are modifications of a region discovery algorithm named CLEVER [2]. CLEVER is a representative-based clustering algorithm that applies randomized hill climbing to maximize the fitness function q . Figure 1 gives the pseudo-code of CLEVER.

Inputs: Dataset O , k' , neighborhood-size, p , p' , β

Outputs: Clustering X , fitness q

Algorithm:

1. Create a current solution by randomly selecting k' representatives from O .
2. Create p neighbors of the current solution randomly using the given neighborhood definition.
3. If the best neighbor improves the fitness q , it becomes the current solution. Go back to step 2.
4. If the fitness does not improve, the solution neighborhood is re-sampled by generating p' more neighbors. If re-sampling does not lead to a better solution, terminate returning the current solution; otherwise, go back to step 2 replacing the current solution by the best solution found by re-sampling.

Fig. 1. Pseudo-code of CLEVER

Given two datasets O_1 and O_2 , the goal of correspondence clustering is to discover sets of clusterings X_1 and X_2 that maximize the compound fitness function $\tilde{q}(X_1, X_2)$. The compound fitness function $\tilde{q}(X_1, X_2)$ is defined as follows:

$$\tilde{q}(X_1, X_2) = (\alpha \times (q(X_1) + q(X_2))) + ((1 - \alpha) \times \text{Corr}(X_1, X_2)) \quad (2)$$

where q is a fitness function that assess the quality of X_1 and X_2 . The correspondence parameter α is a user-defined parameter. The correspondence function $\text{Corr}(X_1, X_2)$ measures the correspondence between X_1 and X_2 .

CLEVER is modified to maximize the compound fitness function \tilde{q} instead of the fitness function q . Two approaches that implement correspondence clustering are introduced in the following: (1) The Interleaved approach (C-CLEVER-I), and (2) The Concurrent approach (C-CLEVER-C). The algorithms of C-CLEVER-I and C-CLEVER-C are given in Figure 2 and 3, respectively.

The C-CLEVER-C uses the compound fitness function (equation 2) to cluster two data sets concurrently. For the C-CLEVER-I, dataset O_1 and O_2 are clustered one at a time—not concurrently—therefore, the compound fitness function (equation 2) simplifies to (3) and (4) when clustering the first and second dataset, respectively.

$$\tilde{q}_1(X_1) = (\alpha \times q(X_1)) + ((1 - \alpha) \times \text{Corr}(X_1, X_2)) \quad (3)$$

$$\tilde{q}_2(X_2) = (\alpha \times q(X_2)) + ((1 - \alpha) \times \text{Corr}(X_1, X_2)) \quad (4)$$

In general, there are many possible choices in selecting initial representatives of C-CLEVER-I and C-CLEVER-C. Our current implementation supports three options: The first option is to randomly select a subset of size k' from O as in CLEVER. The second option uses the final set of representative R from the previous iteration as the

initial set of representatives. The third option uses the set of representatives R' of its counterpart clustering X' to compute a set of “nearby” representatives R taken from the dataset O as follows:

1. For each r' in R' determine its (1-)nearest neighbor in O obtaining a set R
2. Remove duplicates from R .

There are many choices for termination condition ($TCond$). The possible choices are: (1) fix the number of iterations; (2) terminate the program if the compound fitness function in the present iteration does not improve from the previous iteration.

Input: O_1 and O_2 , $TCond$, k' , neighborhood-size, p , p' , α , β

Output: X_1 , X_2 , $q(X_1)$, $q(X_2)$, $\tilde{q}(X_1, X_2)$, $Corr(X_1, X_2)$

Algorithm:

1. Run CLEVER on dataset O_1 with fitness function q and get clustering result X_1 and a set of representative R_1 :
 $(X_1, R_1) := \text{Run CLEVER}(O_1, q)$;
2. Repeat until the Termination Condition $TCond$ is met.
 - a. Run CLEVER on dataset O_2 with compound fitness function \tilde{q}_2 that uses the representatives R_1 to calculate $Corr(X_1, X_2)$:
 $(X_2, R_2) := \text{Run CLEVER}(O_2, R_1, \tilde{q}_2)$
 - b. Run CLEVER on dataset O_1 with compound fitness function \tilde{q}_1 that uses the representatives R_2 to calculate $Corr(X_1, X_2)$:
 $(X_1, R_1) := \text{Run CLEVER}(O_1, R_2, \tilde{q}_1)$

Fig. 2. Pseudo-code of C-CLEVER-I

Input: O_1 and O_2 , $TCond$, k' , neighborhood-size, p , p' , α , β

Output: X_1 , X_2 , $q(X_1)$, $q(X_2)$, $\tilde{q}(X_1, X_2)$, $Corr(X_1, X_2)$

Algorithm:

1. Run CLEVER on dataset O_1 with fitness function q and get clustering result X_1 and a set of representative R_1 :
 $(X_1, R_1) := \text{Run CLEVER}(O_1, q)$;
2. Run CLEVER on dataset O_2 with fitness function q and get clustering result X_2 and a set of representative R_2 :
 $(X_2, R_2) := \text{Run CLEVER}(O_2, q)$;
3. Repeat until the Termination Condition $TCond$ is met.
 - a. Run CLEVER on datasets O_1 and O_2 concurrently maximizing the compound fitness function \tilde{q} :
 $(X_1, R_1, X_2, R_2) := \text{Run CLEVER}(O_1, R_1, O_2, R_2, \tilde{q})$

Fig. 3. Pseudo-code of C-CLEVER-C

4 Agreement Assessment by Forward and Backward Re-clustering Techniques and Co-occurrence Matrices

Using agreement between two clusterings X_1 and X_2 is a popular choice for a correspondence function. In applications such as change analysis [7] or co-location mining [2], domain experts want to discover clusterings that are good and agree at least to some extent. In such case, $Agreement(X_1, X_2)$ would be used as the correspondence

function. In addition, domain experts might be interested to discover regions with disagreement between the two datasets in anti-co-location or novelty detection. In the later case, $Corr(X_1, X_2)$ can be defined as $(1 - Agreement(X_1, X_2))$. For the remaining of the paper, $Agreement(X_1, X_2)$ will be used as correspondence function $Corr(X_1, X_2)$; in the section we will introduce a measure to assess agreement.

First, we introduce re-clustering techniques that use the clustering model of one clustering to cluster the data in another dataset. In case of representative-based clustering, the cluster models are sets of representatives. Given two clusterings X_1 and X_2 of two datasets O_1 and O_2 and two sets of representatives of R_1 and R_2 of the two clusterings X_1 and X_2 , forward and backward re-clusterings can be created using the representatives of one dataset to cluster the other dataset. More formally:

Definition 2. Let O be a dataset and R be an arbitrary set of representatives. Then $\chi_{REC}(O, R)$ denotes the result of re-clustering dataset O using the set of representatives R . The clusters of $\chi_{REC}(O, R)$ are created by assigning objects $o \in O$ to the closest representative $r \in R$ obtaining $|R|$ clusters.

Definition 3. $\chi_{REC}(O_2, R_1)$ is called forward re-clustering and $\chi_{REC}(O_1, R_2)$ is called backward re-clustering.

To assess cluster similarity, the similarity between two representative-based clusterings X_1 and X_2 is computed by comparing X_1 with $\chi_{REC}(O_1, R_2)$ and X_2 with $\chi_{REC}(O_2, R_1)$. To assess the similarity of two clusterings, we construct a co-occurrence matrix M_X for each clustering X of O as follows:

1. If o_j and o_i belong to the same cluster in X , entries (i, j) and (j, i) of M_X are set to 1.
2. If o_i is not an outlier in X , set (i, i) in M_X to 1
3. The remaining entries of M_X are set to 0

Let M_X and $M_{X'}$ be two co-occurrence matrices that have been constructed for two clusterings X and X' of the same dataset O ; then the similarity between X and X' can be computed as follows:

$$Sim(X, X') := \frac{\text{(Number of entries } (i, j) \text{ with } i \leq j \text{ in } M_X \text{ and } M_{X'} \text{ that both are 1 in } M_X \text{ and } M_{X'})}{\text{(Number of entries } (i, j) \text{ with } i \leq j \text{ that contain a 1 in } M_X \text{ or } M_{X'} \text{ or a 1 in both)}} \quad (5)$$

$Sim(X, X')$ in equation (5) is a generalization of the popular Rand Index [8] that additionally takes outliers into consideration. Finally, we define agreement between clustering X_1 and X_2 , by comparing the clusterings of the two datasets with their respective forward and backward re-clusterings as follows:

$$Agreement(X_1, X_2) = \frac{Sim(X_1, \chi_{REC}(O_1, R_2)) + Sim(X_2, \chi_{REC}(O_2, R_1))}{2} \quad (6)$$

The advantage of the proposed agreement assessment method based on re-clustering techniques and co-occurrence matrices is that it can deal with: (1) datasets with unknown object identity, (2) different number of objects in two datasets, (3)

different number of clusters in the two clusterings. Therefore, we claim that it is suitable for most types of spatial data.

5 Experiments

In the first experiment, we show that correspondence clustering provides comparable or better results than the traditional clustering. Moreover, the experimental results show that by enhancing agreement between two corresponding datasets, correspondence clustering produces clusterings that have lower variance than a traditional clustering algorithm. In the second experiment, we evaluate and compare different cluster initialization strategies for correspondence clustering.

5.1 Earthquake Dataset and Interestingness Function

We run our experiments on an earthquake dataset that is available on website of the U.S. Geological Survey Earthquake Hazards Program <http://earthquake.usgs.gov/>. The data includes the location (longitude, latitude), the time, the severity (Richter magnitude) and the depth (kilometers) of earthquakes. We uniformly sampled earthquake events from January 1986 to November 1991 as dataset O_1 and earthquake events between December 1991 and January 1996 as dataset O_2 . Each dataset contains 4132 earthquakes.

Suppose that a domain expert interests in finding regions where deep and shallow earthquakes are in close proximity; that is, he/she is interested in regions with a high variance for the attribute earthquake depth. The following interestingness function captures the domain expert’s notion of interestingness.

$$i(c) = \begin{cases} 0 & \frac{Var(c,z)}{Var(O,z)} \leq th \\ \left(\frac{Var(c,z)}{Var(O,z)} - th\right)^\eta & otherwise \end{cases} \tag{7}$$

where
$$Var(c, z) = \frac{1}{|c|-1} \sum_{o \in c} (z(o) - \mu_z)^2 \tag{8}$$

The attribute of interest $z(o)$ is depth of earthquake o ; $|c|$ is the number of objects in region c and μ_z is the average value of z in region c ; $th \geq 1$ is the reward threshold that captures what degree of earthquake depth variance the domain expert find news worthy; in general, regions with $i(c)=0$ are considered outliers. Finally, η with $\infty > \eta > 0$ is the reward function form parameter.

5.2 Experiment Investigating Variance Reduction

We run the interleaved approach of the representative based correspondence clustering, C-CLEVER-I, and the traditional clustering algorithm, CLEVER, and compare the results with respect to cluster quality and agreement.

First we run CLEVER on dataset O_1 and O_2 five times to generate five clusterings for each dataset. Then we run C-CLEVER-I for five iterations with $\alpha=1.0e-4$ and

$\alpha=2.0e-6$ for five times each. Figure 4 summarizes the experiments conducted. Each circle represents each clustering. The dashed lines between Clustering X_1 and X_2 in CLEVER show that fitness values ($q(X_1)+q(X_2)$), and $Agreement(X_1, X_2)$ of CLEVER are computed from all twenty five possible pairs of X_1 and X_2 . When correspondence clustering is used, those values are only computed for the five pairs of clusterings obtained by C-CLEVER-I (one for each run; indicated by solid lines with two ways arrows). For each clustering of C-CLEVER-I, the representatives from the previous iteration of its own clustering are used as initial representatives of the present iteration. The parameter settings of CLEVER and C-CLEVER-I are shown in Table 1 and Table 2. All parameters for CLEVER and C-CLEVER-I are set to the same values except for the values of p and p' . Since C-CLEVER-I is run for five iterations for each pair of clustering X_1 and X_2 , for a fair comparison, we set the p and p' of CLEVER to be five times higher than C-CLEVER-I. The experiment is evaluated by fitness function (equation (1)), agreement (equation (6)) and similarity (equation (5)). Table 3 shows average values of all the experimental results. The computation time measures the average wall clock time in milliseconds used by the algorithms to generate a pair of clusterings X_1 and X_2 . We use similarity measure $Sim(X, X')$ in equation (5) to assess variance between two clusterings generated using the same dataset. In general, the algorithm that produces higher $Sim(X, X)$ creates clusterings that are more similar in different runs, thus, exhibiting lower variance.

Table 1. Parameter settings of CLEVER

| | | | | |
|-----------------------|--|----------|------------|----------|
| $\beta=2.0$ | $p=100$ | $p'=100$ | $\eta=2.0$ | $th=1.2$ |
| Neighborhood size = 3 | Probabilities for add, delete, and replace representatives : 0.2, 0.2, 0.6 | | | |

Table 2. Parameter settings of C-CLEVER-I

| | | | | | |
|------------------------|--|--------|---------|------------|----------|
| $TCond = 5$ iterations | $\beta=2.0$ | $p=20$ | $p'=20$ | $\eta=2.0$ | $th=1.2$ |
| Neighborhood size = 3 | Probabilities for add, delete, and replace representatives : 0.2, 0.2, 0.6 | | | | |

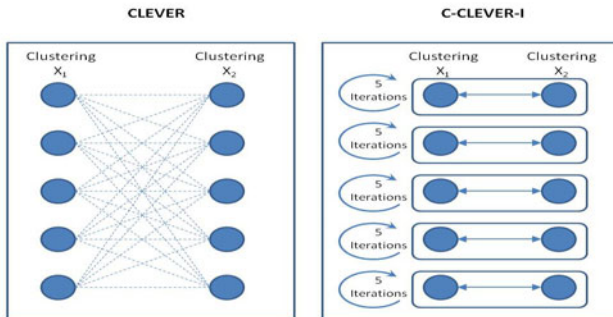


Fig. 4. Illustration of the experiment

Table 3. Comparison of average results of CLEVER and C-CLEVER-I

| | CLEVER | C-CLEVER-I ($\alpha=1.0e-5$) | C-CLEVER-I ($\alpha=2.0e-6$) |
|-----------------------|----------|-----------------------------------|-----------------------------------|
| Fitness $q(X_1)$ | 1896492 | 1896821 | 1870152 |
| Fitness $q(X_2)$ | 1756189 | 1713519 | 1685332 |
| $q(X_1) + q(X_2)$ | 3652681 | 3610341 | 3555485 |
| $Agreement(X_1, X_2)$ | 0.332909 | 0.349231 | 0.776172 |
| $Sim(X_1, X_1)$ | 0.665973 | 0.703770 | 0.663314 |
| $Sim(X_2, X_2)$ | 0.344895 | 0.623614 | 0.619446 |
| Computation Time | 5.48E+06 | 2.18E+06 | 2.30E+06 |

From Table 3, C-CLEVER-I with $\alpha=1.0e-5$ produces higher fitness values for clustering X_1 and but lower fitness values of X_2 than CLEVER. For $Agreement(X_1, X_2)$ and $Sim(X_1, X_1)$, C-CLEVER-I with $\alpha=1.0e-5$ produces slightly higher values than CLEVER but for $Sim(X_2, X_2)$, C-CLEVER-I produces significantly higher value than CLEVER. From this point of view, the higher values of $Sim(X_1, X_1)$ and $Sim(X_2, X_2)$ indicate that each run of C-CLEVER-I creates more similar clustering results for each clustering X_1 and X_2 which means that C-CLEVER-I produces lower variance than CLEVER. With $\alpha=2.0e-6$, C-CLEVER-I is forced to emphasize agreement. Therefore, the fitness values of clustering X_1 and X_2 of C-CLEVER-I are slightly lower than CLEVER but $Agreement(X_1, X_2)$, and $Sim(X_2, X_2)$ of C-CLEVER-I are significantly higher than CLEVER. Moreover, C-CLEVER-I computes its results about half of the runtime CLEVER uses.

From the experimental results, we conclude that correspondence clustering can reduce the variance inherent to representative-based clustering algorithms. Since the two datasets are related to each other, using one dataset to supervise the clustering of the other dataset can lead to more reliable clusterings by reducing variance among clusterings that would have resulted from using traditional representative-based clustering algorithms, as they are more susceptible to initial representatives and outliers. Moreover, obtaining higher agreement could be accomplished with only a very slight decrease in cluster quality.

5.3 Experiment for Representative-Based Correspondence Clustering with Different Methods of Initial Representative Settings

We run experiments to compare results of three initialization strategies for C-CLEVER-I; the three tested strategies are as follows : (1) random representatives (C-CLEVER-I-R), (2) representatives from the nearest neighbor of representatives of the counterpart clustering (C-CLEVER-I-C), and (3) the final representatives from the previous iteration are used as the initial representatives for the next iteration (C-CLEVER-I-O). For each option of the initial representative setting techniques, five pairs of clustering X_1 and X_2 are generated, similar to the previous experiment. Table 4 shows parameter settings used in the experiments. The average values of the experimental results are shown in Table 5.

Table 4. Parameter settings of C-CLEVER-I

| | | | | | | |
|------------------------|--|-------------|--------|---------|------------|----------|
| $TCond = 5$ iterations | $\alpha=2.0e-8$ | $\beta=2.8$ | $p=20$ | $p'=20$ | $\eta=2.0$ | $th=1.2$ |
| Neighborhood size = 3 | Probabilities for add, delete, and replace representatives : 0.2, 0.2, 0.6 | | | | | |

Table 5. Comparison of average results of C-CLEVER-I with different means of initial representative settings

| | C-CLEVER-I-C | C-CLEVER-I-O | C-CLEVER-I-R |
|--|--------------|--------------|--------------|
| Compound Fitness $\tilde{q}(X_1, X_2)$ | 9.857655 | 10.10406 | 9.952686 |
| Fitness $q(X_1)$ | 2.3E+08 | 2.66E+08 | 2.46E+08 |
| Fitness $q(X_2)$ | 2.14E+08 | 2.17E+08 | 2.13E+08 |
| $q(X_1) + q(X_2)$ | 4.44E+08 | 4.82E+08 | 4.59E+08 |
| Agreement (X_1, X_2) | 0.977259 | 0.459206 | 0.771505 |
| Computation Time | 3.23E+06 | 2.72E+06 | 7.10E+06 |

From Table 5, C-CLEVER-I-C produces clusterings with the highest agreement but the lowest compound fitness value. This is because C-CLEVER-I-C uses initial representatives that are closest to the representatives of its counterpart clustering. Then C-CLEVER-I-C generates clusterings X_1 and X_2 that are very similar which results in very high agreement. Though, the agreement is very high, the low fitness values lead to the low compound fitness value. For C-CLEVER-I-O, the initial representatives used are the final representatives from the previous iteration. In contrast to C-CLEVER-I-C, with $\alpha=2.0e-8$, C-CLEVER-I-O favors increasing fitness values rather than increasing agreement between the two clusterings. This is indicated by the highest fitness values but the lowest agreement value. As for C-CLEVER-I-R, it produces comparable fitness values and intermediate agreement value but consumes the highest computation time. This is due to the fact that C-CLEVER-I-R randomizes its initial representatives, which allows the algorithm to explore the dataset more thoroughly than the others but in return, it needs more time to find the solution.

6 Related Work

Correspondence clustering relates to coupled clustering, and co-clustering which both cluster more than one dataset at the same time. Coupled clustering [3] is introduced to discover relationships between two textual datasets by partitioning the datasets into corresponding clusters where each cluster in one dataset is matched with its counterpart in the other dataset. Consequently, the coupled clustering requires that the number of clusters in two datasets be equal. In general, the coupled clustering ignores intra-dataset similarity and concentrates solely on inter-dataset similarity. Our approach, on the other hand, provides no limitation on number of clusters. It considers both intra-dataset and inter-dataset similarities. The intra-dataset similarity is included through interestingness measures and the inter-dataset similarity is included through correspondences in sets of representatives.

Co-clustering has been successfully used for applications in text mining [4], market-basket data analysis, and bioinformatics [5]. In general, the co-clustering clusters two datasets with different schemas by rearranging the datasets. In brief, datasets are

represented as a matrix with one dataset is organized into rows while the other dataset is organized into columns. Then, the co-clustering partitions rows and columns of the data matrix and creates clusters which are subsets of the original matrix. In case of spatial data mining, re-organizing the data into data matrix causes spatial relationships related to location of data points to be lost: clusters are no longer guaranteed to be contiguous. Accordingly, co-clustering is not applicable to spatial clustering.

Correspondence clustering is also related to evolutionary clustering [6] that is used for multi-objective clustering. Evolutionary clustering clusters streaming data based on two criteria: the clustering quality of present data and its conformity to historical data.

In conclusion, the three reviewed clustering techniques cluster data based on distances alone whereas the correspondence clustering approach allows to cluster multiple datasets based on a domain expert's definition of interestingness and correspondence. Consequently, correspondence clustering is more general and can serve a much larger group of applications. For example, none of the three approaches can be used for the earthquake clustering problem we used in the experimental evaluation; in the experiment, clusters are formed by maximizing the variance of a continuous variable and not by minimizing the distances between objects that belong to the same cluster.

7 Conclusion

In this paper, we introduce a novel clustering approach called correspondence clustering that clusters two or more related spatial datasets by maximizing cluster interestingness and correspondence between clusters for the different datasets. A representative-based correspondence clustering framework is introduced and two representative-based correspondence clustering algorithms are proposed. We conducted experiments in which two datasets had to be clustered by maximizing cluster interestingness and agreement between the spatial clusters obtained. The results show that correspondence clustering can reduce the variance inherent to representative-based clustering algorithms. Moreover, high agreements could be obtained by only slightly lowering clustering quality. In general, correspondence clustering is beneficial for many applications, such as change analysis, co-location mining and applications that are interested in analyzing particular, domain-specific relationships between two or more datasets.

In addition, the paper proposes a novel agreement assessment measure that relies on re-clustering techniques and co-occurrence matrices. The agreement assessment technique proposed is applicable for (1) datasets with unknown object identity, (2) different number of objects in two datasets, (3) different number of clusters in two clusterings. Therefore, it is suitable for most types of spatial data.

References

1. Ding, W., Jiamthaphaksin, R., Parmar, R., Jiang, D., Stepinski, T., Eick, C.F.: Towards Region Discovery in Spatial Datasets. In: Proceedings of 12th Pacific-Asia Conference on Knowledge Discovery and Data Mining (2008)
2. Eick, C.F., Parmar, R., Ding, W., Stepinski, T., Nicot, J.-P.: Finding Regional Co-location Patterns for Sets of Continuous Variables in Spatial Datasets. In: Proceedings of 16th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (2008)

3. Marx, Z., Dagan, I., Buhmann, J.M., Shamir, E.: Coupled Clustering: A Method for Detecting Structural Correspondence. *Journal of Machine Learning Research* 3, 747–780 (2002)
4. Dhillon, I.S.: Co-clustering Documents and Words using Bipartite Spectral Graph Partitioning. In: *Proceedings of 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2001)
5. Cheng, Y., Church, C.M.: Biclustering of Expression Data. In: *Proceedings of 8th International Conference on Intelligent Systems for Molecular Biology* (2000)
6. Chakrabarti, D., Kumar, R., Tomkins, A.: Evolutionary Clustering. In: *Proceedings of 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2006)
7. Chen, C.S., Rinsurongkawong, V., Eick, C.F., Twa, M.D.: Change Analysis in Spatial Data by Combining Contouring Algorithms with Supervised Density Functions. In: *Proceedings of 13th Pacific-Asia Conference on Knowledge Discovery and Data Mining* (2009)
8. Rand, W.: Objective Criteria for the Evaluation of Clustering Methods. *Journal of the American Statistical Association* 66, 846–850 (1971)

Mining Trajectory Corridors Using Fréchet Distance and Meshing Grids

Haohan Zhu^{1,2}, Jun Luo², Hang Yin³, Xiaotao Zhou²,
Joshua Zhexue Huang², and F. Benjamin Zhan⁴

¹ Institute of Computing Technology, Chinese Academy of Sciences

² Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences
{hh.zhu,xt.zhou,zx.huang}@siat.ac.cn, {jun.luo,hang.yin}@sub.siat.ac.cn

³ University of Science and Technology of China

⁴ Texas Center for Geographic Information Science, Department of Geography,
Texas State University-San Marcos
zhan@txstate.edu

Abstract. With technology advancement and increasing popularity of location-aware devices, trajectory data are ubiquitous in the real world. Trajectory corridor, as one of the moving patterns, is composed of concatenated sub-trajectory clusters which help analyze the behaviors of moving objects. In this paper we adopt a three-phase approach to discover trajectory corridors using Fréchet distance as a dissimilarity measurement. First, trajectories are segmented into sub-trajectories using meshing-grids. In the second phase, a hierarchical method is utilized to cluster intra-grid sub-trajectories for each grid cell. Finally, local clusters in each single grid cell are concatenated to construct trajectory corridors. By utilizing a grid structure, the segmentation and concatenation need only single traversing of trajectories or grid cells. Experiments demonstrate that the unsupervised algorithm correctly discovers trajectory corridors from the real trajectory data. The trajectory corridors using Fréchet distance with temporal information are different from those having only spatial information. By choosing an appropriate grid size, the computing time could be reduced significantly because the number of sub-trajectories in a single grid cell is a dominant factor influencing the speed of the algorithms.

Keywords: Trajectory, Fréchet Distance, Meshing Grids.

1 Introduction

With the technology progress and popularity of location-aware devices, vast data of moving objects have been collected. Trajectory data are ubiquitous in the real world including tropical cyclones data, transportation system data, flocking sheep data, migrating birds data, to name a few. Consequently, trajectory analysis has become a pragmatic tool to discover moving objects patterns. Trajectory corridor, through which the moving objects frequently pass, is one of the

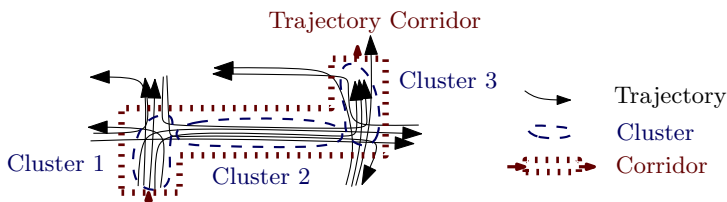


Fig. 1. An example of trajectories, clusters and corridors

moving patterns. In this paper, we address the trajectory corridor as concatenated sub-trajectory clusters which could help identify and predict the behaviors of the moving objects. An example to illustrate trajectories, local clusters and trajectory corridors is shown in Fig. 1.

Because not only may a trajectory belong to multiple trajectory corridors simultaneously, but trajectory corridors are also composed by different groups of trajectories at different parts. In this paper, we adopt a three-phase approach to mine the trajectory corridors using Fréchet distance and meshing grids. Firstly, trajectories are segmented into sub-trajectories according to the meshing-grids. In the second phase, a hierarchical method is utilized in each grid cell separately to cluster intra-grid sub-trajectories. Finally, the local clusters in each single grid cell are concatenated to construct trajectory corridors.

Summarizing, the contributions presented in this paper are:

- We introduce discrete Fréchet distance as a novel dissimilarity measurement between trajectory clustering because it is generally regarded as a more appropriate distance function for polygonal curves and could easily consider not only shapes, relative positions and orientations, but also velocities of trajectories.
- We propose a meshing grid structure. By utilizing grid structure, the segmentation and concatenation need only single traversing of trajectories or grid cells. When choosing appropriate grid size, the computing time could be reduced significantly since the dominant factor of the computing time is the amount of sub-trajectories in a single grid cell.

2 Related Work

Gaffney and Smyth [7] propose a method of trajectory clustering using mixture of regression models. Nanni and Pedreschi [10] propose a density-based trajectory clustering algorithm based on OPTICS [2]. In the above work, the object to be clustered is the whole trajectory, namely, one trajectory can be only in one cluster. Thus, trajectory corridors are not their objective.

In the trajectory clustering algorithm proposed by Lee et al. [9], trajectories are partitioned into a set of line segments which are clustered by using DBSCAN [6] algorithm. However, because Fréchet distance could handle polygonal curves

well, we need not to simplify the trajectories into line segments but only cut off trajectories into several short polygonal curves. And we adopt a hierarchical method for clustering to avoid distance accumulation discussed in Section 5.2.

3 Preliminary Definition

In reality, a moving object trajectory is a finite set of observations at discrete time stamps t_1, t_2, \dots, t_n . It could be assumed that an object moves between two consecutive time steps on a straight line and the velocity is constant along the segment. Hence we define the trajectory τ as a polygonal line with n vertices having time stamps.

Definition 1 (Trajectory). $\tau = \langle (t_1, p_1), (t_2, p_2), \dots, (t_n, p_n) \rangle$, where $p_i \in \mathbb{R}^d$.

The space is partitioned by meshing-grids. Every grid cell G_j has an id j to label it. The sub-trajectory is recorded in $\tau'_{i,j,mark}$ where i represents the original trajectory τ_i it belongs to, j represents the grid cell G_j it falls into and $mark$ is the mark to differentiate the different part of the same trajectory in the same grid. The definition of sub-trajectories is the same as that of trajectories.

Definition 2 (Local Cluster in Grid Cell G_j). $C_j = \langle \tau'_{i_1,j,mark_1}, \tau'_{i_2,j,mark_2}, \dots, \tau'_{i_m,j,mark_m} \rangle$, where $\tau'_{i_k,j,mark_k}$ ($1 \leq k \leq m$) is a sub-trajectory in grid cell G_j .

The local cluster in a certain grid cell is a set of sub-trajectories in that cell, so the cluster has no shape or range. However, the cluster has its own origin and destination. The position and velocity at origin and destination of a cluster are average values of sub-trajectories in that cluster.

Definition 3 (Trajectory Corridor). $TC = \langle C_{j_1}, C_{j_2}, \dots, C_{j_l} \rangle$, where C_{j_i} ($1 \leq i \leq l$) is a local cluster in grid cell G_{j_i} , and the consecutive local clusters C_{j_i} and $C_{j_{i+1}}$ need to follow concatenating criteria discussed in Section 5.3.

The trajectory corridor is a sequence of local clusters and the order of the sequence indicates the direction of the trajectory corridor. Every trajectory corridor is composed of either one local cluster or more. Moreover, different trajectory corridors may share the same local cluster.

4 Computing Fréchet Distance

Distance function is the key component of mining trajectory corridors because dissimilarity measurement is necessary to group similar trajectories. The Fréchet distance is a measurement for curves and surfaces. It is defined using reparameterizations of the shapes. The Fréchet distance is generally regarded as being a more appropriate distance function than the Hausdorff distance or other distances for curves [1] because it takes the continuity of the shapes into account.

Due to high complexity of computing Fréchet distance, the discrete Fréchet distance as a natural variant for polygonal curves is more proper for computing. Eiter and Mannila[5] proposed a dynamic programming algorithm to compute discrete Fréchet distance in $O(mn)$ time.

Consider two polygonal curves P, Q in \mathbb{R}^d given by the sequences of their vertices $\langle p_1, \dots, p_n \rangle$ and $\langle q_1, \dots, q_m \rangle$, respectively. Computing discrete Fréchet distance only uses coupling C which is a sequence of pairs of vertices $C=\langle C_1, \dots, C_k \rangle$ with $C_r=(p_i, q_j)$ for all $r=1, \dots, k$ and some $i \in \{1, \dots, n\}, j \in \{1, \dots, m\}$, fulfilling

- $C_1=(p_1, q_1)$ and $C_k=(p_n, q_m)$
- $C_r=(p_i, q_j) \Rightarrow C_{r+1} \in \{(p_{i+1}, q_j), (p_i, q_{j+1}), (p_{i+1}, q_{j+1})\}$ for $r=1, \dots, k-1$

Definition 4 (Discrete Fréchet Distance[3]). Let P, Q be two polygonal curves in \mathbb{R}^d , and let $|\cdot|$ denote an underlying norm in \mathbb{R}^d . Then the Discrete Fréchet Distance $\delta_{dF}(P, Q)$ is defined as

$$\delta_{dF}(P, Q) = \min_{\text{coupling } C} \max_{(p_i, q_j) \in C} |p_i - q_j|$$

where C ranges over all couplings of the vertices of P and Q .

If the distance computation $|\cdot|$ between vertices ignores velocities of trajectories, the distance function is shape dependent only like DTW[8], LCSS[11], EDR[4] and so on. However, the two trajectories may have similar shapes, but actually they represent totally different moving patterns when considering velocities illustrated in Fig 2. Yanagisawa et al. [12] propose a measurement combined DTW distance with Euclidean distance which considers both velocities and shapes of moving objects, whereas the distance is sensitive to the grouping parameter μ and they require time duration of different trajectories to be the same length. Hence another merit brought by Fréchet distance is that it could easily take account of velocities. By substituting $|\cdot|$ between vertices, not only spatial information of trajectories but also temporal information of trajectories can be considered. In our paper, a simply solution is provided that the distance $|\cdot|$ between two vertices in two dimensions is defined as $\sqrt{\omega_s(\Delta x^2 + \Delta y^2) + \omega_t \Delta v^2}$, and weights ω_s and ω_t differentiate the effects of spatial properties and temporal

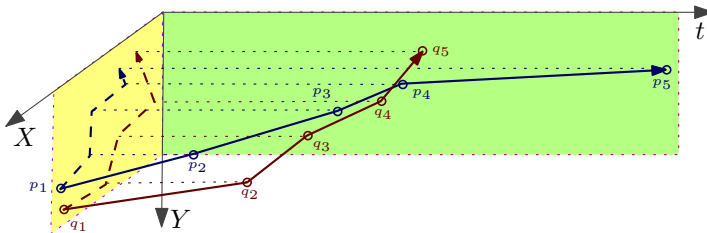


Fig. 2. The trajectories with similar shapes but different velocities

properties. v is the instant velocity at the vertex. If $\omega_t = 0$, $|\cdot|$ is translated into Euclidean norm. The experiments in Section 6 illustrates the trajectory corridors considering velocities are quite different from those using $\omega_t = 0$. Actually, adjusting weights to make spatial and temporal properties equally important is empirical and influenced by spatial and temporal units.

5 Mining Trajectory Corridors

The procedure of mining trajectory corridors which is composed of three phases is illustrated in Fig. 3. In the first phase, trajectories are segmented into sub-trajectories according to the meshing-grids. In the second phase, the hierarchical clustering algorithm based on discrete Fréchet distance is implemented in each grid cell. Finally, the local clusters in abutting grid cells are concatenated according the concatenation criteria to discover trajectory corridors.

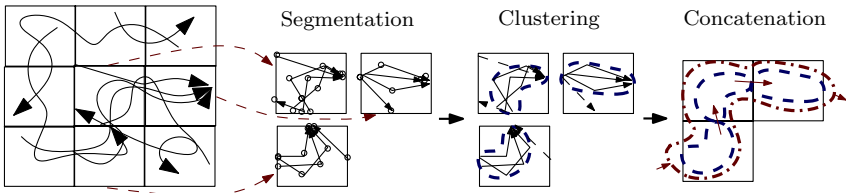


Fig. 3. An example of mining trajectory corridors

In this paper, we use meshing grids structure to segment trajectories and concatenate local clusters because by utilizing grids, the segmentation and concatenation need only single traversing of trajectories or grid cells. Furthermore when changing grid size appropriately, the computing time could be reduced significantly since the dominant factor of the computing time is the amount of sub-trajectories in a single grid cell. This advantage is theoretically and experimentally analyzed and discussed in Section 6.

5.1 Segmentation

Since the different parts of a certain trajectory may belong to different trajectory corridors, segmenting trajectory into sub-trajectories is indispensable. In the process of segmentation, the size of grid cells are assigned in advance. As illustrated in Fig. 4, when traversing each vertex in a trajectory, no segmentation will be executed until consecutive vertices pair (p_i, p_{i+1}) are not in the same grid cell. The intersections between the line segment $p_i p_{i+1}$ and edges of grid cells are computed. The trajectory is partitioned at each intersection. After segmentation, only the sub-trajectories in those grid cells which potentially contain local clusters will be preserved for the next phase. This process may reduce computing time dramatically when many grid cells include sparse sub-trajectories. The algorithm of segmentation costs $O(n)$ time, where n is the sum of the vertices of all trajectories.

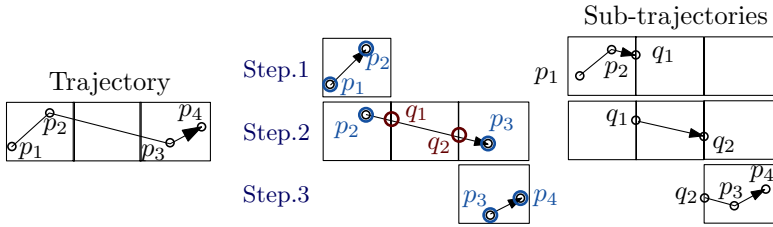


Fig. 4. Illustration of Segmentation

5.2 Intra-grid Sub-trajectory Hierarchical Clustering

In this paper, we adopt an agglomerative hierarchical clustering method and use two cluster distances d_{min} and d_{max} at the same time to avoid distance accumulation illustrated in Fig. 5. In the example, τ_1 and τ_8 are almost in the opposite directions but may be merged into the same cluster because each pair of nearby trajectories τ_i and τ_{i+1} has a small distance. Two cluster distances d_{min} and d_{max} are defined as follow: $d_{min}(C_i, C_j) = \min_d(p, q)$, $d_{max}(C_i, C_j) = \max_d(p, q)$, where $p \in C_i, q \in C_j$, $d(p, q)$ is modified discrete Fréchet distance between p, q . In this phase, the computing is only executed in the grid cells that have sufficient sub-trajectories. For each hierarchy, the nearest clusters are merged according to d_{min} . Namely, two clusters are merged when they have the minimal d_{min} . However, while d_{max} between the nearest clusters exceeds an certain threshold, no merging executes and the algorithm continues to the next hierarchy. Until the minimal d_{min} exceeds an termination condition, clustering ceases. Finally, the local clusters which do not have sufficient sub-trajectories are pruned. The algorithm for each grid can be computed in $O(n^2l^2 + n^2 \log n + n^2m^2)$ time, where n is the amount of sub-trajectories, l is the amount of vertices of one sub-trajectory, m is the amount of sub-trajectories in one cluster. An example of intra-grid sub-trajectory clustering using distance matrix and index is illustrated in Fig. 6.

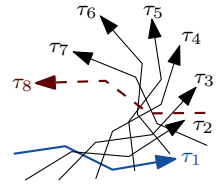


Fig. 5. Illustration of distance accumulation

5.3 Inter-grid Concatenation

In this phase, we propose an algorithm of concatenating local clusters to discover trajectory corridors. The concatenation criteria is defined as follow: When positions and velocities between the origin of one cluster and the destination of the other cluster in the adjacent grid cell are similar, we call the two local clusters are connectable. And we consider concatenation that does not require the adjacent local clusters sharing the sufficient same trajectories or even same amount of trajectories. Trajectory corridors are continuous channels with directions that moving objects frequently visit but enter and leave freely.

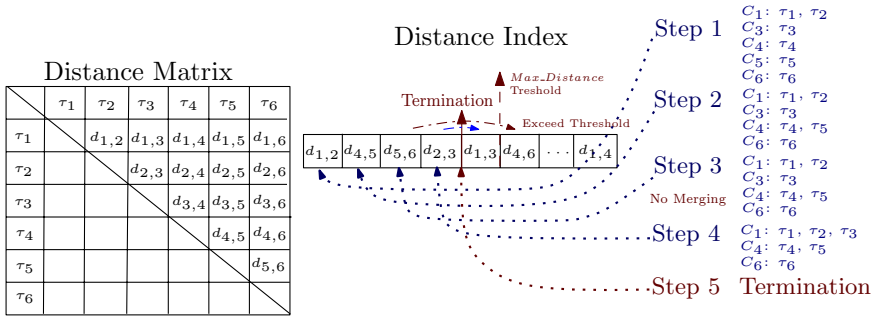


Fig. 6. Illustration of intra-grid sub-trajectory clustering using distance matrix and index

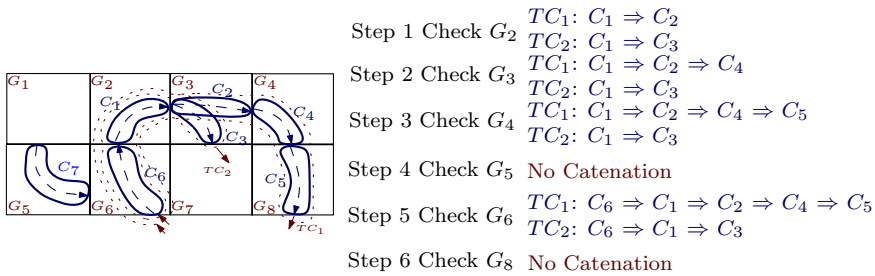


Fig. 7. Illustration of inter-grid concatenation

In the phase of local clusters concatenation, traversing all grid cells that have local clusters only once could find all possible trajectory corridors. In each step, we check one grid cell G_j and the local clusters in it. If there exists local clusters C_q in neighbor cells having the origins connectable with the destinations of the local clusters C_p in G_j , all trajectory corridors including C_p are concatenated to all trajectory corridors including C_q . This approach can be computed in $O(nk)$ time, where n is the amount of local clusters and k is the amount of trajectory corridors. An example of concatenation is illustrated in Fig. 7 and trajectory corridors with only one cluster are hidden.

6 Experiments

In this paper, all experiments were conducted by using the tropical cyclone best track data set¹. And for all experiments, parameters including pruning criteria, termination condition and concatenation criteria are constant.

Fig. 8(a) is the result of clustering which considers both spatial and temporal similarity, whereas, Fig. 8(b) is the result of clustering which ignores the velocity.

¹ <http://www.jma.go.jp/jma/jma-eng/jma-center/rsmc-hp-pub-eg/besttrack.html>

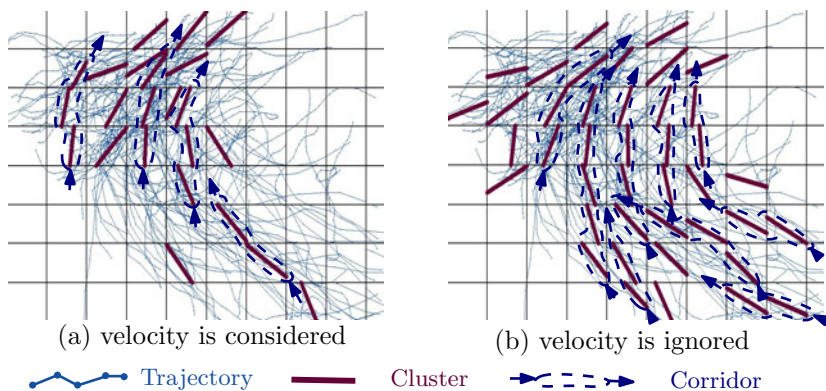


Fig. 8. Trajectory corridors considering velocity or not

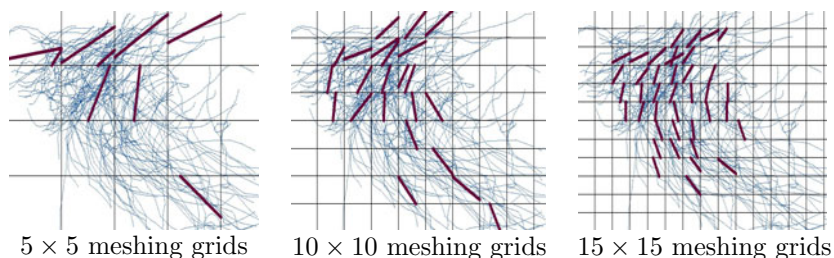


Fig. 9. Trajectory corridors in different grid sizes

The experiments successfully demonstrate that by considering temporal properties, the sub-trajectory clusters are different from those that have only spatial properties.

From Fig. 9, it is obvious that more grid cells produce more sub-trajectories, local clusters and trajectory corridors. However, the computing time is reduced significantly, when the amount of sub-trajectories per cell decreases from 21 to 5. (see Fig. 10(a)). Since building index runtime is the dominant factor in the overall runtime illustrated in Fig. 10(b), the overall running time could be approximated to $O(kn^2 \log n)$, where n is the amount of sub-trajectories in one grid cell and k is the amount of grid cells. Thus when choosing appropriate grid size, the computing time could be reduced, because the amount of sub-trajectories per cell may decrease. The grid size may affect both computing time and clustering quality. The quality of the results may decrease when grid size is larger due to more noises. So to keep the quality of the clustering results and to reduce computing time requires a trade-off which vary from data to data.

The experiments successfully validate our algorithm to discover trajectory corridors. The tropical cyclones in Western North Pacific and the South China Sea often have parts of their trajectories in such a corridor: start from the position

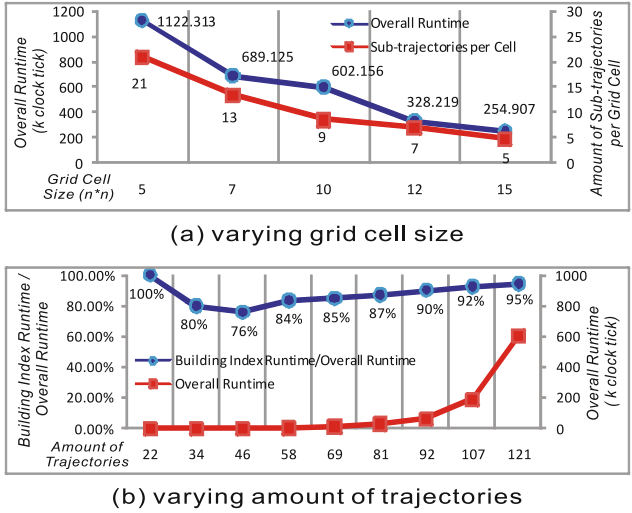


Fig. 10. Runtime in different meshing-grid sizes

around 135 °E–150 °E, 10 °N, move towards northwestern to the location about 120 °E–135 °E, 25 °N, then towards northeastern and finally end approximately at 145 °E, 40 °N.

7 Conclusions

In this paper, we adopt a three-phase approach to discover trajectory corridors using Fréchet distance as a dissimilarity measurement. The experiments successfully discovered tropical cyclone corridors by segmenting, clustering and concatenating various components of a trajectory. The trajectory corridors discovered by using Fréchet distance with temporal information may be more significant. The use of meshing grid structure could reduce computing time effectively by choosing an appropriate grid size.

References

1. Alt, H., Knauer, C., Wenk, C.: Comparison of distance measures for planar curves. *Algorithmica* 38(1), 45–58 (2003)
2. Ankerst, M., Breunig, M.M., Kriegel, H.-P., Sander, J.: Optics: Ordering points to identify the clustering structure. In: *SIGMOD Conference*, pp. 49–60 (1999)
3. Buchin, K., Buchin, M., Gudmundsson, J., Löffler, M., Luo, J.: Detecting commuting patterns by clustering subtrajectories. In: Hong, S.-H., Nagamochi, H., Fukunaga, T. (eds.) *ISAAC 2008*. LNCS, vol. 5369, pp. 644–655. Springer, Heidelberg (2008)
4. Chen, L., Tamer Özsu, M., Oria, V.: Robust and fast similarity search for moving object trajectories. In: *SIGMOD Conference*, pp. 491–502 (2005)

5. Eiter, T., Mannila, H.: Computing discrete fréchet distance. Technical Report CD-TR, 94(64) (1994)
6. Ester, M., Kriegel, H.-P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: KDD, pp. 226–231 (1996)
7. Gaffney, S., Smyth, P.: Trajectory clustering with mixtures of regression models. In: KDD, pp. 63–72 (1999)
8. Keogh, E.J.: Exact indexing of dynamic time warping. In: VLDB, pp. 406–417 (2002)
9. Lee, J.-G., Han, J., Whang, K.-Y.: Trajectory clustering: a partition-and-group framework. In: SIGMOD Conference, pp. 593–604 (2007)
10. Nanni, M., Pedreschi, D.: Time-focused clustering of trajectories of moving objects. *J. Intell. Inf. Syst.* 27(3), 267–289 (2006)
11. Vlachos, M., Gunopulos, D., Kollios, G.: Discovering similar multidimensional trajectories. In: ICDE, pp. 673–684 (2002)
12. Yanagisawa, Y., Satoh, T.: Clustering multidimensional trajectories based on shape and velocity. In: ICDE Workshops, p. 12 (2006)

Subseries Join: A Similarity-Based Time Series Match Approach

Yi Lin¹ and Michael D. McCool²

¹ Faculty of Computer Science, University of New Brunswick,
540 Windsor Street, Fredericton, NB, Canada E3B 5A3
ylin@unb.ca

² Intel of Canada, Ltd., 180 King St S, Suite 500,
Waterloo, Ontario, Canada N2J 1P8
michael.mccool@intel.com

Abstract. Time series data appears in numerous applications including medical data processing, financial analytics, network traffic monitoring, and Web click-stream analysis. An essential task in time series mining is efficiently finding matches between similar time series or parts of time series in a large dataset. In this work, we introduce a new definition of subseries join as a generalization of subseries matching. We then propose an efficient and robust solution to subseries join (and match) based on a non-uniform segmentation and a hierarchical feature representation. Experiments demonstrate the effectiveness of our approach and also show that this approach can better tolerate noise and phase-scaling than previous work.

Keywords: time series, similarity-based match, subseries join.

1 Introduction

Time series are composed of sequences of data items measured at (typically) uniform intervals. Time series arise frequently in many scientific and engineering applications, including finance, medicine, digital audio, and motion capture. Efficiently searching for similarities in a large time series dataset is a challenging problem, especially when partial or subseries matches are needed. Most previous work has focused on either whole match or subseries match problems. Whole matches find a time series similar to a given (query) time series in a dataset consisting of a collection of time series. Such matches are one-to-one. Subseries match, in contrast, finds similar *segments* (which we will call subseries) in a time series dataset to given a single query time series, and is a one-to-many match. However, when time series are very long and contain complex features, whole matches will find meaningless matches. Subseries matches are more specific but still depend on matching the whole of a specific query time series. This form of match still cannot deal with two timeseries that may have subseries pairs that are similar but are not similar as a whole. To deal with this case, we introduce an operation called *subseries join* which matches pairs of similar subseries in a

set of time series. Subseries join is a symmetric operation and produces many-to-many matches. It is a generalization of both whole match and subseries match. It is potentially useful in many data mining applications such as motif detection and anomaly detection [1]. In this work, we also present a non-uniform indexing method for time series based on intrinsic structure, and a search technique based on this structure that can efficiently compute subseries join. When restricted to matching, our approach can also tolerate both impulsive noise and additive noise better than previous approaches to these problems.

2 Related Work

In order to define the similarity of time series, we need to define a distance metric. The Euclidean distance metric has been used widely in the time series mining area. Various normalizations of the Euclidean distance have also been proposed, such as shifting, scaling, linear trend elimination, noise removal, and moving average filtering. However, Euclidean distance can only be used for measuring the similarity of two time series of the same length. It is also sensitive to noise and time shifting. Dynamic time warping (DTW) [2] defines another distance measure based on dynamic programming. It can detect similar time series even when they are subject to transformations along the time axis. DTW can also tolerate a limited amount of length variation, insertions and deletions, and additive noise. However, DTW often fails when the differences of the lengths of time series are large, it handles changes in length by replication or deletion of samples (which is a poor interpolation method), and it cannot handle impulsive noise. An approximate normalized information distance (ANID) based on Kolmogorov complexity has been studied in [3] but was found to be sensitive to impulsive noise. To overcome such limitations above, we will define a new distance function based on a feature segmentation and continuous feature warping.

Given a long time series and a short time series, most existing subseries match methods first segment the long time series. The common segmentation method is based on sliding windows. There are three major sliding-window based segmentation methods: FRM [4], Dual Match [5], and General Match [6]. All these methods are based on uniform segmentation. However, uniform segmentation requires the user to manually select the length of the segments, and is not sensitive to the actual behavior of the data. This arbitrary segmentation may cause undesirable division of important features in the data into different segments and involves an assumption of a particular scale for features. Although overlapping sliding windows can avoid division of features (at least at one scale), it introduces additional costs and a redundant representation.

In this work, we propose an alternative approach that divides time series data into non-uniform segments of arbitrary length based on intrinsic smoothness properties. Pairs of candidate matches are found using a spatial search, and then a dynamic programming approach is used to align candidate subsequences by continuous warping.

3 Subseries Join

Subseries join is a symmetric operation that returns all pairs of subseries drawn from two datasets that satisfy a given similarity threshold relative to some metric d and are also of maximal length:

Definition 1. Subseries join: *Given two sets of time series \mathcal{X} and \mathcal{Y} , the subseries join is the set of all pairs $(X_{i,k}, Y_{j,\ell})$ of subseries $X_{i,k} \subseteq X_k$ for $X_k \in \mathcal{X}$ and $Y_{j,\ell} \subseteq Y_\ell$ for $Y_\ell \in \mathcal{Y}$ such that $d(X_{i,k}, Y_{j,\ell}) \leq \epsilon$, and for which there does not exist any $X'_{i,k} \supset X_{i,k}$ and $Y'_{j,\ell} \supset Y_{j,\ell}$ where $X'_{i,k}$ is longer than $X_{i,k}$ and contains $X_{i,k}$ as a proper subset and where $Y'_{j,\ell}$ is longer than $Y_{j,\ell}$ and contains $Y_{j,\ell}$ as a proper subset for which $d(X'_{i,k}, Y'_{j,\ell}) < \epsilon$ or for which $d(X_{i,k}, Y'_{j,\ell}) < \epsilon$ or for which $d(X'_{i,k}, Y_{j,\ell}) < \epsilon$.*

Note that a subseries join computes a subseries match if one of the input datasets \mathcal{X} is a singleton set $\{X\}$.

We now present our approach to solve the subseries join problem. First, each time series in the dataset is smoothed and segmented by an anisotropic diffusion scale-space analysis [7,8]. Anisotropic diffusion, unlike Gaussian smoothing, ‘‘pins’’ zero crossings of the second derivative across scales: their positions are invariant, although they are progressively eliminated. We exploit this property to create a strict hierarchy of segments that can be used as an index. Zero crossings of the second derivative are present at discontinuities in the data, but not all such zero crossings are useful discontinuities. Therefore, we only segment the data at zero-crossings where the gradient magnitude is also large [9].

Next, to represent features of a segment A of time series X , we use a polynomial $P(A, t)$ to approximate its shape, but with the parameter space rescaled to the interval $[0, 1]$. We can generalize this to an envelope by adding an interval to $P(A, t)$ that allows us to compute conservative distance bounds. To evaluate the similarity between two segments A and B , which may be of different lengths $|A|$ and $|B|$, we define the (square of the) distance function d as follows:

$$d^2(A, B) = \gamma \int_0^1 (P(A, t) - P(B, t))^2 dt + (1 - \gamma) \left(\frac{|A| - |B|}{\max\{|A|, |B|\}} \right)^2, \quad (1)$$

The first term compares shape, the second length; the ratio between these is controlled by varying γ over $[0, 1]$. This distance can be computed analytically from the coefficients of the polynomials and the lengths of the segments. In fact, the polynomial coefficients and the segment lengths can be placed in a single vector, and this vector can be linearly transformed so that the ordinary Euclidean distances on the transformed coefficients can be used [10]. We can also transform polynomial envelopes (polynomials plus intervals which can conservatively bound the original data) into axis-aligned line segments (in the higher-dimensional abstract feature space) and compute minmax distance between them for conservative distance bounds between functions bounded by the original envelopes. Note that we do not apply the Euclidean distance directly into the original time series data, instead we use the Euclidean distance over the feature representations in a

higher-dimensional parameter space. This distance function can deal with time series of different lengths easily and interpolation of shape is continuous.

We then build an index by inserting features at all scales of the dataset into an R-tree in the transformed feature space. Every time series in the dataset is represented as a feature sequence. An R-tree join operation [11] can be used to obtain candidate pairs of features whose distance is less than a predefined threshold. Based on the associated leaves of the R-trees, pairs of feature sequences can be found by counting the number of pairs of matching features from each sequence. If this number is greater than a predefined threshold, these two feature sequences are taken as a pair of candidate matching feature sequences. Finally, we use a sequence alignment algorithm based on dynamic programming [10] to align two candidate matching feature sequences; from this alignment we can compute an overall metric for the entire match by summing the metrics for the matching features.

4 Experimental Evaluations

To evaluate effectiveness of our approach (SJ in the following), we consider two baseline approaches: dynamic time warping (DTW) [2] and approximate normalized information distance (ANID) [3]. We will limit our comparisons to clustering and matching problems since previous solutions did not include the concept of subseries join.

All testing time series data are extracted or synthesized from the UCR Time Series Data Mining Archive [12]. A set of 40 time series \mathbb{D} were extracted from the UCR Time Series Data Mining Archive. The total number of samples in this dataset is 1,091,465. The average number of samples per time series is 27,287. The dataset is used to generate other synthetic testing datasets in a way that lets us know the correct answers for testing purposes.

We first generate data for a clustering problem. For each time series $X \in \mathbb{D}$, 50 variations are generated from X by uniformly scaling them by γ times the original length of X . Given $x_i \in X$, define a variation $x' \in X'$ computed by:

$$x'_i = x_{\lceil i/(\gamma) \rceil} \quad (2)$$

where γ is a random value with $\gamma \geq 1$.

Given this synthetic construction of testing datasets, the correct clustering and ranking results can be represented by an evolutionary tree that can be easily computed in advance. The correct clustering result is that each of the 50 synthetic time series should be clustered into the same set with its seed time series. The correct ranking result is that each of the 50 synthetic time series should be ranked according to the distance to its seed time series. The rank is defined as the order in a sequence sorted according to the Euclidean distance metric in one cluster.

Two metrics are used to evaluate the errors of the search results, where e_c is called the *clustering error* and e_r is called the *rank error*.

$$e_c = \frac{\text{the number of time series that should be in } C_i \text{ but are not}}{\text{the number of time series in } C_i} \quad (3)$$

$$e_r = \frac{\text{the number of time series in } C_i \text{ that are in the wrong rank}}{\text{the number of time series in } C_i} \quad (4)$$

The values of e_c and e_r of DTW, ANID, and SJ are shown in Figure 1. Given two seed time series and two variations computed from (2), Figure 2 shows the clustering results returned by DTW, ANID and SJ. The results show that our approach produces fewer clustering and ranking errors than DTW and ANID.

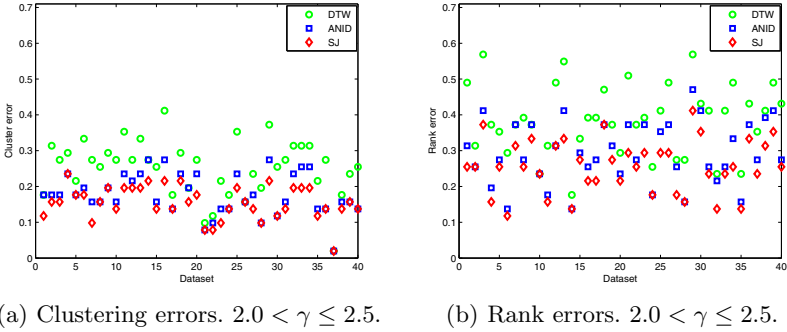


Fig. 1. Clustering errors e_c and rank errors e_r produced by DTW, ANID and SJ

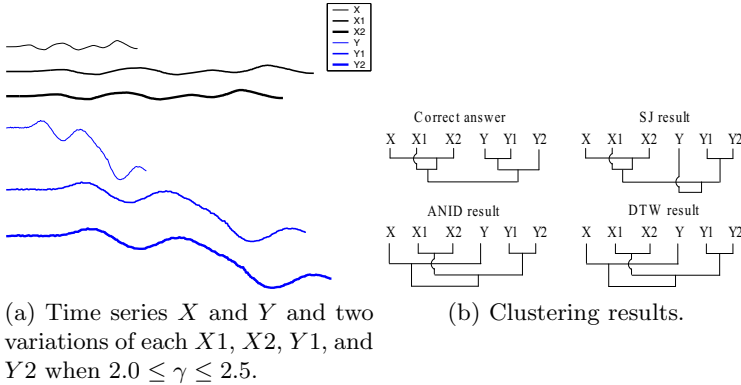


Fig. 2. Clustering results returned by SJ, ANID, and DTW

To evaluate the robustness of the proposed approach to additive noise, we generated 50 variations for each time series $X \in \mathbb{D}$ by adding a factor μ of additive noise. Given $x_i \in X$, define a variation $x' \in X'$ computed by:

$$x'_i = x_i + \mu(\max(X) - \min(X)) \quad (5)$$

where μ is a uniform random value over various ranges to be defined. The values of e_c and e_r for DTW, ANID, and SJ are shown in Figure 3. Figure 4 shows

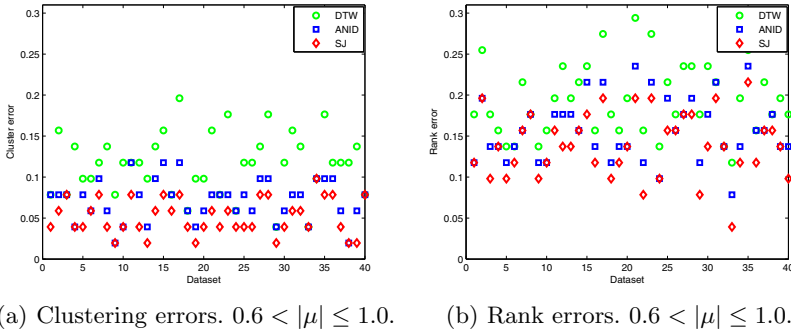


Fig. 3. Clustering errors e_c and rank errors e_r with additive noise

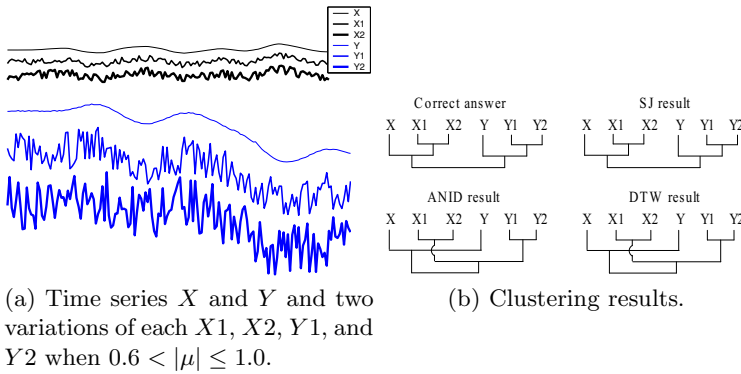


Fig. 4. Clustering results with additive noise

clustering results returned by DTW, ANID, and SJ for two seed time series and two variations computed from (5). Our approach again produces fewer clustering and ranking errors than DTW and ANID. The clustering result of SJ is in fact exactly the same as the correct answer, while the clustering results of DTW and ANID are quite different. This experiment demonstrates that our approach is more robust to additive noise in time series than either DTW or ANID.

Similarly, we generate 50 synthesized time series for each time series $X \in \mathbb{D}$ by adding impulsive noise to some elements of X . Given some element $x_j \in X$,

$$x'_j = x_j + \rho(\max(X) - \min(X)) \tag{6}$$

where ρ is a random value with $\rho > 0$. The number of such elements is much less than the total number of elements in the time series. The values of e_c and e_r of DTW, ANID, and SJ are shown in Figure 5. Our approach produces no clustering errors and the percentages of ranking errors are less than 5%. However, the percentages of both clustering errors and ranking errors produced by DTW and ANID are above 40%. Figure 6 shows clustering results returned by SJ,

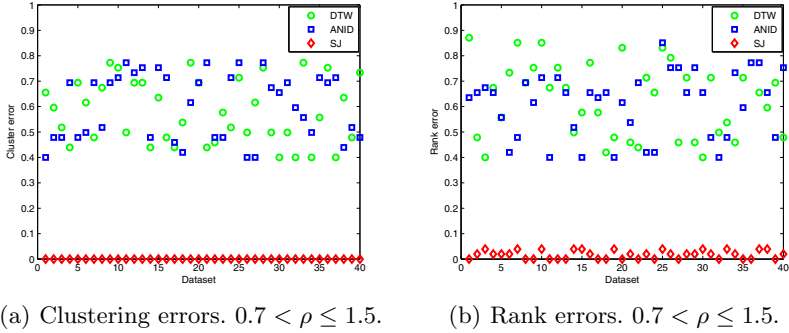


Fig. 5. Clustering errors e_c and rank errors e_r with impulsive noise

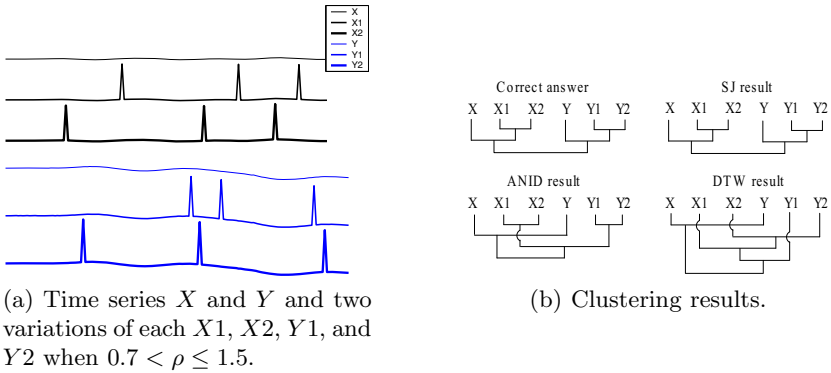


Fig. 6. Clustering results with impulsive noise

ANID, and DTW, given two seed time series and four variations computed from (6). According to the correct answer, the clustering result of SJ is correct, while the clustering results of DTW and ANID are not. This experiment demonstrates that DTW and ANID cannot tolerate impulsive noise but our approach can tolerate impulsive noise effectively.

The experimental results show that our approach produces fewer clustering errors and rank errors than ANID and DTW no matter whether the data is uniformly scaled, has additive noise, or has impulsive noise. ANID can tolerate more uniform scaling and additive noise than DTW. However, both ANID and DTW fail in the presence of impulsive noise. In all experiments our approach demonstrated a much greater tolerance to uniform scaling, additive noise, and impulsive noise than either DTW or ANID and so we conclude that it is a more robust approach.

5 Conclusion

In this work, we proposed a new definition of subseries join that finds similar subseries in two or more time series. We also proposed a method to efficiently compute subseries join based on a hierarchical feature representation and non-uniform segmentation. Experiments have demonstrated the effectiveness and efficiency of our approach by testing on a set of synthetic time series. Compared with previous work, our approach is observed to be relatively immune to changes in length, additive noise, and impulsive noise.

References

1. Lin, Y., McCool, M.D., Ghorbani, A.A.: Motif and anomaly discovery of time series based on subseries join. In: IAENG International Conference on Data Mining and Applications, ICDMA 2010 (2010)
2. Myers, C., Rabiner, L.: A level building programming dynamic time warping algorithm for connected word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing* 29(2), 284–297 (1981)
3. Cilibrasi, R., Vitányi, P.M.: Clustering by compression. *IEEE Transactions on Information Theory* 51(4), 1523–1545 (2005)
4. Faloutsos, C., Ranganathan, M., Manolopoulos, Y.: Fast subsequence matching in time-series databases. In: Proceedings of 1994 ACM SIGMOD International Conference on Management of Data, pp. 419–429 (1994)
5. Moon, Y.S., Whang, K.Y., Loh, W.K.: Duality-based subsequence matching in time-series databases. In: Proceedings of the 17th International Conference on Data Engineering, pp. 263–272 (2001)
6. Moon, Y.S., Whang, K.Y., Han, W.S.: General match: a subsequence matching method in time-series databases based on generalized windows. In: Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data, pp. 382–393 (2002)
7. Lin, Y., McCool, M.D.: Nonuniform segment-based compression of motion capture data. In: Bebis, G., Boyle, R., Parvin, B., Koracin, D., Paragios, N., Tanveer, S.-M., Ju, T., Liu, Z., Coquillart, S., Cruz-Neira, C., Müller, T., Malzbender, T. (eds.) ISVC 2007, Part I. LNCS, vol. 4841, pp. 56–65. Springer, Heidelberg (2007)
8. Perona, P., Malik, J.: Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 12(7), 629–639 (1990)
9. Canny, J.: A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 8(6), 679–698 (1986)
10. Lin, Y.: Subseries join and compression of time series data based on non-uniform segmentation. PhD thesis, University of Waterloo (2008)
11. Shekar, S., Chawla, S.: *Spatial Databases: a Tour*, 1st edn. Prentice-Hall, Englewood Cliffs (2003)
12. Keogh, E.: The UCR time series data mining archive. Department of Computer Science and Engineering, University of California, Riverside (2006), http://www.cs.ucr.edu/~eamonn/time_series_data

TWave: High-Order Analysis of Spatiotemporal Data

Michael Barnathan¹, Vasileios Megalooikonomou¹, Christos Faloutsos²,
Feroze B. Mohamed³, and Scott Faro³

¹ Data Engineering Laboratory (DEnLab), Department of Computer and Information Sciences,
Temple University, 1805 N. Broad St., Philadelphia, PA, USA 19122

² School of Computer Science, Carnegie Mellon University,
5000 Forbes Ave., Pittsburgh, PA 15213

³ Department of Radiology, Temple University School of Medicine,
3401 N. Broad St. Philadelphia, PA 19140

{mbarnath, vasilis}@temple.edu, christos@cs.cmu.edu,
{feroze, faros}@temple.edu

Abstract. Recent advances in data acquisition and sharing have made available large quantities of complex data in which features may have complex interrelationships or may not be scalar. For such datasets, the traditional matrix model is no longer appropriate and may fail to capture relationships between features or fail to discover the underlying concepts that features represent. These datasets are better modeled using tensors, which are high-order generalizations of matrices. However, naive tensor algorithms suffer from poor efficiency and may fail to consider spatiotemporal neighborhood relationships in analysis. To surmount these difficulties, we propose TWave, a wavelet and tensor-based methodology for automatic summarization, classification, concept discovery, clustering, and compression of complex datasets. We also derive TWaveCluster, a novel high-order clustering approach based on WaveCluster, and compare our approach against WaveCluster and k-means. The efficiency of our method is competitive with WaveCluster and significantly outperforms k-means. TWave consistently outperformed competitors in both speed and accuracy on a 9.3 GB medical imaging dataset. Our results suggest that a combined wavelet and tensor approach such as TWave may be successfully employed in the analysis of complex high-order datasets.

Keywords: Tensors, matrix models, wavelets, spatiotemporal mining.

1 Introduction

The traditional approach to data representation utilizes a matrix structure, with observations in the rows and features in the columns. Although this model is appropriate for many datasets, it is not always a natural representation because it assumes the existence of a single target variable and lacks a means of modeling dependencies between other features. Additionally, such a structure assumes that observed variables are scalar quantities by definition. This assumption may not be valid in certain domains, such as diffusion tensor imaging, where higher-order features predominate.

Traditionally, these problems have been solved by reducing the features to scalars and fitting the dataset to a matrix structure. However, as well as potentially losing information, this strategy also employs a questionable approach from a philosophical standpoint: attempting to fit the data to an imprecise model rather than attempting to accurately model the existing structure of the data. Finally, while it may be possible to model dependencies between features by making many runs, each with a different target variable, this yields suboptimal performance and may not be computationally feasible when real-time performance is required or when the dataset is very large.

To address these issues, we propose to model such datasets using *tensors*, which are generalizations of matrices corresponding to r -dimensional arrays, where r is known as the *order* of the tensor. Using a combination of wavelet and tensor analysis tools, we propose a framework for summarization, classification, clustering, concept discovery, and compression, which we call TWave. Applying our technique to analysis of the MNIST digit recognition dataset [6] and a large real-world spatiotemporal dataset, we compare the performance of TWave against voxelwise, SVD-based, wavelet-only, and tensor-only techniques and demonstrate that TWave achieves superior results and reduces computation time vs. competing methodologies.

2 Background

2.1 Tensor Tools

Tensors are defined within the context of data mining as multidimensional arrays. The number of indices required to index the tensor is referred to as the *rank* or *order* of the tensor, while each individual dimension is referred to as a *mode*. The number of elements defined on each mode is referred to as the mode's *dimensionality*. The dimensionality of a tensor is written in the same manner as the dimensionality of a matrix; for example, $20 \times 50 \times 10$. Tensors represent generalizations of scalars, vectors, and matrices, which are respectively orders 0, 1, and 2.

An important operation applicable to our analysis is the *tensor product* (also the *outer product*). This product generalizes from the Kronecker product, but results in another tensor rather than a block matrix. Given order r and s tensors \mathcal{A} and \mathcal{B} , their tensor product $\mathcal{A} \otimes \mathcal{B}$ is a tensor of order $r + s$:

$$(\mathcal{A} \otimes \mathcal{B})_{i_1, i_2, \dots, i_r, j_1, j_2, \dots, j_s} = \mathcal{A}_{i_1, i_2, \dots, i_r} * \mathcal{B}_{j_1, j_2, \dots, j_s}$$

Singular value decomposition (SVD) is a unique factorization by which an $m \times n$ matrix is decomposed into two projection matrices and a core matrix, as follows:

$$\mathbf{A} = \mathbf{U} \times \mathbf{\Sigma} \times \mathbf{V}^T$$

where \mathbf{A} is an $m \times n$ matrix, \mathbf{U} is an $m \times m$ column-orthonormal projection matrix, \mathbf{V} is an $n \times n$ column-orthonormal projection matrix, and $\mathbf{\Sigma}$ is a diagonal $r \times r$ *core matrix*, where r is the (matrix) rank of matrix \mathbf{A} .

SVD is used in Latent Semantic Analysis (LSA), an unsupervised summarization technique [1]. Here \mathbf{A} is treated as a term-document matrix. In this context, singular value decomposition automatically derives a user-specified number of latent *concepts* from the given terms, each representing a linear combination. The projection matrices

\mathbf{U} and \mathbf{V} contain term-to-concept and document-to-concept similarities, respectively. Thus, SVD can be used to provide simple yet powerful automatic data summarization.

The natural extensions of singular value decomposition to tensors are the *Tucker* and *PARAFAC* decompositions [2,3]. Let \mathcal{A} be an order- r tensor. Tucker decomposition is a factorization into a *core tensor* \mathcal{G} and *projection matrices* \mathbf{U}_i :

$$\mathcal{A} = \mathcal{G} \times \mathbf{U}_1 \times \mathbf{U}_2 \times \dots \times \mathbf{U}_r$$

Though the Tucker decomposition provides SVD-like data summarization, evaluating it requires computing \mathcal{A} 's covariance matrix. This can come at a memory cost of $\Omega(n^2)$, which, for large datasets such as ours, may be prohibitive. Fortunately, PARAFAC avoids this problem. PARAFAC is a generalization of PCA [2] and forms the basis of our tensor analysis approach. Given a user-specified number of concepts c , PARAFAC decomposes an order- r tensor \mathcal{A} into a columnwise sum of the tensor product of r projection matrices, denoted $\mathbf{U}^{(1)} \dots \mathbf{U}^{(r)}$, as follows:

$$\mathcal{A} = \sum_{i=1}^c \lambda_i \mathbf{U}_{:,i}^{(1)} \otimes \mathbf{U}_{:,i}^{(2)} \otimes \dots \otimes \mathbf{U}_{:,i}^{(r)}$$

Where the \mathbf{U} matrices represent projection matrices containing mode-to-concept similarities and λ represents a c -element scaling vector, in which each element represents the strength of a concept. The notation $\mathbf{U}_{:,i}$ refers to the i th column of \mathbf{U} .

Both the Tucker and PARAFAC decompositions may be computed using alternating least squares (ALS) [4], as shown below:

1. Given an order- r tensor \mathcal{A} , declare projection matrices $\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \dots, \mathbf{U}^{(r)}$.
2. Let $i = 1$.
3. Holding all other matrices constant, solve the following equation for $\mathbf{U}^{(i)}$:

$$\mathbf{u}^{(i)} = \mathcal{A}^{(i)} \left(\bigodot_{j=1..r \wedge j \neq i} \mathbf{U}^{(j)} \right) \left(\prod_{j=1..r \wedge j \neq i} [\mathbf{U}^{(j)}]^T \mathbf{U}^{(j)} \right)^*$$

Where \odot represents the n -ary Khatri-Rao product, $*$ represents the Moore-Penrose pseudoinverse, and $\mathcal{A}^{(i)}$ represents \mathcal{A} matricized [4] on mode i .

4. Repeat for all i from 1 to r until convergence is attained.

The resulting PARAFAC decomposition is illustrated in Figure 1 below:

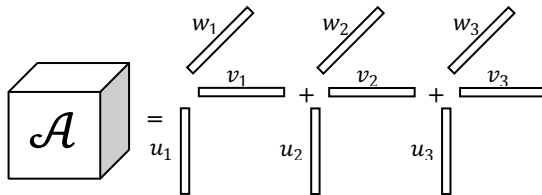


Fig. 1. Illustration of a third-order PARAFAC decomposition

3 Proposed Method

3.1 Overview

Our methodology makes use of both wavelets and tensors. Because spatiotemporal data tends to exhibit a high degree of spatial locality, the spatiotemporal modes of the dataset are first preprocessed using an m -dimensional discrete wavelet transform (obtained through cascading), where m is the number of spatiotemporal modes. For applications other than clustering, we utilize the Daubechies-4 wavelet; clustering itself is optimally paired with a hat-shaped wavelet such as the (2,2) biorthogonal wavelet, as these wavelets boost the strengths of dense clusters of points while suppressing outliers. We then linearize the wavelet coefficients to form a vector representing all spatiotemporal voxels in the dataset, reducing the order of the tensor by $d-1$; this overcomes many of the performance issues associated with a high-order pure tensor approach and allows us to threshold the discovered wavelet coefficients, storing the results in a sparse matrix to achieve a significant compression rate.

PARAFAC is then performed using alternating least squares and the resulting projection matrices are stored and analyzed, either by direct inspection or as input to a classifier. This method provides a general framework for further tensor and wavelet analysis, including concept discovery, compression, clustering, and classification.

3.2 Other Methods

It is also possible to analyze data using wavelets and tensors alone, or by using neither preprocessing method (the voxelwise approach). Singular value decomposition run on the dataset in matrix representation additionally provides a benchmark for comparison of the tensor model and techniques.

We performed voxelwise classification by linearizing each image in the dataset and using the normalized values of an image's voxels as a feature vector in classification. Similarly, we performed wavelet classification by using each image's linearized 3-level wavelet coefficient vector (using the Daubechies-4 wavelet) as a feature vector representing that image. Both approximation and detail coefficients at each resolution were included in this analysis.

3.3 Classification

To perform classification using TWave, we wavelet-transform the dataset, run a tensor decomposition such as PARAFAC or Tucker, and directly use each wavelet coefficient's similarity to each concept as an element in the feature vector. We then perform k -nearest neighbor classification, which assigns a class to each image based on the majority class of that image's k nearest neighbors (using Euclidean distance).

When classifying on a variable other than the principal variable of the dataset, we subtract the mean of the principal variable from the dataset. We have empirically observed this to boost accuracy.

3.4 TWaveCluster

We extended the WaveCluster algorithm to use the PARAFAC decomposition rather than a connected component algorithm to grow the clusters, calling our algorithm

TWaveCluster. Our approach exhibits a number of advantages, including the ability to create a fuzzy clustering (where each voxel’s degree of membership in cluster c is its similarity to concept c in the decomposed tensor), the ability to cluster noncontiguous voxels based on patterns in the projected concept space, and even the ability to discover clusters that extend across modes of the tensor. Our approach also has the advantage of simple cluster validation, as the terms in the λ vector automatically represent cluster variance.

The first few steps of our algorithm are identical to WaveCluster:

- Quantize data, using the counts of each grid cell in place of the original data.
- Apply a wavelet transformation using a hat-shaped wavelet (such as the (2,2) or (4,2) biorthogonal wavelets), retaining the approximation coefficients.
- Threshold cells in the transformed space. Cells with values above a user specified density threshold are considered “significant”.

However, the remaining steps in our algorithm differ:

- Model significant cells as a tensor $\mathcal{X} \in \mathfrak{R}^{d_1 \times d_2 \times \dots \times d_r}$.
- For a user-specified k , run a k -concept PARAFAC-ALS analysis on \mathcal{X} : $\mathcal{X} = \sum_{i=1}^k \lambda_i \mathbf{U}_{:,i}^{(1)} \otimes \mathbf{U}_{:,i}^{(2)} \otimes \dots \otimes \mathbf{U}_{:,i}^{(r)}$.
- For each c from 1 to k , recompose a tensor using only column c of each projection. The resulting tensor \mathcal{X}_c contains voxel similarities to concept c :

$$\mathcal{X}_c = \lambda_c \mathbf{U}_{:,c}^{(1)} \otimes \mathbf{U}_{:,c}^{(2)} \otimes \dots \otimes \mathbf{U}_{:,c}^{(r)}$$

- Assign every voxel the cluster label of its most similar concept:

$$(\forall x \in \mathcal{X}) \mathcal{L}_x = \arg \max_{1 \leq c \leq k} (\mathcal{X}_c)_x$$

4 Results

4.1 Dataset

We analyzed each approach on a high-order motor task fMRI dataset consisting of 11 subjects performing 4 simple motor tasks: left finger-to-thumb, left squeeze, right finger-to-thumb, and right squeeze. Classification was also performed on 10,000 randomly-sampled observations from the low-order MNIST digit recognition dataset, split into 5,000 element training and test sets [6]. Acquisition of the fMRI dataset took place using one scanner and one common set of acquisition parameters. Data was acquired from each subject over 120 time points, each 3 seconds long. The period of each task was 30 seconds. Each acquired volume consisted of $79 \times 95 \times 69$ voxels. Thus, the dataset was most easily represented as a 6th order tensor of dimensionality $79 \times 95 \times 69 \times 120 \times 4 \times 11$, of which the first four modes were spatiotemporal and the remaining two were categorical.

4.2 Discovered Concepts

When summarizing the data using a 2-concept TWave analysis, we noticed two outliers among the subject-to-concept similarities, which we found corresponded exactly to the 2 left-handed subjects in the dataset. This pattern was made even more explicit when subtracting the subject means from each subject's set of images, suggesting that the task residuals discriminate better between left and right handed subjects than when task activations are biased by subjects' means. The results of TWave using the Daubechies-4 wavelet and mean subtraction are shown in Figure 2. These results suggest that PARAFAC may be employed as a powerful concept-discovery and feature extraction tool on complex datasets, though we caution that a larger dataset may be necessary to adequately confirm these findings.

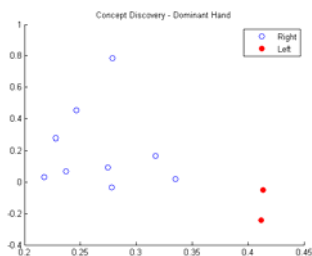


Fig. 2. A 2-concept projection using TWave. The two rightmost points are left-handed.

4.3 Classification

Use of wavelets in particular greatly improved subject classification accuracies, given a complex enough wavelet (to 98% using the Daubechies-4 wavelet but only to 82% for the Haar wavelet). We were able to threshold up to the weakest 98% of wavelet coefficients without any loss of subject or task classification accuracy, greatly improving time and space costs while preserving the discriminative power of the classifier. Further compression is possible in the decomposed tensor through truncation of weak concepts (though computation of these concepts is expensive).

Task classification was more difficult because the intra-subject between-task variance ($\sigma^2 = 179.29$) was less than the between-subject variance ($\sigma^2 = 9066.85$). Initial results yielded only 2% accuracy for voxelwise analysis and 27% accuracy for wavelet-based analysis. However, by subtracting the voxelwise mean of each subject across all tasks, we were able to improve classification substantially. Use of MPCA+LDA [7] as a preprocessing step further improved accuracy. As the sampled MNIST digit recognition dataset is a dense low-order dataset, less difference is seen between low and high-order approaches than in the fMRI dataset, though wavelet preprocessing still did significantly boost accuracy.

4.4 Clustering

We analyzed two subjects on all four spatiotemporal modes of the fMRI tensor using the k -means ($k=4$) and TWaveCluster ($k=5$, density threshold=85th percentile)

approaches. Average running times for each method were 53 seconds and 23 seconds, respectively. Discovered clusters are shown in Figure 3. A demarcation can be seen between the frontal and temporal regions of the brain in the TWaveCluster results; this distinction is less clear in *k*-means. The clusters discovered by TWaveCluster show a greater degree of symmetry and homogeneity than the *k*-means clusters, and also yield a clustering in-line with domain expectations.

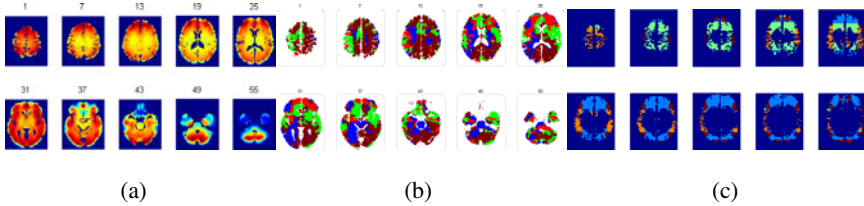


Fig. 3. (a) Activation in a right-handed subject performing a left finger-to-thumb task and the clusters discovered by (b) *k*-means and (c) TWaveCluster. Only significant voxels are shown.

4.5 Speed and Summary of Results

Runtime was assessed for the voxelwise, wavelet-based, and TWAVE approaches on a dual-processor 2.2 GHz Opteron system with 4 GB of memory. The SVD and pure tensor approaches were measured on an 8 processor (16 core) 2.6 GHz Opteron supercomputer with 128 GB of memory. Despite running on a much more powerful system, the tensor and SVD approaches still took significantly longer to complete than other approaches, as shown in Tables 1 and 2:

Table 1. High-order fMRI dataset runtimes, subject and task classification accuracies, compressed dataset size, and ability to automatically identify left-handed subjects

| | Voxels | Wavelets | SVD | PARAFAC | TWave | TWave+MPCA/LDA |
|----------|--------|----------|---------------|---------------|---------|----------------|
| Runtime | 95 min | 112 min | 3 days | 8 days | 117 min | 130 min |
| Subjects | 52% | 98% | 80% | 88% | 96% | 100% |
| Tasks | 34% | 68% | 56% | 52% | 72% | 93% |
| Size | 9.3 GB | 181 MB | 9.3 GB | 9.3 GB | 181 MB | 181 MB |
| Lefties? | No | No | No | Yes | Yes | N/A |

Table 2. Low-order MNIST digit recognition dataset runtimes and classification accuracies (after random sampling to training set size = 5000, test set size = 5000. *k*=2 in all cases)

| | Voxels | Wavelets | SVD | PARAFAC | TWave |
|----------|---------|----------|---------------|-----------------|---------|
| Runtime | 250 sec | 422 sec | 20 min | 25.3 min | 512 sec |
| Accuracy | 47% | 88% | 53% | 53% | 88% |

5 Conclusions

From these results, we may conclude that the combination of wavelets and tensor tools in the analysis of fMRI motor task datasets yields better performance in space,

time, and accuracy than the voxelwise approach or either technique alone, achieving benefits such as sensitivity to locality while avoiding the prohibitive space and time costs of using only tensors. Additionally, such an approach provides powerful automatic data summarization techniques, as demonstrated through discovery of left-handed subjects in our dataset. Potential avenues for future research include use of different wavelet functions, extension of our methods to streaming and sparse tensor data, and applications to high-order datasets in other fields.

Acknowledgments. We would like to thank Charalampos Tsourakakis for assistance with the literature review and experiments. This work was supported in part by the National Science Foundation under grants IIS-0237921, IIS-0705215, IIS-0705359, and the National Institutes of Health under grant R01 MH68066-05 funded by the National Institute of Mental Health, the National Institute of Neurological Disorders and Stroke, and the National Institute on Aging. All agencies specifically disclaim responsibility for any analyses, interpretations, or conclusions.

References

1. Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 391–407 (1999)
2. Harshman, R.: Foundations of the PARAFAC Procedure: Models and Conditions for an Explanatory Multimodal Factor Analysis. In: *UCLA Working Papers in Phonetics*, pp. 1–84 (1970)
3. Carroll, J.D., Chang, J.: Analysis of Individual Differences in Multidimensional Scaling via an n-way Generalization of ‘Eckart-Young’ Decomposition. *Psychometrika*, 283–319 (1970)
4. Sands, R., Young, F.W.: Component Models for Three-way Data: An Alternating Least Squares Algorithm with Optimal Scaling Features. *Psychometrika*, 39–67 (1980)
5. Sheikholeslami, G., Chatterjee, S., Zhang, A.: WaveCluster: a wavelet-based clustering approach for spatial data in very large databases. *The International Journal on Very Large Data Bases*, 289–304 (2000)
6. LeCun, Y., Cortes, C.: The MNIST Database of Handwritten Digits, <http://yann.lecun.com/exdb/mnist>
7. Lu, H., Plataniotis, K.N., Venetsanopoulos, A.N.: MPCA: Multilinear Principal Component Analysis of Tensor Objects. *IEEE Transactions on Neural Networks* 19(1), 18–39 (2008)

Spatial Clustering with Obstacles Constraints by Dynamic Piecewise-Mapped and Nonlinear Inertia Weights PSO

Xueping Zhang¹, Haohua Du², and Jiayao Wang^{1,3}

¹ School of Information Science and Engineering, Henan University of Technology,
Zhengzhou 450052, China

² School of Computer Science and Engineering, Beihang University, Beijing 100191, China

³ School of Surveying and Mapping, PLA Information Engineering University,
Zhengzhou 450052, China

zhang_xpcn@yahoo.com.cn

Abstract. Spatial clustering with constraints has been a new topic in spatial data mining. A novel Spatial Clustering with Obstacles Constraints (SCOC) by dynamic piecewise-mapped and nonlinear inertia weights particle swarm optimization is proposed in this paper. The experiments show that the algorithm can not only give attention to higher local constringency speed and stronger global optimum search, but also get down to the obstacles constraints and practicalities of spatial clustering; and it performs better than PSO K-Medoids SCOC in terms of quantization error and has higher constringency speed than Genetic K-Medoids SCOC.

Keywords: Spatial Clustering with Obstacles Constraints, Particle Swarm Optimization, Dynamic Piecewise Linear Chaotic Map, Dynamic Nonlinear Inertia Weights.

1 Introduction

Spatial clustering with constraints has been a new topic in spatial data mining. Spatial clustering with constraints has two kinds of forms [1]. One kind is Spatial Clustering with Obstacles Constraints (SCOC), such as bridge, river, and highway etc. whose impact on the result should be considered in the clustering process of large spatial data. Ignoring the constraints leads to incorrect interpretation of the correlation among data points. The other kind is spatial clustering with handling operational constraints, it consider some operation limiting conditions in the clustering process. In this paper, we mainly discuss SCOC.

Since K.H.Tung put forward a clustering question COE (Clustering with Obstacles Entities) [2] in 2001, a new studying direction in the field of clustering research have been opened up. To the best of our knowledge, only four clustering algorithms for clustering spatial data with obstacles constraints have been proposed very recently: COD-CLARANS [2] based on the Partitioning approach of CLARANS,

AUTOCLUST+ [3] based on the Graph partitioning method of AUTOCLUST, DBCluC [4-5] based on the Density-based algorithm, and GKSCOC [6] based on Genetic algorithms (GAs) and Partitioning approach of K-Medoids. Although these algorithms can deal with some obstacles in the clustering process, many questions exist in them [6].

PKSCOC based on Particle Swarm Optimization (PSO) and K-Medoids is presented [7] by us. However, the performance of simple PSO depends on its parameters, it often getting into local optimum and fails to converge to global optimum. A lot of improved methods were presented by many scholars, e.g. the paper [8-9] presented the Quantum PSO algorithm, and the paper [10-12] presented the Chaotic PSO algorithm. Recently, Dynamic Piecewise-mapped and Nonlinear Inertia Weights PSO (PNPSO) is proposed in [13]. Experiments and comparisons demonstrated that PNPSO outperformed several other well-known improved PSO algorithms on many famous benchmark problems in all cases.

This article developed a novel spatial clustering with obstacles constraints by PNPSO to cluster spatial data with obstacles constraints, which called PNPKSCOC. The contrastive experiments show that PNPKSCOC is better than PKSCOC in terms of quantization error and has higher constringency speed than GKSCOC.

The remainder of the paper is organized as follows. Section 2 introduces PNPSO algorithm. Section 3 presents PNPKSCOC. The performances of PNPKSCOC are showed in Section 4, and Section 5 concludes the paper.

2 Dynamic Piecewise-Mapped and Nonlinear Inertia Weights PSO

2.1 Classical PSO

PSO is a population-based optimization method first proposed by Kennedy and Eberhart. The mathematic description of PSO is as the following. Suppose the dimension of the searching space is D , the number of the particles is n . Vector $X_i = (x_{i1}, x_{i2}, \dots, x_{iD})$ represents the position of the i^{th} particle and $pbest_i = (p_{i1}, p_{i2}, \dots, p_{iD})$ is its best position searched by now, and the whole particle swarm's best position is represented as $gbest = (g_1, g_2, \dots, g_D)$. Vector $V_i = (v_{i1}, v_{i2}, \dots, v_{iD})$ is the position change rate of the i^{th} particle. Each particle updates its position according to the following formulas:

$$v_{id}(t+1) = wv_{id}(t) + c_1 \text{rand}() [p_{id}(t) - x_{id}(t)] + c_2 \text{rand}() [g_d(t) - x_{id}(t)] \quad (1)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1), \quad 1 \leq i \leq n, \quad 1 \leq d \leq D \quad (2)$$

where c_1 and c_2 are positive constant parameters, $\text{Rand}()$ is a random function with the range $[0, 1]$, and w is the inertia weight. Eq.1 is used to calculate the particle's new velocity, the particle flies toward a new position according to Eq.2.

2.2 Coordinate PSO with Dynamic Piecewise Linear Chaotic Map and Dynamic Nonlinear Inertia Weights

2.2.1 Dynamic Piecewise Linear Chaotic Map

The well-known Piecewise linear chaotic map is defined as follows [10]:

$$x_{r+1} = \begin{cases} x_r / p_c, & x_r \in (0, p_c) \\ (1 - x_r) / (1 - p_c), & x_r \in (p_c, 1) \end{cases} \quad (3)$$

where p_c is the control parameter and X is a variable. Although Eq.3 is deterministic, it exhibits chaotic dynamics in $(0, 1)$ when $p_c \in (0, 0.5) \cup (0.5, 1)$. The newly introduced dynamic Piecewise linear chaotic map [14] is incorporated into the PSO inertia weight which is described in equations (4) and (5).

$$\alpha = \alpha_{\max} - (\alpha_{\max} - \alpha_{\min}) \left(\frac{iter}{iter_{\max}} \right) \quad (4)$$

$$w = \alpha + (1 - \alpha)Pmap \quad (5)$$

where α is the dynamic chaotic inertia weight adjustment factor, α_{\max} and α_{\min} represent the maximum and minimum values of α respectively, $iter$ is the current iteration number, $iter_{\max}$ is the maximum iteration number, and $Pmap$ is the result of Piecewise linear chaotic map.

2.2.2 Dynamic Nonlinear Equations

To achieve trade-off between exploration and exploitation, two types of dynamic nonlinear inertia weights are introduced [13]. In this paper, the first type is proposed in equations (6) and (7):

$$dnl = dnl_{\min} + (dnl_{\max} - dnl_{\min}) \left(\frac{iter}{iter_{\max}} \right) \quad (6)$$

$$w = w_{\min} + (w_{\max} - w_{\min}) \left(\frac{iter_{\max} - iter}{iter_{\max}} \right)^{dnl} \quad (7)$$

where dnl represents the dynamic nonlinear factor, w represents the inertia weight, w_{\max} and w_{\min} represent the maximum and minimum value of w respectively, dnl_{\max} and dnl_{\min} represent the maximum and minimum value of dnl respectively, $iter$ represents the current iteration number, and $iter_{\max}$ represents the maximum iteration number.

2.2.3 Parallel Inertia Weight Adjustment

To avoid the premature convergence problem and to achieve the balance between global exploration and local exploitation, dynamic Piecewise linear chaotic map and dynamic nonlinear equations are used in parallel to dynamically adjust PSO inertia weight w , which is described as follows[13]:

Initialize all the parameters.

repeat

 Evaluate the fitness values of all the particles

 if $f_i > f_{avg}$

 Equations (4), (5), (1) and (2) are employed

 Elseif $f_i \leq f_{avg}$

 Equations (6), (7), (1) and (2) are employed

 endif

until (a termination criterion is met)

where f_i is the fitness value of particle i and f_{avg} is the average fitness value of the whole population.

3 Spatial Clustering with Obstacles Constraints by PNPSO

3.1 Motivating Concepts

To derive a more efficient algorithm for SCOC, the following definitions are first introduced.

Definition 1 (Visibility graph). Given a set of m obstacle, $O = (o_1, o_2, \dots, o_m)$, the visibility graph is a graph $VG = (V, E)$ such that each vertex of the obstacles has a corresponding node in V , and two nodes v_1 and v_2 in V are joined by an edge in E if and only if the corresponding vertices they represent are visible to each other.

To generate VG , we use VPIA (VGRAPH Point Incorporation Algorithm) as presented in [14].

Definition 2 (Obstructed distance). Given point p and point q , the obstructed distance $d_o(p, q)$ is defined as the length of the shortest Euclidean path between two points p and q without cutting through any obstacles.

We can use Dijkstra Algorithm to compute obstructed distance. The simulation result is in Fig.1 and the red solid line represents the obstructed distance we got.

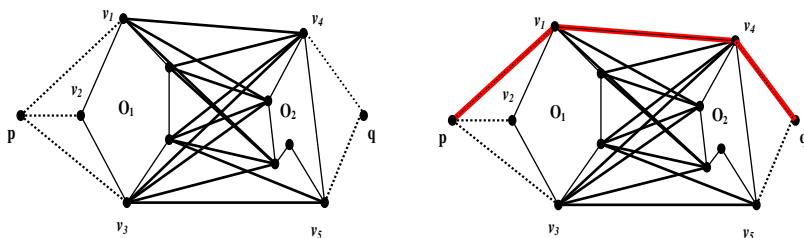


Fig. 1. Visibility graph and Obstructed distance

3.2 Spatial Clustering with Obstacles Constraints by Improved K-Medoids

K-Medoids algorithm is adopted for SCOC to avoid cluster center falling on the obstacle. Square-error function is adopted to estimate the clustering quality, and its definition can be defined as:

$$E = \sum_{j=1}^{N_c} \sum_{p \in C_j} (d(p, m_j))^2 \tag{8}$$

where N_c is the number of cluster C_j , m_j is the cluster centre of cluster C_j , $d(p, q)$ is the direct Euclidean distance between the two points p and q .

To handle obstacle constraints, accordingly, criterion function for estimating the quality of spatial clustering with obstacles constraints can be revised as:

$$E_o = \sum_{j=1}^{N_c} \sum_{p \in C_j} (d_o(p, m_j))^2 \tag{9}$$

where $d_o(p, q)$ is the obstructed distance between point p and point q .

The method of IKSCOC is adopted as follows [4].

1. Select N_c objects to be cluster centers at random;
2. Assign remaining objects to nearest cluster center;
3. Calculate E_o according to Eq.9;
4. While (E_o changed) do {Let current $E = E_o$;
5. Select a not centering point to replace the cluster center m_j randomly;
6. Assign objects to the nearest center;
7. Calculate E according to Eq.8;
8. If $E >$ current E , go to 5;
9. Calculate E_o ;
10. If $E_o <$ current E , form new cluster centers }.

While IKSCOC still inherits two shortcomings, one is selecting initial value randomly may cause different results of the spatial clustering and even have no solution, the other is that it only gives attention to local constringency and is sensitive to an outlier.

3.3 Spatial Clustering with Obstacles Constraints Based on PNPSO and Improved K-Medoids

In the context of clustering, a single particle represents the N_c cluster centroid. That is, each particle X_i is constructed as follows:

$$X_i = (m_{i1}, \dots, m_{ij}, \dots, m_{iN_c}) \tag{10}$$

where m_{ij} refers to the j^{th} cluster centroid of the i^{th} particle in cluster C_{ij} . Here, the objective function is defined as follows:

$$f(x_i) = \frac{1}{J_i} \quad (11)$$

$$J_i = \sum_{j=1}^{N_c} \sum_{p \in C_{ij}} d_o(p, m_j) \quad (12)$$

The PNPkSCOC is developed as follows.

1. Execute the IKSCOC algorithm to initialize one particle to contain N_c selected cluster centroids;
2. Initialize the other particles of the swarm to contain N_c selected cluster centroids at random;
3. For $t = 1$ to t_{\max} do {
 4. For $i = 1$ to no_of_particles do {
 5. For each object p do {
 6. Calculate $d_o(p, m_j)$;
 7. Assign object p to cluster C_{ij} such that $d_o(p, m_j) = \min_{C=1, \dots, N_c} \{d_o(p, m_{ic})\}$;
 8. Evaluate fitness of particle according to Eq.11;
 9. if $f_i > f_{avg}$ Update particles using equations (4), (5), (1) and (2);
 10. Elseif $f_i \leq f_{avg}$ Update particles using equations (6), (7), (1) and (2) ;
 11. Update $Pbest$;
 12. Update $Pgbest$;
 13. If $\|v\| \leq \varepsilon$, terminate }
 14. Select two other particles j and k ($i \neq j \neq k$) randomly;
 15. Optimize new individuals using IKSCOC}
 16. Output.

where t_{\max} is the maximum number of iteration for PNPkSCOC. STEP 16 is to improve the local constringency speed of PNPkSCOC.

4 Results and Discussion

We have made experiments separately by K-Medoids, IKSCOC, GKSCOC, PKSCOC, and PNPkSCOC. $n = 50$, $c_1 = c_2 = 2$, $t_{\max} = 100$. Fig.2 shows the results on real Dataset. Fig.2 (a) shows the original data with river obstacles. Fig.2 (b) shows the results of 4 clusters found by K-Medoids without considering obstacles constraints. Fig.2(c) shows 4 clusters found by IKSCOC. Fig.2(d) shows 4 clusters found by GKSCOC. Fig.2 (e) shows 4 clusters found by PNPkSCOC. Obviously, the results of the clustering illustrated in Fig.2(c), Fig.2 (d), and Fig.2(e) have better practicalities than that in Fig.2 (b), and the ones in Fig.2 (e) and Fig.2 (d) are both superior to the one in Fig.2(c). So, it can be drawn that PNPkSCOC is effective and has better practicalities.

Fig.3 is the value of J showed in every experiment on Dataset1 by IKSCOC, PKSCOC, and PNPKSCOC respectively. It is showed that IKSCOC is sensitive to initial value and it constringes in different extremely local optimum points by starting at different initial value while PNPKSCOC constringes nearly in the same optimum point at each time, and PNPKSCOC is better than PKSCOC.

Fig.4 is the constringency speed in one experiment on Dataset1. It is showed that PNPKSCOC constringes in about 12 generations while GKSCOC constringes in nearly 25 generations. So, it can be drawn that PNPKSCOC is effective and has higher constringency speed than GKSCOC.

Therefore, we can draw the conclusion that PNPKSCOC has stronger global constringent ability than PKSCOC and has higher convergence speed than GKSCOC.

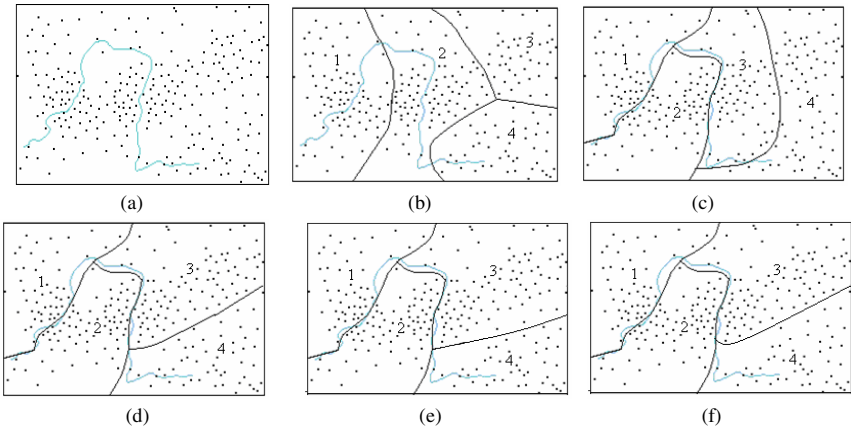


Fig. 2. Clustering Dataset

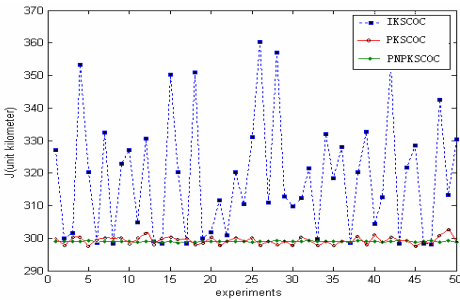


Fig. 3. PNPKSCOC vs. IKSCOC, PKSCOC

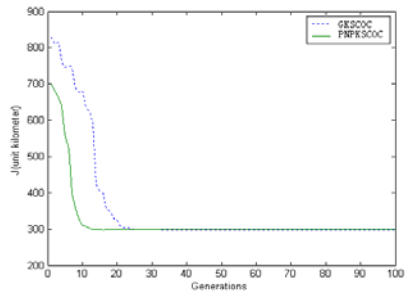


Fig. 4. PNPKSCOC vs. GKSCOC

5 Conclusions

In this paper, we developed a novel spatial clustering with obstacles constraints by dynamic piecewise-mapped and nonlinear inertia weights particle swarm optimization to cluster spatial data with obstacles constraints. The proposed method is also

compared with some other algorithms to demonstrate its efficiency and the experimental results are satisfied.

Acknowledgments. This work is partially supported by Program for New Century Excellent Talents in University (NCET-08-0660), the Supporting Plan of Science and Technology Innovation Talent of Colleges in Henna Province (Number: 2008HASTIT012), and the Opening Research Fund of Key Laboratory of Spatial Data Mining & Information Sharing of Ministry of Education (Number: 200807).

References

1. Tung, A.K.H., Han, J., Lakshmanan, L.V.S., Ng, R.T.: Constraint-Based Clustering in Large Databases. In: Van den Bussche, J., Vianu, V. (eds.) ICDT 2001. LNCS, vol. 1973, pp. 405–419. Springer, Heidelberg (2000)
2. Tung, A.K.H., Hou, J., Han, J.: Spatial Clustering in the Presence of Obstacles. In: Proceedings of International Conference on Data Engineering (ICDE 2001), Heidelberg, Germany, pp. 359–367 (2001)
3. Estivill-Castro, V., Lee, I.J.: AUTOCLUST+: Automatic Clustering of Point-Data Sets in the Presence of Obstacles. In: Proceedings of the International Workshop on Temporal, Spatial and Spatial-Temporal Data Mining, Lyon, France, pp. 133–146 (2000)
4. Zaïane, O.R., Lee, C.H.: Clustering Spatial Data When Facing Physical Constraints. In: Proceedings of the IEEE International Conference on Data Mining (ICDM 2002), Maebashi City, Japan, pp. 737–740 (2002)
5. Wang, X., Hamilton, H.J.: Gen and SynGeoDataGen Data Generators for Obstacle Facilitator Constrained Clustering (2004)
<http://Ftp.cs.uregina.ca/Research/Techreports/2004-08.pdf>
6. Zhang, X., Wang, J., Wu, F., Fan, Z., Li, X.: A Novel Spatial Clustering with Obstacles Constraints Based on Genetic Algorithms and K-Medoids. In: Proceedings of the Sixth International Conference on Intelligent Systems Design and Applications (ISDA 2006), Jinan Shandong, China, pp. 605–610 (2006)
7. Zhang, X., Wang, J.: A Novel Spatial Clustering with Obstacles Constraints Based on Particle Swarm Optimization and K-Medoids. In: Zhou, Z.-H., Li, H., Yang, Q. (eds.) PAKDD 2007. LNCS (LNAI), vol. 4426, pp. 1105–1113. Springer, Heidelberg (2007)
8. Sun, J., Feng, B., Xu, W.: Particle Swarm Optimization with particles having Quantum Behavior. In: Proceedings of Congress on Evolutionary Computation, Portland (OR, USA), pp. 325–331 (2004)
9. Mikki, S.M., Kishk, A.A.: Quantum Particle Swarm Optimization for Electromagnetics. IEEE transactions on antennas and propagation 54(10), 2764–2775 (2006)
10. Baranovsky, A., Daems, D.: Design of one-dimensional chaotic maps with prescribed statistical properties. International Journal of Bifurcation and Chaos 5(6), 1585–1598 (1995)
11. Liu, B., Wang, L., Jin, Y.-H., Tang, F., Huang, D.-X.: Improved particle swarm optimization combined with chaos. Chaos, Solitons and Fractals 25, 1261–1271 (2005)
12. Xiang, T., Liao, X.F., Wong, K.W.: An improved particle swarm optimization algorithm combined with piecewise linear chaotic map. Appl. math. and Compute., 1637–1645 (2007)
13. Liu, H., Su, R., Gao, Y.: Coordinate Particle Swarm Optimization with Dynamic Piecewise-mapped and Nonlinear Inertia Weights. In: Proceedings of the 2009 International Conference on Artificial Intelligence and Computational Intelligence (AICI 2009), Shanghai, China, pp. 124–128 (2009)
14. Tokuta, A.: Extending the VGRAPH Algorithm for Robot Path Planning,
http://wscg.zcu.cz/wscg98/papers98/Tokuta_98.pdf

An Efficient GA-Based Algorithm for Mining Negative Sequential Patterns

Zhigang Zheng¹, Yanchang Zhao^{1,2}, Ziyue Zuo¹, and Longbing Cao¹

¹ Data Sciences & Knowledge Discovery Research Lab
Centre for Quantum Computation and Intelligent Systems
Faculty of Engineering & IT, University of Technology, Sydney, Australia
{zgzheng, zzuo, lbcao}@it.uts.edu.au
² Centrelink, Australia
yanchang.zhao@centrelink.gov.au

Abstract. Negative sequential pattern mining has attracted increasing concerns in recent data mining research because it considers negative relationships between itemsets, which are ignored by positive sequential pattern mining. However, the search space for mining negative patterns is much bigger than that for positive ones. When the support threshold is low, in particular, there will be huge amounts of negative candidates. This paper proposes a Genetic Algorithm (GA) based algorithm to find negative sequential patterns with novel crossover and mutation operations, which are efficient at passing good genes on to next generations without generating candidates. An effective dynamic fitness function and a pruning method are also provided to improve performance. The results of extensive experiments show that the proposed method can find negative patterns efficiently and has remarkable performance compared with some other algorithms of negative pattern mining.

Keywords: Negative Sequential Pattern, Genetic Algorithm, Sequence Mining, Data Mining.

1 Introduction

The concept of discovering sequential patterns was firstly introduced in 1995 [1], and aimed at discovering frequent subsequences as patterns in a sequence database, given a user-specified minimum support threshold. Some popular algorithms in sequential pattern mining include AprioriAll [1], Generalized Sequential Patterns (GSP) [10] and PrefixSpan [8]. GSP and AprioriAll are both Apriori-like methods based on breadth-first search, while PrefixSpan is based on depth-first search. Some other methods, such as SPADE (Sequential Pattern Discovery using Equivalence classes) [12] and SPAM (Sequential Pattern Mining) [4], are also widely used in researches.

In contrast to traditional positive sequential patterns, negative sequential patterns focus on negative relationships between itemsets, in which, absent items are taken into consideration. We give a simple example to illustrate the difference: suppose $p_1 = \langle a b c d \rangle$ is a positive sequential pattern; $p_2 = \langle a b \neg c e \rangle$ is a negative sequential pattern; and each item, a , b , c , d and e , stands for a claim item code in the customer claim database

of an insurance company. By getting the pattern p_1 , we can tell that an insurant usually claims for a , b , c and d in a row. However, only with the pattern p_2 , we are able to find that given an insurant claim for item a and b , if he/she does NOT claim c , then he/she would claim item e instead of d . This kind of patterns cannot be described or discovered by positive sequential pattern mining.

However, in trying to utilize traditional frequent pattern mining algorithms for mining negative patterns, two problems stand in the way. (1) Huge amounts of negative candidates will be generated by classic breath-first search methods. For example, given 10 distinct positive frequent items, there are only 1,000 ($=10^3$) 3-item positive candidates, but there will be 8,000 ($=20^3$) 3-item negative candidates because we should count 10 negative items in it. (2) Take a 3-item data sequence $\langle a b c \rangle$, it can only support candidates $\langle a \rangle$, $\langle b \rangle$, $\langle c \rangle$, $\langle a b \rangle$, $\langle a c \rangle$, $\langle b c \rangle$ and $\langle a b c \rangle$. But in the negative case, data sequence $\langle a b c \rangle$ not only supports the positive candidates as the above, but also can match a large bunch of negative candidates, such as $\langle a \neg a \rangle$, $\langle b \neg a \rangle$, $\langle b \neg b \rangle$, $\langle a \neg a c \rangle$, $\langle a \neg c c \rangle$ etc. There are thus still huge amounts of negative candidates even after effective pruning.

Based on Genetic Algorithm (GA) [5], we propose a new method for mining negative patterns. GA is an evolution method, which simulates biological evolution. A generation pass good genes on to a new generation by crossover and mutation, and the populations become better and better after many generations. We borrow the ideas of GA to focus on the space with good genes, because this always finds more frequent patterns first, resulting in good genes. It is therefore more effective than methods which treat all candidates equally, especially when a very low support threshold is set. It is equally possible to find long negative patterns at the beginning stage of process.

Our contributions are:

- A GA-based algorithm is proposed to find negative sequential patterns efficiently. It obtains new generations by crossover and mutation operations without generating candidates, and uses dynamic fitness to control population evolution. A pruning method is also provided to improve performance.
- Extensive experimental results on 3 synthetic datasets and a real-world dataset show that our algorithm has better performance compared with PNSP [11] and Neg-GSP [14] especially when the support threshold min_sup is very low.

This paper is organized as follows. Section 2 talks about related work. Section 3 briefly introduces negative sequential patterns and presents formal descriptions of them. Our GA-based algorithm is then described in Section 4. Section 5 shows experimental results on some datasets. The paper is concluded in the last section.

2 Related Work

Most research on sequential patterns has focused on positive relationships. In recent years, some research has started to focus on negative sequential pattern mining.

Zhao et al. [13] proposed a method to find negative sequential rules based on SPAM [4]. However the rules are limited to formats such as $\langle A \Rightarrow \neg B \rangle$, $\langle \neg A \Rightarrow B \rangle$, $\langle \neg A \Rightarrow \neg B \rangle$. Ouyang & Huang [7] extended traditional sequential pattern definition

(A, B) to include negative elements such as $(\neg A, B)$, $(A, \neg B)$ and $(\neg A, \neg B)$. They put forward an algorithm which finds both frequent and infrequent sequences and then obtains negative sequential patterns from infrequent sequences. Nancy et al. [6] designed an algorithm PNSPM and applied the Apriori principle to prune redundant candidates. They extracted meaningful negative sequences using the interestingness measure; nevertheless the above works defined some limited negative sequential patterns, which are not general enough. Sue-Chen et al. [11] presented more general definitions of negative sequential patterns and proposed an algorithm called PNSP, which extended GSP to deal with mining negative patterns, but they generated negative candidates in the appending step, which then may produce a lot of unnecessary candidates.

Some existing researches have used GA for mining the negative association rule and positive sequential pattern. Bilal and Erhan [2] proposed a method using GA to mine negative quantitative association rules. They generated uniform initial population, and used an adaptive mutation probability and an adjusted fitness function. [9] designed a GA to mine generalized sequential patterns, but it is based on SQL expressions. It is an instructive work since there are few research works using GA for negative sequential pattern mining.

3 Problem Statement

3.1 Definitions

A *sequence* s is an ordered list of elements, $s = \langle e_1 e_2 \dots e_n \rangle$, where each e_i , $1 \leq i \leq n$, is an element. An *element* e_i ($1 \leq i \leq k$) consists of one or more items. For example, sequence $\langle a b (c, d) f \rangle$ consists of 4 elements and (c, d) is an element which includes two items. The *length* of a sequence is the number of items in the sequence. A sequence with k items is called a *k-sequence* or *k-item sequence*.

Sequence is a general concept. We extend sequence definition to positive/negative sequence. A sequence $s = \langle e_1 e_2 \dots e_n \rangle$ is a *positive sequence*, when each element e_i ($1 \leq i \leq n$) is a positive element. A sequence $s = \langle e_1 e_2 \dots e_n \rangle$ is a *negative sequence*, when $\exists i$, e_i ($1 \leq i \leq n$) is a negative element, which represents the absence of an element. For example, $\neg c$ and $\neg(c, d)$ are negative elements, so $\langle a b \neg c f \rangle$ and $\langle a b \neg(c, d) f \rangle$ are both negative sequences.

A sequence $s_r = \langle e_{r_1} e_{r_2} \dots e_{r_m} \rangle$ is a *subsequence* of another sequence $s_p = \langle e_{p_1} e_{p_2} \dots e_{p_n} \rangle$, if there exists $1 \leq i_1 \leq i_2 \leq \dots \leq i_k \leq p_n$, $e_{r_1} \subseteq e_{p_{i_1}}$, $e_{r_2} \subseteq e_{p_{i_2}}$, ..., $e_{r_k} \subseteq e_{p_{i_k}}$.

A sequence s_r is a *maximum positive subsequence* of another sequence s_p , if s_r is a subsequence of s_p , and s_r includes all positive elements of s_p . For example, $\langle a b f \rangle$ is maximum positive subsequence of $\langle a b \neg c f \rangle$ and $\langle a b \neg(c, d) f \rangle$.

Definition 1: Negative Sequential Pattern. If the support value of a negative sequence is greater than the pre-defined support threshold min_sup , and it also meets the following constraints, then we call it a negative sequential pattern.

1) Items in a single element should be all positive or all negative. The reason is that a positive item and negative item in the same element are unmeaning. For example, $\langle a (a, \neg b) c \rangle$ is not allowed since item a and item $\neg b$ are in the same element.

2) Two or more continuous negative elements are not accepted in a negative sequence. This constraint is also used by other researchers [11].

3) For each negative item in a negative pattern, its positive item is required to be frequent. For example, if $\langle -c \rangle$ is a negative item, its positive item $\langle c \rangle$ is required to be frequent. It is helpful for us to focus on the frequent items.

In order to calculate the support value of a negative sequence against the data sequences in a database, we need to clarify the sequence matching method and criteria. In other words, we should describe what kinds of sequence a data sequence can support.

Definition 2: Negative Matching. A negative sequence $s_n = \langle e_1 e_2 \dots e_k \rangle$ matches a data sequence $s = \langle d_1 d_2 \dots d_m \rangle$, iff:

1) s contains the max positive subsequence of s_n

2) for each negative element $e_i (1 \leq i \leq k)$, there exist integers $p, q, r (1 \leq p \leq q \leq r \leq m)$ such that: $\exists e_{i-1} \subseteq d_p \wedge e_{i+1} \subseteq d_r$, and for $\forall d_q, e_i \not\subseteq d_q$

For example, see Table 1, $s_n = \langle b \neg c a \rangle$ matches $\langle b d a c \rangle$, but does not match $\langle b d c a \rangle$, since the negative element c appears between the element b and a .

Table 1. Pattern matching

| Pattern | match | Sequence |
|------------------------------|-------|---------------------------|
| $\langle b \neg c a \rangle$ | √ | $\langle b d a \rangle$ |
| $\langle b \neg c a \rangle$ | √ | $\langle b d a c \rangle$ |
| $\langle b \neg c a \rangle$ | × | $\langle b d c a \rangle$ |

Table 2. Encoding

| Sequence | Chromosome | | |
|---|--------------------------|--------------------------|--------------------------|
| | <i>gene</i> ₁ | <i>gene</i> ₂ | <i>gene</i> ₃ |
| $\langle a b \neg(c,d) \rangle \Rightarrow$ | +a | +b | -(c,d) |

3.2 Ideas of GA-Based Method

As introduced in Section 1, negative sequential pattern mining may encounter huge amounts of negative candidates even after effective pruning. It will take a long time to pass over the dataset many times to get the candidates' support.

Based on GA, we obtain negative sequential patterns by crossover and mutation, without generating candidates; high frequent patterns are then selected to be parents to generate offspring. It will pass the best genes on to the next generations and will always focus on the space with good genes. By going through many generations, it will obtain a new and relatively high-quality population.

A key issue is how to find all the negative patterns since the GA-based method cannot ensure locating all of them. We therefore use an incremental population, and add all negative patterns, which are generated by crossover and mutation during the evolution process, into population. A dynamic fitness function is proposed to control population evolution. Ultimately, we can secure almost all the frequent patterns. The proportion can reach 90% to 100% in our experiments on two synthetic datasets.

4 GA-Based Negative Sequential Pattern Mining Algorithm

The general idea of the algorithm is shown as Fig. 1. We will describe the algorithm from how to encode a sequence, and then introduce population, selection, crossover, mutation, pruning, fitness function and so on. A detailed algorithm will then be introduced.

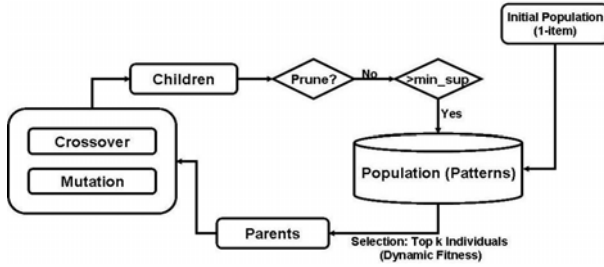


Fig. 1. Algorithm Flow

4.1 Encoding

Sequence is mapped into the chromosome code in GA. Both crossover and mutation operations depend on the chromosome code. We need to define the chromosome to represent the problems of negative sequential pattern mining exactly. There are many different methods to encoding the chromosome, such as binary encoding, permutation encoding, value encoding and tree encoding [5]. The permutation encoding method is suitable for ordering problem and its format is consistent with the format of the sequence data, so we use it for sequence encoding.

Each sequence is mapped into a chromosome. Each element of the sequence is mapped into a gene in the chromosome, no matter whether the element has one item or more. Given a sequence $\langle e_1 e_2 \dots e_n \rangle$, it is transformed to a chromosome which has n -genes. Each gene is composed of a tag and an element. The element includes one or more items, and the tag indicates that the element is positive or negative. For example, a negative sequence $\langle a b \neg(c,d) \rangle$ is mapped into a 3-gene chromosome, see Table 2.

4.2 Population and Selection

In the classical GA method, the number of populations is fixed [5]. We using a fixed number of populations to produce the next generation, but the populations tended to contract into one high frequent pattern, and we can only obtain a small part of frequent patterns. To achieve as many sequential patterns as possible, we potentially needed a population to cover more individuals. We therefore adjusted the basic GA to suit negative sequential pattern mining in the following ways.

Initial Population. All 1-item frequent positive patterns are obtained first. Based on the 1-item positive patterns, we transform all of them to their corresponding 1-item negative sequences, such as transforming the frequent positive sequence $\langle e \rangle$ to the negative sequence $\langle \neg e \rangle$. We then take all positive and negative 1-item patterns as the initial population.

Population Increase. We do not limit population to a fixed number. When we acquire new sequential patterns during the evolvement, new patterns are put into the population for the next selection. If the population has already included the patterns, we ignore them. To improve the performance of this process, a hash table is used to verify whether a pattern has already appeared in the population.

Selection. The commonly used selection method is roulette wheel selection [3]. We have an increased population and the population number depends on the count of sequential patterns; thus, we can not use roulette wheel selection because the selection will be too costly if the population number is huge. We select the top K individuals with high dynamic fitness (see Section 4.5), where K is a constant number showing how many individuals will be selected for the next generation. To improve the performance of this selection method, we sort all individuals in population in descending order by dynamic fitness value. In every generation, we only select the first K individuals.

4.3 Crossover and Mutation

Crossover. Parents with different lengths are allowed to crossover with each other, and crossover may happen at different positions to get sequential patterns with varied lengths. For example, a crossover takes place at a different position, which is shown by '↑↓' in Table 3. After crossover, it may acquire two children. $Child1 \langle b \neg c e \rangle$ consists of the first part of $parent1$ and the second part of $parent2$. $Child2 \langle d a \rangle$ consists of the second part of $parent1$ and the first part of $parent2$. So we get two children with different lengths. If a crossover takes place both at the end/head of $parent1$ and at the head/end of $parent2$, as Table 4 shows, $child2$ will be empty. In that case, we shall set $child2$ by reverse. A *Crossover Rate* is also used to control the probability of crossover when parents generate their children.

Table 3. Crossover

| | | | | |
|-----------|-------------------------|---------------|----------|--------------|
| $parent1$ | $b \neg c \downarrow a$ | \Rightarrow | $child1$ | $b \neg c e$ |
| $parent2$ | $d \uparrow e$ | \Rightarrow | $child2$ | $d a$ |

Table 4. Crossover at head/end

| | | | | |
|-----------|-------------------------|---------------|----------|------------------|
| $parent1$ | $b \neg c a \downarrow$ | \Rightarrow | $child1$ | $b \neg c a d e$ |
| $parent2$ | $\uparrow d e$ | \Rightarrow | $child2$ | $d e b \neg c a$ |

Mutation. Mutation is helpful in avoiding contraction of the population to a special frequent pattern. To introduce mutation into sequence generation, we select a random position and then replace all genes after that position with 1-item patterns. For example, given an individual $\langle b \neg c a \rangle$, after mutation, it may change to $\langle b d \neg e \rangle$ if $\langle d \rangle$ and $\langle \neg e \rangle$ are 1-item patterns. *Mutation Rate* is a percentage to indicate the probability of mutation when parents generate their children.

4.4 Pruning

When a new generation is obtained after crossover and mutation, it is necessary to verify whether the new generation is valid in terms of the constraints for negative sequential patterns before passing over the whole dataset for their supports.

For a new individual $c = \langle e_1 e_2 e_3 \dots e_n \rangle$, $c' = \langle e_i e_j \dots e_k \rangle$ ($0 < i \leq j \leq k \leq n$) is the max positive subsequence of c , that is to say, e_i, e_j, \dots and e_k are all positive elements, and other elements in c are negative. If c' is not frequent, c must be infrequent and should be pruned. This method is simple but effective for pruning invalid candidates without cutting off possible valid individuals by mistake.

4.5 Fitness Function

In order to evaluate the individuals and decide which are the best for the next generation, a fitness function for individuals is implemented in GA. We use the fitness function

shown in Eqn.(1):

$$ind.fitness = (ind.support - min_sup) \times DatasetSize. \quad (1)$$

Fitness. The fitness function is composed of two parts. *Support* is the percentage that indicates how many proportion records are matched by the individual. If support is high, fitness will be high, so that the individual has good characteristics to pass down to next generation. *min_sup* is a threshold percentage value for verifying whether a sequence is frequent. *Dataset size* is the record count of whole dataset.

Dynamic Fitness. Because the characteristics of the individual have been transmitted to the next generation by crossover or mutation, the individual should exit after a few generations. The result will tend to contract to one point if the individual doesn't exit gradually. We therefore set a dynamic fitness *dfitness* to every individual in the population, shown in Eqn.(2). Its initial value is equal to *fitness*, but decreases during the evolvement. It indicates that the individuals in the population will gradually ceased to evolve. It is like a life value. When an individual's dynamic fitness is low or close to 0 (<0.01), we set it to 0 because we regard it as a wasted individual which cannot be selected for the next generation.

$$ind.dfiness = \begin{cases} ind.fitness, & \text{initial set} \\ ind.dfiness \times (1 - DecayRate), & \text{if } ind \text{ is selected} \end{cases} \quad (2)$$

Decay Rate. We set a decay rate to indicate the decrease speed of individual's fitness. The decay rate is a percentage value between 0% and 100%. If an individual is selected by the selection process, its dynamic fitness will decrease by the speed of the decay rate. If the decay rate is high, dynamic fitness will decrease quickly and individuals will quickly cease to evolve. Thus, we may get less frequent patterns through a high decay rate. If we want to obtain the maximum frequent patterns, we can set a low decay rate, such as 5%, but this will give rise to a longer running time.

4.6 Algorithm Description

Our algorithm is composed of the following six steps.

Step 1: We obtain the initial population which includes all frequent 1-item positive and 1-item negative sequences. **Step 2:** Calculate all initial individuals' fitness. Their *dynamic fitness* is set to their *fitness*. **Step 3:** We select the top *K* individuals with high dynamic fitness from the population. After selection, the dynamic fitness of the selected individuals is updated by Eqn.(2). **Step 4:** Crossover and mutation between the selected individuals to produce the next generation. **Step 5:** After obtaining the next generation, we first prune invalid individuals and then calculate the frequency and fitness of remained individuals in new generation. If the frequency of an individual is greater than *min_sup*, we add it into the population, and set its fitness and dynamic fitness, but if the population has included this individual, we ignore it. **Step 6:** Go back to step 3 and iterate the above process until all individuals in the population are dead (i.e., their dynamic fitness has become close to 0). The dead individuals are still in

population, but they ceased to evolve. In the end, we obtain the final result - whole population, which is composed of all dead individuals.

The pseudocode of our algorithm is given as follows.

```

RunGA(min_sup, decay_rate, crossover_rate, mutation_rate){
  pop = initialPopulation();
  for (each individual ind in pop){
    ind.fitness = calculateFitness(ind);
    ind.dfitness = ind.fitness
    pop.sum_dfitness = pop.sum_dfitness + ind.dfitness
  }
  while ( pop.sum_dfitness > 0 ){
    popK = Selection(pop);
    if (Random() < crossover_rate) Crossover(popK);
    if (Random() < mutation_rate) Mutation(popK);
    for (each individual ind in popK)
      if (Prune(ind) != true && ind.sup >= min_sup) pop.add(ind);
  }
  return pop;
}

Selection(pop){ //Subfunction for selecting top K individuals from population
  for (each ind with top K dfitness in pop){
    popK.add(ind);
    ind.dfitness = ind.dfitness * (1-decay_rate);
    if (ind.dfitness < 0.01) ind.dfitness = 0;
  }
  return popK;
}

```

5 Experiments

Our algorithm was implemented with Java and tested with three synthetic sequence datasets generated by an IBM data generator [1] and a real-world dataset. We also implemented the PNSP algorithm [11] and Neg-GSP algorithm [14] with Java for performance comparison. All the experiments were conducted on a PC with Intel Core 2 CPU of 2.9GHz, 2GB memory and Windows XP Professional SP2.

Dataset1(DS1) is C8.T8.S4.I8.DB10k.N1k, which means the average number of elements in a sequence is 8, the average number of items in an element is 8, the average length of a maximal pattern consists of 4 elements and each element is composed of 8 items average. The data set contains 10k sequences, the number of items is 1000.

Dataset2(DS2) is C10.T2.5.S4.I2.5.DB100k.N10k.

Dataset3(DS3) is C20.T4.S6.I8.DB10k.N2k.

Dataset4(DS4) is real application data for insurance claims. The data set contains 479 sequences. The average number of elements in a sequence is 30. The minimum number of elements in a sequence is 1, and the maximum number is 171.

Experiments were done to compare the different *Crossover Rate*, *Mutation Rate* and *Decay Rate* on two synthetic datasets, *DS1* and *DS2*. Each experiment was run 10 times and then the average value was got as the final result. We focused on comparing runtime, the number of patterns and the runtime per pattern, which indicates how long it takes to get one pattern. The total number of all patterns was determined by PNSP and

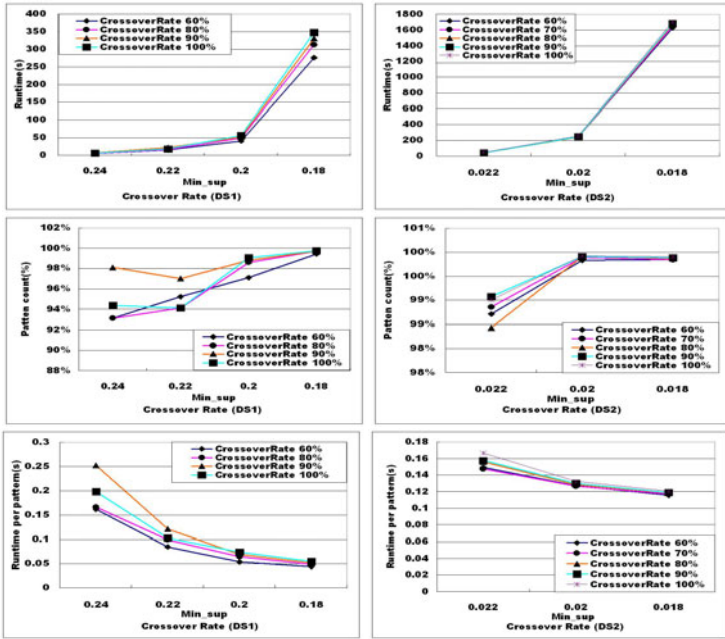


Fig. 2. Different Crossover Rates

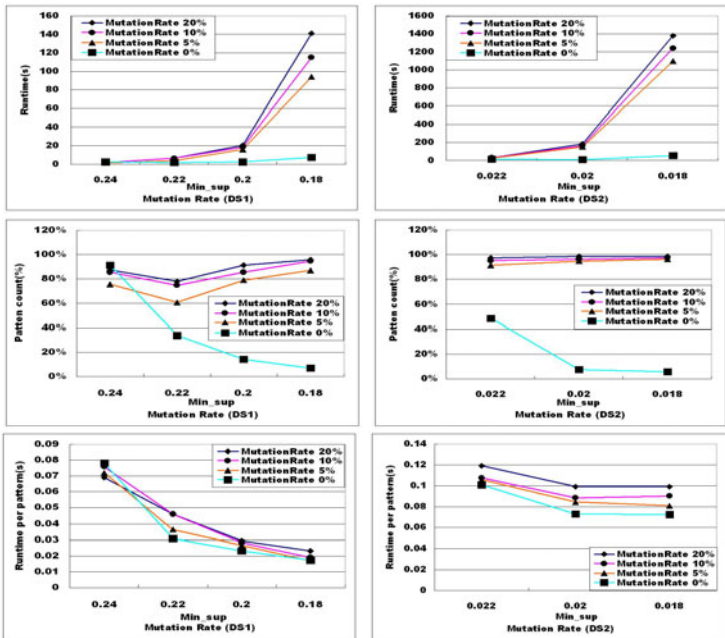


Fig. 3. Different Mutation Rates

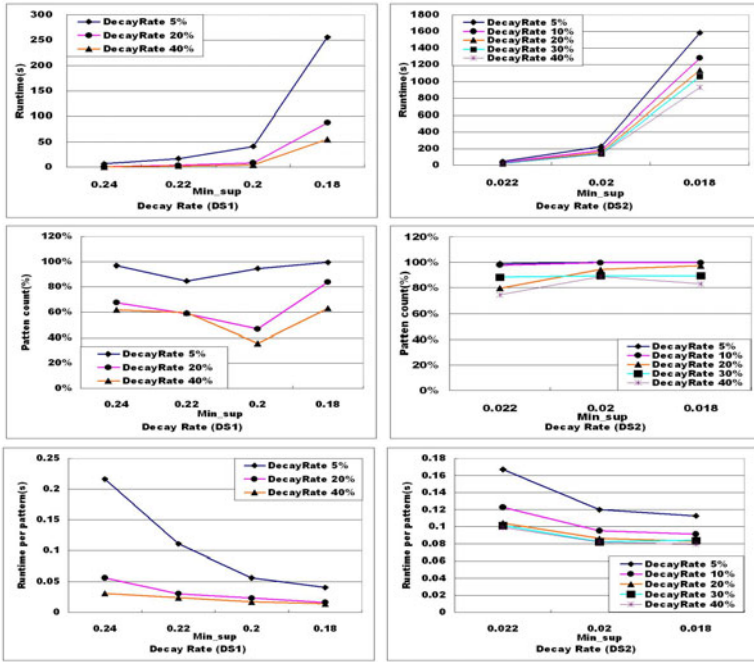


Fig. 4. Different Decay Rates

Neg-GSP algorithm, and it was then easy to know the proportion of patterns we could get by using our algorithm. The Y axis (see the 3rd and 4th chart of Fig. 2) indicates the proportion of patterns.

Crossover Rate. We compared different crossover rates from 60% to 100%. Fig. 2 shows the effect of different crossover rates on *DS1* and *DS2*. With low crossover rates, such as 60%, we obtained almost the same proportion of patterns as with high crossover rates (see the 2nd and 5th charts in Fig. 2). The least runtime per pattern is achieved when the crossover rate is low, so 60% is the best choice for the two datasets in our experiments.

Mutation Rate. We compared different mutation rates from 0% to 20% on *DS1* and *DS2* (see Fig. 3). They show that the mutation rate will not have an outstanding effect, but if it is set to 0%, it will result in missing a lot of patterns. A Mutation rate of 5-10% is a good choice because it can produce around 80% patterns for *DS1* and above 90% patterns for *DS2*. When the mutation rate is 5%, the average runtime per pattern is lower. We therefore set a mutation rate of 5% for the following experiments.

Decay Rate. Decay rate is a variable that we used to control evolution speed. If the decay rate is high, individuals will die quickly, so we can get only small proportion of patterns. If the decay rate is low, we can get more patterns, but a longer runtime is necessary (see Fig. 4). In order to get all negative sequential patterns, we always choose *decayrate*=5%, which enables us to obtain around 90% to 100% patterns on the two datasets.

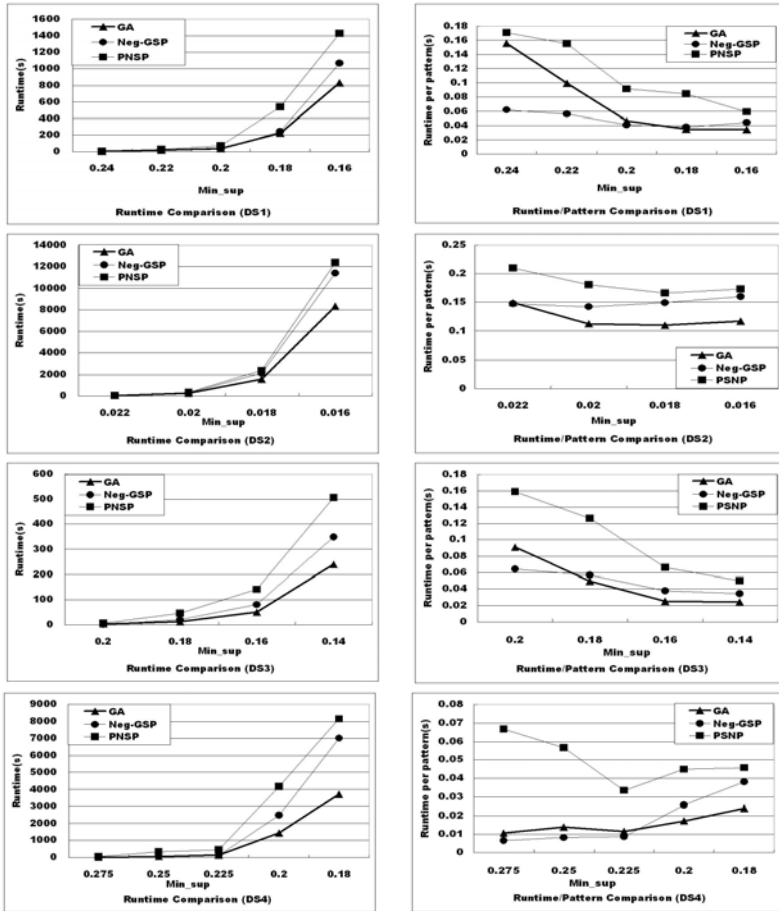


Fig. 5. Comparison with PNSP and Neg-GSP Algorithms

Performance Comparison. We compared our algorithm with PNSP and Neg-GSP, which are two algorithms proposed recently for negative sequential pattern mining. The tests are based on *Crossover Rate*=60%, *Mutation Rate*=5% and *Decay Rate*=5%. The results (see Fig. 5) on 4 different datasets show that the performance of the GA-based algorithm is better than PNSP and Neg-GSP when the support threshold is low. Our algorithm is not better than others when *min_sup* is high, because most patterns are very short and the GA-based method cannot demonstrate its advantage.

When *min_sup* is high, there are not as many patterns and the patterns are short, so it is very easy to find the patterns with existing methods. However, when *min_sup* is low, the patterns are longer and the search space is much bigger, so it is time-consuming to find patterns using traditional methods. Using our GA-based algorithm, it is still can obtain the patterns quickly even though *min_sup* is very low.

6 Conclusions and Future Work

Based on GA, we have proposed a method for negative sequential pattern mining. Extensive experimental results on synthetic datasets and a real-world dataset show that the proposed method can find negative patterns efficiently, and it is better than existing algorithms when the support threshold min_sup is low or when the patterns are long.

In our future work, we will focus on studying new measures including fitness function, selection and crossover method to make our algorithm more efficient. There should also be some better methods for pruning. Other work will be to explore post mining to find interesting patterns from the discovered negative sequential patterns. As we have obtained many negative sequential patterns, the means of finding interesting and interpretable patterns from them is valuable in industry applications.

References

1. Agrawal, R., Srikant, R.: Mining Sequential Patterns. In: Yu, P.S., Chen, A.L.P. (eds.) 11th International Conference on Data Engineering, pp. 3–14. IEEE Computer Soc. Press, Taipei (1995)
2. Bilal, A., Erhan, A.: An Efficient Genetic Algorithm for Automated Mining of Both Positive and Negative Quantitative Association Rules. *Soft. Computing* 10(3), 230–237 (2006)
3. Haupt, R.L., Haupt, S.E.: *Practical Genetic Algorithms*. Wiley, New York (1998)
4. Jay, A., Jason, F., Johannes, G., Tomi, Y.: Sequential PATTERN Mining Using a Bitmap Representation. In: *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, Edmonton (2002)
5. Mitchell, M.: *Introduction to Genetic Algorithms*. MIT Press, Cambridge (1996)
6. Nancy, P.L., Hung-Jen, C., Wei-Hua, H.: Mining Negative Sequential Patterns. In: *Proceedings of the 6th Conference on WSEAS International Conference on Applied Computer Science*, vol. 6, WSEAS, Hangzhou (2007)
7. Ouyang, W.M., Huang, Q.H.: Mining Negative Sequential Patterns in Transaction Databases. In: *2007 International Conference on Machine Learning and Cybernetics*, vol. 2, pp. 830–834 (2007)
8. Pei, J., Han, J., Mortazavi-Asl, B., Jianyong, W., Pinto, H., Qiming, C., Dayal, U., Mei-Chun, H.: Mining Sequential Patterns by Pattern-growth: The PrefixSpan Approach. *IEEE Transactions on Knowledge and Data Engineering* 16, 1424–1440 (2004)
9. Sandra de Amo, A.d.S.R.J.: Mining Generalized Sequential Patterns Using Genetic Programming. In: *ICAI 2003, Las Vegas, Nevada, USA* (2003)
10. Srikant, R., Agrawal, R.: Mining Sequential Patterns: Generalizations and Performance Improvements. In: *Proceedings of the Fifth International Conference on Extending Database Technology*, EDBT (1998)
11. Sue-Chen, H., Ming-Yen, L., Chien-Liang, C.: Mining Negative Sequential Patterns for E-commerce Recommendations. In: *Proceedings of the 2008 IEEE Asia-Pacific Services Computing Conference*, IEEE Computer Society Press, Los Alamitos (2008)
12. Zaki, M.J.: SPADE: An Efficient Algorithm for Mining Frequent Sequences. *Machine Learning* 42, 31–60 (2001)
13. Zhao, Y., Zhang, H., Cao, L., Zhang, C., Bohlscheid, H.: Efficient Mining of Event-Oriented Negative Sequential Rules. In: *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, WI-IAT 2008*, vol. 1, pp. 336–342 (2008)
14. Zheng, Z., Zhao, Y., Zuo, Z., Cao, L.: Negative-GSP: An Efficient Method for Mining Negative Sequential Patterns. In: *The 8th Australasian Data Mining Conference. AusDM 2009. Data Mining and Analytics*, Melbourne, Australia, vol. 101, pp. 63–67 (2009)

Valency Based Weighted Association Rule Mining

Yun Sing Koh, Russel Pears, and Wai Yeap

School of Computing and Mathematical Sciences,
Auckland University of Technology, New Zealand
{ykoh, rpears, wyeap}@aut.ac.nz

Abstract. Association rule mining is an important data mining task that discovers relationships among items in a transaction database. Most approaches to association rule mining assume that all items within a dataset have a uniform distribution with respect to support. Therefore, weighted association rule mining (WARM) was introduced to provide a notion of importance to individual items. Previous approaches to the weighted association rule mining problem require users to assign weights to items. This is infeasible when millions of items are present in a dataset. In this paper we propose a method that is based on a novel Valency model that automatically infers item weights based on interactions between items. Our experimentation shows that the weighting scheme results in rules that better capture the natural variation that occurs in a dataset when compared to a miner that does not employ such a weighting scheme.

Keywords: weighted association rule mining, valency, principal components.

1 Introduction

Association rule mining was introduced by [1] and is widely used to derive meaningful rules that are statistically related. It aims to extract interesting correlations, frequent patterns, associations or casual structures among sets of items in transaction databases. The relationships are not based on the inherent properties of the data themselves but rather based on the co-occurrence of the items within the database. The original motivation for seeking association rules came from the need to analyze supermarket transaction data also known as market basket analysis. An example of a common association rule is $\{bread\} \rightarrow \{butter\}$. This indicates that a customer buying bread would also buy butter. With traditional rule mining techniques even a modest sized dataset can produce thousands of rules, and as datasets get larger, the number of rules produced becomes unmanageable. This highlights a key problem in association rule mining; keeping the number of generated itemsets and rules in check, whilst identifying interesting rules amongst the plethora generated.

In the classical model of association rule mining, all items are treated with equal importance. In reality, most datasets are skewed with imbalanced data. By applying the classical model to these datasets, important but critical rules which occur infrequently may be missed. For example consider the rule: $\{stiff\ neck, fever, aversion\ to\ light\} \rightarrow \{meningitis\}$. Meningitis occurs relatively infrequently in a medical dataset, however if it is not detected early the consequences can be fatal.

Recent research [2,3,4,5] has used item weighting to identify such rules that rarely manifest but are nonetheless very important. For example, items in a market basket dataset may be weighted based on the profit they generate. However, most datasets do not come with preassigned weights, the weights must be manually assigned in a time consuming and error-prone fashion. Research in the area of weighted association rule mining has concentrated exclusively on formulating efficient algorithms for exploiting pre-assigned weights rather than deducing item weights from a given transactional database. We believe that it is possible to deduce the relative importance of items based on their interactions with each other. In application domains where user's input on item weights is either unavailable or impractical, an automated approach to assigning weights to items can contribute significantly to distinguishing high value rules from those with low value.

In this paper we make two contributions to the field of association rule mining. Firstly, we present a novel scheme that automates the process of assigning weights to items. The weights assignment process is underpinned by a "Valency model" that we propose. The model considers two factors: purity and connectivity. The purity of an item is determined by the number of items that it is associated with over the entire transactional database, whereas connectivity represents the strength of the interactions between items. We will elaborate on the Valency model later in the paper in section 3. Secondly, association rules produced by the Valency model are evaluated through a novel scheme based on Principal Components Analysis. The formulation of this interest measure was motivated by the fact that none of the popularly used interest measures such as Confidence and Lift was able to capture differences between rules with highly weighted items from those with lowly weighted ones.

The rest of the paper is organized as follows. In the next section, we look at previous work in the area of weighted association rule mining. In section 3 we give a formal definition of the weighted association rule mining problem. Section 4 describes our proposed Valency model while Section 5 presents the evaluation scheme used to assess the performance of the Valency model. Our experimental results are presented in Section 6. Finally we summarize our research contributions in Section 7 and outline directions for future work.

2 Background

The classical association rule mining scheme has thrived since its inception in [1] with application across a very wide range of domains. However, traditional Apriori-like approaches were not designed to deal with the rare items problem [6,7]. Items which are rare but have high confidence levels are unlikely to reach the minimum support threshold and are therefore pruned out. For example, Cohen [8] noted that in market basket analysis rules such as $\{caviar\} \rightarrow \{vodka\}$ will not be generated by traditional association rule mining algorithms. This is because both caviar and vodka are expensive items which are not purchased frequently, and will thus not meet the support threshold.

Numerous algorithms have been proposed to overcome this problem. Many of these algorithms follow the classical framework but substitute an item's support with a weighted form of support. Each item is assigned a weight to represent the importance of individual

items, with items that are considered interesting having a larger weight. This approach is called *weighted association rule mining* (WARM) [2,4,5,9,10]. Sanjay et al. [9] introduced weighted support to association rule mining by assigning weights to both items and transactions. In their approach rules whose weighted support is larger than a given threshold are kept for candidate generation, much like in traditional Apriori [1]. A similar approach was adopted by [2], but they applied weights to items and did not weigh transactions. They also proposed two different ways to calculate the weight of an itemset, either as the sum of all the constituent items' weights or as the average of the weights. However, both of these approaches invalidated the downward closure property [11].

This led Tao et al. [10] to propose a "weighted downward closure property". In their approach, two types of weights were assigned, item weight and itemset weight. The goal of using weighted support is to make use of the weight in the mining process and prioritize the selection of targeted itemsets according to their perceived significance in the dataset, rather than by their frequency alone.

Yan and Li [5] working in the domain area of Web mining proposed that weights be assigned on the basis of the time taken by a user to view a web page. Unlike the previous approaches [2,4,9,10] that assumed a fixed weight for each item, Yan and Li [5] allowed their weights to vary according to the dynamics of the system, as pages became more popular (or less popular) the weights would increase (or decrease), as the case may be.

Recently Jian and Ming [12] introduced a system for incorporating weights for mining association rules in communication networks. They made use of a method based on a subjective judgements matrix to set weights for individual items. Inputs to the matrix were supplied by domain specialists in the area of communications networks.

Thus it can be seen in previous work that the weight assignment process relies on user's subjective judgements. The major issue with relying on subjective input is that rules generated only encapsulate known patterns, thus excluding the discovery of unexpected but nonetheless important rules. Another issue is that the reliance on domain specific information constrains the range of applicability to only those domains where such information is readily available. There is no published work that is known to the authors that addresses these two issues. This motivated us to formulate a generic solution for the weight assignment problem that can be deployed across different application domains.

3 The Weighted Association Rule Mining (WARM) Problem

Given a set of items, $I = \{i_1, i_2, \dots, i_n\}$, a transaction may be defined as a subset of I and a dataset as a set D of transactions. A set X of items is called an itemset. The support of X , $\text{sup}(X)$, is the proportion of transactions containing X in the dataset. An *association rule* is an implication of the form $X \rightarrow Y$, where $X \subset I$, $Y \subset I$, and $X \cap Y = \emptyset$. The rule $X \rightarrow Y$ has *support* of s in the transaction set D , if $s = \text{sup}(XY)$. The rule $X \rightarrow Y$ holds in the transaction set D with *confidence* c where $c = \text{conf}(X \rightarrow Y) = \text{sup}(XY)/\text{sup}(X)$. Given a transaction database D , a support threshold *minsup* and a confidence threshold *minconf*, the task of association rule mining is to generate all association rules that have support and confidence above the user-specified thresholds.

In weighted association rule mining a weight w_i is assigned to each item i , where $-1 \leq w_i \leq 1$, reflecting the relative importance of an item over other items that it is

associated with. The weighted support of an item i is $w_i \text{sup}(i)$. Similar to traditional association rule mining, a weighted support threshold and a confidence threshold is assigned to measure the strength of the association rules produced. The weight of a k -itemset, X , is given by:

$$\left(\sum_{i \in X} w_i \right) \text{sup}(X) \quad (1)$$

Here a k -itemset, X , is considered a frequent itemset if the weighted support of this itemset is greater than the user-defined minimum weighted support ($w\text{minsup}$) threshold.

$$\left(\sum_{i \in X} w_i \right) \text{sup}(X) \geq w\text{minsup} \quad (2)$$

The weighted support of a rule $X \rightarrow Y$ is:

$$\left(\sum_{i \in X \cup Y} w_i \right) \text{sup}(XY) \quad (3)$$

Algorithm: Weighted Association Rule Mining (WARM)

Input: Transaction database D , weighted minimum support $w\text{minsup}$, universe of items I

Output: Weighted Frequent itemsets

```

 $L_k \leftarrow \{\{i\} | i \in I, \text{weight}(c) * \text{support}(c) > w\text{minsup}\}$ 
 $k \leftarrow 1$ 
while ( $|L_k| > 0$ ) do
   $k \leftarrow k + 1$ 
   $C_k \leftarrow \{x \cup y | x, y \in L_{k-1}, |x \cap y| = k - 2\}$ 
   $L_k \leftarrow \{c | c \in C_k, \text{weight}(c) * \text{support}(c) > w\text{minsup}\}$ 
 $L_k \leftarrow \bigcup_k L_k$ 

```

An association rule $X \rightarrow Y$ is called an interesting rule if $X \cup Y$ is a large itemset and the confidence of the rule is greater than or equal to a minimum confidence threshold. A general weighted association rule mining algorithm [10] is shown above. The algorithm requires a weighted minimum support to be provided. In this algorithm L_k represents the frequent itemsets also known as the large itemsets and C_k represents the candidate itemsets. Candidate itemsets whose weighted support exceeds the weighted minimum support are considered large itemsets and will be included in the rule generation phase.

Thus it can be seen that item weighting enables items with relatively low support to be considered interesting (large) and conversely, items which have relatively high support may turn out to be uninteresting (not large). This adds a new dimension to the classical association rule mining process and enables rules with high weights in their rule terms to be ranked ahead of others, thus reducing the burden on the end user in sifting through and identifying rules that are of the greatest value.

4 Valency Model

The Valency model is based on the intuitive notion that an item should be weighted based on the strength of its connections to other items as well as the number of items that it is connected with. We say that two items are connected if they have occurred together in at least one transaction. Items that appear often together when compared

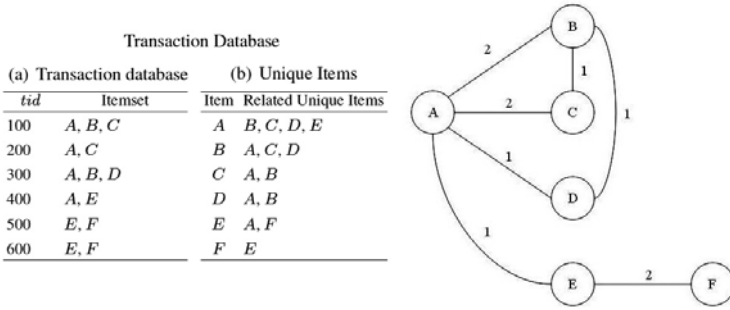


Fig. 1. Items Graph

to their individual support have a high degree of connectivity and are thus weighted higher. At the same time, an item that is contained in a small clique of items is said to have a high degree of purity and is given a proportionally higher weight. We will formally define the notions of *connectivity* and *purity* with the help of the following example.

Figure 1 is an example transaction dataset which can be represented as a graph whereby the nodes represent an item and the edges represent the support of the two items as an itemset. For example, the edge between node A and node B has a strength of 2, meaning that A and B occur together twice in the dataset. The Valency model we developed for our research is inspired by the Inverse Distance Weighting function (which was proposed by Shepard [13]). Inverse distance weighting is an interpolation technique which generates values for unknown points as a function of the values of a set of known points scattered throughout the dataset. In defining purity we took into account the importance of an item being featured in a rule term. In general, we prefer rules that have items that are not merely associated with each other strongly but also are distinctive in the sense that they appear with relatively few items. Such distinctive items add value to rules as they are more likely to capture genuine affinities, and possibly causal effects than an item that is selected only on the basis of a strong statistical correlation [14]. A strong statistical correlation between two items does not always indicate that a natural affinity exists between them. Consider, for example an item X having very high support that ends up being associated with many items Y, Z , etc merely because of the fact that it occurs in a very large fraction of the transactions, thus making the associations between (X, Y) and (X, Z) spurious, even though the correlations between X and Y on the one hand and X and Z on the other hand are high. Keeping these facts in mind, we formally define the purity, p , for a given item k as:

$$p_k = 1 - \frac{\log_2(|I_k|) + \log_2(|I_k|)^2}{\log_2(|U|)^3} \tag{4}$$

Where $|U|$ represents the number of unique items in the dataset and $|I_k|$ represents the number of unique items which are co-occurring with item k . Purity as defined in Equation 4 ensures that the maximum purity value of 1 is obtained when the number of items linked with the given item is 1, whereas the purity converges to the minimum

value of 0 as the number of linkages increases and become close to the number of items in the universal set of items. We chose to model purity with a non-linear logarithmic function as we wanted it to decrease sharply with the number of linkages. The $\log(|U|)^3$ term in the denominator ensures that the rate of decrease in purity is sensitive to the size of the universal set. For databases with a larger number of items (larger $|U|$) the gradient of descent is steeper when compared to databases with a smaller pool of items (smaller $|U|$) and so a smaller number of items will acquire high purity values. The second contribution to an item's valency relies on how strongly it is connected to its neighboring items, or its *connectivity*. Given an item k which is connected to n items in its neighborhood, the connectivity, c_k is defined as:

$$c_k = \sum_i^n \frac{\text{count}(ik)}{\text{count}(k)} \quad (5)$$

We can now define the valency contained by an item k , denoted by v_k as the combination of both the purity and the connectivity components:

$$v_k = \beta.p_k + (1 - \beta). \sum_i^n \frac{\text{count}(ik)}{\text{count}(k)}.p_i \quad (6)$$

where β is a parameter that measures the relative contribution of the item k over the items that it is connected with in the database. The objective of the Valency model is to capture rules over small cliques of items such that items within a given clique have high purity and connectivity with other items contained within that clique. Since all items within a given clique are connected to each other, it follows from our definition of purity that all items within a clique have the same purity. Thus we can re-write the above equation as:

$$v_k = \beta.p_k + (1 - \beta).p_k. \sum_i^n \frac{\text{count}(ik)}{\text{count}(k)} \quad (7)$$

Thus, for a given item k , the relative contribution made to the valency by other items in the clique is dependent on both the value of the parameter β as well as the sum of the connectivity values from item k . We set the value of β as:

$$\beta = \frac{1}{n} \sum_i^n \frac{\text{count}(ik)}{\text{count}(k)} \quad (8)$$

With this setting of β we can re-write Equation 7 as:

$$v_k = \beta.p_k + n\beta(1 - \beta).p_k \quad (9)$$

With this setting of β we can see from the above expression that the relative contribution of the neighboring items of k over itself is $1 - \beta$, which means that as the value of β increases the item k itself assumes more importance in relation to its neighbors. We use the valency of an item as its weight. The weight calculation for an item is thus a computationally straightforward process as the weight for an item is independent of the

weights of other items. Also, the weight assignment process for items can be accomplished in the first pass through the dataset as the local neighborhoods for each item can be computed on the fly together with the reading of the dataset. In the next section we discuss our evaluation criteria for determining the quality of the rules obtained by applying the Valency model.

5 Rule Evaluation Methodology

A vast array of metrics for evaluating the quality of association rules have been proposed in the literature. Apart from the standard metrics of rule Support and Confidence, other measures such as Lift, Information Gain, Conviction, and Correlation have been used. The standard metrics are excellent at evaluating rules at the individual level in terms of the strength of correlation between terms and in assessing predictive accuracy. However, in the context of weighted association rule mining it is necessary that the contribution from each rule item is quantified and the contribution that it makes to the overall rule quality be assessed. Existing metrics tend to operate on the rule level rather than on the individual item level. This motivated us to investigate the use of Principal Components Analysis (PCA) to evaluate the quality of our weighted association rule miner.

PCA is a mathematical technique that has been widely used in the data mining arena. Basically, PCA takes a set of variables and finds a set of independent axes (the Principal Components) which explain all or most of the variation that occurs within the dataset. Its main application is in the area of classification and clustering where it is used as a pre-processing technique for dimensionality reduction. It has also been used before in association rule mining, but in a limited context where items are defined on a true numerical scale [15]. However, our use of PCA is quite different.

We concentrate solely on the right hand sides (RHSs) of rules as they encapsulate the actionable components of rules. Focussing on rule consequents allows us to test the degree of diversity amongst the actionable components discovered by the rule generator without the confounding effect of diversity amongst the left hand sides (LHSs) of rules. A set of rules with exactly the same RHS does not yield as much knowledge as rules that are diverse in their RHSs. For example, a set of rules *egg, bread* → *milk*; *butter, bread* → *milk*; and *tuna, egg* → *milk*, can be considered less interesting than rules with a greater diversity such as *diaper* → *baby food*; *ham* → *cheese*; and *chips* → *soda*. In a medical database containing information about a number of different diseases a rule generator that has poor coverage of the set of diseases (i.e. only identifies a small fraction of the diseases in the RHSs of the rules) is not as useful as one that has a better coverage with respect to the set of diseases.

We first apply PCA to the transaction dataset and obtain the Eigen vectors for the first two principal components. These vectors will be a linear function of the form: $e_{k1}I_1 + e_{k2}I_2 + \dots + e_{kn}I_n$ where e_{kp} is the Eigen value for the p th item on the k th principal component (k is either 1 or 2). We process the rule set by removing LHS of each rule. This results in a collection of rule consequents (RHSs) containing duplicate entries as the LHS terms have been eliminated. After duplicate elimination we obtain a more compact representation of the rule set, R . We project the rule set R on its first two principal components and obtain a quantified version of the rule set, which we

denote by S . The set S contains a set of ordered pairs (X, Y) where X, Y are vectors representing principal components 1 and 2 respectively for each rule.

PCA enables us to capture the amount of variance that is explained by each rule term for each rule. The greater the amount of variance that is captured by a rule term, the higher the value of that term and the higher the contribution it makes to the rule as a whole. Thus PCA provides us with an independent method of evaluating the efficacy of our Valency model. If our Valency model is to outperform an unweighted association mining scheme such as Apriori then the delineation of the rules in PCA space produced by our Valency model should be better. In order to assess the quality of the delineation we applied the K-means clustering algorithm to the (X, Y) vectors and then visualized the clusters. We also quantified the degree of delineation by calculating a cluster purity measure along the axis that provided the better delineation, which happened to be the first principal axis (rather than the second) in most of the experiments that we carried out. In the next section we present the results of our experimentation with our Valency model.

6 Experimental Results

Our motivation in introducing the Valency model was to facilitate the automatic assignment of weights from a given transaction dataset without requiring additional information from users. As such we were interested in examining the impact of the weight assessment process in an environment where user input is not available, and this led us to compare our algorithm with the classical Apriori approach. Our experimentation was conducted in two steps, firstly a performance comparison with Apriori, and secondly an examination of the impact of key parameters of the Valency model. We used seven UCI datasets [16]. We also experimented with synthetic data for which we used the data generator proposed by [11]. Datasets D , were created with the following parameters: number of transactions $|D|$, average size of transactions $|T|$, number of unique items $|I|$, and number of large itemsets $|L|$.

6.1 Principal Components Analysis of the Rule Bases

Table 1 shows the results of running both Apriori and our algorithm on the datasets mentioned above. Each row shows the dataset, the number of rules produced, the number of RHSs produced (bracketed), and the cluster purity obtained by clustering the resulting rule bases on the first two principal components. We see from the results that the effect of weighting is to produce a much more compact rule base as Valency's rule base, with the exception of Soybean, is much smaller than Apriori's. In order to keep the comparison fair we ran the two algorithms at minimum support thresholds so that they produced rule bases which had approximately the same support distributions. The compact nature of Valency's rule base vis-a-vis Apriori is due to the influence of the purity component that reduces the weighted support of an item sharply as the number of items that it interacts with increases. We verified this by substituting the non linear purity function with a linear one. The linear function did not punish highly connected items as severely as its non linear counterpart, thus resulting in a rule base that exploded in size and became very similar to that of Apriori in both qualitative and quantitative

Table 1. Clustering Results for First Two Principal Components

| Dataset | Apriori | | Valency | | Improvement |
|------------------------|--------------|----------------|--------------|----------------|-------------|
| | No. of Rules | Cluster Purity | No. of Rules | Cluster Purity | |
| Bridges | 1875(15) | 100 | 505(15) | 100 | 0.0 |
| Flag | 486500(719) | 86.6 | 119948(121) | 94.2 | 8.8 |
| Flare | 244753(1052) | 90.9 | 715(32) | 100 | 10.0 |
| Hepatitis | 720633(2065) | 87.9 | 45850(233) | 96.6 | 9.9 |
| Mushroom | 61515(1064) | 92.5 | 4005(134) | 92.5 | 0.0 |
| Soybean | 188223(1211) | 82.9 | 456753(1310) | 99.7 | 20.3 |
| Synthetic (T25I200D1K) | 618579(2195) | 89.7 | 266914(853) | 98.4 | 9.7 |
| Zoo | 644890(3128) | 89.7 | 5475(127) | 99.2 | 10.6 |

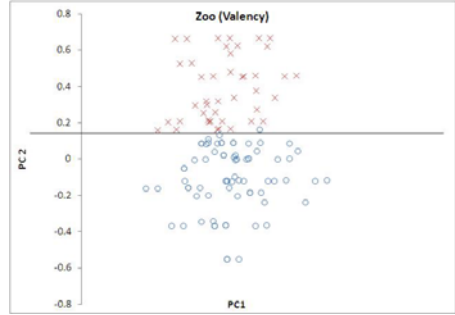
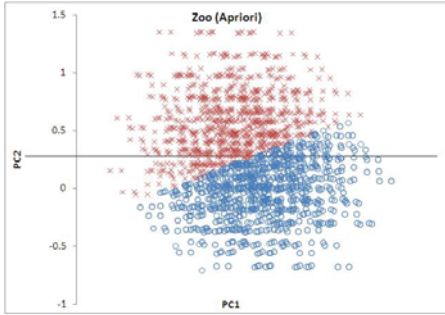


Fig. 2. PC1 and PC2 Clusters for Zoo Dataset based on Apriori

Fig. 3. PC1 and PC2 Clusters for Zoo Dataset based on Valency

terms. The effect of the linear function was to dilute the effect of purity and hence the weighting scheme was not as effective in discriminating between different items, thus resulting in a larger proportion of items assuming higher purity and higher weight values.

The second interesting aspect of the results is that Valency produced a better set of clusters when compared to Apriori. The cluster purity improvement measure for Valency ranges from 0% for the relatively sparse Bridges and Mushroom datasets to 20.3% for the denser Soybean dataset. This improvement is due to the fact that the items that feature in Valency’s rules capture a higher proportion of variance that occurs over the underlying dataset in relation to Apriori. This result confirms our hypothesis that it is possible to automatically deduce weights from the interactions that occur between items in a transactional database. The result for the experimentation with various types of synthetic datasets were broadly similar to that of the real-world datasets. As the density of the dataset increased so did the improvement in cluster purity value between Valency and Apriori. We do not report on all synthetic datasets due to lack of space. Instead we present the result for one such dataset (T25I200D1K) that is representative of the experimentation that we conducted with synthetic data.

Figures 2 and 3 shows the clusters generated in PCA space for the Apriori and Valency schemes on the Zoo dataset. For the Zoo dataset the 2nd principal component produced the cleaner demarcation between the clusters. The figures show that Valency produces a visibly better separation of clusters around the point of intersection with the

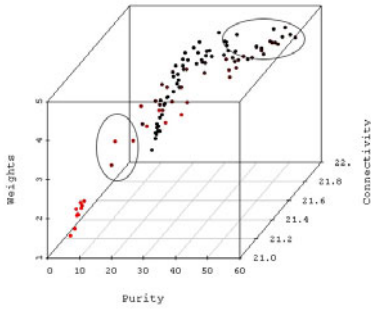


Fig. 4. Weights, Purity, and Connectivity for Mushroom Dataset

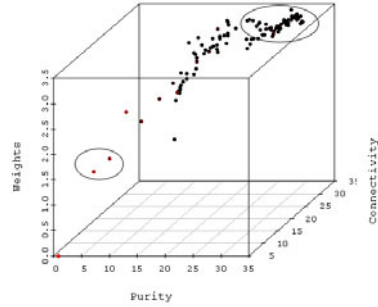


Fig. 5. Weights, Purity, and Connectivity for Soybean Dataset

second principal axis. The other visualizations which we could not include due to lack of space, were similar except that the axis of greater separation was the first principal component for both of the algorithms.

6.2 Impact of Purity and Connectivity on Item Weight

In this part of the experimentation we investigated the simultaneous effects of purity and connectivity on item weight. For each of the datasets that we experimented with we plotted item weight versus purity and connectivity in a 3D representation in order to assess the simultaneous effects of purity and connectivity on weight.

Figures 4 and 5 show the results for two representative datasets, namely Mushroom and Soybean. It is clear from the plots that high weights only result if both purity and connectivity are high at the same time (the ovals on the north east corner of the cubes). We can also see the filtering effect of purity on weight. Items with high connectivity but low purity end up with lower weight (denoted by the ovals on the south west corners of the cubes). Algorithms such as Apriori (and all such un-weighted association rule mining algorithms) that do not discriminate on purity will tend to capture rules that contain items that occur with a large number of other items thus producing rules that are unlikely to be novel or useful to the decision maker.

6.3 Case Study Zoo Dataset

We compare the results on the zoo dataset based on the rule bases produced by the Apriori and the Valency schemes. Using Apriori, we found that the item *toothed = 1* occurs with 33/40 (or 82.5%) of the other items, and that it appears in 431079/644890 (or 66.8%) of the rules. The item *toothed = 1* thus serves to dilute the effects of all rules that it participates in, which happens to be the majority (66.8%) of the Apriori rule base. The subrule $\{eggs = 0, legs = 4\} \rightarrow \{toothed = 1\}$ appeared in all of the top 20 rules when ranked by Lift. Considering that all three items within this subrule have low weights, and given the fact that all of Apriori's top 20 rules embed this subrule, it follows that the degree of diversity captured by Apriori's top ranked rules happens to be low. With the Valency scheme, we noticed that *toothed = 1* occurs with 1508/5474

(27.5%) of the other rules. When we ranked the rules by lift, we noticed that *toothed = 1* first appeared in rule 317 with a lift value of 2.06.

For Apriori, the items which appeared in the first 20 rules were: *eggs = 0*, *legs = 4*, *airborne = 0*, *fins = 0*, *hair = 0*, *toothed = 1*, *milk = 1*, *backbone = 1*, *breathes = 1*, *type = 1*, and *feathers = 0*. Out of the 11 items only 4 items are of high weight. The average weight for the items was 1.16. With the Valency scheme, the items that appeared in the first 20 rules were *backbone = 1*, *breathes = 1*, *milk = 0*, *venomous = 0*, *eggs = 1*, *fins = 0*, *tail = 1*, and *domestic = 0*. The average weight for the items was 1.34.

The above results illustrate the extent to which items that are not distinctive can dilute the efficacy of rules produced by an association miner that does not utilize item weighting. On the other hand the Valency model is more discriminative in its use of items such as *toothed = 1*. Whenever such items feature in its rule base, it tends to include items with higher weight to compensate, thus mitigating the effects of such lowly weighted items.

7 Conclusions and Future Work

In this paper, we propose a new item weighting scheme based on the Valency model. We fit the weights to items based on the notions of connectivity and purity. The valency weighting function consists of two different parts: weights of an item based on its strength of connections and weights of its neighboring items. We used PCA to investigate the degree of variation captured by the rule bases. Overall, the Valency model produces better rules than traditional Apriori. In terms of future work we would like to investigate the effects of not just the immediate neighbors on an item's weight but to also capture the effects of non-neighboring items that are not directly connected to the given item under consideration.

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, pp. 207–216 (1993)
2. Cai, C.H., Fu, A.W.C., Cheng, C.H., Kwong, W.W.: Mining association rules with weighted items. In: IDEAS 1998: Proceedings of the 1998 International Symposium on Database Engineering & Applications, Washington, DC, USA, p. 68, IEEE Computer Society, Los Alamitos (1998)
3. Sun, K., Bai, F.: Mining weighted association rules without preassigned weights. *IEEE Trans. on Knowl. and Data Eng.* 20(4), 489–495 (2008)
4. Wang, W., Yang, J., Yu, P.S.: Efficient mining of weighted association rules (WAR). In: KDD 2000: Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 270–274. ACM, New York (2000)
5. Yan, L., Li, C.: Incorporating pageview weight into an association-rule-based web recommendation system. In: Sattar, A., Kang, B.-h. (eds.) AI 2006. LNCS (LNAI), vol. 4304, pp. 577–586. Springer, Heidelberg (2006)
6. Liu, B., Hsu, W., Ma, Y.: Mining association rules with multiple minimum supports. In: Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 337–341 (1999)

7. Koh, Y.S., Rountree, N.: Finding sporadic rules using apriori-inverse. In: Ho, T.-B., Cheung, D., Liu, H. (eds.) PAKDD 2005. LNCS (LNAI), vol. 3518, pp. 97–106. Springer, Heidelberg (2005)
8. Cohen, E., Datar, M., Fujiwara, S., Gionis, A., Indyk, P., Motwani, R., Ullman, J.D., Yang, C.: Finding interesting association rules without support pruning. *IEEE Transactions on Knowledge and Data Engineering* 13, 64–78 (2001)
9. Sanjay, R., Ranka, S., Tsur, S.: Weighted association rules: Model and algorithm (1997), <http://citeseer.ist.psu.edu/185924.html>
10. Tao, F., Murtagh, F., Farid, M.: Weighted association rule mining using weighted support and significance framework. In: KDD 2003: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 661–666. ACM, New York (2003)
11. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: Bocca, J.B., Jarke, M., Zaniolo, C. (eds.) Proceedings of the 20th International Conference on Very Large Data Bases, VLDB, Santiago, Chile, pp. 487–499 (1994)
12. Jian, W., Ming, L.X.: An effective mining algorithm for weighted association rules in communication networks. *Journal of Computers* 3(4) (2008)
13. Shepard, D.: A two-dimensional interpolation function for irregularly-spaced data. In: Proceedings of the 1968 23rd ACM national conference, pp. 517–524. ACM, New York (1968)
14. Tan, P.N., Steinbach, M., Kumar, V.: Introduction to Data Mining. Pearson Inc., London (2006)
15. Faloutsos, C., Korn, F.A., Labrinidis, Y., Kotidis, A.K., Perkovic, D.: Quantifiable data mining using principal component analysis. Technical Report CS-TR-3754, Institute for Systems Research, University of Maryland, College Park, MD (1997)
16. Asuncion, A., Newman, D.: UCI machine learning repository (2007), <http://www.ics.uci.edu/~mllearn/MLRepository.html>

Ranking Sequential Patterns with Respect to Significance

Robert Gwadera and Fabio Crestani

Universita della Svizzera Italiana
Lugano, Switzerland

Abstract. We present a reliable universal method for ranking sequential patterns (itemset-sequences) with respect to significance in the problem of frequent sequential pattern mining. We approach the problem by first building a probabilistic reference model for the collection of itemset-sequences and then deriving an analytical formula for the frequency for sequential patterns in the reference model. We rank sequential patterns by computing the divergence between their actual frequencies and their frequencies in the reference model. We demonstrate the applicability of the presented method for discovering dependencies between streams of news stories in terms of significant sequential patterns, which is an important problem in multi-stream text mining and the topic detection and tracking research.

1 Introduction

1.1 Motivation

Frequent sequential pattern mining, introduced in [1], has established itself as one of the most important data mining frameworks with broad applications including analysis of time-related processes, telecommunications, bioinformatics, business, software engineering, Web click stream mining, etc [9]. The problem is defined as follows. Given a collection of itemset-sequences (sequence database of transactions) and a minimum frequency (support) threshold, the task is to find all subsequence patterns, occurring across the itemset-sequences in the collection, whose frequency is greater than the minimum frequency threshold. The main focus of the research on sequential pattern mining has been on devising efficient algorithms for discovering frequent sequential patterns (see [9] for a review). Although state of the art mining algorithms can efficiently derive a complete set of frequent sequential patterns under certain constraints, the main problem is that the set of frequent sequential patterns is still too large for effective usage [9]. The two most effective methods for reducing the large set of frequent sequential patterns have been: *closed sequential pattern* mining [12] and *maximal sequential pattern* mining [5]. However no methods for assessing interestingness of sequential patterns have been proposed while such methods are very important to advance the applicability of frequent sequential pattern mining. By comparison, such methods have been proposed for subsequence patterns in the sliding window model [6] and for itemsets (see [11] for a review).

1.2 Overview of the Method

We approach the problem by first building a probabilistic reference model for the collection of itemset-sequences and then deriving an analytical formula for the relative frequency for sequential patterns. Given such a model we discover sequential patterns that are *under-represented* or *over-represented* with respect to the reference model, where a pattern is under-represented if it is too infrequent in the input collection of itemset-sequences and a pattern is over-represented if it is too frequent in the input collection of itemset-sequences. According to this notion a sequential pattern is significant if the probability that it would occur by chance a specific number of times, in the reference model, is very small. Note that the frequency of occurrence alone is not enough to determine significance, i.e., an infrequent sequential pattern can be more significant than a frequent one. Furthermore an occurrence of a subsequence pattern may be *meaningless* [6] if it occurs in an sequence of an appropriately large size. Our algorithm for ranking sequential patterns with respect to significance works as follows: (I) we find frequent sequential patterns using *PrefixSpan* [10] for a given minimum support threshold; (II) we compute their frequencies and variances of the frequencies in the reference model and (III) we rank the frequent sequential patterns with respect to significance by computing the divergence (*Z-score*) between the empirical (actual) frequencies and frequencies in the reference model. Given the reference model a presence of significant divergence between the actual and computed frequency of a sequential pattern indicates that there is a dependency between itemsets/items in that pattern. In order to capture these dependencies our reference model consists of two sub-models: (I) *sequence-wise reference model*: treats itemsets as alphabet symbols and represents an independence model where itemsets occur independently of their order in an itemset-sequence and (II) *itemset-wise reference model*: provides decorrelated frequencies of itemsets for the sequence-wise reference model. By decorrelated frequencies we mean that given an attribute (item) a_1 and attribute a_2 the frequency of itemset (a_1, a_2) is computed using a *maximum entropy model*, where the marginal empirical probabilities are preserved. The reason we use such a model for itemsets is that unlike in the case of frequent itemset mining, we do not consider empty itemsets (empty attribute sets) and therefore the independence model for itemsets [3] is inappropriate as an itemset-wise reference model. In particular, using the independence model for sparse non-empty itemsets (the average number of ones in a row is much smaller than the number of attributes) would artificially overestimate the probability of the empty itemset causing a distortion of proper proportions of probabilities of non-empty itemsets. Note, that the sequence-wise reference model can be easily extended to *Markov models* in the spirit of [7]. The reason we consider the sequence-wise model to be independence model in this paper is because of the following reasons: (I) it is the model of choice if the Markov reference model is not known; (II) it has an intuitive interpretation as a method for discovering dependencies and (III) it leads to exact polynomial formulas for computing the frequencies of sequential patterns.

1.3 Multi-stream of News Stories and the Reference Model

We demonstrate the applicability of the presented method for discovering dependencies between streams of news stories, which is an important problem in multi-stream text mining and the topic detection and tracking research [2]. For this purpose we generated a collection of itemset-sequences from a multi-stream of news stories that was gathered from RSS feeds of major world news agencies [8]. Every itemset-sequence in that collection consists of stream identifiers of stories in a cross-stream cluster of news stories reporting the same news event, where the sequence is ordered according to the timestamps of the stories. Every itemset contains stream identifiers of documents published within the same time granularity. As an example itemset-sequence in that collection consider [(AP, MSNBC), UPI] that corresponds to three articles on the same news event (e.g., an earthquake in Italy), where the first two of them were published by AP and MSNBC within the same time granularity and followed by an article by UPI. Thus, clearly the empty itemset () does not occur in our data set. We stated the following research questions with respect to this collection of itemset-sequences: (I) what is the relationship between frequency, significance and content similarity in the discovered significant sequential patterns? and (II) what are the dependencies between the news sources in terms of sequential patterns of reporting the same news events?

As an example of the application of the reference model consider a case where the input collection of itemset-sequences contains a frequent sequential pattern $s = [(AP, MSNBC), UPI]$, that consist of two itemsets $s_1 = (AP, MSNBC)$ and $s_2 = UPI$ that are correlated by occurring frequently together. Then since the sequence-wise reference model assumes independence between the elements, the frequency of s computed from the sequence-wise reference model will be much smaller than its actual frequency leading to a high significance rank of s . Furthermore, $s_1 = (AP, MSNBC)$ contains two items $a_1 = AP$ and $a_2 = MSNBC$ which are correlated by occurring frequently together in the same itemsets. Then there are two possibilities for computing the frequency of s in the sequence-wise reference model: (I) we use the empirical frequency of s_1 or (II) we use a frequency of s_1 provided by the itemset-wise reference model. Then since the itemset-wise reference model provides decorrelated frequencies of itemsets while preserving marginal frequencies of the items (the publishing rates of AP and MSNBC), the frequency of s_1 computed from the itemset-wise reference model will be smaller than its empirical frequency leading to an even higher significance rank of s .

1.4 Related Work and Contributions

Thus, we present a reliable universal method for ranking sequential patterns with respect to significance that builds on the previous work [6], where a framework for assessing significance of subsequence patterns in the sliding window model was presented. The challenges of analysing itemset-sequences with respect to the previous work on sequences in [6] stems from the following facts: (I) itemset-sequences have variable sizes; (II) itemset-sequences contain itemsets (unordered sets) and (III) we do not consider empty itemsets. We address the

first problem by modeling the frequency of an itemset-sequence using a probabilistic *discrete mixture model* and we approach the second and third problem by using an appropriate *maximum entropy* itemset-wise reference model.

To the best of our knowledge this is the first algorithm for ranking sequential patterns with respect to significance while there has been an extensive research on mining frequent sequential patterns (see [9] for a review).

The paper is organized as follows. Section 2 reviews theoretical foundations, Section 3 defines the problem, Section 4 presents the sequence-wise reference model, Section 5 presents the itemset-wise reference model, Section 6 presents the algorithm for ranking sequential patterns with respect to significance, Section 7 presents experimental results and finally Section 8 presents conclusions.

2 Theoretical Foundations (Review)

In this section we review some concepts that are necessary in order to explain our framework.

2.1 Sequential Pattern Mining

In this section we review the problem of sequential pattern mining [1]. Let $\mathcal{A} = \{a_1, a_2, \dots, a_{|\mathcal{A}|}\}$ be a set of items (alphabet). A subset $\mathcal{I} \subseteq \mathcal{A}$, where $\mathcal{I} = \{a_1, a_2, \dots, a_{|\mathcal{I}|}\}$ is called an *itemset* or *element* and is also denoted by $(a_1, a_2, \dots, a_{|\mathcal{I}|})$. An *itemset-sequence* $s = [s_1, s_2, \dots, s_m]$ is an ordered list of itemsets, where $s_i \subseteq \mathcal{A}$. The size of the itemset-sequence is denoted by $|s|$ and the length of itemset-sequence s is defined as $l = \sum_{i=1}^m |s_i|$. An itemset-sequence $s = [s_1, s_2, \dots, s_m]$ is a *subsequence* of itemset-sequence $s' = [s'_1, s'_2, \dots, s'_m]$, denoted $s \sqsubseteq s'$, if there exist integers $1 \leq i_1 \leq i_2 \dots \leq i_m$ such that $s_1 \subseteq s'_{i_1}$, $s_2 \subseteq s'_{i_2}, \dots, s_m \subseteq s'_{i_m}$. We also say that s' is a *supersequence* of s and s is *contained* in s' . Given a *collection of itemset-sequences* $\mathbf{S} = \{s^{(1)}, s^{(2)}, \dots, s^{(|\mathbf{S}|)}\}$ the *support* (frequency) of an itemset-sequence s , denoted by $sup_{\mathbf{S}}(s)$, is defined as the number of itemset-sequences $s^{(i)} \in \mathbf{S}$ that contain s as a subsequence. The *relative support* (relative frequency) $rsup_{\mathbf{S}}(s) = \frac{sup_{\mathbf{S}}(s)}{|\mathbf{S}|}$ is the fraction of itemset-sequences that contain s as a subsequence. Given a relative support threshold $minRelSup$ an itemset-sequence s is called a *frequent sequential pattern* if $rsup_{\mathbf{S}}(s) \geq minRelSup$. The problem of mining sequential patterns is to find all frequent sequential patterns in \mathbf{S} given $minRelSup$. The support has an *anti-monotonic* property meaning that $sup_{\mathbf{S}}(s) \geq sup_{\mathbf{S}}(s')$ if $s \sqsubseteq s'$. A pattern s is called a *closed frequent sequential pattern* if none of its frequent supersequences has the same support. A pattern s is called a *maximal frequent sequential pattern* if none of its frequent supersequences is frequent. Table 1 presents an example collection of itemset-sequences, where itemset-sequence $id = 1$ has size $s = 3$, length $l = 4$ and consists of three elements (itemsets): $(1, 3)$, 1 and 1 . Given $minRelSup = 0.5$, $s = [(1, 3), 1]$ is a frequent sequential pattern that is contained in itemset-sequences: $id = 1, 3$, where $rsup_{\mathbf{S}}(s) = 0.5$.

Table 1. A collection of itemset-sequences

| id | itemset-sequence |
|----|------------------------|
| 0 | [2, 0] |
| 1 | [(1, 3), 1, 1] |
| 2 | [2, 3, (0, 2)] |
| 3 | [(1, 3), (2, 3), 0, 1] |

2.2 Significance of Subsequence Patterns

In this section we review the framework introduced in [6]. Let $e = [e_1, e_2, \dots, e_m]$ be a sequence of symbols. Let $\Omega_n(e|w) = \sum_{i=1}^n I_i$ be a random variable that represent the actual frequency (support) of size w windows containing at least one occurrence of e as a subsequence in an event sequence of size $n + w - 1$ (n shifts of the window), where I_i is an indicator function equal to 1 if the i -th shift contains e . Clearly $\mathbf{E}[\Omega_n(e|w)] = nP^\exists(e|w)$, where $P^\exists(e|w)$ is the probability that a window ending at a given position in the event sequence contains at least one occurrence of e as a subsequence. The superscript \exists means “at least one occurrence as a subsequence” and is used to distinguish this probability from a probability of e as a string. Clearly, I_1, I_2, \dots, I_n is a sequence of dependent random variables because a given subsequence pattern occurring in the input sequence may occur in many consecutive windows depending on its span. Therefore, because of the sliding window overlap $\Omega_n(e|w)$ does not have a *Binomial* distribution meaning $\mathbf{Var}[\Omega_n(e|w)] \neq nP^\exists(e|w)(1 - P^\exists(e|w))$. Let $\overline{\Omega}_n(e|w) = \frac{\Omega_n(e|w)}{n}$ be a random variable that represents the actual relative frequency of size w windows containing at least one occurrence of e as a subsequence in an event sequence, where $\mathbf{E}[\overline{\Omega}_n(e|w)] = P^\exists(e|w)$ and $\mathbf{Var}[\overline{\Omega}_n(e|w)] \leq \frac{1}{n}P^\exists(e|w)(1 - P^\exists(e|w))$.

Let $\mathcal{W}^\exists(e|w)$ be the set of all distinct windows of length w containing at least one occurrence of pattern e as a subsequence. Then $P^\exists(e|w) = \sum_{x \in \mathcal{W}^\exists(e|w)} P(x)$, where $P(x)$ is the probability of string x in a given Markov model. $\mathcal{W}^\exists(e|w)$ can be enumerated using an enumeration graph. The enumeration graph for a subsequence pattern $e = [e_1, e_2, \dots, e_m]$ is shown in Figure 1. In particular for the 0-order Markov reference model $P^\exists(e|w)$ can be expressed as follows

$$P^\exists(e|w) = P(e) \sum_{i=0}^{w-m} \sum_{\sum_{k=1}^m n_k = i} \prod_{k=1}^m (1 - P(e_k))^{n_k}, \tag{1}$$

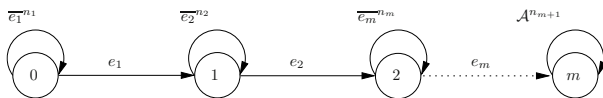


Fig. 1. Enumeration graph for a subsequence pattern $e = [e_1, e_2, \dots, e_m]$, where $\bar{e} = \mathcal{A} - e$ and \mathcal{A} is the alphabet. The exponents n_1, \dots, n_{m+1} above the self-loops denote the number of times the corresponding self-loops are selected.

where $P(e) = \prod_{i=1}^m P(e_i)$ and $P(e_i)$ is the probability of symbol e_i in the reference model. Then $P^\exists(e|w)$ is the probability of getting from state 0 to m in w steps. Paper [6] also presented an efficient $O(w^2)$ dynamic programming algorithm for computing $P^\exists(e|w)$ from (II). It was shown that if $\mathbf{Var}[\overline{\mathcal{O}}_n(e|w)] > 0$ then $\overline{\mathcal{O}}_n(e|w)$ satisfies the *Central limit theorem (CLT)* and this fact was used to set a lower and upper significance thresholds for $\overline{\mathcal{O}}_n(e|w)$.

3 Problem Definition

The problem of ranking sequential patterns (itemset-sequences) with respect to significance can be defined as follows.

Given: (I) collection of itemset-sequences $\mathbf{S} = \{s^{(1)}, s^{(2)}, \dots, s^{(n)}\}$, where $s^{(i)} = [s_1^{(i)}, s_2^{(i)}, \dots, s_{|s^{(i)}|}^{(i)}]$, $s_t^{(i)} \subseteq \mathcal{A} = \{a_1, a_2, \dots, a_{|\mathcal{A}|}\}$ and $M = \max_{1 \leq i \leq n} |s^{(i)}|$ and (II) minimum relative support threshold $minRelSup$ for sequential patterns.

Task: rank the discovered sequential patterns with respect to significance.

Note that in our method the main purpose of the support threshold for sequential patterns is to limit the search space of possible significant patterns.

4 Sequence-Wise Reference Model

The sequence-wise reference model treats itemsets as alphabet symbols and represents an independence model where itemsets occur independently of their order in an itemset-sequence. In order to present the sequence-wise reference model we introduce the *element-wise* representation of a collection of itemset-sequences, that is a sequence of itemset-sequences $\mathcal{R} = [r^{(1)}, r^{(2)}, \dots, r^{(|\mathcal{R}|)}]$ over an itemset alphabet $\Xi = \{\xi_1, \xi_2, \dots, \xi_{|\Xi|}\}$, where $r^{(i)}$ is the i -th itemset-sequence and $r_t^{(i)} \in \Xi$ is the element (itemset) at time point t . As an example, Figure 2 presents the element-wise sequence of itemset-sequences for the collection from Table 1, where $\Xi = \{0, 1, 2, 3, (1, 3), (2, 3)\}$ and the itemsets are represented as decimal numbers. Note that for the sequence-wise reference model Ξ is provided by the itemset-wise reference model and includes all non-empty subsets of \mathcal{A} .

4.1 Generative Process

Now consider the sequence-wise reference model as a generative process, that generates itemset-sequences in \mathcal{R} as follows:

1. it first generates the size of the itemset-sequence from a distribution $\alpha = [\alpha_1, \alpha_2, \dots, \alpha_M]$, where α_m is the probability of generating an itemset-sequence of size m and
2. it generates a sequence of itemsets $r^{(i)}$ of size m from distribution $\theta = [\theta_1, \theta_2, \dots, \theta_{|\Xi|}]$, provided by the itemset-wise reference model, where $\theta_j = P(r_t^{(i)} = \xi_j)$, for $\xi_j \in \Xi$.

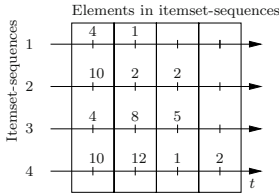


Fig. 2. Element-wise sequence of itemset-sequences representing the collection of itemset-sequences from Table 1. The streams correspond to itemset-sequences, where a decimal number at a given time point corresponds to an itemset (e.g., $10 = 2^{1+3}$ for itemset $(1, 3)$).

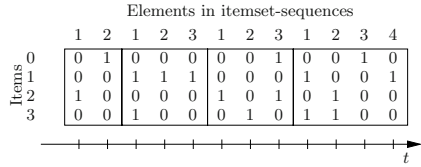


Fig. 3. Item-wise multi-attribute sequence representing the collection from Table 1. The streams correspond to items, every time point corresponds to an itemset, where digit 1 in the i -th stream means a presence of the i -th item.

Let $P(r^{(i)}, m)$ be the joint probability of a particular itemset sequence $r^{(i)}$ of size m to be generated by the process. Then given the independence assumption of the presented generative process we factorize $P(r^{(i)}, m)$ as follows

$$P(r^{(i)}, m) = \alpha_m \cdot P(r^{(i)}|m), \tag{2}$$

where $P(r^{(i)}|m) = \prod_{t=1}^m P(r_t^{(i)}|m)$ is the probability of a particular itemset-sequence $r^{(i)}$ to be generated given the size m .

We compute the parameters of the sequence-wise reference model from \mathbf{S} as follows: (I) $\alpha_m = \frac{N_n(|s^{(i)}|=m)}{n}$ (ML estimator), where $N_n(|s^{(i)}|=m)$ is the number of occurrences of itemset-sequences of size m in \mathbf{S} and (II) θ is computed from the itemset-wise reference model, that is presented in Section 5 and whose purpose is to provide decorrelated frequencies of itemsets. Note that we could compute θ_j as $\theta_j = \frac{N_n(\xi_j)}{n_\Xi}$ (ML estimator), where n_Ξ is the number of itemsets in \mathbf{S} and $N_n(\xi_j)$ is the number of occurrences of itemset ξ_j in \mathbf{S} . However the ML estimator for the itemsets does not provide decorrelated frequencies.

4.2 Relative Frequency

In this section we consider occurrences of a sequential pattern as a *subsequence* (gaps between elements of the sequential pattern are allowed) in the sequence-wise reference model represented by its element-wise representation \mathcal{R} . Let $\overline{\Omega}_n(s)$ be a random variable representing the actual relative frequency of a sequential pattern s occurring as a subsequence in \mathcal{R} . Recall that the relative frequency of a sequential pattern s is equal to the fraction of itemset-sequences in \mathcal{R} that contain it as a subsequence. This means that even if s occurs many times in a given itemset-sequence $s' \in \mathcal{R}$ we count it only as one occurrence. Let $\Omega_n(s) = \sum_{i=1}^n I_i$ be a random variable that represent the actual frequency of s occurring as a subsequence in \mathcal{R} ($sup_{\mathcal{R}}(s)$), where I_i is an indicator function equal to 1 if the i -th itemset-sequence contains s . Then clearly, I_1, I_2, \dots, I_n is

a sequence of independent random variables because occurrences of a pattern as a subsequence in itemset-sequences are independent of each other. Therefore, $\Omega_n(s)$ has the *Binomial distribution* and $\overline{\Omega}_n(s) = \frac{\Omega_n(s)}{n}$ is a *Binomial proportion*, where $\mathbf{E}[\overline{\Omega}_n(s)] = P^\exists(s)$, $\mathbf{Var}[\overline{\Omega}_n(s)] = \frac{1}{n}P^\exists(s)(1-P^\exists(s))$ and $P^\exists(s)$ is the probability that s exists as a subsequence in an itemset-sequence in \mathcal{R} . Thus, clearly $\Omega_n(s)$ and $\overline{\Omega}_n(s)$ both satisfy CLT. However, since itemset-sequences have variable sizes, for a given s , $P^\exists(s)$ depends on the distribution of the sizes of itemset-sequences in \mathcal{R} .

Let $P^\exists(s, m)$ be the joint probability that an itemset-sequence s of size $|s|$ exists as a subsequence in another itemset-sequence s' of size $m \geq |s|$ in \mathcal{R} . Then following (2) we factorize $P^\exists(s, m)$ as follows

$$P^\exists(s, m) = \alpha_m \cdot P^\exists(s|m), \tag{3}$$

where $P^\exists(s|m)$ is the probability that s occurs given an itemset-sequence of size m in \mathcal{R} . In order to obtain the formula for $P^\exists(s)$ we marginalize from (3) as follows:

$$P^\exists(s) = \sum_{m=|s|}^M \alpha_m \cdot P^\exists(s|m). \tag{4}$$

Thus, $P^\exists(s)$ is expressed as a *discrete mixture model*, where the *mixing coefficients* $(\alpha_1, \alpha_2, \dots, \alpha_M)$ model the fact that an occurrence of s as a subsequence in an itemset-sequence s' depends on the size of s' and may possibly occur in any itemset-sequence $s' \in \mathcal{R}$ for which $|s'| \geq |s|$. In other words, $P^\exists(s)$ is a weighted combination of contributions from itemset-sequences of all possible relevant sizes in \mathcal{R} .

Finally, the formula for $P^\exists(s|m)$ for an itemset-sequence $s = [s_1, s_2, \dots, s_{|s|}]$ given an itemset-sequence of size m in \mathcal{R} can be obtained as follows. Let $\mathcal{X}_i = \bigcup_{\xi_j \in \Xi, s_i \subseteq \xi_j} \xi_j$ be the set of all supersets of itemset s_i in itemset alphabet Ξ . Then clearly, the enumeration graph for $\mathcal{W}^\exists(s|m)$ can be obtained from the enumeration graph for a sequence of items $e = [e_1, e_2, \dots, e_m]$ by substituting $\mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_{|s|}$ for items e_1, e_2, \dots, e_m in Figure 1. Then the formula for $P^\exists(s|m)$ can be obtained from (1) by substituting marginal probabilities of itemsets $P_{\mathcal{M}}(s_i) = P(\mathcal{X}_i)$ for probabilities of items in (1). Thus, $P^\exists(s|m)$ in a 0-order Markov sequence-wise reference model, can be obtained from (1) as follows:

$$P^\exists(s|m) = P_{\mathcal{M}}(s) \sum_{i=0}^{m-|s|} \sum_{\sum_{k=1}^{|s|} n_k = i} \prod_{k=1}^{|s|} (1 - P_{\mathcal{M}}(s_k))^{n_k}, \tag{5}$$

where $P_{\mathcal{M}}(s) = \prod_{i=1}^{|s|} P_{\mathcal{M}}(s_i)$ and $P_{\mathcal{M}}(s_i)$ is the marginal probability computed from the itemset-wise reference model. Formula (5) can be computed in $O(m^2)$ using the dynamic programming algorithm given in [6]. Thus computing $P^\exists(s)$ from (4) takes $O(M^3)$ time.

The presented sequence-wise reference model can be easily extended to more application specific models. As a first extension, we could assume that

itemset-sequences are generated using a *Markov model* and use the algorithm for computing $P^{\exists}(s|m)$ from [7] for Markov models. As another extension, we could assume that the distribution of itemsets θ depends on the size of an itemset-sequence (e.g., itemsets having large cardinality are more likely to occur in shorter itemset-sequences).

5 Itemset-Wise Reference Model

The itemset-wise reference model treats itemsets as binary vectors and provides decorrelated frequencies of itemsets. In order to present the itemset-wise reference model we introduce the *item-wise* representation, that is a multi-attribute binary sequence $\mathcal{B} = \{b^{(1)}, b^{(2)}, \dots, b^{(|\mathcal{A}|)}\}$ of size $|\mathcal{A}|$, where: $b^{(i)}$ is a binary sequence corresponding to attribute (item) $a_i \in \mathcal{A}$ and $b_t^{(j)} \in \{0, 1\}$ is the value at time point t . Thus, \mathcal{B} represents \mathbf{S} as a sequence of time ordered itemsets. Figure 3 presents the item-wise multi-attribute sequence for the collection from Table 1.

Note that we do not consider empty itemsets (binary vectors consisting of all zeros in \mathcal{B}) because a lack of attributes is meaningless in our framework. Therefore the streams in \mathcal{B} are inherently dependent, i.e., $P(b_t^{(1)}, b_t^{(2)}, \dots, b_t^{(|\mathcal{A}|)}) \neq \prod_{j=1}^{(|\mathcal{A}|)} P(b_t^{(j)})$ and the independence model is inappropriate in our framework.

Therefore we build a *maximum entropy* model of the form [4]

$$P(b_t^{(1)}, b_t^{(2)}, \dots, b_t^{(|\mathcal{A}|)}) = Z \left(\prod_{i=1}^{|\mathcal{A}|} \mu_i^{I^{(i)}} \right) \mu_c^{|\mathcal{A}| - \sum_{i=0}^{|\mathcal{A}|} I^{(i)}}, \tag{6}$$

where Z is the normalizing constant, $I^{(i)}$ is an indicator function equal to one if $b_t^{(i)} = 1$. We build (6) using *Generalized Iterative Scaling* (GIS) algorithm by finding μ_i for $i = 1, \dots, |\mathcal{A}|$, μ_c and Z under constraints that empirical marginal probabilities of items are preserved, i.e., $\sum P(b_t^{(1)}, b_t^{(2)}, \dots, b_t^{(|\mathcal{A}|)}) I^{(i)} = P(b_t^{(i)} = 1)$, where μ_c is the *correction feature* that ensures that the number of parameters (features) for every binary vector in (6) is constant. Let $Sum = \sum_{b_t^{(i)} \in \{0,1\}, \sum_i b_t^{(i)} > 0} P(b_t^{(1)}, b_t^{(2)}, \dots, b_t^{(|\mathcal{A}|)})$, let $p^{(i)} = P(b_t^{(i)} = 1)$ and let $p_{\mathcal{M}}^{(i)} = Z \cdot u_i \sum_{n_j \in \{0,1\}} \left(\prod_{j \neq i} \mu_j^{n_j} \right) \mu_c^{|\mathcal{A}| - 1 - \sum_{j \neq i} n_j^{(j)}}$ be the marginal probability of attribute i computed from the model given the current estimates of the parameters.

The iterative scaling algorithm proceeds as follows: (I) initialization: $\mu_i = 1$, $\mu_c = 1$, $Z = \frac{1}{Sum}$; and (II) iteration: **repeat for** $i = 1$ **to** $|\mathcal{A}|$ **do begin**
 $\mu_i^{n+1} = \mu_i^n \left(\frac{p^{(i)}}{p_{\mathcal{M}}^{(i)}} \right)^{\frac{1}{|\mathcal{A}|}}$, $\mu_c^{n+1} = \mu_c^n \left(\frac{1}{Z \cdot Sum} \right)^{\frac{1}{|\mathcal{A}|}}$, $Z = \frac{1}{Sum}$ **end until for** $i = 1$ **to** $|\mathcal{A}|$ $\frac{|p^{(i)} - p_{\mathcal{M}}^{(i)}|}{p^{(i)}} < \epsilon$. Thus, the maximum entropy model satisfies our requirements: (I) it preserves empirical marginal probabilities; (II) it is defined only for all

Table 2. Comparison of marginal probabilities of the itemsets of size two and the probability of the empty itemset for the collection from Table 1 obtained using the independence model and the maximum entropy model

| Itemset | Independence model | Maximum Entropy model |
|---------|--------------------|-----------------------|
| (0,2) | 8.33e-02 | 2.35e-01 |
| (1,3) | 1.39e-01 | 3.81e-01 |
| (2,3) | 1.11e-01 | 3.08e-01 |
| () | 1.94e-01 | 0 |

non-empty subsets of \mathcal{A} and (III) it gives as much independence to the attribute streams as possible given the constraints.

Table 2 presents marginal probabilities of the itemsets of size two from Figure 3 obtained using the independence model and the maximum entropy model. Thus, Table 2 shows the following facts: (I) although the empty itemset does not occur in Figure 3 the independence model assigns a bigger probability ($1.94e-01$) to the empty itemset than to the occurring itemsets of size two; and (II) the ME model, as expected, assigns greater probabilities to the occurring itemsets than the independence model.

6 Ranking Algorithm

Given a collection of itemset-sequences $\mathbf{S} = \{s^{(1)}, s^{(2)}, \dots, s^{(n)}\}$, where $s^{(i)} = [s_1^{(i)}, s_2^{(i)}, \dots, s_{|s^{(i)}|}^{(i)}]$, the ranking algorithm proceeds as follows:

1. run *PrefixSpan* for a given value of *minRelSup* to obtain a set of frequent sequential patterns \mathcal{F} .
2. compute $\alpha = [\alpha_1, \alpha_2, \dots, \alpha_M]$, where $\alpha_m = \frac{N_n(|s^{(i)}|=m)}{n}$ and $N_n(|s^{(i)}|=m)$ is the number of itemset-sequences of size m in \mathbf{S} .
3. for every frequent sequential pattern $s = [s_1, s_2, \dots, s_{|s|}]$, where $s \in \mathcal{F}$ and $rsups(s) \geq minRelSup$ do the following:
 - (a) compute the marginal probability vector $[\theta_1, \theta_2, \dots, \theta_{|s|}]$ ($\theta_i = P_{\mathcal{M}}(s_i)$) for elements of s from the itemset-wise reference model.
 - (b) compute $P^\exists(s)$ from (4) and compute the significance rank as follows

$$sigRank(s) = \frac{\sqrt{n}(rsups(s) - P^\exists(s))}{\sqrt{P^\exists(s)(1 - P^\exists(s))}}$$

The reference model will be violated in \mathbf{S} in two cases: (I) the sequence-wise reference model is violated by correlated itemsets and (II) the itemset-wise reference model is violated by correlated items in itemsets.

7 Experiments

In this section we present our experimental results on the multi-stream of news stories of size 224062 stories that have been retrieved, via RSS feeds, from

the following thirteen news sources: ABC news (ABC), Aljazeera (ALJ), Associated Press (AP), British Broadcast Co. (BBC), Canadian Broadcast Co. (CBC), Xinhua News Agency (CHV), Central News Agency Taiwan (CNE), CNN, MSNBC, Reuters (REU), United Press International (UPI), RIA Novosti (RIA) and Deutsche Welle (DW). The stories were retrieved over a period of thirteen months from the 12-th of November 2008 to the 3rd of January 2010. We implemented a clustering algorithm that uses a time-window of a given duration (e.g., 24 hours) and is an incremental variant of a non-hierarchical document clustering algorithm using a similarity measure based on nearest neighbors. We ran the algorithm for the following parameters: (I) the time-window size $w = 24$ hours; (II) the document similarity threshold $\tau_d = 0.5$ that is used to identify nearest neighbors for a new arriving document to the window and (III) the time quantization step size $Q_t = 1$ hour. As a result we obtained a collection of itemset-sequences \mathbf{S} of size $|\mathbf{S}| = 32464$, where there are 109964 itemsets, the maximum itemset-sequence size $M = 25$, the average itemset-sequences size is 3.5 and the average itemset size is 1.2.

7.1 From Clusters to Itemset-Sequences

Let $\mathcal{D} = \{d^{(1)}, d^{(2)}, \dots, d^{(|\mathcal{D}|)}\}$ be a multi-stream of news stories (documents), where $d_t^{(i)}$ is a document in stream i at a *time point* t and has three attributes: (I) the exact publishing timestamp $d_t^{(i)}.timestamp$; (II) stream identifier $d_t^{(i)}.stream = i$; and (III) text content $d_t^{(i)}.content$. The publishing timestamp $d_t^{(i)}.timestamp$ is unique in each stream $d^{(i)}$. Let $\mathcal{C} = [d_1, d_2, \dots, d_{|\mathcal{C}|}]$ be a cluster of documents (reporting the same event in our case) defined as a sequence of documents ordered with respect to publishing timestamp $d_i.timestamp$. We convert \mathcal{C} to an itemset-sequence $s = [s_1, s_2, \dots, s_{|s|}]$, where $s_i \subseteq \mathcal{A}$ and $\mathcal{A} = \{0, 1, \dots, |\mathcal{D}| - 1\}$ is the set of all stream identifiers of the news sources in \mathcal{D} . As a result of the conversion each itemset s_i contains stream identifiers of documents with the same timestamp ($d_i.timestamp$) and the itemset-sequence s is ordered with respect to the timestamps of the itemsets. As an example consider itemset-sequence $[(1, 3), 1, 1]$ in Table 1, where $s_1 = (1, 3)$ means that two documents: the first from source 1 and the second from source 3 were published (within the time granularity Q_t) before a document from streams 1 and 1 respectively. Furthermore, for every itemset-sequence, we recorded content similarity between the stories corresponding to its elements in terms of the *cosine similarity measure*. In order to assess the nature of content similarity between documents in a given itemset-sequence s we define the average content similarity $AvgSim_{\mathbf{S}}(s)$ and the variance of the content similarity $VarSim_{\mathbf{S}}(s)$ between documents in an itemset-sequence s of length l occurring as a subsequence over the whole collection of itemset-sequences \mathbf{S} are expressed as follows

$$AvgSim_{\mathbf{S}}(s) = \frac{2 \cdot \mathit{sup}_{\mathbf{S}}(s)}{l^2 - l} \sum_{s' \in \mathbf{S}, s \sqsubseteq s'} \sum_{k=1}^l \sum_{i=j_k-1}^{j_k-1} \mathit{sim}(s'_i, s'_{j_k}) \quad (7)$$

Table 3. Baseline: Top-20 most frequent sequential patterns of size greater than one

| | Pattern | <i>rsups</i> |
|----|--------------|--------------|
| 1 | [AP, MSNBC] | 1.78e-01 |
| 2 | [MSNBC, UPI] | 1.20e-01 |
| 3 | [BBC, UPI] | 1.06e-01 |
| 4 | [AP, UPI] | 1.06e-01 |
| 5 | [REU, UPI] | 1.06e-01 |
| 6 | [AP, ABC] | 9.03e-02 |
| 7 | [BBC, ALJ] | 8.78e-02 |
| 8 | [REU, BBC] | 8.71e-02 |
| 9 | [CNN, UPI] | 8.56e-02 |
| 10 | [REU, CNE] | 8.55e-02 |
| 11 | [REU, MSNBC] | 8.41e-02 |
| 12 | [BBC, CNE] | 8.15e-02 |
| 13 | [ABC, UPI] | 8.09e-02 |
| 14 | [ABC, MSNBC] | 8.07e-02 |
| 15 | [CNE, UPI] | 7.87e-02 |
| 16 | [AP, REU] | 7.83e-02 |
| 17 | [BBC, REU] | 7.51e-02 |
| 18 | [MSNBC, REU] | 7.49e-02 |
| 19 | [MSNBC, ABC] | 7.20e-02 |
| 20 | [CNE, BBC] | 7.05e-02 |

Table 4. Top-20 most significant sequential patterns for *minRelSup* = 0.01

| | Pattern | <i>sigRank</i> |
|----|----------------------|----------------|
| 1 | [BBC, ALJ, ALJ, ALJ] | 25.5 |
| 2 | [ALJ, ALJ, ALJ, CNE] | 19.1 |
| 3 | [CNE, ALJ, ALJ, ALJ] | 18.6 |
| 4 | [BBC, CNE, ALJ, ALJ] | 18.1 |
| 5 | [ALJ, ALJ, CNE, ALJ] | 17.7 |
| 6 | [CNE, ALJ, ALJ, CNE] | 16.4 |
| 7 | [BBC, ALJ, ALJ, UPI] | 16.3 |
| 8 | [BBC, CNE, ALJ, ALJ] | 16.1 |
| 9 | [AP, MSNBC] | 15.7 |
| 10 | [ALJ, ALJ, BBC, ALJ] | 15.1 |
| 11 | [ALJ, CNE, ALJ, ALJ] | 14.5 |
| 12 | [CNE, BBC, ALJ, ALJ] | 14.2 |
| 13 | [BBC, ALJ, ALJ, BBC] | 14.1 |
| 14 | [ALJ, BBC, ALJ, ALJ] | 13.9 |
| 15 | [BBC, ALJ, ALJ, CNN] | 13.2 |
| 16 | [ALJ, ALJ, CBS] | 13.1 |
| 17 | [ALJ, ALJ, ALJ, UPI] | 12.9 |
| 18 | [REU, ALJ, ALJ, ALJ] | 12.8 |
| 19 | [ALJ, ALJ, ALJ, BBC] | 12.7 |
| 20 | [BBC, ALJ, CNE, ALJ] | 12.4 |

and

$$VarSims(s) = \frac{2 \cdot sups(s)}{l^2 - l} \sum_{s' \in \mathbf{S}, s \sqsubseteq s'} \sum_{k=1}^l \sum_{i=j_1}^{j_k-1} (AvgSims_{\mathbf{S}}(s) - sim(s'_i, s'_k))^2, \quad (8)$$

where $j_1 \leq j_2 \dots \leq j_l$ are the positions where s occurs in s' as a subsequence and $sim(d_i, d_j)$ is the cosine similarity or content similarity between documents i and j . Thus, (7) computes the average content similarity over all itemset-sequences containing s as a subsequence. We also use $StdDevSims(s)$ to denote $\sqrt{VarSims(s)}$.

7.2 Baseline: Most Frequent Patterns

As a baseline against which we compare the performance of the ranking algorithm we use the top-20 most frequent sequential patterns of size greater than one, where we also removed patterns containing the same symbol, which corresponds to frequent updates of the same news event. Table 3 presents the top-20 most frequent sequential patterns of size greater than one.

7.3 Significant Patterns

In the first experiment we rank the top-20 most frequent patterns from Table 3 with respect to significance. Figure 4 presents the results. As it turns out the most frequent pattern in Table 3 is also the most significant one but for the following patterns there is not any obvious relationship between the significance rank and the frequency rank. The dependency between AP and MSNBC can be explained by the fact that as we saw in the recorded stories MSNBC is reusing some content from AP.

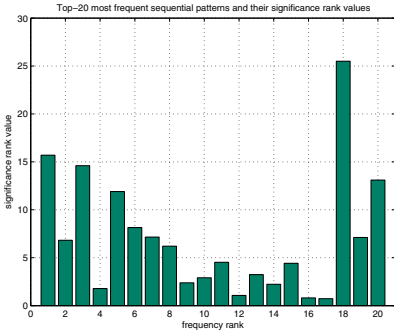


Fig. 4. Frequency rank (x -axis) versus significance rank (y -axis) for the top-20 most frequent sequential patterns from Table 3

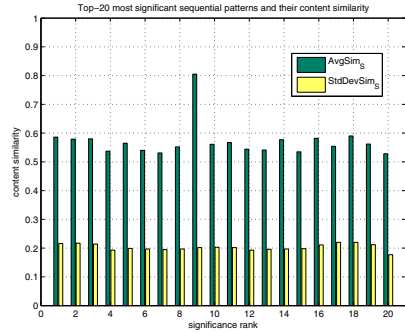


Fig. 5. Significance rank (x -axis) versus content similarity (the average and the standard deviation) (y -axis) for the top-20 most significant sequential patterns from Table 4

In the second experiment we set $minRelSup = 0.01$ and found the most significant over-represented ($sigRank > 0$) sequential patterns. Table 4 presets the top-20 most significant sequential patterns, where among the top patterns we removed patterns containing the same symbol and patterns having significant supersequences. Note however that the top-20 most significant patterns for the whole collection may not be the same since the patterns in Table 4 were obtained using $minRelSup = 0.01$. In general the lower the value of $minRelSup$ the higher the chance that the reference model will discover long significant patterns having low support. By comparing the results from Table 3 and from Table 4 we can make the following observations: (I) the most significant patterns are generally longer than the most frequent ones since the sequence-wise reference model leverages rank of correlated longer patterns and (II) there is a prevalence of patterns involving BBC in the first position and ALJ in the following positions. The dependency between BBC and ALJ may be related to the fact that, as we found out from the BBC web site, BBC signed a news exchange agreement with ALJ and as the pattern suggests this exchange seems to be really “one-way” from BBC to ALJ. Furthermore, ALJ tends to provide many updates of the same news event. Also, although [AP, MSNBC] is the most frequent pattern it has significance rank nine in Table 4 as a result of the reference model leveraging rank of the longer patterns involving BBC and ALJ.

Figure 5 presents a graph of the significance rank (x -axis) versus the average content similarity $AvgSim_S$ and the standard deviation $StdDevSim_S$ (y -axis) for the top-20 most significant sequential patterns from Table 4. Figure 5 shows two facts: (I) the average content similarity is above the document similarity threshold $\tau_d = 0.5$ and (II) the value of the standard deviation is relatively low for all patterns. These results suggest that temporally correlated news streams tend to be also correlated with respect to their content.

8 Conclusions

We presented a reliable general method for ranking frequent sequential patterns (itemset-sequences) with respect to significance. We demonstrated the applicability of the presented method on a multi-stream of news stories that was gathered from RSS feeds of the major world news agencies. In particular we showed that there are strong dependencies between the news sources in terms of temporal sequential patterns of reporting the same news events and content similarity, where the frequency and significance rank are correlated with the content similarity.

References

1. Agrawal, R., Srikant, R.: Mining sequential patterns. In: ICDE, pp. 3–14 (1995)
2. Allan, J.: Topic Detection and Tracking: Event-Based Information Organization. Kluwer Academic Publishers, Norwell (2002)
3. Brin, S., Motwani, R., Silverstein, C.: Beyond market baskets: Generalizing association rules to correlations. In: Proceedings ACM SIGMOD International Conference on Management of Data, May 1997, pp. 265–276 (1997)
4. Darroch, J., Ratcliff, D.: Generalized iterative scaling for log-linear models. *The Annals of Mathematical Statistics* 43(5), 1470–1480 (1972)
5. Guan, E., Chang, X., Wang, Z., Zhou, C.: Mining maximal sequential patterns. In: 2005 International Conference on Neural Networks and Brain, pp. 525–528 (2005)
6. Gwadera, R., Atallah, M., Szpankowski, W.: Reliable detection of episodes in event sequences. In: Third IEEE International Conference on Data Mining, November 2003, pp. 67–74 (2003)
7. Gwadera, R., Atallah, M., Szpankowski, W.: Markov models for discovering significant episodes. In: SIAM International Conference on Data Mining, pp. 404–414 (2005)
8. Gwadera, R., Crestani, F.: Mining and ranking streams of news stories using cross-stream sequential patterns. In: CIKM 2009: Proceedings of the 18th International Conference on Information and Knowledge Management, Hong Kong, October 2009, pp. 1709–1712 (2009)
9. Han, J., Cheng, H., Xin, D., Yan, X.: Frequent pattern mining: current status and future directions. *Data Mining and Knowledge Discovery* 15(1) (2007)
10. Pei, J., Han, J., Mortazavi-Asl, B., Wang, J., Pinto, H., Chen, Q.: Mining sequential patterns by pattern-growth: The prefixspan approach. *TKDE* 16(11) (November 2004)
11. Tatti, N.: Maximum entropy based significance of itemsets. *KAIS* 17(1), 57–77 (2007)
12. Yan, X., Han, J., Afshar, R.: Clospan: Mining closed sequential patterns in large datasets. In: SDM, pp. 166–177 (2003)

Mining Association Rules in Long Sequences

Boris Cule and Bart Goethals

University of Antwerp,
Middelheimlaan 1, 2020 Antwerpen, Belgium
{boris.cule,bart.goethals}@ua.ac.be

Abstract. Discovering interesting patterns in long sequences, and finding confident association rules within them, is a popular area in data mining. Most existing methods define patterns as interesting if they occur frequently enough in a sufficiently cohesive form. Based on these frequent patterns, association rules are mined in the traditional manner. Recently, a new interestingness measure, combining cohesion and frequency of a pattern, has been proposed, and patterns are deemed interesting if encountering one event from the pattern implies with a high probability that the rest of the pattern can be found nearby. It is quite clear that this probability is not necessarily equally high for all the events making up such a pattern, which is why we propose to introduce the concept of association rules into this problem setting. The confidence of such an association rule tells us how far on average from a particular event, or a set of events, one has to look, in order to find the rest of the pattern. In this paper, we present an efficient algorithm to mine such association rules. After applying our method to both synthetic and real-life data, we conclude that it indeed gives intuitive results in a number of applications.

Keywords: association rules, sequences, interesting itemsets.

1 Introduction

Pattern discovery in sequences is a popular data mining task. Typically, the dataset consists of a single event sequence, and the task consists of analysing the order in which events occur, trying to identify correlation among events. In this paper, events are items, and we look for association rules between itemsets. Usually, an itemset is evaluated based on how close to each other its items occur (cohesion), and how often the itemset itself occurs (frequency).

Recently, we proposed to combine cohesion and frequency into a single measure [2], thereby guaranteeing that if we encounter an item from an itemset identified as interesting, there is a high probability that the rest of the itemset can be found nearby. The proposed algorithm suffered from long runtimes, despite the efficient pruning of candidates. We now propose relaxing the pruning function, but making it much easier to compute. As a result, we prune less, but the algorithm runs much faster.

We further propose to introduce the concept of association rules into this setting. We wish to find itemsets that imply the occurrence of other itemsets

nearby. We present an efficient algorithm to mine such rules, taking advantage of two factors that lead to its efficiency — we can mine itemsets and rules in parallel, and we only need to compute the confidence of a simple class of rules, and use them to evaluate all other rules.

2 Related Work

The concept of finding patterns in sequences was first described by Mannila et al. [7]. Patterns are described as episodes, and can be parallel, where the order in which events occur is irrelevant, serial, where events occur in a particular order, or a combination of the two. We limit ourselves to parallel episodes containing no more than one occurrence of any single event type — in other words, itemsets. In this setting, Mannila et al. propose the WINEPI method to detect frequent itemsets. The method slides a window of fixed length over the sequence, and each window containing the itemset counts towards its total frequency, which is defined as the proportion of all windows that contain it. The confidence of an association rule $X \Rightarrow Y$, denoted $c(X \Rightarrow Y)$, is defined as the ratio of the frequency of $X \cup Y$ and the frequency of X . Once the frequent itemsets have been found, rules between them are generated in the traditional manner [1].

Mannila et al. propose another method, MINEPI, to search for frequent itemsets based on their minimal occurrences [7]. Here, however, association rules are of the form $X[win_1] \Rightarrow Y[win_2]$, meaning that if itemset X has a minimal occurrence in a window W_1 of size win_1 , then $X \cup Y$ has a minimal occurrence in a window W_2 of size win_2 that fully contains W_1 . As we plan to develop rules that tell us how likely we are to find some items nearby, we do not wish to use any fixed window lengths, so these are precisely the sort of rules we wish to avoid.

Generating association rules based on a maximum gap constraint, as defined by Garriga [4], was done by Méger and Rigotti [8], but only for serial episodes X and Y , where X is a prefix of Y . Most other related work has been based on the WINEPI definitions, and only attempted to find the same rules (or a representative subset) more efficiently [3,5].

Consider sequence $s = cghefababcdjcgdlcmd$, that will be used as a running example throughout the paper. Assume that the time stamps associated with the items are integers from 1 to 20. This short sequence is enough to demonstrate the unintuitiveness of the WINEPI method for evaluating association rules. We see that for each occurrence of an a , there is a b right next to it. Similarly, for each g , there is a c right next to it. Logically, association rules $a \Rightarrow b$ and $g \Rightarrow c$ should be equally confident (to be precise, their confidence should be equal to 1). WINEPI, with a window size of 2 (the number of possible windows is thus 21, as the first window starts one time stamp before the sequence, and the last one ends one time stamp after the sequence), results in $fr(a) = \frac{4}{21}$, $fr(ab) = \frac{3}{21}$, $fr(g) = \frac{4}{21}$, $fr(cg) = \frac{2}{21}$. and therefore $c(a \Rightarrow b) = 0.75$, $c(g \Rightarrow c) = 0.5$. A larger window always gives similar results, namely $c(g \Rightarrow c) < c(a \Rightarrow b) < 1$.

Due to space restrictions, we only present related work on association rules. A more extensive discussion of related work on discovering patterns in sequences can be found in [2] and [6].

3 Problem Setting

As our work is based on an earlier work [2], we now reproduce some of the necessary definitions and notations that we will use here. An event is a pair (i, t) , consisting of an item and a time stamp, where $i \in I$, the set of all possible items, and $t \in \mathbb{N}$. Two items can never occur at the same time. We denote a sequence of events by \mathcal{S} . For an itemset X , the set of all occurrences of its items is denoted by $N(X) = \{t \mid (i, t) \in \mathcal{S} \text{ and } i \in X\}$. The coverage of X is defined as the probability of encountering an item from X in the sequence, and denoted

$$P(X) = \frac{|N(X)|}{|\mathcal{S}|}.$$

The length of the shortest interval containing itemset X for each time stamp in $N(X)$ is computed as

$$W(X, t) = \min\{t_2 - t_1 + 1 \mid t_1 \leq t \leq t_2 \text{ and } \forall i \in X, \exists (i, t') \in \mathcal{S}, t_1 \leq t' \leq t_2\}.$$

The average length of such shortest intervals is expressed as

$$\overline{W}(X) = \frac{\sum_{t \in N(X)} W(X, t)}{|N(X)|}.$$

The cohesion of X is defined as the ratio of the itemset size and the average length of the shortest intervals that contain it, and denoted

$$C(X) = \frac{|X|}{\overline{W}(X)}.$$

Finally, the interestingness of an itemset X is defined as $I(X) = C(X)P(X)$. Given a user defined threshold *min_int*, X is considered interesting if $I(X)$ exceeds *min_int*. An optional parameter, *max_size*, can be used to limit the output only to itemsets with a size smaller or equal to *max_size*.

We are now ready to define the concept of association rules in this setting. The aim is to generate rules of the form *if X occurs, Y occurs nearby*, where $X \cap Y = \emptyset$ and $X \cup Y$ is an interesting itemset. We denote such a rule by $X \Rightarrow Y$, and we call X the body of the rule and Y the head of the rule. Clearly, the closer Y occurs to X on average, the higher the value of the rule. In other words, to compute the confidence of the rule, we must now use the average length of minimal windows containing $X \cup Y$, but only from the point of view of items making up itemset X . We therefore define this new average as

$$\overline{W}(X, Y) = \frac{\sum_{t \in N(X)} W(X \cup Y, t)}{|N(X)|}.$$

The confidence of a rule can now be defined as

$$c(X \Rightarrow Y) = \frac{|X \cup Y|}{\overline{W}(X, Y)}.$$

A rule $X \Rightarrow Y$ is considered confident if its confidence exceeds a given threshold, *min_conf*.

We now return to our running example. Looking at itemset *cd*, we see that the occurrence of a *c* at time stamp 1 will reduce the value of rule $c \Rightarrow d$, but

not of rule $d \Rightarrow c$. Indeed, we see that $W(cd, 1) = 12$, and the minimal window containing cd for the other three occurrences of c is always of size 3. Therefore, $\overline{W}(c, d) = \frac{21}{4} = 5.25$, and $c(c \Rightarrow d) = \frac{2}{5.25} = 0.38$. Meanwhile, the minimal window containing cd for all occurrences of d is always of size 3. It follows that $\overline{W}(d, c) = \frac{9}{3} = 3$ and $c(d \Rightarrow c) = \frac{2}{3} = 0.67$. We can conclude that while an occurrence of a c does not highly imply finding a d nearby, when we encounter a d we can be reasonably certain that a c will be found nearby. We also note that, according to our definitions, $c(a \Rightarrow b) = 1$ and $c(g \Rightarrow c) = 1$, as desired.

4 Improved Interesting Itemsets Algorithm

The algorithm proposed in [2] and given in Algorithm 1, finds interesting itemsets as defined in Section 3 by going through the search space (a tree) in a depth-first manner, pruning whenever possible. The first call to the algorithm is made with X empty, and Y equal to the set of all items.

Algorithm 1. INIT($\langle X, Y \rangle$) finds interesting itemsets

```

if  $UBI(\langle X, Y \rangle) \geq min\_int$  and  $size(X) \leq max\_size$  then
  if  $Y = \emptyset$  then
    output  $X$ 
  else
    Choose  $a$  in  $Y$ 
    INIT( $\langle X \cup \{a\}, Y \setminus \{a\} \rangle$ )
    INIT( $\langle X, Y \setminus \{a\} \rangle$ )
  end if
end if

```

The algorithm uses the *UBI* pruning function, that returns an upper bound of the interestingness of all itemsets Z such that $X \subseteq Z \subseteq X \cup Y$. If this upper bound is lower than the chosen *min_int*, the subtree rooted at $\langle X, Y \rangle$ can be pruned. The *UBI* function is defined as

$$UBI(\langle X, Y \rangle) = \frac{|N(X \cup Y)|^2 \times |X \cup Y|}{\sum_{t \in N(X)} W(X, t) \times |S|}$$

This pruning function prunes a large number of candidates, but the algorithm still suffers from long runtimes, due to the fact that each time a new itemset X is considered for pruning, $W(X, t)$ needs to be computed for almost each $t \in N(X)$. For large itemsets, this can imply multiple dataset scans just to decide if a single candidate node can be pruned.

We propose a new pruning function in an attempt to balance pruning a large number of candidates with the effort needed for pruning. As the main problem with the original function was computing the exact minimal windows for each candidate, we aim to estimate the length of these windows using a much simpler computation. To do this, we first compute the exact sum of the window lengths for each pair of items, and we then use these sums to come up with a lower bound of the sum of the window lengths for all other candidate nodes.

We first note that

$$\sum_{t \in N(X)} W(X, t) = \sum_{x \in X} \sum_{t \in N(\{x\})} W(X, t).$$

We then note that each window around an item $x \in X$ must be at least as large as the largest such window containing the same item x and any other item $y \in X$. It follows that

$$\sum_{t \in N(\{x\})} W(X, t) \geq \sum_{t \in N(\{x\})} \max_{y \in X \setminus \{x\}} (W(xy, t)).$$

Naturally, it also holds that

$$\sum_{t \in N(\{x\})} W(X, t) \geq \max_{y \in X \setminus \{x\}} (\sum_{t \in N(\{x\})} W(xy, t)).$$

To simplify our notation, from now on we will denote $\sum_{t \in N(\{x\})} W(xy, t)$ by $s(x, y)$. Finally, we see that

$$\sum_{t \in N(X)} W(X, t) \geq \sum_{x \in X} \max_{y \in X \setminus \{x\}} (s(x, y)),$$

giving us a lower bound for the sum of windows containing X around all occurrences of items of X . This gives us a new pruning function:

$$NUBI((X, Y)) = \frac{|N(X \cup Y)|^2 \times |X \cup Y|}{\sum_{x \in X} \max_{y \in X \setminus \{x\}} (s(x, y)) \times |S|}.$$

This new pruning function is easily evaluated, as all it requires is that we store $s(x, y)$, the sum of minimal windows containing x and y over all occurrences of x , for each pair of items (x, y) , so we can look them up when necessary.

The exact windows will still have to be computed for the leaves of the search tree that have not been pruned, but this is unavoidable. Even for the leaves, it pays off to first check the upper bound, and then, only if the upper bound exceeds the threshold, compute the exact interestingness. The new algorithm uses *NUBI* instead of *UBI*, and is the same as the one given in Algorithm [11](#), with

if $I(X) \geq \text{min_int}$ **then output** X

replacing line 3.

Let us now return to our running example, and examine what happens if we encounter node $\langle \{a, b, c\}, \{d, e\} \rangle$ in the search tree. We denote $X = \{a, b, c\}$ and $Y = \{d, e\}$. With the original pruning technique, we would need to compute the exact minimal windows containing X for each occurrence of a, b or c . After a fair amount of work scanning the dataset many times, in all necessary directions, we would come up with the following: $\sum_{t \in N(X)} W(X, t) = 7 + 5 + 4 + 3 + 3 + 3 + 7 + 11 = 43$. The value of the *UBI* pruning function is therefore:

$$\frac{|N(X \cup Y)|^2 \times |X \cup Y|}{\sum_{t \in N(X)} W(X, t) \times |S|} = \frac{144 \times 5}{43 \times 20} = 0.84.$$

Using the new technique, we would first compute $s(x, y)$ for all pairs (x, y) . The relevant pairs are $s(a, b) = 4$, $s(a, c) = 8$, $s(b, a) = 4$, $s(b, c) = 6$, $s(c, a) = 27$, $s(c, b) = 25$. We can now compute the minimal possible sum of windows for each item, giving us

$$\max_{y \in X \setminus \{a\}}(s(a, y)) = 8, \max_{y \in X \setminus \{b\}}(s(b, y)) = 6, \max_{y \in X \setminus \{c\}}(s(c, y)) = 27$$

resulting in a sum of $\sum_{x \in X} \max_{y \in X \setminus \{x\}}(s(x, y)) = 41$. The value of the NUBI pruning function is therefore

$$\frac{|N(X \cup Y)|^2 \times |X \cup Y|}{\sum_{x \in X} \max_{y \in X \setminus \{x\}}(s(x, y)) \times |S|} = 0.88$$

We see that by simply looking up a few precomputed values instead of scanning the dataset a number of times, we get a very good estimate of the sum of the window lengths.

5 Association Rules Algorithm

Unlike the traditional approaches, which need all the frequent itemsets to be generated before the generation of association rules can begin, we are able to generate rules in parallel with the interesting itemsets. When finding an interesting itemset X , we compute the sum of all minimal windows $W(X, t)$ for each $x \in X$ apart, before adding them up into the overall sum needed to compute $I(X)$. With these sums still in memory, we can easily compute the confidence of all association rules of the form $x \Rightarrow X \setminus \{x\}$, with $x \in X$, that can be generated from itemset X . In practice, it is sufficient to limit our computations to rules of precisely this form (i.e., rules where the body consists of a single item). To compute the confidence of all other rules, we first note that

$$\sum_{t \in N(X)} W(X \cup Y, t) = \sum_{x \in X} \sum_{t \in N(\{x\})} W(X \cup Y, t).$$

A trivial mathematical property tells us that

$$\sum_{t \in N(\{x\})} W(X \cup Y, t) = \overline{W}(x, Y \cup X \setminus \{x\})|N(x)|.$$

As a result, we can conclude that

$$\overline{W}(X, Y) = \frac{\sum_{x \in X} \overline{W}(x, Y \cup X \setminus \{x\})|N(x)|}{|N(X)|},$$

which in turn implies that

$$c(X \Rightarrow Y) = \frac{|X \cup Y||N(X)|}{\sum_{x \in X} \overline{W}(x, Y \cup X \setminus \{x\})|N(x)|}.$$

Meanwhile, we can derive that

$$c(x \Rightarrow Y \cup X \setminus \{x\}) = \frac{|X \cup Y|}{\overline{W}(x, Y \cup X \setminus \{x\})},$$

and it follows that

$$c(X \Rightarrow Y) = \frac{|N(X)|}{\sum_{x \in X} \frac{|N(x)|}{c(x \Rightarrow Y \cup X \setminus \{x\})}}. \tag{1}$$

As a result, once we have evaluated all the rules of the form $x \Rightarrow X \setminus \{x\}$, with $x \in X$, we can then evaluate all other rules $Y \Rightarrow X \setminus Y$, with $Y \subset X$, without

Algorithm 2. AR($\langle X, Y \rangle$) finds interesting itemsets and confident association rules within them

```

if  $NUBI(\langle X, Y \rangle) \geq min\_int$  and  $size(X) \leq max\_size$  then
  if  $Y = \emptyset$  then
    if  $I(X) \geq min\_int$  then
      output  $X$ 
      for all  $x \in X$  do
        compute and store  $c(x \Rightarrow X \setminus \{x\})$ 
        if  $c(x \Rightarrow X \setminus \{x\}) \geq min\_conf$  then output  $x \Rightarrow X \setminus \{x\}$ 
      end for
      for all  $Y \subset X$  with  $|Y| \geq 2$  do
        if  $c(Y \Rightarrow X \setminus Y) \geq min\_conf$  then output  $Y \Rightarrow X \setminus Y$ 
      end for
    end if
  else
    Choose  $a$  in  $Y$ 
    AR( $\langle X \cup \{a\}, Y \setminus \{a\} \rangle$ )
    AR( $\langle X, Y \setminus \{a\} \rangle$ )
  end if
end if

```

further dataset scans. The algorithm for finding both interesting itemsets and confident association rules is given in Algorithm 2.

Looking back at our running example, let us compute the confidence of rule $ab \Rightarrow c$. First we compute $c(a \Rightarrow bc) = \frac{3}{4} = 0.75$ and $c(b \Rightarrow ac) = \frac{3}{3.5} = 0.86$. From Eq. 1, it follows that $c(ab \Rightarrow c) = \frac{2 \cdot 4}{0.75 + 0.86} = \frac{4}{5} = 0.8$. It is easy to check that this corresponds to the value as defined in Section 3.

6 Experiments

In our experiments, we aim to show three things:

1. our algorithm for finding interesting itemsets works faster than the one given in 2 and can handle much longer sequences.
2. our algorithm for finding association rules gives meaningful results, without generating spurious output.
3. our algorithm for finding association rules runs very efficiently, even with a confidence threshold set to 0.

To do this, we designed a dataset that allowed us to demonstrate all three claims. To generate a sequence in which some association rules would certainly stand out, we used the Markov chain model given in Table 1. Our dataset consisted of items a , b , c , and 22 other items, randomly distributed whenever the transition table led us to item x . We fine tuned the probabilities in such a way that all items apart from c had approximately the same frequency, while c appeared approximately twice as often. The high probability of transitions from a to b

Table 1. A transition matrix defining a Markov model

| | <i>a</i> | <i>b</i> | <i>c</i> | <i>x</i> |
|----------|----------|----------|----------|----------|
| <i>a</i> | 0 | 0.45 | 0.45 | 0.1 |
| <i>b</i> | 0.45 | 0 | 0.45 | 0.1 |
| <i>c</i> | 0 | 0 | 0.1 | 0.9 |
| <i>x</i> | 0.025 | 0.025 | 0.05 | 0.9 |

and *c*, and from *b* to *a* and *c* should result in rules such as $a \Rightarrow c$ and $b \Rightarrow c$ having a high confidence. However, given that *c* appears more often, sometimes without an *a* or a *b* nearby, we would expect rules such as $c \Rightarrow a$ and $c \Rightarrow b$ to be ranked lower. Later on we show that our algorithm gave the expected results for all these cases.

First, though, let us examine our first claim. We used our Markov model to generate ten sequences of 10 000 items and ran the two algorithms on each sequence, varying *min_int*, at first choosing *max_size* of 4. Figures I(a) and I(c) show the average runtimes and number of evaluated candidate nodes for each algorithm, as well as the actual number of identified interesting itemsets. While our algorithm pruned less (Figure I(c)), it ran much faster, most importantly at the thresholds where the most interesting itemsets are found (see Figure I(b) for a zoomed-in version). Naturally, if *min_int* = 0, the algorithms take equally long, as all itemsets are identified as interesting, and their exact interestingness must be computed. In short, we see that the runtime of the original algorithm is proportional to the number of candidates, while the runtime of our new algorithm is proportional to the number of interesting itemsets.

To support the second claim, we ran our association rules algorithm with both *min_int* and *min_conf* equal to 0. We set *max_size* to 4. We now simply had to check which rules had the highest confidence. In repeated experiments, with various 500 000 items long datasets, the results were very consistent. The most interesting rules were $a \Rightarrow c$ and $b \Rightarrow c$, with a confidence of 0.52. Then followed rules $a \Rightarrow bc$, $b \Rightarrow ac$ and $ab \Rightarrow c$, with a confidence of 0.34. Rules $a \Rightarrow b$ and $b \Rightarrow a$ had a confidence of 0.29, while rules $ac \Rightarrow b$ and $bc \Rightarrow a$ had a confidence of 0.2. Rule $c \Rightarrow ab$ had a confidence of around 0.17, while rules not involving *a*, *b* or *c* had a confidence between 0.13 and 0.17. We can safely conclude that our algorithm gave the expected results.

To confirm this claim, we ran our algorithm on three text datasets: *English*¹, *Italian*² and *Dutch*³. In each text, we considered the letters of the alphabet and the space between words (denoted \sqcup) as items, ignoring all other symbols. In all three languages, rule $q \Rightarrow u$ had a confidence of 1, as *q* is almost always followed by *u* in all these languages. In all three languages, there followed a number of rules with \sqcup in the head, but the body varied. In Italian, a space is often found near *f*, as *f* is mostly found at the beginning of Italian words, while the same is true for *j* in English. Rules involving two letters revealed some patterns inherent

¹ <http://www.gutenberg.org/files/1999/1999.txt>

² <http://www.gutenberg.org/dirs/etext04/7clel10.txt>

³ <http://www.gutenberg.org/files/18066/18066-8.txt>

Table 2. Some interesting rules found in three different languages

| | <i>English</i> | <i>Italian</i> | <i>Dutch</i> |
|---------------|---|---|---|
| two letters | $c(q \Rightarrow u) = 1$ $c(z \Rightarrow i) = 0.61$ $c(v \Rightarrow e) = 0.58$ $c(x \Rightarrow e) = 0.53$ | $c(q \Rightarrow u) = 1$ $c(h \Rightarrow c) = 0.75$ $c(h \Rightarrow e) = 0.51$ $c(z \Rightarrow a) = 0.5$ | $c(q \Rightarrow u) = 1$ $c(j \Rightarrow i) = 0.75$ $c(d \Rightarrow e) = 0.61$ $c(g \Rightarrow e) = 0.57$ |
| three letters | $c(q \Rightarrow eu) = 0.63$ $c(z \Rightarrow ai) = 0.52$ $c(q \Rightarrow nu) = 0.4$ | $c(q \Rightarrow eu) = 0.6$ $c(h \Rightarrow ce) = 0.57$ $c(q \Rightarrow au) = 0.52$ | $c(q \Rightarrow iu) = 1^4$ $c(j \Rightarrow ei) = 0.46$ $c(g \Rightarrow en) = 0.44$ |
| with \sqcup | $c(y \Rightarrow \sqcup) = 0.85$ $c(w \Rightarrow \sqcup) = 0.84$ $c(j \Rightarrow \sqcup) = 0.83$... $c(\sqcup \Rightarrow e) = 0.35$ | $c(q \Rightarrow \sqcup) = 0.84$ $c(f \Rightarrow \sqcup) = 0.7$ $c(a \Rightarrow \sqcup) = 0.7$... $c(\sqcup \Rightarrow a) = 0.39$ | $c(z \Rightarrow \sqcup) = 0.78$ $c(w \Rightarrow \sqcup) = 0.74$ $c(v \Rightarrow \sqcup) = 0.73$... $c(\sqcup \Rightarrow n) = 0.35$ |

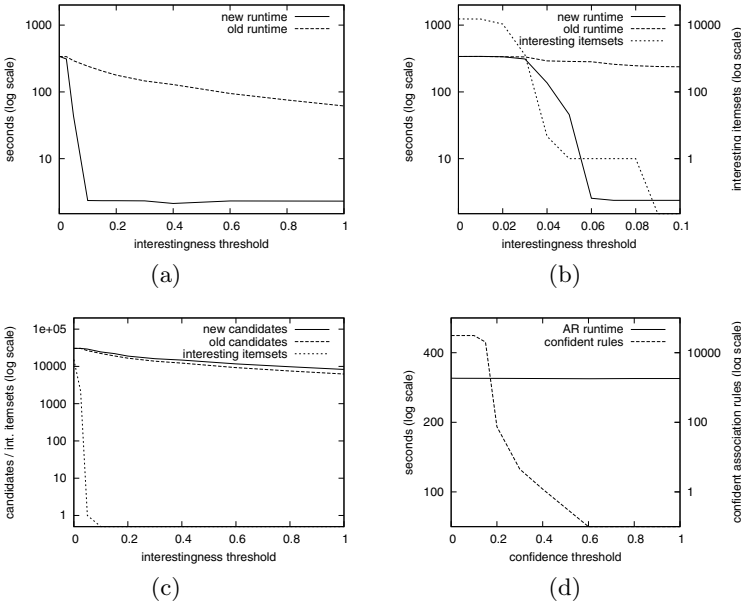


Fig. 1. (a) Runtime comparison of the two itemset algorithms with *max_size* set to 4. (b) Zoomed-in version of Figure 1(a). (c) Pruning comparison with *max_size* set to 4. (d) Runtime of the *AR* algorithm with a varying confidence threshold.

in these languages. For example, rule $h \Rightarrow c$ was ranked high in Italian (where h appears very rarely without a c in front of it), while rule $j \Rightarrow i$ scored very high in Dutch (where j is often preceded by an i). A summary of the results is given in Table 2.

To prove our third claim, we ran the *AR* algorithm on ten Markov chain datasets (each 10 000 items long), using *min_int* of 0.025 and *max_size* of 4, each time varying *min_conf*. The average runtimes and number of found association

⁴ This is due to a short dataset, rather than an actual rule in the Dutch language.

rules are shown in Figure 1(d). We can see that the exponential growth in the number of generated rules had virtually no effect on the runtime of the algorithm.

7 Conclusion

In this paper we presented a new way of identifying association rules in sequences. We base ourselves on interesting itemsets and look for association rules within them. When we encounter a part of an itemset in the sequence, our measure of the rule's confidence tells us how likely we are to find the rest of the itemset nearby.

On our way to discovering association rules, we found a way to improve the runtime of the algorithm that finds the interesting itemsets, too. By relaxing the upper bound of an itemset's interestingness, we actually prune fewer candidates, but our new algorithm runs much faster than the old one. To be precise, the runtime of the new algorithm is proportional to the number of identified itemsets, while the runtime of the old algorithm was proportional to the number of evaluated nodes in the search tree. Due to being able to generate association rules while mining interesting itemsets, the cost of finding confident association rules is negligible.

Experiments demonstrated the validity of our central claims — that our algorithm for finding interesting itemsets runs much faster than the original one, particularly on long datasets, and that our algorithm for finding association rules gives meaningful results very efficiently.

References

1. Agrawal, R., Imielinski, T., Swami, A.: Mining Association Rules between Sets of Items in Large Databases. In: Proc. ACM SIGMOD Int. Conf. on Management of Data, pp. 207–216 (1993)
2. Cule, B., Goethals, B., Robardet, C.: A new constraint for mining sets in sequences. In: Proc. SIAM Int. Conf. on Data Mining (SDM), pp. 317–328 (2009)
3. Das, G., Lin, K.-I., Mannila, H., Renganathan, G., Smyth, P.: Rule discovery from time series. In: Proc. Int. Conf. on Knowledge Discovery and Data Mining (KDD), pp. 16–22 (1998)
4. Garriga, G.C.: Discovering Unbounded Episodes in Sequential Data. In: Lavrač, N., Gamberger, D., Todorovski, L., Blockeel, H. (eds.) PKDD 2003. LNCS (LNAI), vol. 2838, pp. 83–94. Springer, Heidelberg (2003)
5. Harms, S.K., Saquer, J., Tadesse, T.: Discovering Representative Episodal Association Rules from Event Sequences Using Frequent Closed Episode Sets and Event Constraints. In: Proc. IEEE Int. Conf. on Data Mining (ICDM), pp. 603–606 (2001)
6. Laxman, S., Sastry, P.S.: A survey of temporal data mining. In: SADHANA, Academy Proceedings in Engineering Sciences, vol. 31, pp. 173–198 (2006)
7. Mannila, H., Toivonen, H., Verkamo, A.I.: Discovery of Frequent Episodes in Event Sequences. *Data Mining and Knowledge Discovery* 1(3), 259–289 (1997)
8. Méger, N., Rigotti, C.: Constraint-Based Mining of Episodic Rules and Optimal Window Sizes. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) PKDD 2004. LNCS (LNAI), vol. 3202, pp. 313–324. Springer, Heidelberg (2004)

Mining Closed Episodes from Event Sequences Efficiently

Wenzhi Zhou¹, Hongyan Liu¹, and Hong Cheng²

¹ Department of Management Science and Engineering
Tsinghua University, Beijing, China, 100084
{zhouwzh.05, liuhy}@sem.tsinghua.edu.cn

² Department of Systems Engineering and Engineering Management,
The Chinese University of Hong Kong, Shatin, N. T., Hong Kong
hcheng@se.cuhk.edu.hk

Abstract. Recent studies have proposed different methods for mining frequent episodes. In this work, we study the problem of mining closed episodes based on minimal occurrences. We study the properties of minimal occurrences and design effective pruning techniques to prune non-closed episodes. An efficient mining algorithm *Clo_episode* is proposed to mine all closed episodes following a breadth-first search order and integrating the pruning techniques. Experimental results demonstrate the efficiency of our mining algorithm and the compactness of the mining result set.

Keywords: Episode, closed episode, frequent pattern, sequence.

1 Introduction

Frequent episode mining in event sequences is an important mining task with broad applications such as alarm sequence analysis in telecommunication networks [1], financial events and stock trend relationship analysis [4], web access pattern analysis and protein family relationship analysis [5]. An event sequence is a long sequence of events. Each event is described by its type and a time of occurrence. A frequent episode is a frequently recurring short subsequence of the event sequence.

Previous studies on frequent itemset mining and sequential pattern mining show that mining closed itemsets [10] or closed sequential patterns [11] is not only an information lossless compression of all frequent patterns, but also an effective way to improve the mining efficiency. A frequent itemset (or sequence) is closed if there is no super itemset (or super sequence) with the same support. To the best of our knowledge, there are some research work on frequent episode or generalized frequent episode mining [2, 3, 7-9], but there is no existing method for mining closed episodes. Therefore, it is natural to raise two questions: (1) what is a closed episode? and (2) how to find closed episodes efficiently?

Following the definitions of closed itemset and closed sequence, we can define closed episode similarly. For example, in Table 1, as event *B* always co-occurs with event *A*, we say event *B* is not closed (the formal definition is given in Section 2)

given a time span constraint of 4 (i.e., the maximum window size threshold) and a frequency constraint of 2 (i.e., the minimum support threshold).

Table 1. An example sequence of events

| | | | | | | | | | | | | | | | | | | | |
|-------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|
| Time | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| Event | A | B | C | A | B | D | F | E | F | C | E | D | F | A | B | C | E | A | B |

Though the concept of closed episode is similar with that of closed itemset and closed sequence, mining closed episode efficiently is much more challenging. The challenge is caused by two properties in frequent episode mining: (1) there is temporal order among different events in an episode; and (2) we take into account repeated occurrences of an episode in the event sequence.

In this paper, we define a concept of minimal occurrence (a similar concept was defined in [1]) and use minimal occurrence to model the occurrences of an episode in the event sequence. Based on this model, we will develop several pruning techniques to reduce the search space on closed episodes. We propose an efficient algorithm to mine all closed episodes by exploiting the pruning techniques.

The remainder of this paper is organized as follows. We give the problem definition in Section 2. Section 3 presents the novel mining algorithm and our proposed pruning techniques. We demonstrate the performance of our algorithm through extensive experiments in Section 4. Section 5 concludes the paper.

2 Problem Definition

The framework of mining frequent episode was first proposed by Mannila et al. [1]. In [1], two kinds of episodes, serial and parallel episodes, were formally defined. In this paper, we focus on serial episode mining, and we call it episode for simplicity.

Given a set E of event types (or elements), $E = \{e_1, e_2, \dots, e_m\}$, an event sequence (or event stream) S is an ordered sequence of events, denoted by $S = \langle (A_1, t_1), (A_2, t_2), \dots, (A_N, t_N) \rangle$, where $A_i \in E$ is the event that happens at time t_i (we call it the occurrence time of A_i), and $T_s \leq t_i \leq t_{i+1} < T_e$ for $i = 1, 2, \dots, N-1$. T_s is called the starting time of S , and T_e the ending time. Thus an event sequence can be denoted by (S, T_s, T_e) . An **episode** α is an ordered collection of event types, $\alpha = \langle B_1, B_2, \dots, B_n \rangle$, where $B_i \in E$ ($i = 1, 2, \dots, n$) and $n < N$. An episode with n events is called an n -length episode, or n -episode. A **window** on an event sequence (S, T_s, T_e) is an event sequence (w, t_s, t_e) , where $t_s > T_e$ and $t_e < T_s$. Window w consists of those events (A, t) from S where $t_s \leq t < t_e$. The time span $t_e - t_s$ is called the **size** of the window w .

Definition 1 (Minimal occurrence). Given an event sequence S and an episode α , a **minimal occurrence** of α is such a time interval $[t_s, t_e)$ that (1) α occurs in $[t_s, t_e)$; (2) there exists no smaller time interval $[t'_s, t'_e) \subset [t_s, t_e)$ that α occurs in it. t_s is the starting time of this occurrence and t_e the ending time; and (3) $t_e - t_s < win$, where win is a user-defined maximum window size threshold, and $t_e - t_s$ is called the **time span** of this minimal occurrence. That is, each occurrence of an episode must be within a window no larger than win .

For an episode α , all of its minimal occurrences are denoted as a set $MO(\alpha) = \{[t_s, t_e] \mid [t_s, t_e] \text{ is a minimal occurrence of } \alpha\}$. The **support** of an episode is the number of its minimal occurrences, i.e., $sup(\alpha) = |MO(\alpha)|$. Given a minimum support threshold $minsup$, if $sup(\alpha) \geq minsup$, α is called a frequent episode, or α is frequent.

Definition 2 (Sub-episode and super episode). An episode $\beta = \langle B_1, B_2, \dots, B_m \rangle$ where $m < n$ is called a **sub-episode** of another episode $\alpha = \langle A_1, A_2, \dots, A_n \rangle$, if there are m integers $1 \leq i_1 < i_2 < \dots < i_m \leq n$ such that $B_1 = A_{i_1}, B_2 = A_{i_2}, \dots, B_m = A_{i_m}$. Episode α is called the **super episode** of episode β . If we have $i_1 = 1, i_2 = 2, \dots, i_m = m$, then α is called the **forward-extension super episode** of β . If we have $i_1 = n - m + 1, i_2 = n - m + 2, \dots, i_m = n$, then α is called the **backward-extension super episode** of β . If α is a super episode of β but is neither a forward-extension nor backward-extension super episode of β , α is called the **middle-extension super episode** of β .

Definition 3 (Closed episode). Given an event sequence S , an episode α is closed if (1) α is frequent; and (2) there exists no super episode $\beta \supset \alpha$ with the same support as it.

Definition 4 (Forward-closed episode). Episode α is **forward-closed** if (1) α is frequent; and (2) there is no forward-extension super episode of α with the same support as α . Similarly we can define another two kinds of closed episodes.

Definition 5 (Backward-closed episode). Episode α is **backward-closed** if (1) α is frequent; and (2) there is no backward-extension super episode of α with the same support as α .

Definition 6 (Middle-closed episode). Episode α is **middle-closed** if (1) α is frequent; and (2) there is no middle-extension super episode of α with the same support as α .

A closed episode should be forward-closed, backward-closed and middle-closed simultaneously.

Mining Task: Given an event sequence S , a minimum support threshold $minsup$, and a maximum window size threshold win , the mining task is to find the complete set of closed episodes from S .

3 Mining Closed Episodes

To enumerate potential closed episodes, we perform a search in a prefix tree as shown in Figure 1. We assume that there is a predefined order, denoted by \langle , among the set of distinct event types. In our example sequence, we use the lexicographical order, i.e., $A \langle B \langle C \langle D \langle E \langle F$.

The mining process follows a breadth-first search order. Starting from the root node with the empty episode $\alpha = \emptyset$, we can generate length-1 episodes at level 1 by adding one event to α . Similarly we can grow an episode at level k by adding one event to get length- $(k+1)$ episodes at level $k+1$. For each candidate episode, we compute its minimal occurrence set from the minimal occurrence sets of its sub-episodes through a “join” operation. Since the episode mining and checking are based on minimal occurrences of episodes, we will first give some properties of minimal occurrences and show how they can be used for pruning search space.

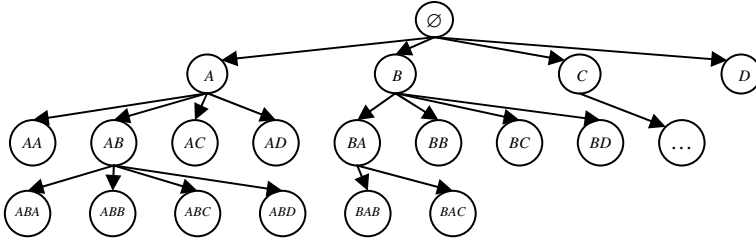


Fig. 1. A tree of episode (sequence) enumeration

Property 1. In an event sequence S , if $[t_s, t_e)$ is a minimal occurrence of episode $\alpha = \langle A_1, A_2, \dots, A_n \rangle$, then there must exist two events, (A_1, t_s) and $(A_n, t_e - 1)$ in S .

Property 2. Assume $[t_s, t_e)$ and $[u_s, u_e)$ are two minimal occurrences of episode α . If $t_s < u_s$, then we have $t_e < u_e$; if $t_e < u_e$, then we have $t_s < u_s$.

Based on Property 2, minimal occurrences in $MO(\alpha)$ can be sorted in the ascending order of their starting time. Then the order among an episode's minimal occurrences is strict. If one occurrence starts ahead of another, then it must end earlier.

According to Definition 1, the time span of an episode's minimal occurrence is bounded by the window size threshold win . Therefore, given a minimal occurrence $[t_s, t_e)$, if $t_e - t_s = win$, the episode cannot be extended forward or backward in the time window $[t_s, t_s + win)$, although some events might be inserted in the middle of it. We define such minimal occurrences as *saturated* minimal occurrences.

Definition 7 (Saturation and expansion). A minimal occurrence $[t_s, t_e)$ of an episode α is **saturated** if the time span $t_e - t_s = win$, the user-specified maximum window size threshold. Otherwise, it is an unsaturated minimal occurrence. An episode α 's **saturation**, denoted as $\alpha.saturation$, is defined as the number of saturated minimal occurrences, and its **expansion**, denoted as $\alpha.expansion$, is the number of unsaturated minimal occurrences.

Apparently, $\alpha.saturation + \alpha.expansion = sup(\alpha)$ holds. For a saturated minimal occurrence of an episode, no additional events can be inserted before the first event or after the last event; otherwise, the maximum window size constraint will be violated. This means, for an episode α we can find its *forward-extension* or *backward-extension* super episode γ only in the unsaturated minimal occurrences. The upper bound of γ 's support is the number of α 's unsaturated minimal occurrences. Therefore, if γ is frequent, the number of unsaturated minimal occurrences in $MO(\alpha)$ must be no smaller than $minsup$. If one episode has less than $minsup$ unsaturated minimal occurrences, then none of its *forward-extension* or *backward-extension* super episodes is frequent. In this case, it is unnecessary to generate and check its super episodes. Therefore, we have following two lemmas.

Lemma 1: If the expansion of an episode α is smaller than the minimum frequency threshold $minsup$, i.e., $\alpha.expansion < minsup$, then all forward-extension and backward-extension super episodes of α are infrequent.

Lemma 2: Given a frequent episode α and its minimal occurrence set $MO(\alpha)$, if (1) $\alpha.saturation > 0$, and (2) there is a saturated minimal occurrence $\omega \in MO(\alpha)$ that cannot be extended in the middle, then α must be closed.

The algorithm, *Clo_episode*, is developed to find all closed episodes in an event sequence. Its major steps are given in Figure 2.

The algorithm structure is as follows. It first generates the set of length-1 frequent episodes F_1 (lines 1-4). The event sequence S is scanned and all distinct single events are generated as length-1 episodes with their minimal occurrences. Then, *Clo_episode* generates closed episodes level by level in an iterative way through the *while* loop (lines 5-19).

Each iteration in the *while* loop generates the set of candidate episodes C_k and the set of frequent episodes F_k . This process iterates until F_k is empty. For a non-empty set F_k , it produces F_{k+1} which is the frequent episode set of length-($k+1$) and CF_k which is the set of length- k closed episodes. Episodes in F_k are processed in a predefined order as we explained above.

| Algorithm <i>Clo_episode</i> | |
|---|---|
| Input: Event sequences S , maximum window size threshold win , minimum support threshold $minsup$; | |
| Output: the set of closed episodes CF | |
| Method: | |
| 1. | $C_1 =$ the set of length-1 episodes in S |
| 2. | scan S and compute $MO(\alpha)$ for all $\alpha \in C_1$; |
| 3. | $k=1$; |
| 4. | $F_1 = \{\alpha \in C_1 \mid \alpha \text{ is frequent}\}$ |
| 5. | while $F_k \neq \Phi$ do |
| 6. | for each episode $\alpha \in F_k$ such that $\alpha.forward \neq 0$ do |
| 7. | $C_{k+1} = Gen_candidate(\alpha, F_k)$; |
| 8. | $C_{k+1} = Prune(C_{k+1})$; |
| 9. | $C_{k+1} = Gen_MO(C_{k+1})$; |
| 10. | for each candidate episode $\gamma \in C_{k+1}$ do |
| 11. | if $sup(\gamma) \geq minsup$, put γ into F_{k+1} |
| 12. | if there is length- k subepisode β of γ such that $\beta.closed = -1$ and $sup(\gamma) = sup(\beta)$ do |
| 13. | set $\beta.closed = 0$ |
| 14. | if γ is a middle-extension super episode of β , call $Clo_prune(\alpha, \beta, \gamma)$; |
| 15. | end for |
| 16. | end for |
| 17. | $CF_k = \{\alpha \in F_k \mid \alpha.closed \neq 0\}$ |
| 18. | $k=k+1$; |
| 19. | end while |
| 20. | $CF = \{\alpha \mid \alpha \in CF_i (1 \leq i < k) \text{ and there is no super episode } \beta \text{ of } \alpha \text{ such that } sup(\beta) = sup(\alpha)\}$; |
| | Output CF |

Fig. 2. Major steps of Algorithm *Clo_episode*

For each episode $\alpha \in F_k$ which can be extended forward (line 6), function $Gen_candidate(\alpha, F_k)$ is called to generate α 's candidate super episodes of length-($k+1$) (line 7). A super episode is generated by combining α and another length- k episode in F_k that shares the first ($k-1$) events with α . Then function $Prune(C_{k+1})$ is called to prune those candidates that cannot be frequent (line 8). Next $Gen_MO(C_{k+1})$ is invoked to generate the minimal occurrences for each candidate episode. It also computes saturation and expansion of

each candidate episode and checks whether they can be extended forward or backward and whether they are closed. Due to space limitation, we omit the details of the two functions. Based on the computed information, each frequent episode γ is added into F_{k+1} (line 11); the sub-episodes of γ that have the same support as γ will be marked not closed (lines 12-13). For those sub-episodes β of γ that share the first and last events with γ , function $Clo_prune(\alpha, \beta, \gamma)$ is invoked to see if it can be pruned (line 14). The details of function Clo_prune are given in Figure 3.

```

Clo_prune( $\alpha, \beta, \gamma$ )
1 Sort minimal occurrences in  $MO(\beta)$  and  $MO(\gamma)$  respectively by starting time ascendingly
2 Initialize  $flag=true$ 
3 for  $i=1$  to  $|MO(\beta)|$  do
4   if  $MO(\gamma)[i].t_s \neq MO(\beta)[i].t_s$  or  $MO(\gamma)[i].t_e \neq MO(\beta)[i].t_e$  do
5      $flag=false$ 
6     break
7 end for
8 if  $flag=true$  do
9   Remove  $\beta$  from  $F_k$ ;
10  if ( $\beta < \alpha$ ) remove all  $\beta$ 's forward-extension super episodes from  $F_{k+1}$ ;
11 return

```

Fig. 3. Major steps of Function Clo_prune

Function Clo_prune compares the minimal occurrences of episode β and that of its middle-extension super episode γ . If $MO(\gamma)=MO(\beta)$, then β is not closed. All of β 's forward-extension super episodes are not closed either. Due to space limitation, we omit the detail of proof.

4 Experiments

In this section we will present experimental results on synthetic datasets. We also conducted experiments on a real dataset. But due to space limitation, we will only report the results of synthetic datasets. We evaluate the efficiency of our proposed algorithm as well as the reduction in the number of episodes generated, comparing with *MINEPI* proposed by Mannila et al. [1] for frequent episode mining. To test the effectiveness of the pruning techniques for pruning non-closed episodes, we disable the function Clo_prune and get another algorithm called $Clo_episode_NP$ without the pruning techniques. All of our experiments were performed on a PC with 3Ghz CPU, 2GB RAM, and 300GB hard disk running windows XP.

We design a synthetic dataset generator, by which we can evaluate the performance of different algorithms with different data characteristics. The generator takes five parameters. The parameter *NumOfPatterns* is the number of potentially closed episodes in the final sequence, *MaxLength* and *MinLength* are the maximum and minimum length of potential episodes respectively, *NumOfWindows* is used to control the length of the whole sequence, and *win* is the size of a window. Due to space limitation, we omit the detail of the generator.

To generate event sequences, we fix the value of *MinLength*, and vary the other five parameters: *NumOfPatterns(P)*, *MaxLength(L)*, *NumOfWindows(N)*, *win(W)* and *minsup*. Thus we create five groups of datasets, each of which is generated by varying one parameter and fixing the other four parameters as shown in Table 2.

Table 2. Five groups of synthetic datasets

| Group | Datasets | <i>P</i> | <i>L</i> | <i>N</i> | <i>W</i> | <i>Minsup</i> |
|-------|-----------------|----------|----------|-----------|----------|---------------|
| 1 | PxL12N4000W14 | 100-200 | 12 | 4000 | 14 | 15 |
| 2 | P200L12NxW14 | 200 | 12 | 3200-4800 | 14 | 20 |
| 3 | P200LxN4000W16 | 200 | 9-14 | 4000 | 16 | 15 |
| 4 | P200L12N4000Wx | 200 | 12 | 4000 | 10-16 | 20 |
| 5 | P200L12N4000W14 | 200 | 12 | 4000 | 14 | 10-35 |

For each group of datasets, we plot the running time and the number of frequent/closed episodes in two figures. The results are shown in Figures 4 to 13. The y-axes are in logarithmic scales in Figures 5, 7, 9, 11 and 13.

From Figure 4 we can see that *Clo_episode* significantly outperforms *MINEPI* and *Clo_episode_NP*. This shows that the pruning techniques in *Clo_episode* are very effective. In addition, the running time of *Clo_episode* is not affected much as *P* increases, but that of the other two algorithms increases dramatically. Figure 5 shows that the number of closed episodes found by *Clo_episode* is much smaller than the number of frequent episodes output by *MINEPI*.

Figures 6 and 7 show a similar result. As *N* increases, the sequence becomes longer. For a fixed *minsup* threshold, the number of frequent episodes by *MINEPI* increases a lot, but the number of closed episodes does not increase as much.

Figures 8 and 9 show that the number of frequent episodes output by *MINEPI* increases significantly with *L*. Accordingly the running time increases. Figure 8 shows that *Clo_episode* is much more efficient than *MINEPI* and *Clo_episode_NP*.

Figures 10 and 11 show that when *win* increases from 10 to 14, the running time becomes longer, the number of frequent episodes becomes larger, but the number of closed episodes does not increase much. When *win* is greater than 14, the running time of all three algorithms does not increase. This is because the average length of potential closed episodes is fixed at 12 in the sequence.

Figures 12 and 13 show that, as *minsup* increases, the number of frequent and closed episodes decreases. The running time decreases accordingly.

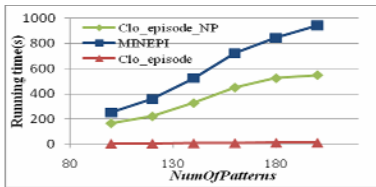


Fig. 4. Running time vs *P*

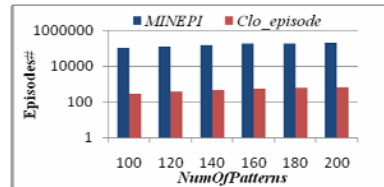
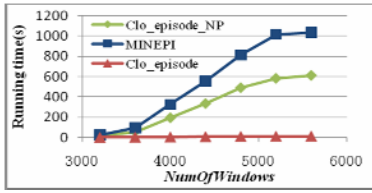
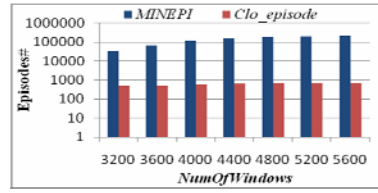
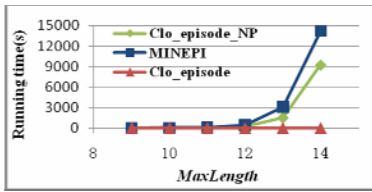
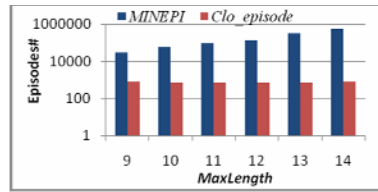
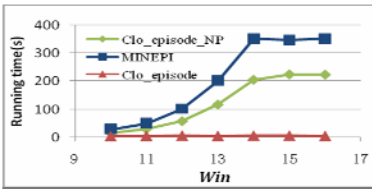
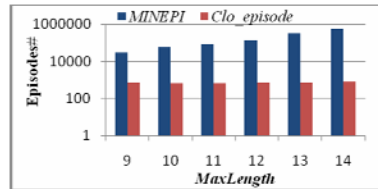
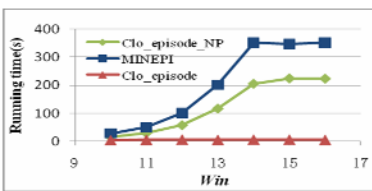
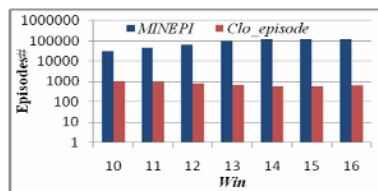


Fig. 5. No. frequent/closed episodes vs *P*

Fig. 6. Running time vs N Fig. 7. No. frequent/closed episodes vs N Fig. 8. Running time vs L Fig. 9. No. frequent/closed episodes vs L Fig. 10. Running time vs W Fig. 11. No. frequent/closed episodes vs W Fig. 12. Running time vs $minsup$ Fig. 13. No. frequent/closed episodes vs $minsup$

Overall, from these five groups of experiments we can see that our algorithm *Clo_episode* performs much better than *MINEPI* and *Clo_episode_NP*.

5 Conclusion

Frequent episode mining is an important mining task in data mining. In this paper, we study how to mine closed episodes efficiently. We proposed a novel algorithm

Clo_episode, which incorporates several effective pruning strategies and a minimal occurrence-based support counting method. Experiments demonstrate the effectiveness and efficiency of these methods.

Acknowledgments. This work was supported in part by the National Natural Science Foundation of China under Grant No. 70871068, 70621061 and 70890083.

References

- [1] Mannila, H., Toivonen, H., Verkamo, A.I.: Discovery of Frequent Episodes in Event Sequences. *Data Mining and Knowledge Discovery* 1(3), 259–289 (1997)
- [2] Mannila, H., Toivonen, H.: Discovering generalized episodes using minimal occurrences. In: *KDD 1996*, Portland, OR (1996)
- [3] Laxman, S., Sastry, P.S., Unnikrishnan, K.P.: A Fast Algorithm For Finding Frequent Episodes In Event Streams. In: *KDD 2007*, SanJose, CA (2007)
- [4] Ng, A., Fu, A.: Mining Frequent Episodes for Relating Financial Events and Stock Trend. In: *PAKDD 2003*, Seoul, Korea (2003)
- [5] Baixeries, J., Casas-Garriga, G., Balcázar, J.L.: Mining unbounded episodes from sequential data
- [6] Meger, N., Rigotti, C.: Constraint-based mining of episode rules and optimal window sizes. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) *PKDD 2004*. LNCS (LNAI), vol. 3202, pp. 313–324. Springer, Heidelberg (2004)
- [7] Gwadera, R., Atallah, M.J., Szpankowski, W.: Reliable detection of episodes in event sequences. In: *ICDM 2003*, Melbourne, FL (2003)
- [8] Gwadera, R., Atallah, M.J., Szpankowski, W.: Markov models for identification of significant episodes. In: *SDM 2005*, Newport Beach, CA (2005)
- [9] Laxman, S., Sastry, P.S., Unnikrishnan, K.P.: Discovering frequent episodes and learning Hidden Markov Models: A formal connection. *IEEE TKDE* 17(11), 1505–1517 (2005)
- [10] Wang, J., Han, J., Pei, J.: CLOSET+: Searching for the Best Strategies for Mining Frequent Closed Itemsets. In: *KDD 2003*, Washington, DC (2003)
- [11] Yan, X., Han, J., Afshar, R.: CloSpan: Mining Closed Sequential Patterns in Large Databases. In: *SDM 2003*, San Francisco, CA (2003)

Most Significant Substring Mining Based on Chi-square Measure

Sourav Dutta and Arnab Bhattacharya

Department of Computer Science and Engineering, Indian Institute of Technology,
Kanpur, India
{sdutta,arnabb}@iitk.ac.in

Abstract. Given the vast reservoirs of sequence data stored worldwide, efficient mining of string databases such as intrusion detection systems, player statistics, texts, proteins, etc. has emerged as a great challenge. Searching for an unusual pattern within long strings of data has emerged as a requirement for diverse applications. Given a string, the problem then is to identify the substrings that differs the most from the expected or normal behavior, i.e., the substrings that are statistically significant (i.e., less likely to occur due to chance alone). To this end, we use the chi-square measure and propose two heuristics for retrieving the top-k substrings with the largest chi-square measure. We show that the algorithms outperform other competing algorithms in the runtime, while maintaining a high approximation ratio of more than 0.96.

1 Motivation

A recent attractive area of research has been detecting statistically relevant sequences or mining interesting patterns from a given string [1,2]. Given an input string composed of symbols from an alphabet set with a probability distribution defining the chance of occurrence of the symbols we would like to find the portions of the string which deviate from the expected behavior and can thus be potent sources of study for hidden pattern and information. An automated monitoring system such as a cluster of sensors sensing the temperature of the surrounding environment for fire alert, or a connection server sniffing the network for possible intrusion detection provides a few of the applications where such pattern detection is essential. Other applications involve text analysis of e-mails and blogs to predict terrorist activities or judging prevalent public sentiments and studying trends of the stock market. Similarly, another interesting field of application can be the identification of good and bad career patches of a sports icon. For example, given the runs scored by Sachin Tendulkar in each innings of his one-day international cricket career, we may be interested in finding his in-form and off-form patches.

A statistical model is used to determine the relationship of an experimental or observed outcome with the factors affecting the system, or to establish the occurrence as pure chance. An observation is said to be statistically significant

if its presence cannot be attributed to randomness alone. The degree of uniqueness of a pattern can be captured by several measures including the p-value and z-score [3,4]. For evaluating the significance of a substring, it has been shown that the p-value provides a more precise conclusion as compared to that by the z-score [1]. However, computing the p-value entails enumerating all the possible outcomes, which can be exponential in number, thus rendering it impractical. So, heuristics based on branch-and-bound techniques have been proposed [5]. The *log-likelihood ratio* G^2 provides such a measure based on the extent of deviation of the substring from its expected nature [6]. For multinomial models, the χ^2 statistic approximates the importance of a string more closely than the G^2 statistic [6,7]. Existing systems for intrusion detection use multivariate process control techniques such as Hotelling’s T^2 measure [8], which is again computationally intensive. The chi-square measure, on the other hand, provides an easy way to closely approximate the p-value of a sequence [6]. To simplify computations, the χ^2 measure, unlike Hotelling’s method, does not consider multiple variable relationship, but is as effective in identifying “abnormal” patterns [2]. Thus, in this paper, we use the Pearson’s χ^2 statistic as a measure of the p-value of a substring [6,7]. The χ^2 distribution is characterized by the *degrees of freedom*, which in the case of a string, is the size of the alphabet set minus one. The larger the χ^2 value of a string, the smaller is its p-value, and hence more is its deviation from the expected behavior.

Formally, given a string S composed of symbols from the alphabet set Σ with a given probability distribution P modeling the chance of occurrence of each symbol, the problem is to identify and extract the top- k substrings having the maximum chi-square value or the largest deviation within the framework of p-value measure for the given probability distribution of the symbols. Naïvely we can compute the χ^2 value of all the substrings present in S and determine the top- k substrings in $O(l^2)$ time for a string of length l . We propose to extract such substrings more efficiently.

Related Work: The blocking algorithm and its heap variant [9] reduce the practical running time for finding such statistically important substrings, but suffers from a high worst-case running time. The number of blocks found by this strategy increases with the size of the alphabet set and also when the probabilities of the occurrence of the symbols are nearly similar. In such scenarios, the number of blocks formed can be almost equal to the length of the given string, thereby degenerating the algorithm to that of the naïve one. The heap variant requires a high storage space for maintaining the separate *max* and *min* heap structures and also manipulates a large number of pointers. Further, the algorithm cannot handle top- k queries. In time-series databases, categorizing a pattern as surprising based on its frequency of occurrence and mining it efficiently using suffix trees has been proposed in [10]. However, the χ^2 measure, as discussed earlier, seems to provide a better parameter for judging whether a pattern is indeed interesting.

In this paper, we propose two algorithms, *All-Pair Refined Local Maxima Search* (ARLM) and *Approximate Greedy Maximum Maxima Search* (AGMM)

to efficiently search and identify interesting patterns within a string. We show that the running time of the algorithms are better than the existing algorithms with lesser space requirements. The procedures can also be easily extended to work in streaming environments. ARLM, a quadratic algorithm in the number of local maxima found in the input string, and AGMM, a linear time algorithm, both use the presence of local maxima in the string. We show that the approximation ratio of the reported results to the optimal is 0.96 or more. Empirical results emphasize that the algorithms work efficiently.

The outline of the paper is as follows: Section 2 formulates the properties and behavior of strings under the χ^2 measure. Section 3 describes the two proposed algorithms. Section 4 discusses the experimental results, before Section 5 concludes the paper.

2 Definition and Properties

Let $S = s_1s_2 \dots s_l$ be a given string of length l composed of symbols s_i from the alphabet set $\Sigma = \{\sigma_1, \sigma_2, \dots, \sigma_m\}$, where $|\Sigma| = m$. To each symbol $\sigma_i \in \Sigma$ is associated a p_{σ_i} (henceforth represented as p_i), denoting the probability of occurrence of that symbol, such that $\sum_{i=1}^m p_i = 1$. Let $\theta_{\sigma_i, S}$ (henceforth represented as $\theta_{i, S}$) denote the observed number of symbol σ_i in string S .

The chi-square value of a string $S \in \Sigma^*$ of length l is computed as

$$\chi_S^2 = \sum_{i=1}^m \frac{(p_i l - \theta_{i, S})^2}{p_i l} \tag{1}$$

We now state certain observations and lemmas, the formal proofs of which are in the full version of the paper [11].

Observation 1. *Under string concatenation operation (\cdot), for two arbitrary strings S and T drawn from the same alphabet set and probability distribution of the symbols (henceforth referred to as the same universe), the χ^2 measure of the concatenated string is commutative, i.e., $\chi_{S \cdot T}^2 = \chi_{T \cdot S}^2$.*

Lemma 1. *The χ^2 value of the concatenation of two strings drawn from the same universe is less than or equal to the sum of the χ^2 values of the individual strings.*

Lemma 2. *The χ^2 value of a string composed of only a single type of symbol increases with the length of the string.*

We next define the term *local maxima* and describe the procedure for finding such a local maxima within a given string.

Definition 1 (Local maxima). *The local maxima is a substring, such that while traversing through it, the inclusion of the next symbol does not decrease the χ^2 value of the resultant sequence.*

Let $s_1 s_2 \dots s_n$ be a local maxima of length n . Then the following holds: $\chi_{s_1 s_2}^2 \geq \chi_{s_1}^2, \dots, \chi_{s_1 s_2 \dots s_n}^2 \geq \chi_{s_1 s_2 \dots s_{n-1}}^2$ and $\chi_{s_1 s_2 \dots s_{n+1}}^2 \leq \chi_{s_1 s_2 \dots s_n}^2$.

The process of finding the local maxima involves a single scan of the entire string. We consider the first local maxima to start at the beginning of the string. We keep appending the next symbol to the current substring until there is a decrease in the chi-square value of the new substring. The present substring is then considered to be a local maxima ending at the previous position. The last symbol appended that decreased the chi-square value signifies the start of the next local maxima. Thus, the running time is $O(l)$ time for a string of length l .

Lemma 3. *The expected number of local maxima in a string of length l is $O(l)$.*

However, practically the number of local maxima is less than l , as all adjacent positions of dissimilar symbols may not correspond to a local maxima boundary. We further optimize the local maxima finding procedure by initially *blocking* the string S , as described in [9], and then searching for the local maxima. A contiguous sequence of the same symbol is considered to be a block, and is replaced by a single instance of that symbol representing the block. If a symbol is selected, the entire block associated with it is considered to be selected. The next lemma shows that a local maxima cannot end in the middle of a block.

Lemma 4. *If the insertion of a symbol of a block increases the chi-squared value of the current substring, then the chi-squared value will be maximized if the entire block is inserted.*

3 Algorithms

Based on the observations, lemmas and local maxima extracting procedure discussed previously, in this section we explain the *All-Pair Refined Local Maxima* (ARLM) and *Approximate Greedy Maximum Maxima* (AGMM) search algorithms for mining the most significant substring based on the chi-square value.

3.1 All-Pair Refined Local Maxima Search (ARLM)

Given a string S of length l and composed of symbols from the alphabet set Σ , we first extract all the local maxima present in it in linear time, as described earlier. With S partitioned into its local maxima, the global maxima can either start from the beginning of a local maxima or from a position within it. Thus, it can contain an entire local maxima, a suffix of it or itself be a substring of a local maxima. It is thus intuitive that the global maxima should begin at a position such that the subsequent sequence of characters offer the maximum chi-square value. Otherwise, we could keep adding to or deleting symbols from the front of such a substring and will still be able to increase its χ^2 value. Based on this, the ARLM heuristic finds within each local maxima the suffix having the maximum

chi-square value, and considers the position of the suffix as a potential starting point for the global maxima.

Let xyz be a local maxima, where x is a prefix, y is a single symbol at position ψ , and z is the remaining suffix. For a local maxima the chi-square value of all its suffixes is computed. The starting position of the suffix having the maximum chi-square value provides the position ψ for the component y , i.e, yz will be the suffix of xyz having the maximum chi-square value.

For each local maxima, the position ψ is inserted into a list α . If no such proper suffix exists for the local maxima, the starting position of the local maxima xyz is inserted in the list. After populating α with position entries of y for each of the local maxima of the input string, the list contains the prospective positions from where the global maxima may start.

The string S is now reversed and the same algorithm is re-run. This time, the β list is similarly filled with positions y' relative to the beginning of the string.

For simplicity and efficiency of operations, we maintain a table C having m rows and l columns, where m is the cardinality of the alphabet set. The rows of the table contain the observed number of each associated symbols present in the length of the string denoted by the column. The observed count of a symbol between two given positions of the string can be easily computed from this table in $O(1)$ time.

Given the two α and β lists, we now find the chi-square value of substrings from position $g \in \alpha$ to $h \in \beta$, and $g \leq h$. The substring having the maximum value is reported as the global maxima. While computing the chi-square values for all the pairs of positions in the two list, the top- k substrings can be maintained using a heap of k elements.

Conjecture 1. The starting position of the global maxima is always present in the α list.

Corollary 1. *From the above conjecture, it follows that the ending position of the global maxima is also present in the β list.*

Finding all the local maxima in the string requires a single pass, which takes $O(l)$ time for a string of length l . Let the number of local maxima in the string be d . Finding the maximum valued suffix for each local maxima using the C table, requires another pass of each of the local maxima, and thus also takes $O(l)$ time. Since, each local maximum contributes one position to the lists, the number of elements in both the lists is d . We then evaluate the substrings formed by each possible pair of start and end positions, which takes $O(d^2)$. So, in total, the time complexity of the algorithm is $O(l + d^2)$.

3.2 Approximate Greedy Maximum Maxima Search (AGMM)

In this section, we propose a linear time greedy algorithm for finding the maximum substring, which is linear in the size of the input string S . We extract all the local maxima of the input string and its reverse, and populate the α and

β lists as discussed previously. We identify the local maxima suffix max having the maximum chi-square value among all the local maxima present in the string. AGMM assumes this local maxima suffix to be completely present within the global maxima. We then find a position $g \in \alpha$ for which the new substring starting at g and ending with max as a suffix has the maximum χ^2 value, for all g . Using this reconstructed substring, we find a position $h \in \beta$ such that the new string starting at the selected position g and ending at position h has the maximum chi-square measure for all positions of h . This new substring is reported by the algorithm as the global maxima.

The intuition here is that the global maxima will contain the maximum of the local maxima to maximize its value. Although this is a heuristic, the assumption is justified by empirical results in Section 4, which shows that we always obtain an approximation ratio of 0.96 or more.

Using the C table, AGMM takes $O(d)$ time, where d is the number of local maxima found. The total running time of the algorithm is thus $O(d + l)$. Thus, being a linear time algorithm, it provides a order of increase in the runtime as compared to the other algorithms.

4 Experiments

To assess the performance of the two proposed heuristics ARLM and AGMM, we conduct tests on multiple datasets and compare it with the results of the naïve algorithm and the blocking algorithm [9]. The heap variant of the blocking algorithm is not efficient as it has a higher running time and uses more memory, and hence has not been reported. We compare the results based on the following parameters: (i) search time for top-k queries, (ii) number of local maxima found, and (iii) accuracy of the result based on the ratio of the optimal χ^2 value to that returned by the algorithms [1].

Table 1. Results of (a) Sachin’s records. (b) Uniform dataset

| Form | Date | Avg. | Runs scored |
|-------------|-------------|-------|---|
| Best patch | 22/04/98 | 84.31 | 143,134,33,18,100* |
| | to 13/11/98 | | 65,53,17,128,77 127*,29,2,141,8,3 118*,18,11,124* |
| Worst patch | 15/03/92 | 21.89 | 14,39,15 |
| | to 19/12/92 | | 10,22,21 32,23,21 |

(a)

| Parameters | Variable | Blocks | Local maxima |
|----------------------------|-------------------|--------|--------------|
| m=5, k=1 | l=10 ³ | 831 | 742 |
| | l=10 ⁴ | 7821 | 6740 |
| | l=10 ⁵ | 77869 | 66771 |
| l=10 ⁴ , k=1 | m=5 | 7821 | 6740 |
| | m=25 | 8104 | 7203 |
| | m=50 | 8704 | 7993 |

(b)

¹ The experiments were conducted on a 2.1GHz desktop PC with 2GB of memory using C++ in Linux.

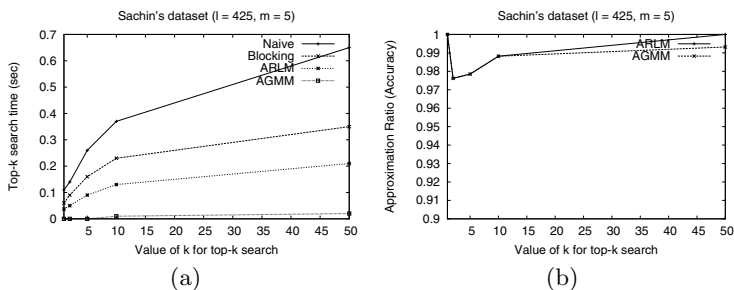


Fig. 1. (a) Time for finding the top-k query in Sachin's run dataset. (b) Approximation ratio of the top-k query in Sachin's run dataset.

4.1 Real Datasets

We used the innings-by-innings runs scored by Sachin Tendulkar in one-day internationals (ODI)² as a real dataset. We quantized the runs scored into 5 symbols as follows: 0-9 is represented by *A*, 10-24 by *B*, 25-49 by *C*, 50-99 by *D*, and 100+ by *E*. The actual probability of occurrences of the different symbols were 0.28, 0.18, 0.22, 0.22 and 0.10 respectively for the five symbols, and the overall average is 45.5 runs per innings. Table 1(a) summarizes the results. We find that during his best patch he had scored 8 centuries and 3 half-centuries in 20 innings with an average of 84.31, while in the off-form period he had an average of 21.89 in 9 innings without any score of above 40.

Figure 1(a) show the times taken by the different algorithms for different values of *k*. The AGMM algorithm requires the least running time as compared to the other procedures, while the ARLM is faster than the naïve and the blocking ones. The number of local maxima found was 281, which is lesser than 319, the number of blocks constructed by the blocking algorithm. So, the heuristic prunes the search space more efficiently. Figure 1(b) plots the approximation factor for the heuristics. The accuracy of the ARLM heuristic is found to be 100% for the top-1 query, i.e., it provides the correct result validating the conjecture we proposed in Section 3. As the value of *k* increases we find an increase in the approximation ratio of both the heuristics.

4.2 Synthetic Datasets

We now benchmark the ARLM and AGMM heuristics against datasets generated randomly using a uniform distribution. To simulate the deviations from the expected characteristics as observed in real applications, we perturb the random data with chunks of data generated from a geometric distribution with parameter $p = 0.3$. The parameters that affect the performance of the heuristics are: (i) length of the input string, *l*, (ii) size of the alphabet set, *m*, and (iii) number of top-k values. For different values of these parameters we compare our

² <http://stats.cricinfo.com/ci/engine/player/35320.html?class=2;template=results;type=batting;view=innings>

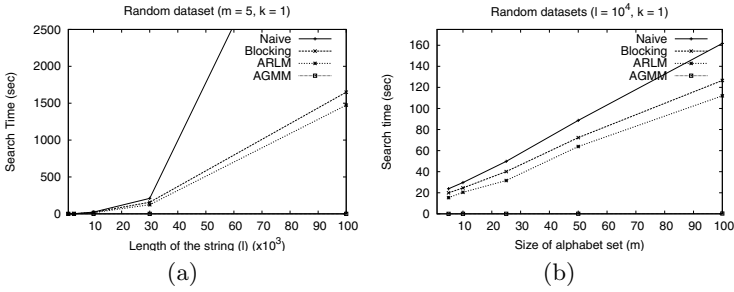


Fig. 2. (a) Effect of length on search time. (b) Effect of size of alphabet on search time.

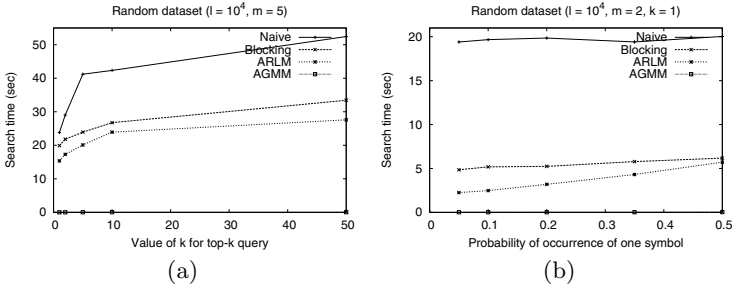


Fig. 3. (a) Effect of value of k for top-k query on search time. (b) Effect of probability in two symbol string on search time.

algorithms with the existing ones on the basis of (a) time to search, (b) approximation ratio of the results, and (c) the number of blocks evaluated in case of blocking algorithm to the number of local maxima found by our algorithm.

Fig 2(a) shows that with the increase in the length of the input string l , the time taken for searching the top- k queries increases. The number of blocks or local maxima increases with the size of the string and, hence, the time increases. The time increases more or less quadratically for ARLM and the other existing algorithms according to the analysis shown in Section 3.1. ARLM takes less running time than the other techniques, as the number of local maxima found is less than the number of blocks found by the blocking algorithm (see Table 1(b)). Hence, it provides better pruning of the search space and is faster. On the other hand, AGMM being a linear time heuristic, runs an order of time faster than the others. We also find that the accuracy of the top- k results reported by AGMM show an improvement with the increase in the string length (graph not shown). The approximation factor for ARLM is 1 for the top-1 query in all the cases tested, while for other top- k queries and for AGMM it is always above 0.96.

The time taken for searching the top- k query as well as the number of blocks formed increases with the size of the alphabet m (Table 1(b) and Fig 2(b)). As m increases, the number of blocks increases as the probability of the same symbol occurring contiguously falls off. A local maxima can only end at positions containing adjacent dissimilar symbols. So the number of local maxima found

increases as well. There seems to be no appreciable effect of m on the approximation ratio of the results returned by the algorithms. We tested with varying values of m with $l = 10^4$ and $k = 2$, and found the ratio to be 1 in all cases.

Fig 3(b) shows the effect of varying probability of occurrence of one of the symbols in a string composed of two symbols only. The approximation ratio remained 1 for both heuristics for the top-1 query.

We next show the scalability of our algorithms by conducting experiments for varying values of k for top- k substrings. Fig 3(a) shows that search time increases with the increase in the value of k . This is expected as more computations are performed. The accuracy of the results for the heuristics increases with k . For $k = 2$, it is 0.96, and increases up to 1 when k becomes more than 10.

5 Conclusions

In this paper, we proposed the problem of finding top- k substrings within a string with the maximum chi-square value for mining interesting patterns. The chi-square value represents the deviation of the observed from the expected. We used the concept of local maxima and proposed two efficient heuristics that run in time quadratic and linear in the number of local maxima. Experiments showed that the heuristics are faster than the existing algorithms, are scalable, and return results that have an approximation ratio of more than 0.96.

References

1. Denise, A., Regnier, M., Vandenbogaert, M.: Accessing the statistical significance of overrepresented oligonucleotides. In: *Work. Alg. Bioinf. (WABI)*, pp. 85–97 (2001)
2. Ye, N., Chen, Q.: An anomaly detection technique based on chi-square statistics for detecting intrusions into information systems. *Quality and Reliability Engineering International* 17(2), 105–112 (2001)
3. Rahmann, S.: Dynamic programming algorithms for two statistical problems in computational biology. In: *Work. Alg. Bioinf. (WABI)*, pp. 151–164 (2003)
4. Regnier, M., Vandenbogaert, M.: Comparison of statistical significance criteria. *J. Bioinformatics and Computational Biology* 4(2), 537–551 (2006)
5. Bejerano, G., Friedman, N., Tishby, N.: Efficient exact p-value computation for small sample, sparse and surprisingly categorical data. *J. Comp. Bio.* 11(5), 867–886 (2004)
6. Read, T., Cressie, N.: *Goodness-of-fit statistics for discrete multivariate data*. Springer, Heidelberg (1988)
7. Read, T., Cressie, N.: Pearson's χ^2 and the likelihood ratio statistic G^2 : a comparative review. *International Statistical Review* 57(1), 19–43 (1989)
8. Hotelling, H.: Multivariate quality control. *Techniques of Statistical Analysis* 54, 111–184 (1947)
9. Agarwal, S.: On finding the most statistically significant substring using the chi-square measure. Master's thesis, Indian Institute of Technology, Kanpur (2009)
10. Keogh, E., Lonardi, S., Chiu, B.: Finding surprising patterns in a time series database in linear time and space. In: *SIGKDD*, pp. 550–556 (2002)
11. Dutta, S., Bhattacharya, A.: Mining most significant substrings based on the chi-square measure. arXiv:1002.4315 [cs.DB] (2010)

Probabilistic User Modeling in the Presence of Drifting Concepts

Vikas Bhardwaj and Ramaswamy Devarajan

Department of Computer Science
Columbia University, New York NY 10027, USA
{vsb2108,rd2446}@columbia.edu

Abstract. We investigate supervised prediction tasks which involve multiple agents over time, in the presence of drifting concepts. The motivation behind choosing the topic is that such tasks arise in many domains which require predicting human actions. An example of such a task is recommender systems, where it is required to predict the future ratings, given features describing items and context along with the previous ratings assigned by the users. In such a system, the relationships among the features and the class values can vary over time. A common challenge to learners in such a setting is that this variation can occur both across time for a given agent, and also across different agents, (i.e. each agent behaves differently). Furthermore, the factors causing this variation are often hidden. We explore probabilistic models suitable for this setting, along with efficient algorithms to learn the model structure. Our experiments use the Netflix Prize dataset¹, a real world dataset which shows the presence of time variant concepts. The results show that the approaches we describe are more accurate than alternative approaches, especially when there is a large variation among agents. All the data and source code would be made open-source under the GNU GPL.

1 Introduction

In this article we investigate prediction problems in which there are multiple evolving entities, which we refer to as *agents*. We address two complications that frequently arise in problems that involve modeling agents: variation among (1) agents and (2) within an agent across time. Let's consider the problem of estimating the probability of a movie reviewer's rating. Here we have features describing the context such as genre, release time, reviewer's preferences etc. In this problem we may have both kinds of variation. Clearly, there will be variation among the agents, as every person is different. Also, there will be variation in the user's ratings because of his tastes, mood etc. which vary with time.

These variations pose problems to the modeler. One approach would be to learn a single model that applies to all agents. This approach has the advantage that training examples from all agents can be pooled together. Given plentiful

¹ <http://archive.ics.uci.edu/ml/datasets/Netflix+Prize>

training observations for an agent, another approach is to learn a different model for each agent from only that agent's examples, and then use the appropriate agent specific model when making predictions. However, if training examples are scarce, this agent specific approach is susceptible to overfitting. Even when training data is relatively plentiful for a given agent there may be contexts that have not been observed very often.

Our approach takes the middle ground between these two extremes. Like the agent specific or heterogeneous approach, we learn a different model for each agent, but unlike it, training observations of multiple agents may have an influence on the learned probabilities for a given agent. We do this by identifying a neighborhood of agents with similar probability profiles and then combining their models if training data is scarce.

In addition to variation across agents, we consider patterns of change within a single agent over time. Problems in which class distributions change with time are said to exhibit concept drift. See, for example, Widmer [15]. Our approach to modeling concept drift utilizes hidden Markov models [12], that have found widespread use in many sequential processing tasks.

2 Related Work

We investigate the approaches to model multiple evolving users or entities over time. Our approach is closely related to previous work in Concept drift and Collaborative Filtering.

Concept Drift has received considerable attention both in the field of Data Mining [16,11,15,14] and Computational Learning Theory [15]. Previous work on concept drift have addressed the problem of concept drift over time, but concept drift across agents has not received particular attention. Incremental (*or online*) concept drift was discussed by [15]. Widmer and Kubat's *FLORA* framework used a window based learning system. The *FLORA2* algorithm and the *FLORA3* algorithm addressed the issue of recurring contexts. [7], addressed concept drift with the *CVFDT* algorithm, which uses flexible windowing and decision trees. [11] have also used Decision Trees for online concept learning, the algorithm *OnlineTree2* introduces multiple flexible windows. Concept Drift has also been applied in the field of Network Security to model the user behavior and to use it for Anomaly Detection. [9].

Collaborative Filtering Methods have become a popular system on the Internet and are often used as techniques to complement content-based filtering systems [6]. [10] uses a Collaborative Framework based on Fuzzy Association Rules and Multiple-level Similarity (FARAMS). Recommender Systems often rely heavily on Collaborative filtering, where past transactions are analyzed to establish connections between users and products [8]. [8] introduces an approach which uses Neighborhood models and Latent factor models to do Collaborative Filtering. The filtering methods uses both implicit and explicit feedback from user (*agents*), to improve results. Current Recommender Systems consider the relations between the various users (*agents*), but do not consider the Concept Drift over time.

Other work similar to what we present here include approaches for training probabilistic models by combining examples from multiple contexts. Interpolated Markov models [3] are models that combine different order Markov models based, in part, on the number of training examples observed in each case. This approach was originally developed for finding genes in microbes. More recently variable length HMMs [14] were introduced. These models dynamically adapt the length of an HMM memory using context trees. Interpolated hidden Markov models have also been used in many domains such as finding genes and in automatic instrument recognition [13].

3 Overview

Figure 1 shows diagrams of some of the graphical probability models we use here. In this example there are N agents. For each agent we have multiple observations at various time points. We use Y_a^t to represent the output random variable for agent a at time t . Each Y is associated with M observable features, denoted by the feature vector X .

When we are learning a classifier to predict the output Y of each agent, there can be two extreme cases, (i) the Homogeneous Case, where we learn a single model for all agents (Fig. 1 A) and (ii) the Heterogeneous Case, where we learn a model for each agent, (Fig. 1 B).

Both these approaches are flawed in many real world scenarios, as there is usually some difference between the behavior of each agent, but yet its never

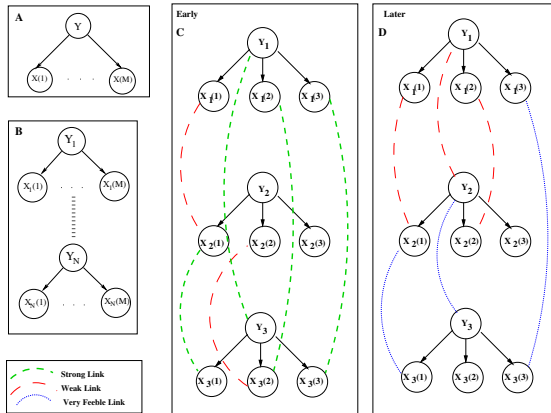


Fig. 1. Illustration of Bayesian Probabilistic Models used in our approach. Solid lines with arrows represent probabilistic dependencies. Dashed lines in C and D represent influences between agent models on learned probability parameters. **A:** A single model is learned for all agents *viz.* *Homogeneous Model*. **B:** A model is learned for each agent *viz.* *Heterogeneous Model*. **C:** An Interpolated Model near beginning of training. **D:** An Interpolated Model later during training, strength of ties have weakened, and new ties are found. Influences change with more training. See difference between C and D.

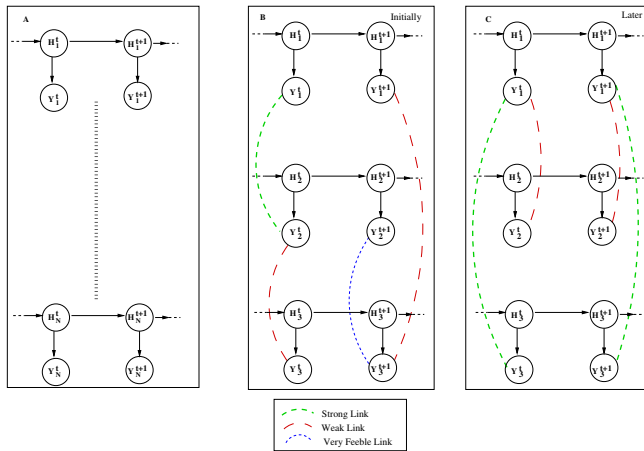


Fig. 2. Illustration of Hidden Markov Models used in our approach. **A:** An HMM is learned for each agent (similar to the Heterogeneous Model, but captures concept drift over time). **B:** An Interpolated HMM near beginning of training. **C:** An Interpolated HMM later during training, strength of ties have weakened, and new ties are found.

true that all agents are entirely different from each other. Thus to model predictive classifiers close to real world scenarios, we choose an approach which is somewhere in between the two extreme cases.

Figure 1 C and D illustrate our approach to modeling variation among agents. In this example we have three agents and a set of three features per agent. Early on during training, or when a new agent appears, and training data for individual agent is scarce, its probabilities are influenced heavily by neighboring agents. This is illustrated in the figure with dashed lines connecting the nodes. Outputs(class) of Agent₁ and Agent₃ are similar, but $X_{(1)}$ of Agent₁ is closer to $X_{(1)}$ of Agent₂. As we have more training data, we learn that output of Agent₁ is closely tied to Agent₂, and closeness between $X_{(3)}$'s of Agent₁ and Agent₃ has grown weaker.

To model concept drifts over time, we investigate Hidden Markov Models, as shown in Fig. 2. Figure 2 A illustrates the case where we learn one HMM for each agent. Variables H_i^t represent the Hidden Variable for Agent _{i} at time t .

The Hidden Variables capture the drift over time as each hidden variable relearns its probability distribution for state transitions and emissions, as new training data is encountered. Figure 2 B and C, illustrate an Interpolated HMM, in which we have a model for each agent and an agent's prediction probabilities are influenced both by its own training data and that of other agent's in its neighborhood similar to the static HMM explained above.

Thus, these Interpolated models attempt to capture the real world scenario, where there are varying relationships among agents. Agent₁ may be closer to Agent₂ in some ways and closer to Agent₃ in some ways.

4 Methods

As our setting involves multiple agents over time, we assume that we have labeled data for N agents through time t . For our models in Figure 1, we estimate a naive Bayes model for each agent.

For agent a at time t' we have the labeled data set $\{(\mathbf{x}_a^t, y_a^t) | t \leq t'\}$ where $1 \leq y_a^t \leq C$ is a discrete class value (C possible class values) with feature values $\mathbf{x}_a^t = (x_a^t(1), \dots, x_a^t(M))$ where $x_a^t(j)$ is the value of feature j for an observation at time t .

Assumptions

- Throughout this article, we shall assume that each of the features $X(j)$ take discrete values $1, 2, \dots, s_j$ where s_j is the size of the domain for $X(j)$. Our approach, however, is general and can be applied to the continuous case as well.
- We assume the number of features and their domains are constant across agents and time.
- We use a value of 0 to denote a missing feature value.
- We do not require the observations of different agents to be aligned in time.

Homogeneous and Heterogeneous Models

To learn the homogeneous model we pool all examples from all agents into a single training set $T = \{(\mathbf{x}_a^t, y_a^t) | \forall a, t < t'\}$. We compute counts $n_j(k, c)$, the number of examples in which feature j had a value of k with class c , and $n(c)$, the total number of observations of class c . Then we compute the maximum *a-posteriori* (MAP) estimates with pseudo-counts of 1 for all parameters:

$$\hat{P}(X(j) = k | Y = c) = \frac{n_j(k, c) + 1}{\left(\sum_{1 \leq k' \leq s_j} n_j(k', c)\right) + s_j} \tag{1}$$

and

$$\hat{P}(Y = c) = \frac{n(c) + 1}{\left(\sum_{1 \leq c' \leq C} n(c')\right) + C} \tag{2}$$

Here we have dropped the agent subscripts because we use the same model for each agent.

Learning in the heterogeneous case is the same as the homogeneous case with the exception that each agent’s model is learned from a training set consisting of only that agent’s examples.

Interpolated Model

Given plentiful training data (and assuming stationary concepts) we would expect the heterogeneous model $\hat{P}_a(Y_a | X_a)$ to be more accurate than the homogeneous model $\hat{P}(Y | X)$ on predictions for any agent a . On the other hand, if training data for agent a is scarce there is concern of over-fitting. Even if training data for a is scarce, however, we may have many examples for other agents.

This happens if the numbers of observations for each agent is skewed so that some agents have many observations while others have few. In this case, we'd like to be able to use training examples from the other agents to help estimate the model for a . Our approach is to form a model for a by combining its own MAP estimate with the MAP estimates of other agents in the neighborhood of a . We call this the interpolated approach.

Since learning naive Bayes models from complete data is just a set of separate estimation problems, one for each conditional distribution and one for the prior of the class variable, we describe our approach in terms of estimating just a single distribution for a discrete domain.

Let W be the random variable we are estimating a distribution for (*i.e.*, $P(W)$ is either $P_a(Y)$ or $P(X_a(j)|Y_a = c)$) and K be the number of values in W 's domain. We begin with the table of counts, where $n(a, k)$ is the number of observations of value k for agent a , and first compute the agent specific MAP estimates $\hat{P}_a(W)$ using equations analogous to Equations 1 or 2 (but of course using only a 's observations). This estimate is identical to the estimate used in the homogeneous model for a .

We construct the interpolated distribution for a , $\hat{P}_a^I(W)$ as a mixture of its MAP estimate and its "neighborhood" distribution $P_a^N(W)$ (described below):

$$\hat{P}_a^I(W) = \lambda_a \times \hat{P}_a(W) + (1 - \lambda_a)P_a^N(W) \quad (3)$$

where $0 \leq \lambda_a \leq 1$ is the mixing coefficient that determines the relative contributions of a 's MAP distribution and its neighborhood distribution. We set λ_a using the logistic function so that it smoothly varies as its number of observations grows. M here is a tuneable parameter that controls how fast we transition from the neighborhood distribution to the MAP distribution. A large value of M causes a slow transition.

$$\lambda_a = \frac{1}{1 + \exp(M - \sum_{1 \leq k \leq K} n(a, k))} \quad (4)$$

An agent's neighborhood distribution is a combination of the MAP distributions of its *eligible* neighbors and an average agent distribution. We represent the "distance" $d(a, a')$ between agents a and a' (for estimating W) with the Kullback-Leibler divergence:

$$d(a, a') = \sum_{1 \leq k \leq K} \hat{P}_a(W = k) \log \left(\frac{\hat{P}_a(W = k)}{\hat{P}_{a'}(W = k)} \right) \quad (5)$$

Next, we form an "average agent" model $\bar{P}(W)$ that is the average of the MAP models for all agents:

$$\bar{P}(W) \propto \prod_a \hat{P}_a(W) \quad (6)$$

where the product is a normalized point-wise product of distributions. Notice that \bar{P} is not the same as the estimate learned by the homogeneous model because \bar{P} is influenced equally by all agents whereas in the homogeneous model

an agent’s influence is proportional to how many observations it has. Our reasoning is that when little or nothing is known about an agent, which is when \bar{P} has influence, we believe it is more reasonable to model it more like the average agent than the average training example.

An agent a' is an eligible neighbor of a if both (i) $\lambda_{a'} > 0.5$ and (ii) $d(a, a')$ is less than the distance between a' ’s MAP distribution $\hat{P}_{a'}(W)$ and the agent average distribution $\bar{P}(W)$. The first condition ensures that eligible neighbors have seen enough training examples so that their MAP estimates are more likely to be reliable and the second condition prevents distant “neighbors” from having any influence. We define the neighborhood of a , \mathcal{N}_a , to be its D closest eligible neighbors. If there are less than D eligible neighbors then \mathcal{N}_a contains all eligible neighbors, which could be zero. Finally, we construct the neighborhood distribution P_a^n by taking the normalized point-wise product of the average agent distribution with the MAP distributions of all neighbors:

$$P_a^n(W) = \left(\prod_{a' \in \mathcal{N}} \hat{P}_{a'}(W) \right) \times \bar{P}(W) \quad (7)$$

We now compute the interpolated model with Equation 3. In this way we compute all distributions for all agents. Before we move on, we stress a few important details about this approach:

- The neighborhood for a is distribution specific. It is possible that a has a different set of neighbors when estimating $P(X(i)|Y = c)$ than it has when estimating $P(X(i)|Y = c')$. Although, this can be seen as a virtue, this may be exasperating problems associated with using a generative model for discriminative classification.
- The neighborhood distribution behaves like an evolving bias or prior. Implicit in this formulation is that while agents may be diverse, the distributions tend to cluster.
- If the MAP distribution for an agent is near the center of its neighborhood the interpolated distribution will be similar to the MAP. On the other hand if most of an agent’s neighbors are in the same direction then the neighborhood distributions combine to pull the interpolated distribution toward that center.

Hidden Markov Models

We now turn toward our approach for representing changes within an agent over time. For this we use hidden Markov models. We explore both a regular HMM with one model for each agent and an Interpolated version which applies the interpolated approach to an agent’s HMM.

Our HMM consists of two states. These state represent different modes of the agent’s distribution over class labels. In the domain of our dataset, the hidden states could correspond to changes in a reviewers’ calibration or even changes in actual human reviewer associated with a given user ID, though we have no definite knowledge of what these states maybe, we can learn their behavior. To learn the parameters of the HMM for agent a at time t we train an HMM using

the expectation-maximization algorithm [4] using the single sequence y_a^1, \dots, y_a^t . Thus, this approach is similar to the heterogeneous method, in which each agent's model is trained using only examples from that agent. But, of course, it is different, as here we allow the agent's probability distribution over the class to change with time.

In the Interpolated hidden Markov model, we learn an interpolated distribution for the emissions, similar to the method explained above. For each agent, the probability distribution of the emissions are learned by using its distribution and the distributions of its "neighbors".

The HMMs we consider here, is in some ways the simplest possible in that the emissions involve a single variable. More complex HMMs in which emissions include both the class and features are possible as well.

5 Evaluation

We conducted a set of experiments to assess that how well our approaches model evolving heterogeneous agent behavior in real world domains. We wish to (i) compare the predictive accuracies of the various models, (ii) investigate the effect of the training set size, and (iii) use HMMs to model concept drift over time We have assembled a publicly available real world data set involving humans. Our set is derived from the Netflix prize² data. This data set contains over 100 million movie ratings (integer values 1-5) for 400,000+ users and 17,770 movies from 1999 to 2005. We use a subset of this data in our experiments. We extracted all the ratings from 2000 randomly selected users (the agents in this problem) for a total of 422,692 ratings. The rating is the class variable, and we divided ratings such that ratings (1,2,3) are classified as low or '0' and ratings (4,5) are classified as high or '1'. Making it a binary class problem We have 103 features for each rating: (i) The day of week of the rating, (ii) The month of the rating (iii) The movie's year of release. The remaining features are the ratings (if available) of 100 "heavy" users who have rated more than 1,000 movies. These 100 users are disjoint from the agents.

The first experiment was designed to see how well the interpolated method models the prior distribution $P(Y_a)$. In this experiment the 30% of the examples with greatest time index were held aside for evaluation. We trained models with successively more training data. We use all data from the time of the earliest instance through t where t is set so that 1%, 2%, ..., 70% of the available labeled examples are in the training set. Each trained model predicts the value of the held aside 30%. We measure the performance of each trained model with the test set log-likelihood. The parameter M was set to 20.

Figure 3 shows the results of this experiment. As expected, the heterogeneous model shows the most improvement as the size of the training set increases and the homogeneous model shows the least with the interpolated model in between. The interpolated model surprisingly continues to slightly improve throughout the whole range. Given the large amount of data it is unlikely that this trend is caused

² <http://archive.ics.uci.edu/ml/datasets/Netflix+Prize>

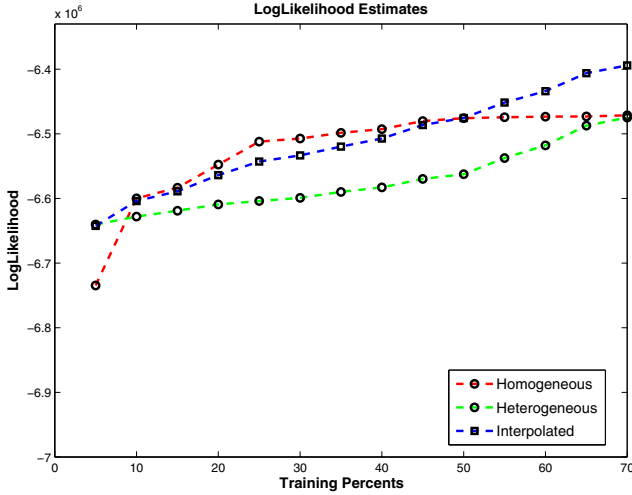


Fig. 3. Log-Likelihood Estimates for Homogeneous, Heterogeneous and Interpolated Model

by estimation error, but suggests that drift may be present. Comparing the interpolated curve to the others, we see that performance begins close to the homogeneous model and then improves as the individual agent models begin to diversify.

These trends suggest high inter-agent variability. We see that the interpolated models perform best in settings with diverse agents. From these results we conclude that the interpolated approach can lead to more accurate estimation of probability distributions, especially if agents are diverse.

To compare the predictive accuracy of the interpolated model, we compute ROC curves (Fig. 4) for each of our static models. In this experiment we use all features (as well as class labels) to train the models with the first 70% of the data and predict the labels on the final 30%. Although the interpolated model was the most accurate in terms of test-set log likelihood this doesn't translate into a clear win in the ROC curves. It is only slightly better than the homogeneous and heterogeneous models.

Our investigations into using HMMs to model concept drift focus on the predicting the class values using only the priors and not the features. Similar to the first experiment, 30% of the examples with greatest time (in total, not per agent) index were held aside for evaluation. A separate HMM is trained from the first 40% using only that agent's observed label sequence. We also learn an Interpolated HMM for each agent by applying the interpolated approach to the emission distributions. We compare the results of the HMM to the heterogeneous model and the Interpolated-HMM to the Interpolated Model. We measure the performance of each trained model with the test set log-likelihood. Figure 5 shows the results of the 2 types of HMMs learned. The results showed strong evidence of concept drift, as the difference in the test set log likelihoods (summed over

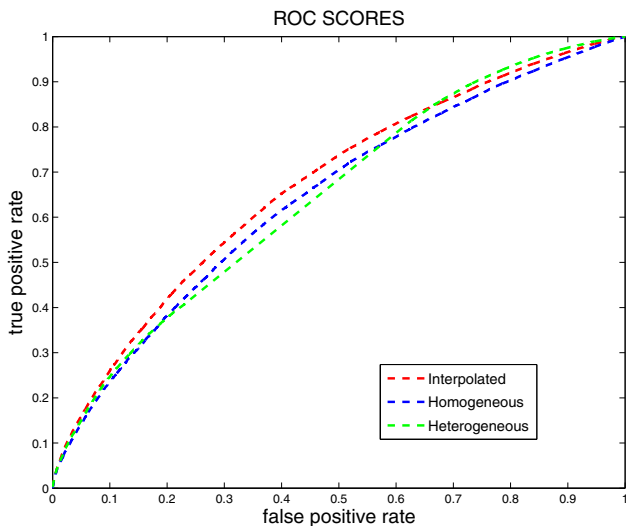


Fig. 4. ROC Curves for Homogeneous, Heterogeneous and Interpolated Models

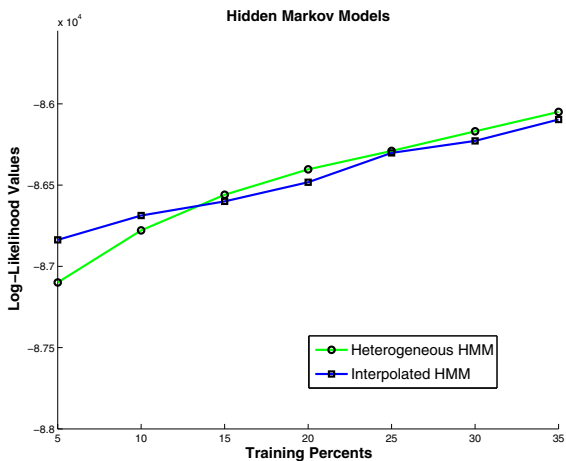


Fig. 5. Log-Likelihood of Heterogeneous and Interpolated Hidden Markov Models

all agents) of the interpolated model and the HMMs was in favor of the HMMs. We see that initially the heterogeneous HMM starts lower than the interpolated HMM, which can be attributed to lack of training data for each agent. As more training data is made available, both models almost tie with each other, but still increasing, which suggests the presence of concept drift. Observing the agent specific log likelihoods shows that most of the difference between that static models and the HMMs is accounted for by a relatively small number of agents.

It might be that these agents are strongly drifting while the others are weakly or not drifting. This suggests an alternate approach in which some agents are modeled with HMMs while others are modeled with static models.

6 Conclusion and Future Work

In this paper we investigated probabilistic models for multiple evolving agents. We addressed the common complications that may arise in such a setting, viz. (i) concept drift over time for a given agent, (ii) varying relationships among different agents. We compare and contrast 3 different static models, viz. (i) the homogeneous model in which we learn a single model for each agent, (ii) a heterogeneous model where we learn a different model for each agent. (iii) an interpolated model where similar to the heterogeneous approach, we have a different model for each agent, but unlike it, observations from other agents can have an influence on an agent's model. Then we investigate hidden Markov models to address the issue of concept drift over time. Lastly we applied the interpolated approach to HMM to get an interpolated hidden Markov model for each agent. The contribution of this paper is the introduction of a novel method for learning agent models that involves combining the models of multiple agents in the same neighborhood.

Importantly, the concept of neighborhood is context specific, so the neighboring agents that combine to form one distribution may be distinct from those that combine in another. We demonstrated the validity and need of such models in real world scenarios. In settings with diverse agents, models learned with our interpolated approach proved to be empirically better than purely homogeneous or purely heterogeneous models. Concept drift over time was investigated using hidden Markov models and strong evidence of concept drift was found.

Our Interpolated Hidden Markov Model, that is used in conjunction with hidden variables, can be used to represent cases where there are time varying concepts affected by unknown factors. For example, such models may find use in risk assessment in financial data such as stocks. Thus, modeling the evolving relationship among various agents together with accounting for time varying concepts can prove to be very useful in several fields like finance, marketing or medicine. This work can be further extended by considering models where there are more than one type of hidden variable, each type affecting different observable features of the agent. Hence this can effectively model cases where certain features of an agent vary over time and others remain consistent. A typical setting would be a behavioral analysis of a multi-agent system which integrates models into an intelligent structure that improves the efficiency of an agent [2].

References

1. Case, J., Jain, S., Kaufmann, S., Sharma, A., Stephan, F.: Predictive learning models for concept drift. In: Richter, M.M., Smith, C.H., Wiehagen, R., Zeugmann, T. (eds.) ALT 1998. LNCS (LNAI), vol. 1501, pp. 276–290. Springer, Heidelberg (1998)

2. Coulondre, S., Simonin, O., Ferber, J.: Dynamo: a behavioural analysis model for multi-agent systems. In: Proceedings 1999 International Conference on Information Intelligence and Systems, pp. 614–621 (1999)
3. Delcher, A., Kasif, S., Fleischmann, R., Peterson, J., White, O., Salzberg, S.: Alignment of whole genomes. *Nucleic Acids Research* 27(11), 2369–2376 (1999)
4. Dempster, A., Laird, N., Rubin, D.: Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B* 39, 1–38 (1977)
5. Helmbold, D.P., Long, P.M.: Tracking drifting concepts by minimizing disagreements. *Machine Learning*, 27–45 (1994)
6. Herlocker, J.L., Konstan, J.A., Borchers, A., Riedl, J.: An algorithmic framework for performing collaborative filtering. In: SIGIR 1999: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval, pp. 230–237. ACM, New York (1999)
7. Hulten, G., Spencer, L., Domingos, P.: Mining time-changing data streams. In: KDD 2001: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 97–106. ACM, New York (2001)
8. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: KDD 2008: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 426–434. ACM, New York (2008)
9. Lane, T., Brodley, C.E.: Approaches to online learning and concept drift for user identification in computer security. In: KDD, pp. 259–263. AAAI Press, Menlo Park
10. Leung, C.W.-k., Chan, S.C.-f., Chung, F.-l.: A collaborative filtering framework based on fuzzy association rules and multiple-level similarity. *Knowl. Inf. Syst.* 10(3), 357–381 (2006)
11. Núñez, M., Fidalgo, R., Morales, R.: Learning in environments with unknown dynamics: Towards more robust concept learners. *J. Mach. Learn. Res.* 8, 2595–2628 (2007)
12. Rabiner, L.R.: A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE* 77(2), 257–286 (1989)
13. Virtanen, T., Heittola, T.: Interpolating hidden markov model and its application to automatic instrument recognition. In: ICASSP 2009: Proceedings of the 2009 IEEE International Conference on Acoustics, Speech and Signal Processing, Washington, DC, USA, pp. 49–52. IEEE Computer Society, Los Alamitos (2009)
14. Wang, Y., Zhou, L., Feng, J., Wang, J., Liu, Z.-Q.: Mining complex time-series data by learning markovian models. In: ICDM 2006: Proceedings of the Sixth International Conference on Data Mining, Washington, DC, USA, pp. 1136–1140. IEEE Computer Society, Los Alamitos (2006)
15. Widmer, G., Kubat, M.: Learning in the presence of concept drift and hidden contexts. *Mach. Learn.* 23(1), 69–101 (1996)
16. Zhang, P., Zhu, X., Shi, Y.: Categorizing and mining concept drifting data streams. In: KDD 2008: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 812–820. ACM, New York (2008)

Using Association Rules to Solve the Cold-Start Problem in Recommender Systems

Gavin Shaw, Yue Xu, and Shlomo Geva

Faculty of Science & Technology, Queensland University of Technology, Australia
g4.shaw@student.qut.edu.au,
{yue.xu,s.geva}@qut.edu.au

Abstract. Recommender systems are widely used online to help users find other products, items etc that they may be interested in based on what is known about that user in their profile. Often however user profiles may be short on information and thus it is difficult for a recommender system to make quality recommendations. This problem is known as the cold-start problem. Here we investigate using association rules as a source of information to expand a user profile and thus avoid this problem. Our experiments show that it is possible to use association rules to noticeably improve the performance of a recommender system under the cold-start situation. Furthermore, we also show that the improvement in performance obtained can be achieved while using non-redundant rule sets. This shows that non-redundant rules do not cause a loss of information and are just as informative as a set of association rules that contain redundancy.

1 Introduction

Recommender systems are designed to understand a user's interests, learn from them and recommend items (whether they be products, books, movies etc) that will be of interest to the user. This requires them to personalise their recommendations. Recommendation systems usually work most effectively when user profiles are extensive and/or the applicable dataset has a high information density. When the dataset is sparse or user profiles are short, then recommender systems struggle to provide quality recommendations. This is often known as the cold-start problem.

We propose expanding a user profile (eg. so it contains more ratings) through the use of association rules derived from the dataset. By doing so we expand profiles based on patterns and associations of items, topics, categories etc and thus give more information to a recommender system. This would reduce the effect of the cold-start problem and result in better quality recommendations earlier on. We also investigate the performance of both non-redundant rules and rules that contain redundancy. The idea behind non-redundant rules is that the removed redundant rules should not cause a loss of information [11] [12]. If there is no information loss, then the performance should be similar to that of a ruleset with redundant rules.

The paper is organised as follows. Section 2 discusses related works. Our proposed approach for solving the cold-start problem is presented in Section 3. In Section 4 we outline our experiments to test our approach. Lastly, Section 5 concludes the paper.

2 Related Work

Here we briefly review works related to our proposed approach. We consider works that have focused on recommender systems and redundancy in association rules.

Much work has been done in the area of recommender systems. A survey undertaken in [1] details many different approaches that have been proposed. It is shown that the cold-start problem heavily affects content-based and collaborative-based systems [1]. In the case of collaborative-based systems, proposed solutions include getting the user to rate specific items [1]. However this places a burden on the user. Our proposed approach does not. Thus work focusing on solving the cold-start problem includes collaborative & content hybrids [2] [7], ontology based systems [5] and taxonomy driven recommender systems [10] [13]. However, all of these proposals have drawbacks. Hybrid systems can lack novelty, resulting in recommendations that are excessively content centric [13]. Ontology based system requires a well defined ontology to be created, something that can be difficult and limiting. Taxonomy based systems work better, but still have low performance. Also the HTR system proposed in [10] performs only marginally better than the TPR system proposed in [13], although it is more time efficient. The taxonomy based approach in [13] does have the advantage of being able to be applied to many domains. Work in [4] proposed a system that uses fuzzy cross-level rules to enhance a collaborative based recommender to solve this cold-start problem. Our work focuses on the cold-start problem of users not items.

Much work in the field of association rule mining has focused on finding more efficient ways to discover all of the rules. However, complete rule enumeration is often intractable in datasets with a very large number of multi-valued attributes. One approach is to determine which rules are redundant and remove them, reducing the number of rules a user has to deal with while not reducing information content [6] [11] [12]. The MinMax algorithm proposed in [6] uses the closure of the Galois connection to define non-redundant rules. These non-redundant rules have a minimal antecedent and maximal consequent and were selected as the most relevant because they are the most general. The ReliableBasis algorithm proposed in [11] [12] argued that MinMax still contained redundant rules. They propose that by using the same technique as MinMax, but relaxing the definition for redundancy, further redundant rules could be removed. Work in [11] shows a reduction of over 80% can be achieved in some situations. Recent work in [8] [9] proposed taking the MinMax and ReliableBasis approaches developed for single level datasets and extend them to remove hierarchical redundancy found in multi-level datasets. The proposed extensions, MinMax with HRR and

ReliableBasis with HRR [8] [9] were developed to find rules that not only had a minimal antecedent and maximum consequent, but also comprised of high level concepts or items. These approaches have been shown to yield further reductions in the size of rule sets. In this paper we demonstrate an application where non-redundant rule sets can be used in place of other rule sets which are redundant.

3 Using Association Rules to Expand User Profiles

In this section we outline our proposed approach and investigation into solving the cold-start problem in recommender systems.

For our investigation we use the Taxonomy-driven Product Recommender (TPR) system first proposed in [13]. The user profiles are created through the process described in [13] (which has been omitted here due to space) to generate taxonomy-driven profiles. A taxonomy T containing topics (or categories) t in a multi-level structure, where each topic has one parent or supertopic, but may have many children or subtopics. Thus the taxonomy can be visualised as a tree. By doing this the profiles represent the user's interests in topics, rather than items. Although we have used the taxonomy in the profile generation, we still have the issue of short profiles. By using the rules to expand the profile we bring other topics in that similar users have shown an interest in.

From the user profiles we can construct a transactional dataset, where each transaction is a user and each topic is an attribute. Thus each transaction consists of the topics that a user is interested in. We then mine the transactional dataset for frequent patterns and derive association rules from these patterns. This will give us association rules between topics that interest users. These rules allow us to discover topics that frequently appear together as part of a user's interest. This rule set will then be used to expand user profiles to solve the cold-start problem.

Finally, we expand the user profiles. For this we take the set of user profiles P and the association rule set we derived in the second step. For each user profile $p(u_x)$ we extract all of the topics t within and generate a list of all the combinations possible from the group of topics. Each combination represents a possible antecedent of an association rule. We take each combination and search the set of association rules for any rules that have a matching antecedent. If such a rule exists we can then take the topics in its consequent and add them to the profile $p(u_x)$. Each new topic added is assigned a weight, which is calculated based on the weights of the topics in that rule's antecedent.

$$Weight_{t_x} = \frac{\sum_{i=1}^{|A|} Weight_{t_i}}{|A|} \times R_{conf} \quad (1)$$

where $|A|$ represents the number of topics in the antecedent of the rule $R : A \rightarrow C$. Then as per the design of the TPR approach the values of the topics in the expanded profile need to be normalised. All topic scores within a profile p are normalised through the following formula.

$$NormalisedWeight_{t_x} = \frac{Weight_{t_x}}{\sum_{i=1}^n Weight_{t_i}} \times Limit \quad (2)$$

where *Limit* is the value to which a profile of normalised topic values is to sum to. Thus this generates a set of expanded user profiles which we show in our experiments have the potential to improve recommendations over profiles that have not been expanded.

We have outlined our proposal for using association rules to expand user profiles in order to improve recommender system quality. However, it is possible that we may want to place limitations on the expansion of user profiles.

1. Restrict the expansion to short profiles. The idea behind this proposal is to expand users who have very few ratings and thus suffer from the cold-start problem. Users with many ratings do not have this problem. Thus a restriction should be imposed on how many topics can be in the user profile p before there is too many to warrant expansion.
2. Restrict the number of rules used when expanding a user profile. It is entirely possible that when deriving the association rules from the transactional dataset that a large list may be generated. It is also possible that from this, when expanding a user profile that a large number of rules and their consequents will be considered for inclusion in the expanded user profile. This may lead to poorer performance as many more topics are added and more items from a wider selection become recommended. Our experiments will test the effect of using 1 to 5 rules during expansion.

4 Experiments and Evaluation of User Profile Expansion

Here we outline the experiments we undertook to study the value of our proposal to use association rules in expanding user profiles to improve recommendation quality, as well as the effect redundant and non-redundant rule sets have. We aim to show that non-redundant rule sets give a similar improvement when compared to rule sets containing redundant rules.

4.1 Evaluation Metrics and Dataset

In order to evaluate the performance of the baseline set of profiles and the expanded set of profiles we follow the same approach detailed in [10]. The past ratings of each user $u \in U$ have been randomly divided into training and test components. For the experiments, the recommender system will recommend a list of n items for user u_i based on the training set and will be evaluated against the test set. For our experiments we use exactly the same training and test sets as used in [10].

To evaluate redundant and non-redundant rule sets we use 4 different rule mining algorithms to extract a set of rules. The algorithms used are the ones previously introduced in Section 2; MinMax [6], ReliableBasis [11] [12], MinMax with HRR [8] [9] and ReliableBasis with HRR [8] [9].

In our work we use precision, recall and F1-measure to determine the overall performance of the recommender system [3]. This allows us to compare the standard approach against our proposal of using association rules for user profile expansion.

For this investigation we use the BookCrossing dataset (obtained from <http://www.informatik.uni-freiburg.de/~chiegler/BX/>) which contains users, books and the ratings given to those books by the user. The taxonomy tree and descriptors are originally sourced from Amazon.com and are exactly the same as those used in [10]. From this we build a transactional dataset that contains 92,005 users (transactions) and 12,147 topics from the taxonomy. The dataset is populated using the descriptors that belonged to 270,868 unique books. This dataset is then mined to derive the association rules from it. For our experiments here, all ratings of items are considered to be positive. From the BookCrossing dataset we also build the base set of user profiles P . This set of profiles contains 85,415 distinct users. As already mentioned the ratings for each user are divided into a training set and a test set. The set of user profiles P is based on the training set. The average number of ratings in a user profile is 27.08. This set of user profiles will serve as the baseline input in our experiments and is also the set that will be expanded.

4.2 Experiment Results

To validate our proposal we conducted a series of experiments to see whether using association rules to expand user profiles improves recommendation quality. From the transactional dataset we set the minimum confidence threshold to 50% and are able to derive 37,827 association rules using the MinMax algorithm [6]. We then go through the user profiles in the training set and for any profile $p \in P(\text{train})$ that has 5 or less ratings we attempt to expand our approach. This yields a total of 15,912 user profiles which we consider to be short profiles. We chose to restrict profile expansion to those with 5 ratings or less as these are the users most likely to suffer from the cold-start problem. This falls in line with the first restriction proposed in section 3. Long profiles do not usually suffer from the cold-start problem, so expanding them is likely to result in a high computation cost for minimal gain. We then make up to 10 recommendations for these 15,912 users and measure the overall performance of the recommender system. We compare our approach against the baseline of the same 15,912 user profiles with no expansion. All experiments use the TPR recommender [13].

As shown in Table 1 the baseline set of user profiles (no expansion) scores only 0.00619, 0.0571 and 0.0112 for precision, recall and F1-measure respectively. When using expanded profiles we manage to achieve up to 0.00815, 0.0754 and 0.0147 for precision, recall and F1-measure. This is an improvement of around 31.5% over the baseline. Also the efficiency of the recommender is not negatively impacted, as while our expanded profiles naturally take longer to make recommendations for, it is no different to that of a longer profile without expansion.

Table 1. Results for TPR using the short user profiles with rules ranked by confidence

| Approach | Precision | % | Recall | % | F1-Measure | % |
|------------------------|-----------|--------|--------|--------|------------|--------|
| Baseline | 0.00619 | | 0.0571 | | 0.0112 | |
| Expanded (1 Top Rule) | 0.00649 | 4.77% | 0.0595 | 4.28% | 0.0117 | 4.72% |
| Expanded (2 Top Rules) | 0.00714 | 15.21% | 0.0655 | 14.66% | 0.0128 | 15.16% |
| Expanded (3 Top Rules) | 0.00732 | 18.15% | 0.0672 | 17.77% | 0.0132 | 18.12% |
| Expanded (4 Top Rules) | 0.00792 | 27.79% | 0.0729 | 27.75% | 0.0143 | 27.79% |
| Expanded (5 Top Rules) | 0.00815 | 31.54% | 0.0749 | 31.22% | 0.0147 | 31.51% |

To test the hypothesis that non-redundant rule sets perform as well as rule sets that contain redundancy we conducted a series of experiments to determine the improvement in a recommender system obtained using various rule sets.

We mine the transactional dataset using four different rule mining algorithms all with the same minimum support and confidence thresholds. Initially we used the MinMax algorithm to extract all of the possible rules, including redundant ones by using the proposed recovery algorithms [6]. However, the entire ruleset and the non-redundant ruleset generated by the MinMax algorithm are actually identical. This means that none of the rules discovered by the MinMax algorithm are considered redundant and thus the ruleset derived using MinMax becomes our baseline ruleset, which based on the ReliableBasis redundancy definition, contains redundant rules.

The other three algorithms all derive smaller rule sets indicating that they deem some of the rules that MinMax derived to actually be redundant. The ReliableBasis with HRR [8] [9] derives the smallest set of rules and thus ReliableBasis and MinMax with HRR still contain some redundant rules. Table 2 shows the size of each ruleset derived using these algorithms. Again we follow the same procedure previously outlined. The same 15,912 'short profile' users are then used to test the performance of the TPR recommender.

Table 3 clearly shows that the performance of the four algorithms is not that different, except for the case of the top 3 rules, where ReliableBasis with HRR (RBHRR) outperformed the worst rule mining algorithm, MinMax (MM) by 8%. We believe Table 3 strongly support our hypothesis that non-redundant rule sets can be used in place of larger rule sets which contain redundancy, without degrading performance. It also supports the theory behind these algorithms.

Table 2. Size of ruleset derived for each algorithm

| Algorithm | No. of rules | Reduction |
|------------------------|--------------|-----------|
| MinMax | 37,827 | |
| ReliableBasis | 36,852 | 2.58% |
| MinMax with HRR | 37,555 | 0.72% |
| ReliableBasis with HRR | 36,604 | 3.23% |

Table 3. Results for TPR using the short user profiles with different derived rule sets and rules ranked by confidence

| Approach | Precision | % | Recall | % | F1-Measure | % |
|--------------------------------|-----------|--------|--------|--------|------------|--------|
| Baseline | 0.00619 | | 0.0571 | | 0.0112 | |
| Expanded (1 Top Rule) - MM | 0.00649 | 4.77% | 0.0596 | 4.28% | 0.01171 | 4.72% |
| Expanded (1 Top Rule) - RB | 0.00649 | 4.77% | 0.0596 | 4.28% | 0.01171 | 4.72% |
| Expanded (1 Top Rule) - MMHRR | 0.0066 | 6.49% | 0.0606 | 6.04% | 0.0119 | 6.45% |
| Expanded (1 Top Rule) - RBHRR | 0.0066 | 6.49% | 0.0606 | 6.04% | 0.0119 | 6.45% |
| | | | | | | |
| Expanded (2 Top Rules) - MM | 0.00714 | 15.21% | 0.0655 | 14.66% | 0.0129 | 15.16% |
| Expanded (2 Top Rules) - RB | 0.00714 | 15.21% | 0.0655 | 14.66% | 0.0129 | 15.16% |
| Expanded (2 Top Rules) - MMHRR | 0.00717 | 15.72% | 0.0658 | 15.21% | 0.01293 | 15.67% |
| Expanded (2 Top Rules) - RBHRR | 0.0072 | 16.13% | 0.066 | 15.65% | 0.01298 | 16.08% |
| | | | | | | |
| Expanded (3 Top Rules) - MM | 0.00732 | 18.15% | 0.0673 | 17.77% | 0.0132 | 18.12% |
| Expanded (3 Top Rules) - RB | 0.00734 | 18.46% | 0.0674 | 18.1% | 0.01323 | 18.42% |
| Expanded (3 Top Rules) - MMHRR | 0.00772 | 24.65% | 0.0711 | 24.57% | 0.01393 | 24.64% |
| Expanded (3 Top Rules) - RBHRR | 0.00782 | 26.17% | 0.0721 | 26.17% | 0.01411 | 26.17% |
| | | | | | | |
| Expanded (4 Top Rules) - MM | 0.00792 | 27.79% | 0.073 | 27.75% | 0.01428 | 27.79% |
| Expanded (4 Top Rules) - RB | 0.00798 | 28.8% | 0.0736 | 28.8% | 0.0144 | 28.8% |
| Expanded (4 Top Rules) - MMHRR | 0.00805 | 29.92% | 0.0741 | 29.78% | 0.0145 | 29.91% |
| Expanded (4 Top Rules) - RBHRR | 0.00802 | 29.41% | 0.0738 | 29.21% | 0.01446 | 29.39% |
| | | | | | | |
| Expanded (5 Top Rules) - MM | 0.00815 | 31.54% | 0.0749 | 31.22% | 0.0147 | 31.51% |
| Expanded (5 Top Rules) - RB | 0.00819 | 32.15% | 0.0754 | 31.97% | 0.0148 | 32.13% |
| Expanded (5 Top Rules) - MMHRR | 0.00808 | 30.43% | 0.0743 | 30.07% | 0.01458 | 30.39% |
| Expanded (5 Top Rules) - RBHRR | 0.00811 | 30.83% | 0.0745 | 30.51% | 0.01462 | 30.8% |

5 Conclusions

In this paper we proposed the idea of using association rules to expand user profiles in order to improve recommendations. We outline an approach whereby the rules can be discovered and used, increasing the number of topics in a user profile that only has a few existing ratings. Our experiments show that the proposed approach can improve the performance of a recommender system under the cold-start problem. We also argued that the performance of non-redundant and redundant rulesets in this application should be very similar. Results obtained show that non-redundant rulesets, which contain fewer rules, performing on par with larger rulesets still containing redundancy.

Acknowledgements. Computational resources and services used in this work were provided by the HPC and Research Support Unit, Queensland University of Technology, Brisbane, Australia.

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, 734–749 (2005)
2. Burke, R.: Hybrid Recommender Systems: Survey and Experiments. *User Modelling and User-Adapted Interaction* 12, 331–370 (2002)
3. Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating Collaborative Filtering Recommender Systems. *ACM Transactions on Information Systems* 22, 5–53 (2004)
4. Leung, C.W., Chan, S.C., Chung, F.: Applying Cross-Level Association Rule Mining to Cold-Start Recommendations. In: *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology - Workshops*, Silicon Valley, California, USA, November 2007, pp. 133–136 (2007)
5. Middleton, S.E., Alani, H., Shadbolt, N.R., Roure, D.C.D.: Exploiting Synergy Between Ontologies and Recommender Systems. In: *The Semantic Web Workshop, World Wide Web Conference (WWW 2002)*, Hawaii, USA, May 2002, pp. 41–50 (2002)
6. Pasquier, N., Taouil, R., Bastide, Y., Stumme, G.: Generating a Condensed Representation for Association Rules. *Journal of Intelligent Information Systems* 24, 29–60 (2005)
7. Schein, A.I., Popescul, A., Ungar, L.H., Pennock, M.: Methods and Metrics for Cold-Start Recommendations. In: *25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2002)*, Tampere, Finland, August 2002, pp. 253–260 (2002)
8. Shaw, G., Xu, Y., Geva, S.: Eliminating Association Rules in Multi-level Datasets. In: *4th International Conference on Data Mining (DMIN 2008)*, Las Vegas, USA, July 2008, pp. 313–319 (2008)
9. Shaw, G., Xu, Y., Geva, S.: Extracting Non-Redundant Approximate Rules from Multi-Level Datasets. In: *20th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2008)*, Dayton, Ohio, USA, November 2008, pp. 333–340 (2008)
10. Weng, L.T., Xu, Y., Li, Y., Nayak, R.: Exploiting Item Taxonomy for Solving Cold-start Problem in Recommendation Making. In: *IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2008)*, Dayton, Ohio, USA, November 2008, pp. 113–120 (2008)
11. Xu, Y., Li, Y.: Generating Concise Association Rules. In: *16th ACM International Conference on Information and Knowledge Management (CIKM 2007)*, Lisbon, Portugal, November 2007, pp. 781–790 (2007)
12. Xu, Y., Li, Y., Shaw, G.: Concise Representations for Approximate Association Rules. In: *IEEE International Conference on Systems, Man & Cybernetics (SMC 2008)*, Singapore, October 2008, pp. 94–101 (2008)
13. Ziegler, C.N., Lausen, G., Schmidt-Thieme, L.: Taxonomy-driven Computation of Product Recommendations. In: *International Conference on Information and Knowledge Management (CIKM 2004)*, Washington, D.C., USA, November 2004, pp. 406–415 (2004)

Semi-supervised Tag Recommendation - Using Untagged Resources to Mitigate Cold-Start Problems

Christine Preisach, Leandro Balby Marinho, and Lars Schmidt-Thieme

Information Systems and Machine Learning Lab, University of Hildesheim, Germany
{preisach,marinho,schmidt-thieme}@ismll.uni-hildesheim.de

Abstract. Tag recommender systems are often used in social tagging systems, a popular family of Web 2.0 applications, to assist users in the tagging process. But in cold-start situations i.e., when new users or resources enter the system, state-of-the-art tag recommender systems perform poorly and are not always able to generate recommendations. Many user profiles contain untagged resources, which could provide valuable information especially for cold-start scenarios where tagged data is scarce. The existing methods do not explore this additional information source. In this paper we propose to use a purely graph-based semi-supervised relational approach that uses untagged posts for addressing the cold-start problem. We conduct experiments on two real-life datasets and show that our approach outperforms the state-of-the-art in many cases.

1 Introduction

Recently Web 2.0 applications like social tagging systems (or folksonomies) are getting more and more popular. One service often provided by these sites are tag recommender systems that help simplifying the process of tagging for the user. Given that users are free to tag, i.e., the same resource can be tagged differently by different people, it is important to personalize the recommended tags for an individual user. But state-of-the-art methods are not always able to suggest personalized tags for a new user or a new resource. Often these situations are handled by using the content of the new resource or just recommending the most popular tags. But these approaches have several drawbacks, first the recommended tags are not personalized and second, using content is not a generic approach, one needs to use different algorithms for each type of resource, e.g., in Last.fm¹ information needs to be extracted from the audio files, in Flickr² images need to be analyzed, in YouTube³ knowledge from videos has to be extracted and for Delicious⁴ and BibSonomy⁵ the text of bookmarked web pages or publications belonging to a BIBTEX entry needs to be assessed. Thus, this is a costly solution if one needs different approaches for several types of resources.

We propose a content independent, purely graph-based approach, which is based on the observation that user profiles usually contain many untagged posts that could be

¹ <http://www.last.fm>

² <http://www.flickr.com>

³ <http://www.youtube.com>

⁴ <http://delicious.com>

⁵ <http://www.bibsonomy.org>

exploited for improving the recommendations, especially when there are only a few tagged examples available. We investigate two scenarios, first where a new user enters the system and second, where among the untagged posts, there are new resources, i.e., resources that were not tagged by any other user in the system. We will address these problems by means of semi-supervised relational classification, whereby we can benefit from the structural information of untagged posts.

As presented in [1], we first cast the problem of personalized tag recommendations as a relational classification problem, where we use relational semi-supervised classification to profit from the potentially valuable information carried by other untagged resources. In contrast to our approach submitted to the ECML/PKDD Discovery Challenge 2009 (task 2) that achieved the second place, in this paper we focus on the *cold-start* problem (that did not occur in the challenge dataset).

In this paper our contributions are as follows:

1. We formally define the *cold-start* (in terms of new user/resource) problem in social tagging systems.
2. To address this problem we propose and compare different semi-supervised relational methods, which exploit the structural information of untagged posts in the post graph.
3. Finally, we show empirically that our approaches outperform the current state-of-the-art algorithms (FolkRank and PITF, a tensor factorization model), as well as other simpler baselines such as KNN and most popular tags in many cases.

2 Related Work

In [2] the authors compared several personalized tag recommendation algorithms, the best results, were achieved by the FolkRank algorithm [3], an adaptation of PageRank for retrieving information and recommending tags in social tagging systems. More recently Rendle et al. [4] introduced RTF (Ranking with Tensor Factorization), a method for learning a tensor factorization model optimized for the best personalized tag ranking. The model also handles missing values by introducing a new interpretation of the data and learns from pairwise ranking constraints through a gradient descent algorithm. The prediction runtime is independent of the number of observations and only depends on the factorization dimensions but the training time is huge. Another new factorization model for tag recommendation PITF (Pairwise Interaction Tensor Factorization) was introduced in [5,6], it tries to find latent interactions between users, items and tags by factorizing the observed tagging data. Similar to [4] the model is learned by optimizing the Bayesian Personal Ranking method with gradient decent. Although the methods discussed above provide high quality recommendations, they are not robust against new user/resource scenarios. Furthermore, RTF and PITF strictly operate over ternary relations, and thereby are not able to exploit the information of untagged posts. For item recommendation a semi-supervised approach for cold-start problems has been recently published, the authors of [7] have introduced the tied Boltzmann machine that captures pairwise interactions between items. To our best knowledge for cold-start problems in tag recommendation no graph-based, semi-supervised approach has been introduced so far.

Since tagging data forms relations between users, resources and tags, it is natural to exploit these relations by adapting relational methods to the tag recommendation

scenario, in [1] we showed that relational classification methods perform very well on the ECML/PKDD Discovery Challenge 2009 dataset (which did not contain new users or resources). Originally relational methods have been applied to areas where entities are linked in an explicit manner, like hypertext documents and scientific publications. Especially iterative semi-supervised relational methods, which use *collective inference* and exploit relational autocorrelation of class labels of related entities, received attention. One of the earliest iterative semi-supervised relational approaches was proposed by Chakrabarti et al. [8], where a probabilistic model for classification of web pages was introduced. In [9,10] the authors presented different semi-supervised iterative models and showed that collective inference increases accuracy.

Here we will focus on the *cold-start problem*, where we expect that simple iterative semi-supervised relational methods outperform supervised approaches, since they allow the usage of unlabeled data, which is particularly important for *cold-start* scenarios.

3 Tag Recommendations

In this section we formalize the general problem of tag recommendations in social tagging systems, and formalize the new user/resource problem.

3.1 Problem Formulation

Social tagging systems data usually comprises a set of users U , a set of resources R , a set of tags T , and a set Y of ternary relations between them, i.e., $Y \subseteq U \times R \times T$. Let

$$X := \{(u, r) \in U \times R \mid \exists t \in T : (u, r, t) \in Y\}$$

be the set of all posts in the data. Let $T(x = (u, r)) := \{t \in T \mid (u, r, t) \in Y\}$ be the set of all tags assigned to a given post $x \in X$. We consider train/test splits based on posts, i.e., $X_{\text{train}}, X_{\text{test}} \subset X$ are disjunct and covering all of X : For training, the learner has access to the set X_{train} of training posts and their true tags $T|_{X_{\text{train}}}$. Semi-supervised methods also could exploit the set X_{test} of untagged posts, but of course not their associated true tags. The tag recommendation task is then to predict, for a given $x \in X_{\text{test}}$, a set $\hat{T}(x) \subseteq T$ of tags that are most likely to be used by the respective user for the respective resource.

3.2 New User/Resource

An issue that remains unaddressed by the current literature on personalized tag recommendations refers to the new user/resource problems. A new user u refers to the user who posted for the first time in the system, i.e.,

$$|X_{\text{test}} \cap (\{u\} \times R)| \geq 1 \text{ and } X_{\text{train}} \cap (\{u\} \times R) = \emptyset$$

In other words, all posts of a new user are in the test set. A new resource r , on the other hand, refers to a resource that has never been tagged before by any other user:

$$X_{\text{train}} \cap (U \times \{r\}) = \emptyset$$

Currently, there are no suitable purely graph-based⁶ approaches for providing recommendations whenever these situations occur. Unpersonalized content-based approaches are usually used in such scenarios, but since the resource' format can vary across different social tagging systems one would need to develop a specific method for each possible kind of resource.

4 Semi-supervised Relational Methods

In this section we present the types of relations we use and introduce several semi-supervised relational methods for tag recommendation. Here we especially focus on the *new user/resource* scenario.

We propose to represent folksonomy data as a homogeneous, undirected relational graph over the post set, i.e., $G := (X, E)$ in which edges are annotated with a weight $w : X \times X \rightarrow \mathbb{R}$ denoting the strength of the relation. Besides making the input data more compact – we have only a binary relation $\mathcal{R} \subseteq X \times X$ between objects of the same type – this representation will allow us to cast the problem of personalized tag recommendations as a relational classification problem.

Relational classifiers usually consider, relations between objects instead of only taking into account the conventional attribute-value data of objects. A scientific paper, for example, can be related to another paper that has been written by the same author or because they share common citations. It has been shown that relational classifiers usually perform better than purely attribute-based classifiers [8,11,12].

In our case, we assume that posts are related to each other if they share the same user: $\mathcal{R}_{\text{user}} := \{(x, x') \in X \times X \mid \text{user}(x) = \text{user}(x')\}$ or the same resource: $\mathcal{R}_{\text{res}} := \{(x, x') \in X \times X \mid \text{res}(x) = \text{res}(x')\}$ as an alternative we can use both relations together, i.e., posts either share the same user or resource (see Figure 2): $\mathcal{R}_{\text{user}}^{\text{res}} := \mathcal{R}_{\text{user}} \cup \mathcal{R}_{\text{res}}$. For convenience, let $\text{user}(x = (u, r)) := u$ and $\text{res}(x = (u, r)) := r$ denote the user and resource of post x respectively. Iterative relational methods have been shown to work very well because of the following three assumptions [13]: first, the first-order Markov assumption, i.e., in the tag recommendation scenario, only the direct neighborhood is necessary for accurate tag recommendations, second, the assumption of homophily, i.e., similar posts are more likely to be tagged alike and third, the assumption of simple belief propagation, i.e., tags can be propagated to untagged posts.

We are especially interested in the situation where related posts are untagged, thus differently from other approaches (e.g., [24]) that use X_{train} and $T|_{X_{\text{train}}}$ allowing us to exploit the structural information of untagged posts using semi-supervised iterative relational methods. One simple relational classification method for tag recommendation is the *WeightedAverage* (WA) which sums up all weights of neighboring posts $x' \in N_x$ that share the same tag $t \in T$ and normalizes this by the sum over all weights in the neighborhood:

$$P(t|x) = \frac{\sum_{x' \in N_x | t \in T(x')} w(x, x')}{\sum_{x' \in N_x} w(x, x')} \quad (1)$$

⁶ By graph-based we mean algorithms that do not rely on resources' content but only on the graph induced by the folksonomy data.

with $N_x := \{x' \in X \mid (x, x') \in \mathcal{R}\}$ being the neighborhood. This algorithm (in the following denoted as *WA*) is similar to collaborative filtering, which is based on the *k*-Nearest Neighbor algorithm, the difference here is that *k*, does not need to be determined but is given by the number of neighbors.

But what if some of the neighboring posts are untagged, how should we handle this situation? State-of-the-art methods just ignore untagged posts. Semi-supervised iterative methods in contrast do exploit them and thus, increase classification accuracy. One simple way of considering untagged posts is to transform *WA* into an iterative algorithm, i.e., in the first iteration we classify test posts by only using direct neighbors from the training set, in the second iteration, the still unclassified test posts are classified by extracting tag information from neighbors that have been classified in the previous iteration. The procedure stops when all the test instances are classified. This iterative version of *WA* is denoted as *WAOneShot* since all test posts are classified only once, i.e., already classified posts are not re-classified in the following iteration. Note that eq. (1) considers the tags of the neighborhood in a deterministic way i.e., probabilities are not taken into account, so that even if the probability of a estimated tag is very low it is considered in the same way as a high probability tag. To overcome this limitation one can extend eq. (1) to *PWA* (*Probabilistic Weighted Average*):

$$P(t|x) = \frac{\sum_{x' \in N_x} w(x, x') P(t|x')}{\sum_{x' \in N_x} w(x, x')} \quad (2)$$

Now, instead of only summing up edge weights of direct neighbors, we additionally take into account the probability of the tags belonging to those neighbors. Combining eq. (2) with the aforementioned iterative algorithm leads to a probabilistic semi-supervised iterative method which makes use of the uncertainty of tag estimations in previous iterations. This algorithm is denoted as *PWAOneShot*. The algorithms, *WA*, *WAOneShot*, *PWA* and *PWAOneShot* use only the first two properties of relational methods, namely the first-order assumption and the homophily but not the third property. Thus, for both algorithms test posts are not re-classified even if tags of neighbors have changed, i.e., the information cannot be spread in the graph.

To resolve this problem relaxation labeling [8] can be used, i.e., we make use of the third property of simple belief propagation, here the tag probability of all test posts are re-estimated simultaneously in each iteration, i.e., the information spreads in the graph which helps to increase accuracy. *PWA* combined with relaxation labeling is denoted as *PWA**. The algorithm for *PWA** is depicted in Figure 1. There we first initialize the untagged posts with the prior probability calculated using the training set, then we compute the probability of each tag *t* given *x* iteratively using *PWA*. The procedure stops when the algorithm converges (i.e., the difference of the tag probability between iteration *i* and *i* + 1 is less than a very small ϵ) or a certain number of iterations is reached. Note that eq. (1) and eq. (2) have been introduced in [14] and applied to relational datasets. The weight *w* in eq. (1) and (2) is an important factor in the estimation of tag probabilities, since it describes the strength of the relation between *x* and *x'*. We used the weight schemes described in [1]. Since we want to recommend more than one tag we need to cast the tag recommendation problem as a multilabel classification problem, i.e., assign one or more tags to a test post. We accomplish the multilabel problem by

```

(1) learn-PWA*(Xtrain, Xtest, T, ε) :
(2) P(t|x)0 := 1, P(t'|x) := 0 for all x ∈ Xtrain, t ∈ T(x), t' ∉ T(x)
(3) P(t|x)0 := prior(t) for all x ∈ Xtest, t ∈ T
(4) for i := 0, ..., I do
(5)   for x ∈ Xtest do
(6)     for t ∈ T do
(7)       P(t|x)i+1 := 1/2 ∑x' ∈ Nx w(x, x') P(t|x')i
(8)     od
(9)   od
(10) od until √(1/|Xtest| ∑x ∈ Xtest ∑t ∈ T (p(t|x)i+1 - p(t|x)i)2) < ε
(11) return (P(t|x))x ∈ Xtest, t ∈ T

```

Fig. 1. Algorithm PWA*

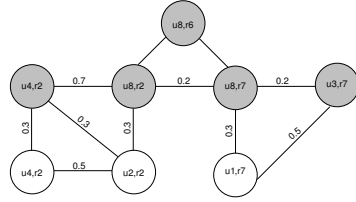


Fig. 2. Relational graph for the new user u_8 and new resource r_6

sorting the calculated probabilities $P(t|x)$ for all $x \in X_{\text{test}}$ and recommend the top n tags with highest probabilities.

In terms of runtime complexity, PWA* is in $O(I \cdot (|T| \cdot N_x))$ for prediction and $O(1)$ for training. I.e., the runtime is only dependent on the number of iterations, number of tags and the size of the neighborhood.

4.1 Cold-Start Problem

Semi-Supervised relational classification is especially useful for addressing cold-start problems where users have untagged resources in their profiles, since, in contrast to the current state-of-the-art, it is able to exploit the structural information of untagged posts and to propagate this information in the post graph. In general our semi-supervised approach is able to extract information from two sources and two relations, from the tagged posts and untagged posts over \mathcal{R}_{res} or $\mathcal{R}_{\text{user}}$ which is very beneficial for the coldstart problem.

In figure 2 we illustrate the new user/resource problem. The gray nodes in the given post graph represent the untagged posts, the white nodes belong to the training set. In our example user u_8 is a new user, she has several untagged posts. In order to recommend tags for post (u_8, r_7) for example, we can make use of both information sources, training and test set. First, through \mathcal{R}_{res} with (u_1, r_7) from the training set and (u_3, r_7) from the test set (this post is an untagged post of user u_3). Second, through $\mathcal{R}_{\text{user}}$ with her own untagged posts (u_8, r_2) and (u_8, r_6) . Thus, the system can profit from both, the training set over the resource relation and from the untagged posts belonging to the users own profile. For the new resource problem in contrast we cannot use \mathcal{R}_{res} , but only $\mathcal{R}_{\text{user}}$. In figure 2 r_6 is a new resource, so if we want to recommend tags for (u_8, r_6) one can exploit (u_8, r_2) and (u_8, r_7) . Although (u_8, r_2) and (u_8, r_7) are initially untagged, our methods still benefit from them, because they are connected to other posts and this information spreads over the graph. Since our graph can be composed by two kinds of relations at the same time, when one relation is missing (e.g., new user or resource), there is always another relation as backup. The only scenario where this does not hold is when all the resources uploaded by a new user are new.

5 Experiments

The main issues we want to address here is the new users/resources scenario. We conduct two main experiments to show that our semi-supervised methods are able to cope

Table 1. Characteristics of 5-core BibSonomy and 10-core Last.fm

| Dataset | $ U $ | $ R $ | $ T $ | $ Y $ | $ X $ | $ E_{\text{user}} $ | $ E_{\text{res}} $ | $ E_{\text{user}}^{\text{res}} $ |
|-----------|-------|-------|-------|---------|--------|---------------------|--------------------|----------------------------------|
| BibSonomy | 116 | 361 | 412 | 10,148 | 2,522 | 64,669 | 9,108 | 73,777 |
| Last.fm | 2,917 | 1,853 | 2,045 | 219,702 | 75,565 | 1,088,023 | 4,149,862 | 5,237,885 |

with these issues. We compare several semi-supervised relational models (*WAOneShot*, *PWAOneShot*, *PWA**) with state-of-the-art methods like WA, FolkRank⁷, PITF (tensor factorization model), and the most popular tags on two real-world datasets.

5.1 Datasets

In order to evaluate our approach we use two real-life datasets, BibSonomy and Last.fm⁸. BibSonomy is a social tagging system that allows users to manage and annotate bookmarks and publication references simultaneously. Last.fm on the other hand, is a social online radio station where people can upload, share and tag music/artists/albums they like. Since these systems represent different domains and are evtl. used by different people, we assume that our findings can also be carried over to other social tagging systems. We follow the conventional approach of using the dense part of Y by means of a p -core⁹. Similarly to [24], we used the 5-core for BibSonomy and the 10-core for Last.fm. Table 1 summarizes the characteristics of the datasets we used. For convenience, let $|E_{\text{res}}|$, $|E_{\text{user}}|$ and $|E_{\text{user}}^{\text{res}}|$ denote the number of edges according to the \mathcal{R}_{res} , $\mathcal{R}_{\text{user}}$, $\mathcal{R}_{\text{user}}^{\text{res}}$ relations respectively.

5.2 Experiment Setting

We analyzed two situations, one where only new users, and a second where new users and new resources were present in the data. For the first situation (new user problem), the test set is composed by only new users (we sampled 30% of the users in U to be in the test set, the rest is used for training) but no new resources. For the second scenario, i.e., new user and new resource problem, we sampled for each user a percentage of test resources to be new. We evaluated our methods on data where 1% and 10% of the test posts contain new resources. In reality many users have untagged resources in their profiles but those untagged posts are usually removed from the standard datasets, thus we simulated this situation. We use the standard *LeavePostOut* [2] protocol, but additionally exploit the untagged posts, i.e., while recommending tags for one post, the other sampled posts are used as untagged posts (their tags are removed).

We used the standard $F1$ measure on top-5 tag lists, similar to [5] we estimate the optimal number of tags to be recommended (i.e., we do not always recommend 5 tags). As in [1] we rewarded the best relation by a weight of d ¹⁰. We optimized c as well as

⁷ Parameters $d := 0.4$, #iterations:=10.

⁸ We have used the same data snapshots as in [24].

⁹ A p -core of X is the largest subset of X where each user, resource and tag must occur in at least p posts.

¹⁰ BibSonomy $c = 2.5$ for \mathcal{R}_{res} , Last.fm $c = 2.5$ for $\mathcal{R}_{\text{user}}$.

the hyperparameters for PITF on a holdout set. Moreover, we restricted the maximum number of iterations for the PWA^* algorithm to 75.

5.3 Results and Discussion

New User Problem. Figure 3 shows the results achieved with PWA^* , $WAOneShot$, $PWAOneShot$, FolkRank, PITF, WA and the most popular [1] algorithm for various numbers of recommendations (1-5) applied to BibSonomy. All users in the test set are new but all resources are known in the system already. This is reflected in the results, the difference among the results is small, $PWAOneShot$, $WAOneShot$ and WA perform very similar. WA performs well since every test post is connected to at least one training post, thus recommendation of tags is possible for each test post. But even in this situation where there are more connections between test and training set, PWA^* outperforms the other algorithms, i.e., can still profit from label propagation and initialization of test posts. The situation changes for Last.fm (see Figure 4). Here both FolkRank and PITF outperform all other methods. PWA^* performs best among the semi-supervised methods, but seems not to profit so much from label propagation and initialization of test posts. $PWAOneShot$, $WAOneShot$ and WA achieve again very similar results. The reason lies in the nature of this dataset, here the user relation contains the more valuable information, the same phenomena was observed in [2]. So, in this case, proposing tags that the user already used in the past instead of tags other users attached to the resource, may provide a better chance to suggest the tags the user finally chose. Since, the most valuable tag information is contained in \mathcal{R}_{user} , but all the users in the test set are new and \mathcal{R}_{res} yields low quality recommendations, unpromising labels are propagated, hence leading to poor results. FolkRank and PITF on the other hand, does not suffer from ill propagated labels and moreover, FolkRank can explore other users, resources and tags that are only indirectly connected to the test posts, which in this dataset at least, seems to yield a great benefit.

New User/Resource. Figure 5 shows the results for the second scenario on the BibSonomy dataset, where both the new user and new resource problem occurs. In general one can see that as the number of new resources increases the results deteriorate, but in both cases (1% and 10% of new resources) the semi-supervised relational methods outperform the state-of-the-art methods, while PWA^* achieves the best result. As expected WA does not perform very well since some test posts are only connected to other test posts, so that in some cases it cannot recommend any tags. The reason why PWA^* performs particularly good in this dataset, is because here the \mathcal{R}_{res} relation already contains usefull tag assignments, and since this relation is the only training information available, it leads to the propagation of promising labels. FolkRank performs similar to WA and $WAOneShot$ but is slightly outperformed by PITF. Again, for Last.fm (see figure 6) things are a little different. For both situations (1% and 10% of the resources are new), PWA^* still achieves the best result. In the 1% case $PWAOneShot$ too, achieves better results than the state-of-the-art. PWA^* and $PWAOneShot$ perform better because they use probabilities, which is not the case for $WAOneShot$. WA performs poorly, since, for many

¹¹ This baseline refers to the *most popular tags* of the folksonomy, i.e., it recommends, for any user $u \in U$ and any resource $r \in R$, the same set: $\hat{T}(u, r) := \operatorname{argmax}_{t \in T}^n (|Y_t|)$.

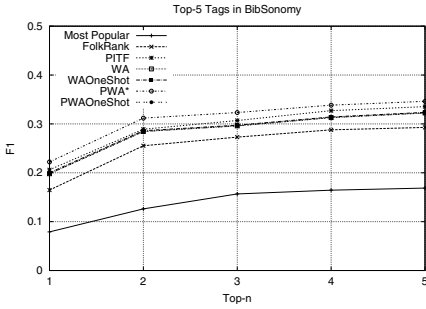


Fig. 3. New user problem BibSonomy

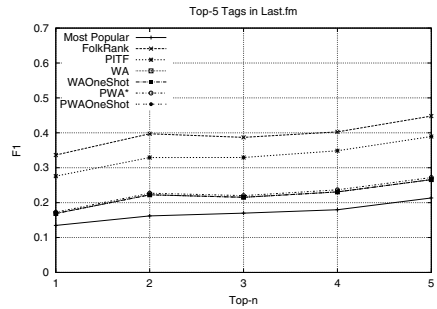


Fig. 4. New user problem Last.fm

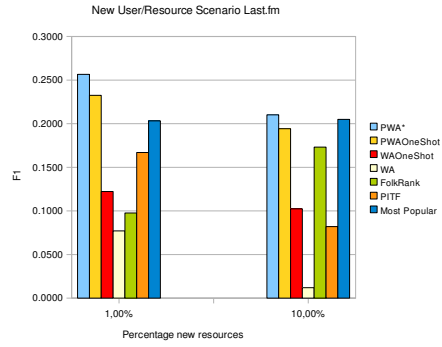
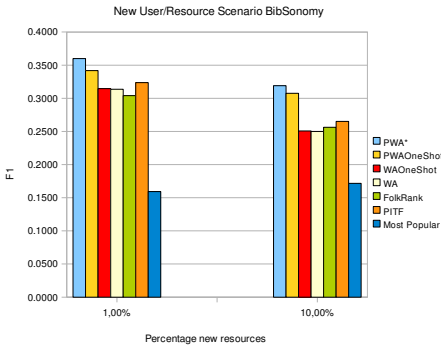


Fig. 5. New user/resource problem BibSonomy

Fig. 6. New user/resource problem Last.fm

test posts no tags can be recommended since many test instances are only connected to test posts. FolkRank and PITF on the other hand, do not maintain the same performance achieved on the new user problem. This happens because the number of connected tag assignments to the test posts decreases proportionally to the number of new resources inside the posts of a new user, thus making it difficult to compute a good set of tags. In this particular situation, the simple most popular method performs surprisingly good, showing that in special cold-start cases, like this, it is a good alternative.

In general the re-estimation and propagation of labels in the post graph as well as the initialization of test posts seems to be the main reason for the good results of *PWA** (since *PWA** performed better than *PWAOneShot*). Furthermore, we see that the new user problem is easier to handle than situations where both new users and new resources occur, since the graph is less sparse, and therefore supervised methods work almost as well (or better) as semi-supervised methods.

6 Conclusions and Future Work

In this paper we have introduced an approach for tag recommendations that is particularly suitable for the cold-start problem. Our model is based on semi-supervised relational classification, that allows to exploit the structural information of untagged

posts. We evaluated our approach against state-of-the-art methods in two real world datasets. We showed that semi-supervised relational methods which are based on label propagation are achieving very good results. In some special cases though, where the available training relations are of low quality, unpromising labels can be propagated thus deteriorating the results. In future work we want to investigate automatic ways of detecting more informative relations as well as other semi-supervised methods and new kinds of relations between the posts (e.g. content-based) for further improvement of cold-start related issues.

Acknowledgements

This work was funded by the X-Media project (www.x-media-project.org) sponsored by the European Commission as part of the Information Society Technologies (IST) programme under EC grant number IST-FP6-026978. The authors gratefully acknowledge the partial co-funding of their work through the European Commission FP7 project MyMedia (www.mymediaproject.org) under the grant agreement no. 215006. For your inquiries please contact info@mymediaproject.org.

References

1. Marinho, L.B., Preisach, C., Schmidt-Thieme, L.: Relational classification for personalized tag recommendation. In: ECML/PKDD Discovery Challenge 2009 at ECML/PKDD (2009)
2. Jaeschke, R., Marinho, L., Hotho, A., Schmidt-Thieme, L., Stumme, G.: Tag recommendations in social bookmarking systems. *AI Communications*, 231–247 (2008)
3. Hotho, A., Jaeschke, R., Schmitz, C., Stumme, G.: Information retrieval in folksonomies: Search and ranking. In: Sure, Y., Domingue, J. (eds.) *ESWC 2006*. LNCS, vol. 4011, pp. 411–426. Springer, Heidelberg (2006)
4. Rendle, S., Marinho, L.B., Nanopoulos, A., Schimdt-Thieme, L.: Learning optimal ranking with tensor factorization for tag recommendation. In: *KDD 2009*, pp. 727–736. ACM, New York (2009)
5. Rendle, S., Schmidt-Thieme, L.: Factor models for tag recommendation in bibsonomy. In: *ECML/PKDD Discovery Challenge 2009 at ECML/PKDD (2009)*
6. Rendle, S., Schmidt-Thieme, L.: Pairwise interaction tensor factorization. In: *WSDM 2010 (2010)*
7. Gunawardana, A., Meek, C.: Tied boltzmann machines for cold start recommendations. In: *RecSys 2008*, pp. 19–26. ACM, New York (2008)
8. Chakrabarti, S., Dom, B.E., Indyk, P.: Enhanced hypertext categorization using hyperlinks. In: Haas, L.M., Tiwary, A. (eds.) *SIGMOD 1998*, pp. 307–318. ACM Press, New York (1998)
9. Jensen, D., Neville, J., Gallagher, B.: Why collective inference improves relational classification. In: *KDD 2004*, pp. 593–598. ACM, New York (2004)
10. Sen, P., Namata, G., Bilgic, M., Getoor, L., Gallagher, B., Eliassi-Rad, T.: Collective classification in network data. *AI Magazine* (2008)
11. Lu, Q., Getoor, L.: Link-based classification using labeled and unlabeled data. In: *Workshop on the continuum from labeled to unlabeled data, ICML 2003 (2003)*
12. Preisach, C., Schmidt-Thieme, L.: Relational ensemble classification. In: *ICDM 2006*, pp. 499–509 (2006)
13. Macskassy, S.A., Provost, F.: Simple models and classification in networked data. In: *CeDER Working Paper 03-04*, Stern School of Business (2004)
14. Macskassy, S., Provost, F.: A simple relational classifier. In: *Workshop on Multi-Relational Data Mining, KDD 2003*, pp. 64–76 (2003)

Cost-Sensitive Listwise Ranking Approach

Min Lu¹, MaoQiang Xie^{2,*}, Yang Wang¹, Jie Liu¹, and YaLou Huang^{1,2}

¹ College of Information Technology Science, Nankai University, Tianjin, China

² College of Software, Nankai University, Tianjin, China

{lumin,wangyang022,jliu}@mail.nankai.edu.cn,

{xiemq,huangyl}@nankai.edu.cn

Abstract. This paper addresses listwise approaches in learning to rank for Information Retrieval(IR). The listwise losses are built on the probability of ranking a document highest among the documents set. The probability treats all the documents equally. However, the documents with higher ranks should be emphasized in IR where the ranking order on the top of the ranked list is crucial. In this paper, we establish a framework for cost-sensitive listwise approaches. The framework redefines the probability by imposing weights for the documents. The framework reduces the task of weighting the documents to the task of weighting the document pairs. The weights of the document pairs are computed based on Normalized Discounted Cumulative Gain(NDCG). It is proven that the losses of cost-sensitive listwise approaches are the upper bound of the NDCG loss. As an example, we propose a cost-sensitive ListMLE method. Empirical results shows the advantage of the proposed method.

1 Introduction

Learning to rank is a popular research area due to its widespread applications. This paper focuses on document retrieval. When learning to rank has been applied to document retrieval, it aims to learn a real-valued ranking function that induces a ranking order over the documents set of a query.

The existing ranking approaches can be summarized into three categories: pointwise, pairwise and listwise. The pointwise and pairwise methods[1,2] transform ranking problem into regression or classification problem. The listwise approaches[3,4,5] minimize the loss functions defined between the ranked list and the ground truth list. Theoretical analysis about the properties of the listwise loss functions are also conducted. Fen Xia[3] studied the consistency and soundness of the listwise loss functions, and Yanyan Lan[6] investigated the generalization bound of the listwise loss functions based on Rademacher Averages.

Many listwise algorithms are developed, but no research is conducted to elaborate the common characteristic of the listwise loss functions. What is more, the listwise losses are inadequate for IR where the high performance of the top documents in the ranked list is preferred, measure by NDCG. In this paper, we propose a framework for cost-sensitive listwise approaches. The cost-sensitive

* Corresponding author.

listwise loss functions are constructed based on the documents with weights. The framework reduces the task of setting weights for the documents to the task of setting weights for the document pairs. The weights of the document pairs are computed based on the NDCG. As an example, we develop a cost-sensitive ListMLE algorithm. Experimental results show that the proposed method outperforms ListMLE[3], RankCosine[5], AdaRank[7] and Ranking SVM[2].

2 Normalized Discount Cumulative Gain (NDCG)

NDCG[8] evaluates the performance of the top documents in the ranked list. Suppose n candidate documents are retrieved for a query. Each candidate document d_i is represented as a pair (\mathbf{x}_i, y_i) , where \mathbf{x}_i denotes the query-document pair feature vector and y_i is the relevance level. The NDCG@k is defined as:

$$NDCG@k = \frac{DCG_{\pi}@k}{DCG_g@k} = \frac{\sum_{j=1}^n \frac{2^{y_j} - 1}{\log_2(1 + \pi(j))} I[\pi(j) \leq k]}{\sum_{j=1}^n \frac{2^{y_j} - 1}{\log_2(1 + g(j))} I[g(j) \leq k]} \tag{1}$$

where k denotes the truncation level. π is the ranked list generated by a ranking function. g denotes the ground truth list obtained in the document relevance level descending order. $g(j)$ and $\pi(j)$ denote the ground truth ranking position and the ranked position of the document d_j respectively. $I[x]$ yields one if x is true and zero otherwise. Assume that the ranking function is f , then the ranked position $\pi(j)$ is computed from:

$$\pi(j) = 1 + \sum_{i=1}^n I[f(\mathbf{x}_i) > f(\mathbf{x}_j)] \tag{2}$$

where $f(\mathbf{x}_i)$ denotes the rank score of the document d_i . If there are many documents sharing the same relevance level with the document d_j , it is impossible to state the definite value of $g(j)$. So the approximate value $\hat{g}(j)$ is given.

$$\hat{g}(j) = 1 + \sum_{i=1}^n I[y_i > y_j] \tag{3}$$

It is obvious that $\hat{g}(j) \leq g(j)$ for each j , which implies the inequality.

$$DCG_g@k \leq DCG_{\hat{g}}@k \tag{4}$$

Then the NDCG@k loss, abbreviated $L_{ndcg@k}$, has a upper bound.

$$L_{ndcg@k} = 1 - NDCG@k \leq \frac{1}{DCG_{\hat{g}}@k} (DCG_{\hat{g}}@k - DCG_{\pi}@k) \tag{5}$$

3 Listwise Approach

3.1 Existing Listwise Loss Function

The listwise approaches take documents lists as an instance, and minimize the loss functions defined between the ranked list and the ground truth list. Typical methods include ListNet [4], RankCosine [5] and ListMLE [3]. We review their loss functions. For the sake of describing simply, the documents have been ranked by the relevance level in descending order, *i.e.*, $y_1 \succeq y_2 \dots \succeq y_n$.

$$\begin{aligned}
 \text{RankCosine} \quad L &= \frac{1}{2} \left(1 - \frac{\sum_{j=1}^n \phi(y_j) \phi(f(\mathbf{x}_j))}{\sqrt{\sum_{j=1}^n \phi(y_j)^2} \sqrt{\sum_{j=1}^n \phi(f(\mathbf{x}_j))^2}} \right) \\
 \text{ListMLE} \quad L &= -\log \prod_{i=1}^n \frac{\exp(\phi(f(\mathbf{x}_i)))}{\sum_{j=i}^n \exp(\phi(f(\mathbf{x}_j)))} \\
 \text{ListNet} \quad L &= -\sum_{\pi \in \mathcal{Y}} \prod_{i=1}^n \frac{\phi(\psi_y(\mathbf{x}_{\pi(t_i)}))}{\sum_{j=i}^n \phi(\psi_y(\mathbf{x}_{\pi(t_j)}))} \log \prod_{i=1}^n \frac{\phi(f(\mathbf{x}_{\pi(t_i)}))}{\sum_{j=i}^n \phi(f(\mathbf{x}_{\pi(t_j)}))}
 \end{aligned}$$

where ϕ is a positive and strictly increasing function. $f(\mathbf{x}_i)$ is the rank score of the document d_i computed by the ranking function f . ψ_y is a mapping function defined on the relevance level and preserves ground truth list, *i.e.*, $\psi_y(y_1) \geq \psi_y(y_2) \dots \psi_y(y_n)$. $\mathbf{x}_{\pi(t_i)}$ denotes the query-document pair feature vector of the document with ranked position being at i .

3.2 Analysis of Listwise Loss Function

Several listwise approaches are developed, but no analysis is studied to elaborate the common characteristic of the listwise loss functions. In this study, we point out that the listwise loss functions are in essence built upon the probability of ranking a document highest among the documents set, defined as

$$h(\mathbf{x}_i | \mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n; \psi) = \frac{\exp(\psi(f(\mathbf{x}_i)))}{\sum_{j=1}^n \exp(\psi(f(\mathbf{x}_j)))} \tag{6}$$

where ψ denotes an increasing function. The existing listwise loss functions can be expressed in terms of the function h .

$$\begin{aligned}
 \text{RankCosine} \quad L &= \frac{1}{2} \left(1 - \sum_{j=1}^n \frac{\phi(y_j)}{\sqrt{\sum_{i=1}^n \phi(y_i)^2}} \cdot \sqrt{h(\mathbf{x}_j | \mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n; 2 \log \phi)} \right) \\
 \text{ListMLE} \quad L &= -\log \prod_{i=1}^n h(\mathbf{x}_i | \mathbf{x}_i, \mathbf{x}_{i+1}, \dots, \mathbf{x}_n; \phi) \\
 \text{ListNet} \quad L &= -\sum_{\pi \in \mathcal{Y}} \prod_{i=1}^n \frac{\phi(\psi_y(\mathbf{x}_{\pi(t_i)}))}{\sum_{j=i}^n \phi(\psi_y(\mathbf{x}_{\pi(t_j)}))} \log \prod_{i=1}^n h(\mathbf{x}_{\pi(t_i)} | \mathbf{x}_{\pi(t_{i+1})} \dots, \mathbf{x}_{\pi(t_n)}; \phi)
 \end{aligned}$$

To minimize the listwise loss function is equivalent to minimizing the function h . The reason for the goodness of these loss functions is that h is closely related to the pairwise classification error. The conclusion is simply proved in the following.

$$\begin{aligned} \frac{1}{n-i} \sum_{j=i+1}^n I[\phi(f(\mathbf{x}_j)) > \phi(f(\mathbf{x}_i))] &\leq \frac{1}{n-i} \sum_{j=i+1}^n \log_2 [1 + \exp(\phi(f(\mathbf{x}_j)) - \phi(f(\mathbf{x}_i)))] \\ &\leq \log_2 \left(1 + \frac{1}{n-i} \sum_{j=i+1}^n \exp(\phi(f(\mathbf{x}_j)) - \phi(f(\mathbf{x}_i))) \right) \leq -\log_2 h(\mathbf{x}_i | \mathbf{x}_i, \mathbf{x}_{i+1}, \dots, \mathbf{x}_n; \phi) \end{aligned}$$

The $I[\phi(f(\mathbf{x}_j)) > \phi(f(\mathbf{x}_i))]$ is the pairwise classification error because of $y_i \succeq y_j$. The above inequalities can be verified with $I[z > 0] \leq \log_2(1 + \exp(z))$ and $\frac{1}{n} \sum_{j=1}^n \log_2(z_j) \leq \log_2(\sum_{j=1}^n z_j/n)$ when each z_j is nonnegative. The [1] shows that the pairwise classification error is inadequate for IR where the ranking order on the top of the ranked list is crucial. Hence, the listwise losses are also inadequate for IR since their key component h is closely related to the pairwise classification error.

4 Cost-Sensitive Listwise Approach Framework

4.1 Cost-Sensitive Listwise Loss Function

To make the listwise losses focus on the ranking order on the top of the ranked list, a good way is to take account of cost-sensitive in the listwise losses, more precisely, to set different weights for the documents. The function h is redefined by imposing weights for the documents:

$$h(\mathbf{x}_i | \mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n; \psi, \boldsymbol{\alpha}_i) = \frac{\alpha_{i,i} \exp(\psi(f(\mathbf{x}_i)))}{\sum_{j=1}^n \alpha_{i,j} \exp(\psi(f(\mathbf{x}_j)))} \tag{7}$$

where $\boldsymbol{\alpha}_i = (\alpha_{i,1}, \dots, \alpha_{i,n})$ and its components are nonnegative. $\alpha_{i,j}$ is the weight of the document d_j . The function h can be also expressed as in the form based on the document pairs with weights.

$$h(\mathbf{x}_i | \mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n; \psi, \boldsymbol{\alpha}_i) = \frac{1}{\sum_{j=1}^n \alpha_{i,j} / \alpha_{i,i} \exp(\psi(f(\mathbf{x}_j)) - \psi(f(\mathbf{x}_i)))} \tag{8}$$

where $\alpha_{i,j} / \alpha_{i,i}$ is the weight of the document pair (d_i, d_j) . The cost-sensitive listwise approaches focus on the performance of the top documents in the ranked list, measured by NDCG. Therefore, one important issue is to relate the weights of the documents with the NDCG. We formulate the problem of setting weights for the documents into the problem of setting weights for the document pairs. In this study, the weights of the document pairs are theoretically given.

4.2 Bound NDCG Errors by Cost-Sensitive Listwise Loss Function

In this section, it is proven that the cost-sensitive listwise losses are the upper bound of NDCG loss. The theorem states definite values of the document pairs' weights. Theoretical proof is based on the lemma 1. For notational simplicity, let $a(i) = 2^{y_i} - 1$, $b(j)$ and its gradient $\nabla b(j)$ are defined as:

$$b(j) = \begin{cases} 1/\log_2(1+j) & j \leq k \\ 1/\log_2(1+k) & j > k \end{cases} \quad \nabla b(j) = \begin{cases} \frac{-\log 2}{(1+j)[\log(1+j)]^2} & j \leq k \\ 0 & j > k \end{cases} \quad (9)$$

It is simple to prove that the NDCG@k loss $L_{ndcg@k}$ has the following upper bound on the basis of equation (5). Due to space limitation, we omit the proof.

$$L_{ndcg@k} \leq \frac{1}{DCG_{\hat{g}@k}} \left(\sum_{i=1}^n a(i) (b(\hat{g}(i)) - b(\pi(i))) \right) + \frac{1}{DCG_{\hat{g}@k}} \sum_{i=1}^n \frac{a(i)}{\log_2(1+k)}$$

Lemma 1. *The NDCG@k loss $L_{ndcg@k}$ is upper bounded by weighted pairwise classification loss.*

$$L_{ndcg@k} \leq \frac{1}{DCG_{\hat{g}@k}} \sum_{y_j \succ y_i} [-a(j)\nabla b(\hat{g}(j)) - a(i)\nabla b(\hat{g}(i))] I[f(\mathbf{x}_j) < f(\mathbf{x}_i)] + C_2$$

where C_2 denotes a constant. The proof is given in the Appendix A. For each query in the training set, the ranking position $\hat{g}(j)$ of the document d_j is easily computed. Therefore, the weights of the document pairs of each query can be calculated at once in training. Based on the lemma, we can justify the correlation between cost-sensitive listwise loss and NDCG@k loss 1.

Theorem 1. *The cost-sensitive listwise loss is the upper bound of NDCG loss $L_{ndcg@k}$.*

$$L_{ndcg@k} \leq -\frac{1}{DCG_{\hat{g}@k}} \sum_{j=1}^n \beta_j \log_2 h(\mathbf{x}_j | \mathbf{x}_{j+1}, \mathbf{x}_{j+2}, \dots, \mathbf{x}_n; \psi, \alpha_j) + C_2 \quad (10)$$

C_2 denotes a constant that is same to the constant in the lemma 1. Based on lemma 1, the above inequality is easily verified with $I[z > 0] \leq \log_2(1 + \exp(z))$ and $\sum_{j=1}^n w_j \log_2(z_j) \leq \left(\sum_{j=1}^n w_j\right) \cdot \left(\log_2\left(\sum_{j=1}^n w_j z_j / \sum_{t=1}^n w_t\right)\right)$ when each z_j and w_j are nonnegative. The theorem demonstrates that many cost-sensitive listwise approaches can be proposed to directly optimize NDCG. For example, we can transform the existing listwise approaches to cost-sensitive listwise methods. Meanwhile, the theorem states definite values for all the weights β_j and α_j .

4.3 Differences from Cost-Sensitive Ranking SVM

The cost-sensitive Ranking SVM 1 and the framework for cost-sensitive listwise approaches both make use of cost-sensitive learning in learning to rank, but there are several differences between them.

¹ In this paper, we use NDCG loss and NDCG@k ranking loss exchangeably.

First, the training instance assigned weights is different. The cost-sensitive Ranking SVM weights on the document pairs. The framework credits the weights for the documents. To solve the problem of setting weights for the documents, the framework reduces it into the task of setting weights for document pairs.

Second, the way to calculate the weights of the document pairs is different. The cost-sensitive Ranking SVM sets the weights by the heuristic method. The framework directly computes the weights based on NDCG@k.

Third, the relationship between the loss functions and NDCG loss is also different. The cost-sensitive Ranking SVM do not solve the problem. The framework proves that the listwise losses are the upper bound of the NDCG loss.

4.4 A Case: Cost-Sensitive ListMLE Ranking Approach

To verify the framework, we propose a novel cost-sensitive approach named CS-ListMLE(cost-sensitive ListMLE). The loss function on a query is defined as:

$$L = \frac{1}{DCG_{\hat{g}@k}} \sum_{j=1}^n \beta_j \log_2 \left(1 + \sum_{t=j+1, y_j > y_t}^n \alpha_{j,t} \exp(f(\mathbf{x}_t) - f(\mathbf{x}_j)) \right) \quad (11)$$

The weights are computed according to Theorem 1. The ranking function of CS-ListMLE is linear, *i.e.*, $f(x) = \langle w, x \rangle$, where $\langle \cdot, \cdot \rangle$ is the inner product and w denotes model parameters. We takes gradient descent method to optimize the loss function. Since the loss function is convex, the model parameters converge to a global optimal solution. The algorithm is provided in Figure 1.

Input: training set, learning rate η , tolerance rate ε , NDCG@k
Initialize: w and compute the weights with respect to each query using Eq. (10)
Repeat: do gradient descent until the change of the loss function is below ε
Return w

Fig. 1. Cost-sensitive ListMLE Ranking Approach

5 Experiments

The experiments are conducted on three datasets OHSUMED, TD2003 and TD2004 in Lector2.0². The experiments validate the two points. The one is that whether CS-ListMLE outperforms the ListMLE. The other is that whether CS-ListMLE can obtains higher performance than the other baselines on the top documents of the ranked list. MAP and NDCG@k(N@k) are used as evaluation measures. The truncation level k in NDCG@k is usually 1, 3, 5 and 10. The performances of ListMLE and RankCosine are directly taken from [9], where both approaches do not provide their performance at NDCG@5 and MAP.

² <https://research.microsoft.com/en-us/um/beijing/projects/lector/Lector2.0/dataset.aspx>

To validate the effectiveness of CS-ListMLE, we train the algorithm with different parameters. CS-ListMLE@ k claims that cost-sensitive ListMLE focuses on the top k documents rank order in the ranked list. It means that CS-ListMLE@ k directly optimizes NDCG@ k . In the experiments, the k takes 1, 3, 5 and 10.

5.1 Ranking Accuracy of ListMLE and Cost-Sensitive ListMLE

We report the performance of cost-sensitive ListMLE and ListMLE on three datasets in Table 1, 2 and 3. The cost-sensitive ListMLE significantly outperforms ListMLE on the datasets TD2003 and TD2004 in terms of all evaluation measures, while their performance is comparable on OHSUMED.

Table 1. Ranking accuracies on OS-HUMED

| Methods | N@1 | N@3 | N@5 | N@10 | MAP |
|---------------|-------|-------|-------|-------|-------|
| ListMLE | 0.548 | 0.473 | — | 0.446 | — |
| CS-ListMLE@1 | 0.555 | 0.482 | 0.464 | 0.446 | 0.442 |
| CS-ListMLE@3 | 0.539 | 0.480 | 0.458 | 0.446 | 0.444 |
| CS-ListMLE@5 | 0.548 | 0.468 | 0.453 | 0.438 | 0.443 |
| CS-ListMLE@10 | 0.536 | 0.471 | 0.452 | 0.439 | 0.443 |

Table 2. Ranking accuracies on TD2003

| Methods | N@1 | N@3 | N@5 | N@10 | MAP |
|---------------|------|-------|-------|-------|-------|
| ListMLE | 0.24 | 0.253 | — | 0.261 | — |
| CS-ListMLE@1 | 0.48 | 0.400 | 0.362 | 0.359 | 0.262 |
| CS-ListMLE@3 | 0.48 | 0.400 | 0.364 | 0.358 | 0.253 |
| CS-ListMLE@5 | 0.48 | 0.391 | 0.358 | 0.355 | 0.264 |
| CS-ListMLE@10 | 0.48 | 0.398 | 0.362 | 0.350 | 0.262 |

Table 3. Ranking accuracies on TD2004

| Methods | N@1 | N@3 | N@5 | N@10 | MAP |
|---------------|-------|-------|-------|-------|-------|
| ListMLE | 0.4 | 0.351 | — | 0.356 | — |
| CS-ListMLE@1 | 0.467 | 0.427 | 0.422 | 0.444 | 0.364 |
| CS-ListMLE@3 | 0.467 | 0.429 | 0.419 | 0.449 | 0.366 |
| CS-ListMLE@5 | 0.467 | 0.420 | 0.407 | 0.444 | 0.366 |
| CS-ListMLE@10 | 0.453 | 0.409 | 0.406 | 0.445 | 0.364 |

Table 4. Test Results on OSHUMED

| Methods | N@1 | N@3 | N@5 | N@10 | MAP |
|--------------|--------------|--------------|--------------|--------------|--------------|
| ListNet | 0.523 | 0.478 | 0.466 | 0.449 | 0.450 |
| RankCosine | 0.523 | 0.475 | — | 0.437 | — |
| AdaRank | 0.514 | 0.462 | 0.442 | 0.437 | 0.442 |
| RSVM | 0.495 | 0.465 | 0.458 | 0.441 | 0.447 |
| CS-ListMLE@1 | 0.555 | 0.482 | 0.464 | 0.446 | 0.442 |

Table 5. Test Results on TD2003

| Methods | N@1 | N@3 | N@5 | N@10 | MAP |
|--------------|-------------|--------------|--------------|--------------|--------------|
| ListNet | 0.46 | 0.408 | 0.382 | 0.374 | 0.273 |
| RankCosine | 0.36 | 0.346 | — | 0.322 | — |
| AdaRank | 0.42 | 0.291 | 0.242 | 0.194 | 0.137 |
| RSVM | 0.42 | 0.379 | 0.347 | 0.341 | 0.256 |
| CS-ListMLE@1 | 0.48 | 0.400 | 0.362 | 0.359 | 0.262 |

Table 6. Test Results on TD2004

| Methods | N@1 | N@3 | N@5 | N@10 | MAP |
|--------------|--------------|--------------|--------------|--------------|--------------|
| ListNet | 0.440 | 0.437 | 0.420 | 0.458 | 0.372 |
| RankCosine | 0.439 | 0.397 | — | 0.405 | — |
| AdaRank | 0.413 | 0.402 | 0.393 | 0.406 | 0.331 |
| RSVM | 0.44 | 0.410 | 0.393 | 0.420 | 0.350 |
| CS-ListMLE@1 | 0.467 | 0.427 | 0.422 | 0.444 | 0.364 |

We explain why CS-ListMLE remarkably outperforms ListMLE on datasets TD2003 and TD2004. On one hand, each query in datasets TD2003 and TD2004 has 1000 documents. The ratio of the queries containing at most 15 relevant documents in all queries is 95% and 89.3% respectively. Since ListMLE considers all the document pairs, 95% and 89.3% queries loss functions of ListMLE induces the losses caused by 484620 irrelevant document pairs. As is well known, the NDCG loss is not sensitive to the losses generated by the irrelevant document

pairs. Thus, ListMLE introduces a large deviation from the NDCG loss. However, CS-ListMLE only cares about the document pairs composed of relevant documents and irrelevant documents. On the other hand, ListMLE treats the all documents equally. But CS-ListMLE assigns the weights for the document based on NDCG@k. In summary, the loss of CS-ListMLE is more close to NDCG@k loss than ListMLE on datasets TD2003 and TD2004.

As far as dataset OHSUMED, 81.1% queries have less than 200 documents, while 85.89% queries contain at least 10% relevant documents. Under such situation, the weights of the documents does not affect much in CS-ListMLE. The loss of CS-ListMLE is approximate to the loss of ListMLE.

5.2 Comparison with the Other Baselines

We take CS-ListMLE@1 as an example to compare the performance with the other baselines, including RankCosine [5], ListNet [4], RSVM [2] and AdaRank [7]. Experimental results are presented in Table 4, 5 and 6. CS-ListMLE almost outperforms RankCosine, AdaRank and Ranking SVM on three datasets at all evaluation measures. Compared to ListNet, CS-ListMLE obtains higher performance at NDCG@1. We conduct t-test on the improvement of CS-ListMLE over ListNet, Ranking SVM and AdaRank on the three datasets in terms of NDCG@1. The results indicate that the improvement of NDCG@1 over Ranking SVM and AdaRank on dataset OHSUMED is statistically significant ($p\text{-value} < 0.05$). There is no statistically significant difference on dataset TD2003 in spite of rising 6% over AdaRank and Ranking SVM.

Experiments results demonstrate that the CS-ListMLE can achieve high performance on NDCG@1, which meets the goal that the CS-ListMLE focuses on the top one documents ranking order of the ranked list. Meanwhile, the cost-sensitive ListMLE obtains comparable performance to the baselines at MAP.

6 Conclusion

In this paper, we point out that the existing listwise losses are inadequate IR where the documents with higher ranks should be emphasized. To address the issue, we propose a framework for cost-sensitive listwise approaches. The framework credits weights for the documents. The framework reduces the problem of setting weights for the documents to the problem of setting weights for the document pairs. The weights of the document pairs are computed based on the NDCG. It is proven that the cost-sensitive listwise loss is the upper bound of NDCG loss. As an example, we develop a cost-sensitive ListMLE approach. Experimental results show that the cost-sensitive ListMLE outperforms ListMLE on two benchmark datasets in terms of all evaluation measures. In addition, the cost-sensitive ListMLE almost outperforms the baselines, such as RankCosine, AdaRank and Ranking SVM, on the three datasets at all evaluation measures.

References

1. Xu, J., Cao, Y., Li, H., Huang, Y.: Cost-sensitive learning of svm for ranking. In: ECML, pp. 833–840 (2006)
2. Joachims, T.: Optimizing search engines using clickthrough data. In: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 133–142. ACM, New York (2002)
3. Xia, F., Liu, T.Y., Wang, J., Zhang, W., Li, H.: Listwise approach to learning to rank: theory and algorithm. In: Proceedings of the 25th international conference on Machine learning, pp. 1192–1199. ACM, New York (2008)
4. Cao, Z., Qin, T., Liu, T.Y., Tsai, M.F., Li, H.: Learning to rank: from pairwise approach to listwise approach. In: Proceedings of the 24th international conference on Machine learning, pp. 129–136. ACM, New York (2007)
5. Qin, T., Zhang, X.D., Tsai, M.F., Wang, D.S., Liu, T.Y., Li, H.: Query-level loss functions for information retrieval. *Inf. Process. Manage.* 44(2), 838–855 (2008)
6. Lan, Y., Liu, T.Y., Ma, Z., Li, H.: Generalization analysis of listwise learning-to-rank algorithms. In: Proceedings of the 26th Annual International Conference on Machine Learning, pp. 577–584. ACM, New York (2009)
7. Xu, J., Li, H.: Adarank: a boosting algorithm for information retrieval. In: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval, pp. 391–398. ACM, New York (2007)
8. Järvelin, K., Kekäläinen, J.: Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.* 20(4), 422–446 (2002)
9. Xia, F., Liu, T.Y., Li, H.: Statistical consistency of top-k ranking. In: Advances in Neural Information Processing Systems, pp. 2098–2106 (2009)

A Theoretical Justification of Lemma 1

$$\begin{aligned}
 L_{ndcg@k} &\leq \frac{1}{DCG_{\hat{g}}@k} \left(\sum_{i=1}^n a(i) (b(\hat{g}(i)) - b(\pi(i))) \right) + \frac{1}{DCG_{\hat{g}}@k} \sum_{i=1}^n \frac{a(i)}{\log_2(1+k)} \\
 C_1 &= \frac{1}{DCG_{\hat{g}}@k} \sum_{i=1}^n \frac{a(i)}{\log_2(1+k)} \quad C_2 = C_1 + \frac{1}{DCG_{\hat{g}}@k} \sum_{y_j=y_i} [-a(j)\nabla b(\hat{g}(j)) - a(i)\nabla b(\hat{g}(i))] \\
 &\because b \text{ is convex function} \iff b(x) - b(y) \leq \nabla b(x)(x - y) \\
 &\leq \frac{1}{DCG_{\hat{g}}@k} \sum_{j=1}^n a(j) \cdot \nabla b(\hat{g}(j)) \cdot (\hat{g}(j) - \pi(j)) + C_1 \\
 &= \frac{1}{DCG_{\hat{g}}@k} \sum_{j=1}^n a(j) \cdot (-\nabla b(\hat{g}(j))) \cdot \left\{ \left(1 + \sum_{i=1}^n I[f(\mathbf{x}_i) > f(\mathbf{x}_j)] \right) - \left(1 + \sum_{i=1}^n I[y_i > y_j] \right) \right\} + C_1 \\
 &\because I[x > 0] - I[y > 0] \leq I[xy < 0] + I[y = 0] \\
 &\leq \frac{1}{DCG_{\hat{g}}@k} \sum_{y_j > y_i} [-a(j)\nabla b(\hat{g}(j)) - a(i)\nabla b(\hat{g}(i))] I[f(\mathbf{x}_j) < f(\mathbf{x}_i)] + C_2
 \end{aligned}$$

Mining Wikipedia and Yahoo! Answers for Question Expansion in Opinion QA

Yajie Miao and Chunping Li

Tsinghua National Laboratory for Information Science and Technology (TNList)
School of Software, Tsinghua University,
Beijing 100084, China
yajiemiao@gmail.com, cli@tsinghua.edu.cn

Abstract. Opinion Question Answering (Opinion QA) is still a relatively new area in QA research. The achieved methods focus on combining sentiment analysis with the traditional Question Answering methods. Few attempts have been made to expand opinion questions with external background information. In this paper, we introduce the *broad-mining* and *deep-mining* strategies. Based on these two strategies, we propose four methods to exploit Wikipedia and Yahoo! Answers for enriching representation of questions in Opinion QA. The experimental results show that the proposed expansion methods perform effectively for improving existing Opinion QA models.

Keywords: Opinion QA, Question Expansion, Wikipedia, Yahoo! Answers.

1 Introduction

Question Answering (QA), which aims to retrieve answers to human-generated questions automatically, is an important research area in text mining and information retrieval. Many of the methods in this area have been proposed mostly for the task of answering fact-based questions, e.g., “When was the Kyoto Protocol adopted?”. However, in many cases, users are more interested in the opinions towards specific events or objects. Questions querying about opinions or attitudes are defined as *opinion questions*, e.g., “How do the Chinese regard the human rights record of the United States?”.

The existing methods for Opinion QA focus on utilizing sentimental information to obtain desirable results. However, a key problem for Opinion QA is that the information needs expressed by an opinion question is much more complicated than a fact-based question. The lexical elements (i.e., words) in the opinion questions are usually unable to express such needs completely. One way to address this problem is to enrich the representation of an opinion question with information from some external knowledge repositories.

In this paper, we exploit Wikipedia and Yahoo! Answers to expand the questions in Opinion QA. We adopt two mining strategies, i.e., broad-mining and

deep-mining in both Wikipedia and Yahoo! Answers, and propose four expanding methods: *Wiki-broad*, *Wiki-deep*, *Yahoo-broad* and *Yahoo-deep*. Experiments show that all of these four methods boost the performance of the state-of-the-art Opinion PageRank [4] model. Also, we observe that Wiki-deep is the most effective method for question expansion in Opinion QA.

The rest of the paper is organized as follows. Section 2 reviews previous work. Section 3 formulates the proposed expansion methods. In Section 4, we present and discuss the experimental results. We have the concluding remarks and future work in Section 5.

2 Previous Work

Opinion QA is still a new area in QA research. Stoyanov et al. [1] trained a *subjectiveness classifier* which filters out the objective sentences from the answer candidates and therefore improves the quality of the answers to opinion questions. Kim et al. [2] proposed that the opinion holders in the opinion question and its answers should be the same. Based on this, they improved the performance of Opinion QA by identifying opinion holders in the sentences. In the TAC 2008 Opinion QA track [3], most participants found answers to opinion questions through combining linearly the topic and opinion weights of answer candidates. Li et al. [4] proposed the Opinion PageRank and Opinion HITS models for answering opinion questions. In both models, the topical relevance and sentimental information are combined in a unified graph-based framework.

There has been a growing amount of research on employing Wikipedia to enhancing traditional text mining tasks. Gabrilovich et al. [5] proposed a method to improve text classification through enriching document representation with Wikipedia concepts. Banerjee et al. [6] proposed to improve clustering of short texts by using Wikipedia concepts as additional features. Hu et al. [10] proposed two mapping strategies for enriching document representation with Wikipedia concept and category information. The enriched documents are used for the task of text clustering.

Wikipedia and Yahoo! Answers have also been applied for Question Answering. Ye et al. [7] proposed to summarize a series of definitions from Wikipedia, which serve as answers to definition questions. Wang et al. [8] proposed an *analogical reasoning* approach for measuring the linkage between questions and their answers. Through exploiting the previous Yahoo! Answers data, their approach can build links between a new question-answer pair. Wang et al. [9] proposed a *syntactic tree matching* method for retrieving similar questions from Yahoo! Answers when given a new question. However, these works have made no attempts to use Wikipedia or Yahoo! Answers for question expansion.

3 Question Expansion

In this section, we first formulate a method for generating topic words for opinion questions. Then, we present our methods for question expansion.

3.1 Topic Word Generation

Wikipedia and Yahoo! Answers can receive queries from users and return a list of relevant (Wiki) articles or (Yahoo) questions. However, their retrieval modules are design for the query which consists of several keywords. If the query is sentences in natural language, Wikipedia and Yahoo! Answers are quite likely to return no relevant results. Therefore, the opinion questions should not be submitted to Wikipedia and Yahoo! Answers directly. To address this problem, we first generate topic words for the questions and use these topic words as the query.

Usually several questions can be about the same topic, though they are pertaining to various aspects. For a topic T , there are n questions which make up the *question set* $Q_T = \{q_1, q_2, \dots, q_n\}$. All the words in the n questions are ranked according to their frequencies of appearing in Q_T . The top K non-stop words are chosen as the topic words for T . For instance, in the MPQA dataset [11], there are totally 4 questions under the topic “kyoto”. With the above procedures, we can get topic words such as “Kyoto”, “US”, “Protocol”, etc.

3.2 Expansion with Wikipedia

Wikipedia is a huge document corpus which contains more than three millions articles. In addition, Wikipedia undergoes constant development, so its breadth and depth steadily increase over time. The topic words generated in Section 3.1 are combined into the Wikipedia query. After searching in its database, Wikipedia returns relevant Wiki articles, which are ranked according to their relevance to the query.

Wiki-broad. We adopt a broad-mining strategy for exploring the ranking list. The M most relevant articles are selected out as the Wikipedia article set WA . In Wikipedia, each article has a title which summarizes the most essential ideas. From WA , we only extract the titles to form the title set WT . Also, the *redirect titles* in WA , which show explicitly the articles which redirect to the ones in WA , are also included in WT . All the non-stop words in WT are extracted to form the title word set TW . If a word appears more than once in TW , it is considered to be a single word (rather than multiple words). Then the set TW is viewed as the *expansion set* for the questions.

Wiki-deep. In addition to text contents, Wikipedia also constructs links between articles. These links provide valuable information about the relation between concepts¹. If there is a link from the article p_1 to the article p_2 , we can conclude that p_2 presents relevant information about the concept of p_1 . Therefore, p_2 can be further exploited to extend the contents of p_1 . This is the basic idea for our Wiki-deep method. In Wiki-broad, the top M articles are selected from the retrieval results. However, in Wiki-deep, we only focus on the first article which is also the most relevant one. The first paragraph of a Wiki article

¹ In Wikipedia research, the title of a Wikipedia article is usually referred to as a concept.

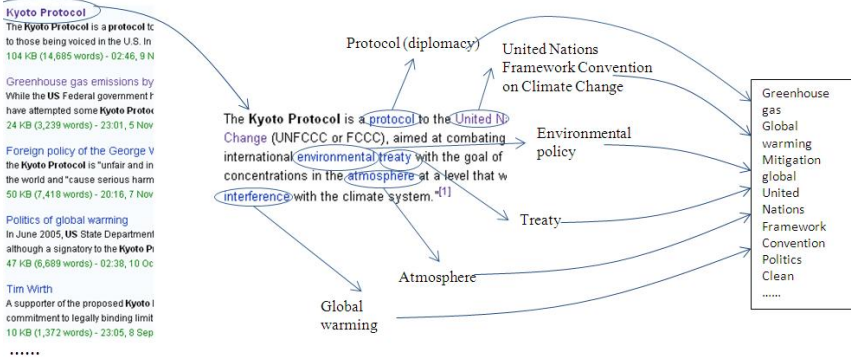


Fig. 1. An example of Wiki-deep

usually serves to summarize the definition of the concept or the main points of the article. All the Wiki articles, which the links in the first paragraph of the most-relevant article point to, are selected to form the link article set LA . Then, all the non-stop words in the titles of the articles in LA and in the title of the most-relevant article are extracted to form a word set. Duplicate words are considered to be a single word. This set is used as the expansion set for the questions. Figure 1 gives the process of Wiki-deep for the questions whose topic words are “Kyoto”, “US” and “Protocol” (see Section 3.1).

3.3 Expansion with Yahoo! Answers

Besides Wikipedia, we also use Yahoo! Answers as external knowledge for question expansion. The topic words generated in Section 3.1 are combined into the query which is submitted to Yahoo! Answers. With the APIs provided by Yahoo! Developer Network², we get a list of Yahoo questions which are also ranked according to their relevance to the query. For each question, Yahoo! Answers returns various forms of information, e.g., *subject* (a brief statement of the question), *content* (a more detailed statement of the question), *chosen-answer* (the best answer chosen by users), etc.

Yahoo-broad. The broad-mining strategy is adopted for expanding questions with the retrieved Yahoo questions. The N most relevant questions in the ranking list are selected and their subjects form the set YS . We only use the subjects in order to cover more Yahoo questions. Then, all the non-stop words in YS are used as the expansion set for the opinion questions. Similarly, duplicate words are considered to be a single one.

Yahoo-deep. Also, we propose the Yahoo-deep method to mine the retrieved Yahoo questions. In this method, we only focus on the most relevant question retrieved from Yahoo! Answers, e.g., “Why dont the US government ratify Kyoto protocol?” in Fig. 2. The subject, content and chosen-answer of this

² <http://developer.yahoo.com/answers/>

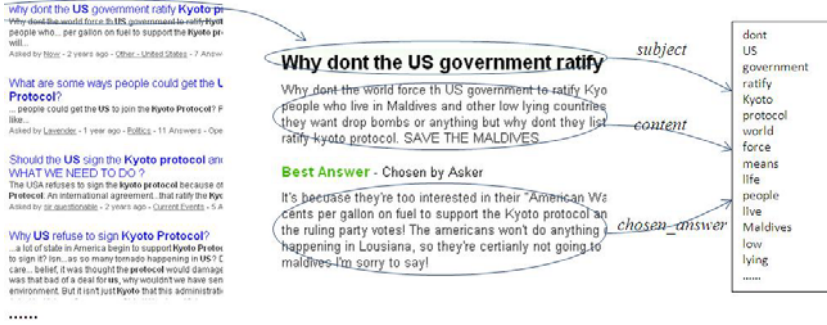


Fig. 2. An example of Yahoo-deep

most-relevant question are concatenated together as the expansion of this question. All the non-stop words in the concatenation are extracted to form the expansion set for the opinion questions. In this method, by exploiting more details about the most-relevant Yahoo question, we mine the Yahoo! Answers archive at the deeper level. Figure 2 shows an example for the process of Yahoo-deep.

With each of the above methods, we get the expansion set for the opinion questions. Note that the questions under one topic have the same expansion set.

4 Experiments

4.1 Experimental Setup

In [4], the experimental results show that the Opinion PageRank model outperforms all the systems in TAC 2008. Therefore, Opinion PageRank is currently one of the most effective methods for Opinion QA. In our experiments, we use Opinion PageRank as the Opinion QA method.

The MPQA dataset [1] is used as the benchmark in this study. MPQA contains 15 opinion questions and has been widely used in Opinion QA research. We adopt the evaluation metrics used in the TAC Opinion QA track [3]. For each opinion question in MPQA, annotators have given a list of answer segments. Each segment is assigned a confidence value which shows its relative importance. The *Recall* of the answers is calculated as $Recall = r/R$, where r is the sum of the confidence of segments in the answers, and R is the sum of the confidence of segments for the question. The *Precision* of the answers is calculated as

$$Precision = 1 - ((l - A) / l). \quad (1)$$

where l is the number of non-whitespace characters in the answers, A is the allowance of the answers, and $A = 100 * a$ (a is the number of segments in the answers). The final *F-score* is calculated with the TAC official value $\beta=3$ [3], which means Recall is three times as important as Precision.

$$F - score = (\beta^2 + 1) * Recall * Precision / (\beta^2 * Precision + Recall) \quad \beta = 3. \quad (2)$$

The overall Recall, Precision and F-score are the average of their corresponding values over the 15 questions.

4.2 Performance Evaluation

We expand each opinion question in MPQA with Wiki-broad, Wiki-deep, Yahoo-broad and Yahoo-deep respectively. For each method, the expanded questions are “answered” by Opinion PageRank and the retrieved answers are evaluated. We take the *no-expansion* method, in which the original questions are inputted into Opinion PageRank without any expansion, as our baseline.

The parameters are set in the following way. The parameter K , which denotes the number of selected topic words, is set to 3. The parameters M (Wiki-broad) and N (Yahoo-broad) are both set to 10. Figure 3 gives the results for the various methods. In the figure, we can see that all of the four expansion methods outperform the no-expansion method which adopts no question expansion operations. When using the Wiki-deep method, Opinion PageRank performs best (F-score: 0.1872) and achieves around 12.5% improvements over no-expansion. This demonstrates that our methods indeed take effects in improving the performance of Opinion Question Answering. Also, we notice from the figure that Wiki-deep is the most effective expansion method when considering F-score. From Section 3.2, we know that when using Wiki-deep, we put more emphasis on links between articles than textual contents. These links are able to represent the relation between concepts at the semantic level. Therefore, a question expanded by Wiki-deep can embody the information needs of the original question more accurately and comprehensively.

Another observation from Fig. 3 is that expansion with Wikipedia (Wiki-broad and Wiki-deep) obtains higher F-score than expansion with Yahoo! Answers (Yahoo-broad and Yahoo-deep). This is partly because the contents in Yahoo! Answers are generated by users in a free way. Therefore, the expansion sets generated

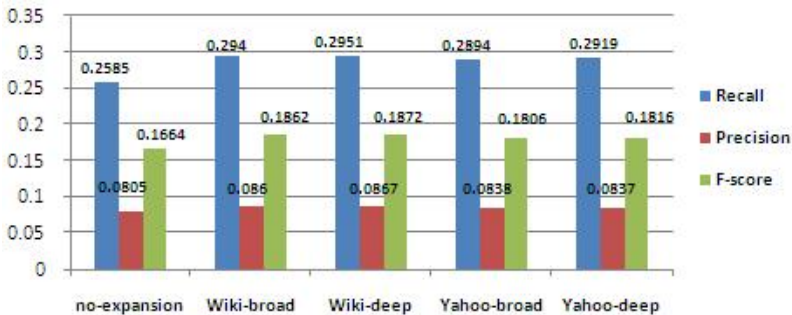


Fig. 3. Performance comparison among the methods

Table 1. P-values in t-tests

| Expansion Methods | P-value |
|-------------------|---------|
| Wiki-broad | 0.048 |
| Wiki-deep | 0.039 |
| Yahoo-broad | 0.118 |
| Yahoo-deep | 0.086 |

Table 2. F-score for the combination models

| Combination Methods | F-score |
|--------------------------|---------|
| Wiki-deep + Yahoo-broad | 0.1872 |
| Wiki-deep + Yahoo-deep | 0.1872 |
| Wiki-broad + Yahoo-broad | 0.1862 |
| Wiki-broad + Yahoo-deep | 0.1862 |

with Yahoo-broad and Yahoo-deep contain noisy words (e.g., “guys”, “because”, etc.), which contribute little to enriching the representation of the questions. On the contrary, Wikipedia articles are created and edited under strict guidelines, and thus the expansion sets are relatively “purer”.

To determine whether these improvements are statistically significant, we perform several single-tailed t-tests. Table 1 shows the P-values of various methods compared with the no-expansion baseline on the F-score metric. Wiki-deep achieves the most significant improvements (the lowest P-value) over no-expansion. Both Wiki-deep and Wiki-broad perform significantly better than the baseline at a 95% confidence level, while the improvements of Yahoo-broad and Yahoo-deep are not significant.

In the above evaluations, we consider the four methods separately. Next, we will investigate whether combining these methods can achieve better results. When combining two methods, we simply merge the expansion sets of the two methods together and get the new set. Table 2 shows the F-score values for the combination methods. From the table, we can see that these four combination methods fail to outperform the best non-combination method, i.e., Wiki-deep. Each combination method achieves the same performance as its corresponding Wikipedia method, e.g., Wiki-deep+Yahoo-broad has the same F-score as Wiki-deep. Moreover, each combination method performs better than its corresponding Yahoo! Answers method. This further proves that expansion with Wikipedia is more effective than that with Yahoo! Answers for opinion questions.

5 Conclusion and Future Work

In this paper, we propose various methods to exploit Wikipedia and Yahoo! Answers for enriching question representation in Opinion QA. The experimental results show that these methods boost the performance of Opinion QA to a

great extent. Also, performance comparison reveals that Wiki-deep is the most effective expansion method.

In our future work, we will consider applying our methods to other types of questions. Also, we will investigate other forms of information, such as Outlines and Infobox in Wikipedia, to enrich sentences in Question Answering.

Acknowledgments. This work was supported by National Natural Science Funding of China under Grant No. 90718022.

References

1. Stoyanov, V., Cardie, C., Wiebe, J.: Multi-perspective Question Answering using the OpQA Corpus. In: Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing, pp. 923–930 (2005)
2. Kim, S., Hovy, E.: Identifying Opinion Holders for Question Answering in Opinion Texts. In: Proceedings of AAAI Workshop on Question Answering in Restricted Domains (2005)
3. Dang, H.T.: Overview of the TAC 2008: Opinion Question Answering and Summarization Tasks. In: Proceeding of Text Analysis Conference (2008)
4. Li, F., Tang, Y., Huang, M., Zhu, X.: Answering Opinion Questions with Random Walks on Graphs. In: Proceedings of the 47th Annual Meeting of the Association of Computational Linguistics, pp. 733–745 (2009)
5. Gabrilovich, E., Markovitch, S.: Overcoming the Brittleness Bottleneck using Wikipedia: Enhancing Text Categorization with Encyclopedic Knowledge. In: Proceedings of the 21st National Conference on Artificial Intelligence, pp. 1301–1306 (2006)
6. Banerjee, S., Ramanathan, K., Gupta, A.: Clustering Short Texts using Wikipedia. In: Proceedings of the 30th ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 787–788 (2007)
7. Ye, S., Chua, T., Lu, J.: Summarizing Definition from Wikipedia. In: Proceedings of the 47th Annual Meeting of the Association of Computational Linguistics, pp. 199–207 (2009)
8. Wang, X., Tu, X., Feng, D., Zhang, L.: Ranking Community Answers by Modeling Question-Answer Relationships via Analogical Reasoning. In: Proceedings of the 32th ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 179–186 (2009)
9. Wang, K., Ming, Z., Chua, T.: A Syntactic Tree Matching Approach to Finding Similar Questions in Community-based QA Services. In: Proceedings of the 32th ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 187–194 (2009)
10. Hu, X., Zhang, X., Lu, C., Park, E.K., Zhou, X.: Exploiting Wikipedia as External Knowledge for Document Clustering. In: Proceedings of the 15th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 389–396 (2009)

Answer Diversification for Complex Question Answering on the Web

Palakorn Achananuparp¹, Xiaohua Hu¹, Tingting He²,
Christopher C. Yang¹, Yuan An¹, and Lifan Guo¹

¹College of Information Science and Technology, Drexel University, Philadelphia PA

²Department of Computer Science, Central China Normal University, Wuhan, China
pkorn@drexel.edu, thu@cis.drexel.edu, the@mail.ccnu.edu.cn,
{chris.yang,yuan.an,lifan.guo}@ischool.drexel.edu

Abstract. We present a novel graph ranking model to extract a diverse set of answers for complex questions via random walks over a negative-edge graph. We assign a negative sign to edge weights in an answer graph to model the redundancy relation among the answer nodes. Negative edges can be thought of as the propagation of negative endorsements or disapprovals which is used to penalize factual redundancy. As the ranking proceeds, the initial score of the answer node, given by its relevancy to the specific question, will be adjusted according to a long-term negative endorsement from other answer nodes. We empirically evaluate the effectiveness of our method by conducting a comprehensive experiment on two distinct complex question answering data sets.

Keywords: Answer diversification, answer reranking, random walk, negative-edge graph, complex question answering.

1 Introduction

Automatically generating a set of answers for *complex questions*, e.g. definition, opinion, and online community questions, remains a challenging task for several reasons. First, the information needs underlying this type of questions are often subjective and ill-defined [15]. Hence, it requires one or more answer passages to generate a complete response to complex questions. Furthermore, the answers themselves do not easily fall into predictable semantic classes [10]. So, name-entity style answer extraction techniques are not likely to be effective. Moreover, it is more desirable for the automatic response to return a set of factually diverse answers.

Suppose that there are two sets of answers, A and B , generated by two different systems that response to a given question “*what effect does steroid use have on athletes’ performance?*” Set A consists of two answers {*steroid helps boost athletic performance by improving muscle mass, steroids can cause many harmful effects*} while set B contains {*steroids enhance athletic performance, athletes use steroids to improve their performance*}. An information seeker would find answers in set A to be more useful than those in set B . As illustrated, the two facts in the first set are relatively more novel than those in the second set. In other words, the factual coverage of the second set is less diverse than that of the first one.

1.1 Contributions

In this research, we utilize a graph topology to rerank the answers according to their relevance and novelty. The summary of our contributions is as follows:

1. We propose a graph ranking model called *DiverseRank* to extract a diverse set of answers for complex questions. Our method is motivated by the ideas that a good answer set should contain facts which are highly relevant as well as novel. The main contribution of our work is in the use of a graph topology to reduce redundancy among answers. We represent a set of answers as a set of vertices whose edges correspond to the similarity between answer nodes. A negative sign is assigned to the edge weights to model the redundancy relationship between nodes. Then, the final ranking score for each answer node is derived from its long-term negative endorsement.
2. We conduct a comprehensive experiment on two distinct question answering data sets: a subset of Yahoo! Answers data and TREC 2006's complex questions data, to evaluate the performance of the proposed method. To measure the quality of the extracted answers, we use a *nugget pyramid* [14] evaluation and a recall-by-length curve as the performance metrics. Specifically, we measure diversity in terms the amount of common *information nuggets*, a small text fragment that describes a certain fact about a given question, between the extracted sets and the gold standard set.

1.2 Paper Organization

The rest of the paper is organized as follows. First, we review related work in section 2. Next, we describe the proposed method in section 3. In section 4, we present the experimental evaluation, including data sets, evaluation metrics, and procedures. Finally, we discuss about the results and conclude the paper in section 5 and 6, respectively.

2 Related Work

Several issues in web community question answering have been investigated, e.g. finding high-quality answers [9][19], maximizing the facet coverage [16], etc. However, answer diversity issue has not been explored as much. Diversity in ranking has long been one of the major issues in many research areas, such as text summarization and information retrieval. Many information retrieval researchers have attempted to establish several theoretical frameworks of diversity ranking and evaluation [2][7]. There are a growing amount of works in text summarization area [6][9][12][20] which try to integrate diversity as part of the ranking function's properties. For example, Zhu et al. [20] proposes a unified ranking algorithm called GRASSHOPPER which is based on random walks over an absorbing Markov chain. Their method works by iteratively transforming the top-ranking nodes into absorbing nodes, effectively reducing the transition probabilities to zero. The absorbing nodes will drag down the scores of the adjacent nodes as the walk gets absorbed. On the other hand, the nodes which are far away from the absorbing nodes still get visited by the random walk, thus ensuring that the novel nodes will be ranked higher. Li et al [12]

approaches the diversity ranking from the optimization under constraints perspective. They propose a supervised method based on structural learning which incorporates diversity as a set of subtopic constraints. Then, they train a summarization model and enforce diversity through the optimization problem.

Our method differs from other diversity ranking methods in the rank propagation. Since most graph ranking models [6] [18][20] are inspired by the PageRank algorithm [4], they rely on eigenvector centrality to measure the importance of nodes. In contrast, our method uses the negative edges to propagate negative endorsements among nodes. High redundant nodes are those which receive a substantial amount of disapproval votes. Furthermore, the applications of negative edges in ranking model have been explored in related domains, such as trust ranking [8], social network mining [11][12], and complex question answering [1]. On the other hand, our method considers the negative edges to represent redundancy relationship among answers.

3 The Proposed Method

Our method focuses on two key aspects to find a diverse set of answer. First, a set of answer should contain many relevant facts pertaining to an information need. Second, Each answer should be novel or contain a distinct fact with respect to the other answers. To achieve that, we employ a graphical model to rank the relevant answers according to the balance between relevance and novelty. Two set of relations are represented by two signed graphs. First, the relevance relation is denoted by the positive edges between the question node and the answer nodes. On the other hand, the redundancy relation is represented by the negative edges between answer nodes. The negative sign is used to propagate a negative endorsement or disapproval vote between the nodes. The absolute value of edge weight represents the degree of similarity between answers. A formal description of our method is described as follows.

Given a question q and a set of n relevant answers, we first define $G = (V, E)$ as an undirected graph where V is a set of vertices representing the relevant answers, E is a set of edges representing the similarity between vertices where $E \subset V \times V$. Next, we can represent G as an $n \times n$ weighted matrix S where S_{ij} is a similarity score $sim(i, j)$ of node i and j and $sim(i, j)$ has a non-negative value. If i and j are unrelated, then $S_{ij} = 0$. Given S , we can derive an $n \times n$ normalized adjacency matrix A such that each element A_{ij} in A is the normalized value of S_{ij} such that $A_{ij} = S_{ij} / \sum_{k=1}^n S_{ik}$ and $\sum_{j=1}^n A_{ij} = 1$. Next, given the question q , we define a vector r where each element r_i is the relevance score $rel(i, q)$ of node i given q . Then, we transform r into an $n \times n$ matrix B from the outer product of an all-1 vector and r^T such that each element $B_{ij} = r_i / \sum_{k=1}^n r_k$ and $\sum_{j=1}^n B_{ij} = 1$. Given two probability distributions, we define the transition matrix P as:

$$P = dA + (1 - d)B \quad (1)$$

where d is a damping factor with a real value from $[0, 1]$, A is the initial answer adjacency matrix, B is the question-answer relevance matrix. Since all rows in P have non-zero probabilities which add up to 1, P is a stochastic matrix where each element

P_{ij} corresponds to the transition probability from state i to j in the Markov chain. Thus, P satisfies ergodicity properties and has a unique stationary distribution $\vec{\pi}P = \vec{\pi}$. At this stage, each answer can then be ranked according to its stationary distribution. At this point, we have derived a random walk model over the answer graph which incorporates both answer relevance and answer similarity. However, it does not take into account factual redundancy between answers.

In order to employ the negative edges to reduce factual redundancy, we modify the aforementioned graph G such that all edge weights in G have a negative sign. As such, we define $G = (V, E)$ as an undirected graph where V is a set of n relevant answer vertices, E is a set of negative edges where $E \subset V \times V$. Then, an adjacency matrix M , corresponding to edge weights in G , is defined as an all-negative matrix of S where $M_{ij} = -S_{ij} \sum_{j=1}^n S_{ij}$ and $\sum_{j=1}^n M_{ij} = -1$.

Next, similar to the process of deriving the transition matrix P , we define a modified transition matrix Q to incorporate the negative adjacency weights defined in M and the answer relevance defined in B . To ensure that Q is still ergodic, we multiply matrix B with a scaling factor c . The value of c is determined by the conditions that all elements in Q should be non-negative and each i -th row of Q should add up to 1. That is, $\sum_{j=1}^n Q_{ij} = 1$. Since all rows of M sum to -1 and all rows of B add up to 1, c is a function of d where $c = 1 + d/1 - d$.

$$Q = dM + (1 - d)cB \quad (2)$$

where M is an all-negative answer adjacency matrix. Since ergodicity properties still hold, Q has a unique stationary distribution $\vec{\pi}Q = \vec{\pi}$. Finally, we rank each node i according to its stationary probability $\vec{\pi}_i$. From the matrix notations, the simplified equation of the DiverseRank score can be formulated as follow:

$$DR^t(i) = (1-d) \cdot c \cdot \frac{rel(i, q)}{\sum_{i=1}^n rel(i, q)} - d \sum_{j \in adj(i)} \frac{sim(i, j)}{\sum_{k=1}^n sim(j, k)} DR^{t-1}(j) \quad (3)$$

Where d is a damping factor with a real value in $[0,1]$ range. Additionally, d serves as a penalty factor of redundancy. c is a scaling factor defined as a function of d where $c = 1 + d/1 - d$. $rel(i, q)$ is the relevance score of answer i given the question q . And $sim(i, j)$ is a similarity score of answer i and j . To estimate the value of $rel(i, q)$, we employ a sentence weighting function described in Allen et al. [3] as it is shown to consistently outperform other relevance models at the sentence level.

4 Experimental Evaluation

4.1 Data Sets

Two question answering data sets are used in the evaluation: a subset of Yahoo! Answers data set (YahooQA) used in Liu et al.'s work [15] and a complex interactive question answering test set (ciQA) used in TREC 2006 [10]. YahooQA data comprises

subjective and ill-defined information needs formulated by the community members. The subjects of interests span widely from mathematics, general health, to wrestling. In contrast, ciQA data largely focus on the complex entity-relationship questions. Their information needs reflect those posed by intelligence analysts. From data quality perspective, YahooQA data are much noisier than ciQA data as they contain mostly informal linguistic expressions. To prepare YahooQA data set, we randomly select 100 questions and 10,546 answers from the top 20 most frequent categories (measured in terms of a number of responded answers) to use as a test set. A set of information nuggets for YahooQA is automatically created by matching relevant answers with the corresponding questions. The best answer chosen by askers for each question is marked as a vital nugget while the other answers are marked as an okay nugget. In the case of ciQA data set, 30 question topics and their free-form description are prepared by human assessors at National Institute of Standards and Technology (NIST).

4.2 Evaluation Settings

We employ the nugget pyramid metric to evaluate the diversity of the answer set. Generally, we assume that the factual diversity of the extracted answers can be measured in terms of a number of information nuggets the extracted sets have in common with the benchmark nuggets. The formulas to compute the pyramid F-score are described in [14]. In summary, the pyramid F-score is computed as a weighted harmonic mean (F-score) between nugget recall (NR) and nugget precision (NP). NR and NP are derived from summing the unigram co-occurrences between terms in each information nugget and terms from each extracted answer set. Following the standard procedure in TREC 2006, we set the evaluation parameters to $\beta = 3$ and $l = 7,000$ and use Pourpre [14] script version 1.1c to automatically compute the scores. Additionally, we further perform a fine-grained analysis of the algorithmic performance at varying sizes of answer set using a recall-by-length performance curve [13]. Better methods will produce curves that rise faster as more relevant and novel facts are included in the answer set.

Starting from the preprocessing step, we extract word features from a collection of answers by splitting answers into unigram tokens, removing non content-bearing words, e.g., articles, conjunctions, prepositions, etc., and stemming the tokens using Porter Stemmer. After the preprocessing step, we use a vector-space model to retrieve the relevant answers. Free-form narrative field associated with each question is used as a query. The relevance scores between the answers and query are computed from the TFIFS weighting function [3]. The next step is to rerank the list of relevant answers obtained from the retrieval step. To achieve that, we first transform the list of relevant answers into an undirected graph with negative edges. Different edge weighting schemes based on inter-sentence similarity measures are employed. Next, the relevance scores of the retrieved answers and a query are calculated using various relevance models.

We compare the effectiveness of four baseline methods against the proposed method. The baselines include maximal marginal relevance (MMR) [5], SumBasic [17],

topic-sensitive LexRank [18], and a backward ranking of topic-sensitive LexRank (henceforth LexRankInv). The last baseline method is used to test whether diversity can be promoted by simply reversing the ranking of eigenvector centrality method. Once the ranking scores are calculated, we select top- k answers into the answer set. The default cut-off level for answer length is 7,000 characters.

5 Results and Discussion

5.1 Pyramid F-Scores

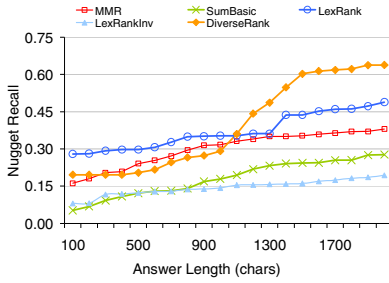
Table 1 shows the average pyramid F-scores of the baseline methods and the best DiverseRank variants. In both data sets, the proposed method significantly outperforms all baseline methods, p -value <0.05 . Considering the performance between random-walk based methods (LexRank vs. DiverseRank), DiverseRank also outperforms LexRank in both data sets although the improvements are relatively minor (6.65% and 2.69%), compared to those of other baselines. Furthermore, LexRankInv produces inferior scores to DiverseRank in both data sets.

Table 1. The average F-Scores of the baseline and DiverseRank methods. The best methods are in bold.

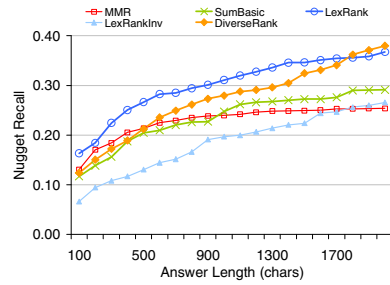
| Method | YahooQA | | ciQA | |
|-------------|---------------|---------------|---------------|---------------|
| | F-Score | % Improvement | F-Score | % Improvement |
| MMR | 0.2946 | +14.52% | 0.2486 | +48.30% |
| SumBasic | 0.2895 | +16.54% | 0.2956 | +24.71% |
| LexRank | 0.3163 | +6.65% | 0.3590 | +2.69% |
| LexRankInv | 0.2391 | +41.12% | 0.3516 | +4.82% |
| DiverseRank | 0.3374 | - | 0.3686 | - |

5.2 Recall-by-Length Performance Curves

Figure 1 shows recall-by-length curves of the baselines and the proposed method in each data set. In YahooQA case (figure 1a), DiverseRank starts to produce a significantly better performance than the best baseline method (LexRank) at the answer length of 1,200 characters. On the other hand, DiverseRank does not perform quite well in ciQA case (figure 2b). In this case, DiverseRank starts to outperform LexRank after the answer length of 1,800 characters. This outcome is not entirely unexpected. As a smaller answer set contains fewer number of information nuggets, therefore there are fewer items to be diversified. As the answer set continues to grow, DiverseRank continues to gain a better performance. Note that our results confirm the previous results in [10] in which a method that produces the best F-score at a predefined answer length does not necessarily perform equally effective across all incremental lengths.



1a. YahooQA



1b. ciQA

Fig. 1. The recall-by-length performance curve on YahooQA (A) and ciQA (B) data sets

6 Conclusion

We propose a graph ranking model to find a diverse set of answers based on random walks over a negative-edge graph. Our main contribution is in the utilization of a graphical model to reduce redundancy within the answer set. First, given a complex question and a set of relevant answers, we represent the relevant answers as an answer graph whose nodes correspond to answers and edges correspond to the similarity between answers. Then, we assign a negative sign to edge weights in the answer graph to model the redundancy relationship between nodes. Then, the final ranking score for each answer node is derived from its long-term negative endorsement. The evaluation results show that our method outperforms most baseline methods. The analysis of recall-by-length performance curves suggests that the best baseline method performs better than DiverseRank at smaller answer lengths. This is explained by the fact that a smaller set of answers contains fewer facts, thus its content can hardly be diversified further. As the size of answer set increases, DiverseRank eventually outperforms the baseline methods.

Acknowledgments. This research work is supported in part from the NSF Career grant IIS 0448023, NSF CCF 0905291, NSF IIP 0934197, NSFC 90920005 and the Program of Introducing Talents of Discipline to Universities B07042 (China).

References

1. Achananuparp, P., Yang, C.C., Chen, X.: Using Negative Voting to Diversify Answers in Non-Factoid Question Answering. In: Proc. of CIKM 2009, Hong Kong (2009)
2. Agrawal, R., Gollapudi, S., Halverson, A., Ieong, S.: Diversifying Search Results. In: Proc. of WSDM 2009, pp. 5–14 (2009)
3. Allan, J., Wade, C., Bolivar, A.: Retrieval and novelty detection at the sentence level. In: Proc. of SIGIR 2003, pp. 314–321. ACM, New York (2003)
4. Brin, S., Page, L.: The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems* 30(1-7) (1998)

5. Carbonell, J., Goldstein, J.: The Use of MMR, Diversity-Based Reranking for Reordering Documents and Producing Summaries. In: Proc. of SIGIR 1998, pp. 335–336 (1998)
6. Chen, S.Y., Huang, M.L., Lu, Z.Y.: Summarizing Documents by Measuring the Importance of a Subset of Vertices within a Graph. In: Proc. of WI 2009 (2009)
7. Clarke, C.L., Kolla, M., Cormack, G.V., Vechtomova, O., Ashkan, A., Büttcher, S., MacKinnon, I.: Novelty and diversity in information retrieval evaluation. In: Proc. of SIGIR 2008, pp. 659–666 (2008)
8. de Kerchove, C., Dooren, P.V.: The PageTrust algorithm: how to rank web pages when negative links are allowed? In: Proc. SDM 2008, pp. 346–352 (2008)
9. Jurczyk, P., Agichtein, E.: Discovering authorities in question answer communities by using link analysis. In: Proc. of CIKM 2007, Lisbon, Portugal, November 6-10 (2007)
10. Kelly, D., Lin, J.: Overview of the TREC 2006 ciQA Task. SIGIR Forum 41(1), 107–116 (2007)
11. Kunegis, J., Lommatzsch, A., Bauckhage, C.: The Slashdot zoo: Mining a social network with negative edges. In: Proc. of WWW 2009, pp. 741–750 (2009)
12. Li, L., Xue, G.R., Zha, H., Yu, Y.: Enhancing Diversity, Coverage and Balance for Summarization through Structure Learning. In: Proc. of WWW 2009, pp. 71–80 (2009)
13. Lin, J.: Is Question Answering Better Than Information Retrieval? Toward a Task-Based Evaluation Framework for Question Series. In: Proc. of NAACL HLT 2007, Rochester, NY, pp. 212–219 (2007)
14. Lin, J., Demner-Fushman, D.: Automatically Evaluating Answers to Definition Questions. In: Proc. of HLT/EMNLP, Vancouver, pp. 931–938 (2005)
15. Liu, Y., Bian, J., Agichtein, E.: Predicting Information Seeker Satisfaction in Community Question Answering. In: Proc. of SIGIR 2008, Singapore, July 20–24 (2008)
16. MacKinnon, I., Vechtomova, O.: Improving Complex Interactive Question Answering with Wikipedia Anchor Text. In: Macdonald, C., Ounis, I., Plachouras, V., Ruthven, I., White, R.W. (eds.) ECIR 2008. LNCS, vol. 4956, pp. 438–445. Springer, Heidelberg (2008)
17. Nenkova, A., Vanderwende, L.: The impact of frequency on summarization. MSR-TR-2005-101 (2005)
18. Otterbacher, Erkan, G., Radev, D.R.: Using Random Walks for Question-focused Sentence Retrieval. In: Proc. of the HLT/EMNLP 2006, Vancouver, pp. 915–922 (2005)
19. Suryanto, M.A., Lim, E.P., Sun, A., Chiang, R.H.: Quality-aware collaborative question answering: methods and evaluation. In: Proc. of WSDM 2009, Barcelona, Spain, pp. 142–151 (2009)
20. Zhu, X., Goldberg, A., Van Gael, J., Andrzejewski, D.: Improving Diversity in Ranking using Absorbing Random Walks. In: Proc. of NAACL-HLT 2007 (2007)

Vocabulary Filtering for Term Weighting in Archived Question Search

Zhao-Yan Ming, Kai Wang, and Tat-Seng Chua

Department of Computer Science,
School of Computing, National University of Singapore
{mingzy, kwang, chuats}@comp.nus.edu.sg

Abstract. This paper proposes the notion of vocabulary filtering in a term weighting framework that consists of three filters at the document level, collection level, and vocabulary level. While term frequency and document frequency along with their variations are respectively the dominant term weighting factors at the document level and collection level, vocabulary level factors are seldom considered in current models. In a way, stopword removal can be seen as a vocabulary level filter, but it is not well integrated into the current term-weighting models. In this paper, we propose a vocabulary filtering and multi-level term weighting model by integrating point-wise divergence based measure into the commonly used TF-IDF model. With our proposed model, the specificity of the vocabulary is captured as a new factor in term weighting, and stopwords are naturally handled within the model rather than being removed according to a separately constructed list. Experiments conducted on searching for similar questions in a large community-based question answering archive show that: (a)our proposed term weighting model with multiple levels is consistently better than those with single level for retrieval task; (b)the proposed vocabulary filter well distinguishes salient and trivial terms, and can be utilized to construct stopword lists.

1 Introduction

As large Community-based Question Answering (cQA) archives are built up through user collaboration, the knowledge is accumulated and made ready for sharing. Research on archived questions has emerged recently [5,14,15]. An application that facilitates knowledge sharing and diversity maintaining is *Archived Question Search* (AQS). It is a function that makes the huge resource reusable by returning relevant answered questions given a new question as a query. If good matches are found, the lag time involved with waiting for a personal response can be avoided, thus improving user satisfaction and avoiding repeating questions.

As a specific application of IR, AQS in cQA repository is distinct from the search of web pages or news articles because it deals with long queries and short documents that are both in the form of questions (for simplicity, we call query questions in AQS as queries, and candidate questions to be searched as documents thereafter):

Long query: each AQS queries consist of natural language sentences that are supposed to be understood and answered by other community members. This kind of queries is

usually longer, noisier, and more verbose than keyword queries. Thus the salient terms and trivial terms are weaved together and the information needs are usually more specific.

Short document: AQS documents are essentially the same as its queries since both are cQA questions. Shorter document length means that most terms might appear only once in a document, resulting in term frequency (tf) approximates a weak binary factor.

This characteristic of cQA data makes the existing term weighting schemes, such as TF-IDF model [13], Okapi BM25 [10], and Divergence From Randomness [2], less capable if directly applied. The major difficulty is that documents and collections statistics are not adequate to provide enough information. The proper functioning of the existing term weighting schemes is under the assumption that documents are long and queries are short. The same difficulty may also be encountered by other forms of community content, like blogs and forums, which have the concise and noise-prone nature.

This motivates us to explore beyond the document and collection. We propose the notion of vocabulary filtering as a complementary dimension of term weighting. By definition, a vocabulary refers to the body of words used on a particular setting or in a particular domain. Although different vocabularies may share a similar set of terms, their respective weights are specific. Given the underlying setting or domain, the weights can be determined, independent of any specific collection and document that instantiates the vocabulary. In a way, stopword removal can be seen as a simple binary vocabulary filter, in that it assigns 0 or 1 score to a term in addition to a weighting scheme.

We propose to measure term saliency in a vocabulary by estimating a heuristic evaluation function that accepts terms' point-wise divergence feature as input. The assumption under the point-wise divergence feature is that terms that have distinct distributions in a specific vocabulary vs. the general vocabulary are important. For instance, we expect "ipod" to have a much higher frequency in a vocabulary about *music & music players* than it does in the general vocabulary, whereas the universal stopword "the" would have similar frequencies in the two vocabularies. The vocabulary filter, in the form of a heuristic term saliency evaluation function, is integrated into the existing term weighting schemes as an enhancement.

2 Related Work

Archived Question Search (AQS) in cQA repository was investigated recently by [5] and [15] using translation-based language model. The translation probability trained on similar collections can be seen as a form of collection-level filtering of term weights. We attribute the success of their proposed model to the integration of the collection-level evidence into the document-level language model.

In related research on term weighting models, the TF-IDF model has been widely used and accepted. Recently, the justification and interpretation of TF-IDF has been studied in [14,12], from the perspectives of information theory, probabilistic language modeling, binary independence retrieval, and Poisson distribution. [11] tried to interpret the Okapi BM25 model from the perspective of poisson model, the language model, the TF-IDF model, and a Divergence From Randomness model. However, none of these efforts attempt to separate the evidences between document-level and the collection-level in term weighting. Since our investigation is focused on the vocabulary-level evidences,

it further raises question on the roles of document level, collection level as well as vocabulary level analysis in term weighting and IR effectiveness.

As for the vocabulary-level filtering for term weighting, the building of a customized stopword list [7,8] and the extraction of domain-specific keywords [3] can be seen as the most relevant work. [6] evaluated the interestingness of a term using the KL divergence and the JS divergence from the distribution of the human interested corpora. This work inspires the method we use for vocabulary filtering.

3 Proposed Vocabulary-Level Filter

Our goal to build a vocabulary-level filter is to quantitatively measure term saliency for a specific vocabulary. We thus emphasize the specificity of vocabularies in constructing the vocabulary-level filters. Term distribution in a specific collection is biased as compared to that in a general collection. This specificity of term distribution in a collection reflects the specificity of its vocabulary, and enables us to highlight the specific important terms for a vocabulary.

3.1 Divergence Feature for Vocabulary Filtering

To capture the vocabulary level term importance, we propose to take a novel point-wise divergence feature for each individual term, rather than divergence of two distributions. We see the term distribution of a vocabulary as the background knowledge to instantiate vocabulary filtering. More broadly, the combining of all vocabularies consists of a general background that can be used to compare against a specific vocabulary.

Jensen-Shannon(JS) divergence is a well adopted distance measure between two probability distributions. It is defined as the mean of the relative entropy of each distribution to the mean distribution, with the following formula:

$$D_{JS}(S||G) = \frac{1}{2} \sum p_s \log \frac{p_s}{\frac{1}{2}(p_s + p_g)} + \frac{1}{2} \sum p_g \log \frac{p_g}{\frac{1}{2}(p_s + p_g)} \quad (1)$$

where S and G denote the specific and general vocabularies, and $p_s(t_i)$ and $p_g(t_i)$ denote their corresponding probability distribution.

As we evaluate the divergence at term level rather than at the whole sample set, we examine the point-wise function as follows:

$$d_{JS}(t) = \frac{p_s(t) \log \frac{2p_s(t)}{p_s(t)+p_g(t)} + p_g(t) \log \frac{2p_g(t)}{p_s(t)+p_g(t)}}{2} \quad (2)$$

The point-wise JS function is an appropriate choice since it is symmetric and ranges over \mathbb{R}^+ . Specifically, $d_{JS}(t)$ assigns a point-wise divergence score to term t highest, when either $p_s(t)$ is much higher than $p_g(t)$, or $p_g(t)$ is much higher than $p_s(t)$, which means the specialized terms in the vocabulary and generally recognized content representative terms are ranked high; lower, when $p_s(t)$ and $p_g(t)$ get closer to each other. These properties suggest that d_{JS} emphasizes divergence at both the most frequent terms in the specific vocabulary and the most frequent terms in the general vocabulary. $p_s(t)$ and $p_g(t)$ are estimated using the Maximum Likelihood Estimator over the specific vocabulary and the general vocabulary respectively.

3.2 Estimating Term Saliency from Divergence Feature

Given a point-wise divergence feature, we aim to estimate the term saliency score, which can be integrated with any existing term weighting scheme. More specifically, we define a mapping function $f_v : d_{JS} \rightarrow W_v$, which produces as output an estimation W_v (denotes the term saliency score on \mathfrak{R}^+) given d_{JS} as input.

We propose a heuristic evaluation function based on logistic function $L(x)$ as below.

$$f_v(x) = 1 + \tau L(x), \quad (3)$$

where $L(x) = \frac{1}{1+e^{-x}}$ is the logistic function that maps \mathfrak{R}^+ to $[0.5, 1)$ monotonically.

Logistic function grows more slowly as $|x|$ increases. This property enables us to control the rate of normalization by shifting the curve along the horizontal axis. Thus a tuning parameter α is introduced. Equation 4 represents the final form of the the heuristic evaluation function of vocabulary level term saliency.

$$f_v(d_{JS}) = 1 + \tau \frac{1}{1 + e^{-(d_{JS} + \alpha)}} \quad (4)$$

where the parameters τ and α are tuned in Section 5.2. Since there is no direct observation of term saliency available, we tune the parameters by using the retrieval performance as an indirect guidance.

4 Three-Level Filtering for Term Weighting

After discussing the method for term weighting accounts for the vocabulary-level informativeness of a term, we next investigate on how it can be naturally integrated into an IR framework.

Term weighting lies in the core of current bag-of-word IR algorithms. Terms weights discriminate the importance of terms for content representation as document descriptors. The general form of the bag-of-words retrieval function with regards to term weighting is given as:

$$Score(Q, D) = \sum_{t_i \in (Q \cap D)} W(t_i) \quad (5)$$

where t_i is the i^{th} query term that appears in both the query Q and the document D , and $W(\cdot)$ is the term weighting model. Generally, $W(\cdot)$ is a function that takes in evidences such as term frequency, document length, document frequency, etc.

Figure 1 illustrates the pipeline of our proposed three-level filtering framework. Given a term t_i as input, the pipeline of three filters outputs $W_t(t_i)$, a quantity that indicates t_i 's importance. Accordingly, we break down the term weighting model into three components. The scoring function in Equation 5 is thus rewritten as

$$Score(Q, D) = \sum_{t_i \in (Q \cap D)} f_v(t_i) \times f_c(t_i) \times f_d(t_i) \quad (6)$$

where $f_v(t_i)$, $f_c(t_i)$, and $f_d(t_i)$ are the *Vocabulary-Level Filter*, *Collection-Level Filter*, and *Document-Level Filter* respectively. The sequence of filters in the pipeline is naturally decided by the scope of the filters and the implementation process.

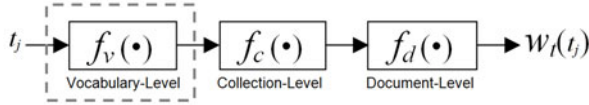


Fig. 1. A general framework for term weighting schemes is represented in a pipeline of three filters, the vocabulary level filter, collection level filter, and document level filter

5 Experiment

5.1 Data Collection and Evaluation Method

We assembled a collection of 325,274 questions posted on Yahoo! Answers(YA) category *Consumer Electronics* from March 2008 to December 2008 using YA APIs¹. The archived questions have an average length of approximately 60 words, consisting of “subject” and “content”. The statistic for replicating the term distribution in the general vocabulary was acquired from a project called Web Term Document Frequency and Rank, a joint effort of the UC Berkeley and Stanford WebBase Projects².

For evaluation, 100 questions were assembled by randomly selecting questions from the whole collection. 93 questions were finally used after manually removing the noisy and redundant ones. The top 20 results of the 93 queries by all the experimented models were labeled by two independent assessors to be relevant or not. The evaluation system, as well as the testing set and the archive, are publicly accessible³.

We use Terrier⁴ for indexing and retrieving, and Porter Stemmer to stem the cQA collection, the queries, and the general web vocabulary. The evaluation metrics are Mean Average Precision(MAP) and Mean Reciprocal Rank (MRR).

5.2 Archived Question Search with Vocabulary Filters

Experimental Setup. In this suit of experiment, we use vocabulary filtering to replace the role of stopword lists in order to test the overall quality of the new retrieval function. The efficiency aspect of stopword removal is not considered here. The comparison systems are (1) $D.(tf)$; (2) $C.D.(idf-tf)$; (3) $V.D.(js-tf)$; and (4) $V.C.D.(js-idf-tf)$, where V . denotes the proposed vocabulary-level filter $f_v(d_{JS})$; C . denotes the collection-level filtering ($f_c(t) = \ln(1 + \frac{N}{df})$), D . denotes the document-level filtering ($f_d(t) = 1 + \ln \frac{tf}{df}$).

Parameter Tuning for Term Saliency Function. A small set of 20 queries are used for tuning the parameters of the term saliency estimation function as in in Equation⁴. For simplicity, we fix τ to be 1.0 as it does not influence the shape of the function. Therefore, only the optimal α is studied in this section.

Figure² shows the influence of α on MAP for $V.C.D$. term weighting scheme. At the point $\alpha = 2.0$, the MAP starts to approach its maximum. This is because the logistic

¹ <http://developer.yahoo.com/answers/>

² <http://www.comp.nus.edu.sg/~rpnlpir/>

³ <http://www.comp.nus.edu.sg/~g0601820/aqs/>

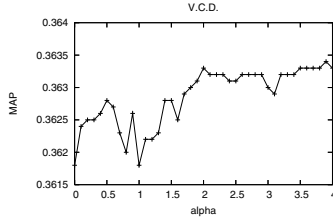


Fig. 2. MAP at different normalization level of α for V.C.D.

function begins to saturate around 2 and grows slowly thereafter as the input increases on \mathbb{R}^+ . This slow growth satisfies the requirement for a deep normalization on $d_{JS}(t)$. It is worth noticing that this optimal α ranges over [2.0, 2.9]. This relatively wide range suggests the stability and tolerance of the proposed heuristic term salience estimation function. For the rest of the experiments, α is set to be 2.0.

Table 1. The MAP and MRR of V.C.D. and improvement over two baselines $D.$ and $C.D.$

| MAP | | $D.$ | $C.D.$ | MRR | | $D.$ | $C.D.$ |
|----------|--------|--------|--------|----------|--------|--------|--------|
| | | 0.1874 | 0.3182 | | | 0.5856 | 0.6987 |
| $V.D.$ | 0.2942 | 56.99% | -7.54% | $V.D.$ | 0.7396 | 26.30% | 5.85% |
| $V.C.D.$ | 0.3622 | 93.28% | 13.83% | $V.C.D.$ | 0.7616 | 30.05% | 9.00% |

Overall Results and Discussion. Table 1 presents the overall results in term of MAP and MRR. MAP is based on the top 20 returned results and MRR evaluates the quality of the top results returned. We draw following observations from the table:

1) Vocabulary Filter in conjunction with collection and document level filters (*i.e.* V.C.D), boosts tf and $idf.tf$ significantly in both MAP and MRR. MAP comparison in Table 1 shows that V.C.D. improves over $D.$ by 93.28%, and $C.D.$ by 13.83%. The consistent improvement suggests that vocabulary level evidence is complimentary to collection level and document factors for term weighting. The scale of improvement over $C.$ is higher than that over $C.D.$, which indicates V.D. is a much stronger baseline than $D.$. In other words, collection level evidence is also critical for measuring term importance. The only negative improvement is V.D. over $C.D.$, which shows that V.D. is not as effective as the classical $tf.idf$ model. This also suggests that $V.$, $C.$, and $D.$ are three orthogonal factors critical for term weighting.

2) Vocabulary Filter improves the top results when the baselines are already very high. By examining MRR comparison in Table 1, we find that two baseline systems both have MRR of over 0.5, which suggests that YA archives have considerable number of similar questions and it is relatively easy to find a similar one with a high ranking. The two systems with $V.$, *i.e.*, $V.D.$ and $V.C.D.$ both have MRR of over 0.7, which shows that both systems have most of their top retrieval results correct. We also notice that $V.D.$ has a higher MRR than $C.D.$, while the latter has higher MAP, which confirms our assertion of the orthogonal of $V.$, $C.$, and $D.$.

Table 2. Highest and lowest ranked 10 terms in *music & music players* category by $f_v(d_{JS})$

| Top 10 | | | | Lowest 10 | | | |
|--------|--------|----|-----------|-----------|-----|-----|-----|
| 1 | ipod | 6 | home | -1 | us | -6 | at |
| 2 | itunes | 7 | servic | -2 | of | -7 | thi |
| 3 | page | 8 | provid | -3 | in | -8 | on |
| 4 | my | 9 | mail | -4 | by | -9 | all |
| 5 | song | 10 | copyright | -5 | new | -10 | be |

5.3 Study on Rank Terms Using Vocabulary Filtering

To examine the effect of the two divergence kernels, we utilize them to rank terms from a subcategory of *Consumer Electronics*, i.e., the *music & music players* question archive. From the top 10 terms in Table 2 we find that the vocabulary filter has successfully captured the salient terms of the recent *music & music players* vocabulary, such as “ipod”, “itunes”, and “sync”. We may guess that if the archive was collected years earlier, the top terms might be “walkman”, “tape” and the like. It suggests that the vocabularies are evolving, or more generally, are specific. We notice that terms like “my” is ranked high, because of the user-collaborative nature of the YA archive. $f_v(d_{JS})$ also ranks some *general* terms high, such as “mail” and “copyright”. This is because “mail” and “copyright” have high probabilities in the general web vocabulary, but low probabilities in the *music & music players* vocabulary. They are considered to be informative though relatively less frequent in the specific vocabulary. This shows that $f_v(d_{JS})$ is capable of capturing term importance in a vocabulary.

The stopwords are expected to be among the lowest in a descending ranked list. Left part of Table 2 lists the lowest 10 terms by $f_v(d_{JS})$. Generally this lowest 10 terms meet our expectation of the commonly recognized stopwords. We thus think of eliminating the lowest ranked terms from indexing, as what stopword removal does, for the purpose of improving the efficiency of the whole retrieval system. As a complementary exploration, we construct stopword lists by taking the lowest 5%, 10%, and 15% $f_v(d_{JS})$ ranked terms. In stead of implementing the full-fledged *V.C.D.* filtering term weighting scheme, we use the 3 stopword lists of different size upon *C.D.* term weighting scheme and find that the retrieval performance at 5% removal actually improves over those without stopword removal and with standard stopword list removal. Moreover, 10% removal is slightly worse than 5% removal since less terms are used for indexing, but still acceptable considering that the efficiency is improved at a small price.

6 Conclusions

In this paper, we proposed a novel notion of vocabulary-filtering to capture term importance by using the whole vocabulary as the background knowledge. JS divergences are utilized to characterize a specific vocabulary by contrasting its term distribution to that of a general vocabulary. The normalized vocabulary filters are integrated into a framework that consists of a pipeline of three filters at the document level, collection level, and vocabulary level. Our proposed model has been empirically shown to be significantly better than TF-IDF model in tackling the archived question search problem.

In future work, we plan to explore the use of vocabulary filtering and the three-level term weighting schemes in other text processing tasks like document clustering and categorization.

References

1. Aizawa, A.: An information-theoretic perspective of tf-idf measures. *Inf. Process. Manage.* 39(1), 45–65 (2003)
2. Amati, G., Joost, C., Rijsbergen, V.: Probabilistic models for information retrieval based on divergence from randomness. *ACM TOIS* 20, 357–389 (2002)
3. Frank, E., Paynter, G.W., Witten, I.H., Gutwin, C., Nevill-manning, C.G.: Domain-specific keyphrase extraction, pp. 668–673. Morgan Kaufmann Publishers, San Francisco (1999)
4. Hiemstra, D.: A probabilistic justification for using tf idf term weighting in information retrieval. *International Journal on Digital Libraries* 3, 131–139 (2000)
5. Jeon, J., Croft, W.B., Lee, J.H.: Finding similar questions in large question and answer archives. In: *Proc. of CIKM*, pp. 84–90. ACM, New York (2005)
6. Kor, K.-W., Chua, T.-S.: Interesting nuggets and their impact on definitional question answering. In: *Proc. of SIGIR*, pp. 335–342. ACM, New York (2007)
7. Lo, R., He, B., Ounis, I.: Automatically building a stopword list for an information retrieval system. In: *Proc. of Dutch-Belgian DIR, Utrecht, Netherlands* (2005)
8. Makrehchi, M., Kamel, M.S.: Automatic extraction of domain-specific stopwords from labeled documents. In: Macdonald, C., Ounis, I., Plachouras, V., Ruthven, I., White, R.W. (eds.) *ECIR 2008. LNCS*, vol. 4956, pp. 222–233. Springer, Heidelberg (2008)
9. Ounis, I., Amati, G., Plachouras, V., He, B., Macdonald, C., Lioma, C.: Terrier: A High Performance and Scalable Information Retrieval Platform. In: *Proc. of SIGIR Workshop on OSIR* (2006)
10. Robertson, S.E., Walker, S.: Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In: *Proc. of SIGIR*, pp. 232–241. Springer, New York (1994)
11. Robertson, S.E., Zaragoza, H.: The probabilistic relevance model: Bm25 and beyond. In: *Tutorial of SIGIR*, ACM, New York (2008)
12. Roelleke, T., Wang, J.: Tf-idf uncovered: a study of theories and probabilities. In: *Proc. of SIGIR*, pp. 435–442. ACM, New York (2008)
13. Salton, G., Yang, C.: On the specification of term values in automatic indexing. *Journal of Documentation* 29, 351–372 (1973)
14. Wang, K., Ming, Z., Chua, T.-S.: A syntactic tree matching approach to finding similar questions in community-based qa services. In: *Proc. of SIGIR*, pp. 187–194. ACM, New York (2009)
15. Xue, X., Jeon, J., Croft, W.B.: Retrieval models for question and answer archives. In: *Proc. of SIGIR*, pp. 475–482. ACM, New York (2008)

On Finding the Natural Number of Topics with Latent Dirichlet Allocation: Some Observations

R. Arun, V. Suresh, C.E. Veni Madhavan, and M. Narasimha Murty

Department of Computer Science and Automation, Indian Institute of Science,
Bangalore 560 012, India

{arun_r,vsuresh,cevm,mmm}@csa.iisc.ernet.in

Abstract. It is important to identify the “correct” number of topics in mechanisms like Latent Dirichlet Allocation(LDA) as they determine the quality of features that are presented as features for classifiers like SVM. In this work we propose a measure to identify the correct number of topics and offer empirical evidence in its favor in terms of classification accuracy and the number of topics that are naturally present in the corpus. We show the merit of the measure by applying it on real-world as well as synthetic data sets(both text and images). In proposing this measure, we view LDA as a matrix factorization mechanism, wherein a given corpus C is split into two matrix factors M_1 and M_2 as given by $C_{d*w} = M1_{d*t} \times Q_{t*w}$. Where d is the number of documents present in the corpus and w is the size of the vocabulary. The quality of the split depends on “ t ”, the right number of topics chosen. The measure is computed in terms of symmetric KL-Divergence of salient distributions that are derived from these matrix factors. We observe that the divergence values are higher for non-optimal number of topics – this is shown by a ‘dip’ at the right value for ‘ t ’.

Keywords: LDA Topic SVD KL-Divergence.

1 Introduction

Topic Modelling is a widely used technique in information retrieval, data mining etc. The idea behind it is the fact that a small number of latent topics are enough to effectively represent a large corpus. As this is often the case with real world corpus such as text which have a large vocabulary, such models have proved to be very effective. However finding the right number of latent topics in a given corpus has remained an open ended question. Almost all previous methods including Latent Semantic Analysis [1], Probabilistic Latent Semantic Analysis [2], Latent Dirichlet Allocation [3], Non-Negative Matrix Factorization [4] which try to model the latent topics either as probability distributions or as a set of basis vectors in the topic space make the implicit assumption that the number of topics is known beforehand. While estimating the right number of topics for a small image or text corpus might seem easy, it becomes unreasonable to guess the same when the corpus size is huge. However the accuracy of all of the above mentioned methods is sensitive to the number of topics.

In this paper, we consider the Latent Dirichlet Allocation (LDA) [3] model as the basis for our work. We view LDA as a matrix factorization method which factorizes a document-word frequency matrix M into two matrices $M1$ and $M2$ of order $T*W$ and $D*T$ respectively where T is the number of topics and W is the size of the vocabulary of the corpus. We propose a new measure that computes the symmetric Kullback-Leibler divergence of the Singular value distributions of matrix $M1$ and the distribution of the vector $L * M2$ where L is a $1 * D$ vector containing the lengths of each document in the corpus. We show that under certain conditions these distributions are comparable and these conditions are expected to determine the ‘right’ number of topics. We also present empirical results that indicate that the proposed measure dips down and hits a low for the ‘right’ number of topics and increases again as the number of topics increase. The number of topics that is considered ‘right’ is any number in a small range that gives the best accuracy on a held out dataset.

This work is organized into the following sections: In Section 2, we review some related work in topic modelling and some methods proposed to choose the ‘right’ number of topics. In section 3, we motivate the rationale behind the measure proposed and explain how it is computed. In section 4, we give experimental evidence to illustrate the robustness of the measure across text and image corpus. Finally we conclude in section 5 with a few points of discussion.

2 Background

2.1 Latent Dirichlet Allocation

LDA is a probabilistic generative model which assumes that every document is a distribution over topics and every topic is a distribution over words. Each word in a document is generated by first sampling a topic from the topic-distribution associated with the document and then sampling a word from the word distribution associated with the topic. Thus, given a corpus, LDA tries to find the right assignment of topic to every word such that the parameters of the generative model are maximized.

Topic Similarity. There have been a couple of approaches in the past which have tried to take advantage of the fact that the topics arising in real world data are correlated. Correlated Topic Models [12] is one such approach which tries to capture relation between topics using a covariance matrix. The Pachinko Allocation Model [14] on the other hand considers an acyclic graph where a topic is a node and is considered as a distribution over not only words but also other topics.

There have also been approaches like Hierarchical Dirichlet Process (HDP) [11] which try to find the right number of topics by assuming that the data has a hierarchical structure to it. Here, both HDP as well as LDA models for the same dataset are built and compared to find the right number of topics.

More recently [10] proposes a method to learn the right size of an ontology by measuring the change in the average cosine distance between topics found as

the number of topics increase. A similar idea is found in [5] where the average correlation between $\binom{n}{2}$ pairs of topics at each stage is considered.

The disadvantage we feel with both [10] and [5] is that they consider only the information in the stochastic topic-word $T * W$ matrix and ignore the document-topic $D * T$ matrix. In the present work, we make use of properties of both these matrices to come up with a robust measure that will help in identifying the right number of topics.

3 Proposed Measure

3.1 Matrix Row Sums

Though LDA is a probabilistic generative model, it can be viewed as a non-negative matrix factorization method that breaks a given Document-Word Frequency Matrix M into a Topic-Word matrix $M1$ of order $T * W$ and a Document-Topic matrix $M2$ of order $D * T$ where D , T and W represent the number of documents, topics and words respectively. Both $M1$ and $M2$ are stochastic matrices where the k th row in $M1$ is a distribution over words in the k^{th} topic and n^{th} row in $M2$ is a distribution of topics in the n^{th} document. If these were not stochastic matrices, but just represented counts i.e if the $(i, j)^{th}$ element in matrix $M1$ indicated the number of the times word j has been assigned topic i and if the (i, j) th element in matrix $M2$ indicated the number of times topic j is assigned to a word in document i , then it is clear that

$$\sum_{v=1}^W M1(t, v) = \sum_{d=1}^D M2(d, t) \quad \forall t = 1 \text{ to } T .$$

This is nothing but the number of words assigned to each topic looked in two different ways - one as row sum over words and other as column-sum over documents. However, when both these matrices are row normalized (as done by LDA), this equality will not hold anymore.

The idea behind the proposed measure is to take advantage of the simple fact that both these sums represent proportion of topics assigned to the corpus and hence can be compared with each other. However, a mere comparison between these values is useless as they always will be the same irrespective of the number of topics considered. Hence, we seek a measure which while trying to compare similar properties of these matrices will also be low only when the ‘right’ number of topics is reached.

3.2 Distribution over Singular Values

Singular Value Decomposition (SVD) [8] is a matrix factorization technique that breaks (uniquely) any rectangular matrix M of order $m * n$, $m \leq n$ into three matrices U , Σ and V or orders $m * m$, $m * n$ and $n * n$ respectively such that

$$M = U * \Sigma * V'$$

where V' denotes the transpose of V . The matrix Σ is diagonal matrix and the matrices U and V are unitary. Also U contains the eigenvectors of matrix $M * M'$ and V contains the eigenvectors of the matrix $M' * M$.

The diagonal entries of Σ are called the singular values, denoted by σ_i , $i = 1$ to m .

Geometrically, the singular values represent the length of the semi-axes of the hyper-ellipsoid that encloses all the vectors in the matrix. So, a distribution of these singular values will give the distribution of the variance in direction of each axis of the hyper-ellipsoid. Now, if we consider the $T * W$ matrix $M1$, its distribution of singular values will be the distribution of variance in topics.

For simplicity, let us consider a case where the words in the vocabulary set are *well separated* in $M1$. By this we mean that the words in the vocabulary are partitioned into T sets V_i , $i = 1$ to T such that $V_i \cap V_j = \emptyset$ when $i \neq j$. Now if each topic T_i (a row in matrix $M1$) contains words only from set V_i , then the following proposition holds for $M1$

Proposition: If the words in the vocabulary are well separated, then the singular value σ_i is equal to the L_2 norm of row- i vector of $M1$, $\forall i = 1$ to m

Proof: This is easy to see both geometrically and algebraically. Geometrically, we observe that when the rows are well separated, the row vectors are orthogonal to each other. Thus, the axes of the hyper-ellipsoid will be row vectors themselves which means the singular values will be their distance from origin, or in other words the L_2 norm.

Algebraically, we see that for any well separated matrix $M1$, the matrix $(M1) * (M1)'$ will be a diagonal matrix with the standard basis as the eigenvectors. Thus if SVD of $M1 = U * \Sigma * V'$, then as columns of U are the eigenvectors of $(M1) * (M1)'$ i.e the standard basis. Hence the $(i, j)^{th}$ entry $(i, j = 1$ to $m)$ in V is given by

$$(V)_{ij} = (M1)_{ij} / \sigma_i$$

As the matrix V is a unitary matrix, we need $V * V'$ to be I and equating each element of $V * V'$ to elements of I proves the proposition.

Thus if in the ideal case, when the topics are well separated, the Singular values will equal the row L_2 norms. Also note that the distribution over singular values will not change by making the matrix stochastic if and only if the number of words assigned to each topic is the same. This can be proved by breaking the matrix into product of a diagonal matrix and the normalized matrix and considering when the distributions of singular values will be the same. But unfortunately both the notion of 'well-separatedness' and same number of words getting assigned to each topic are not likely to hold in real world datasets. Still we can hope that increasing the number of topics will make the topic vectors more and more orthogonal to each other, and hence the distribution over singular values will be close enough to the distribution over the row L_2 norms when the 'right' number of topics is reached.

3.3 Topic Splitting

We observe that the following phenomenon holds for the vectors got from LDA for well separated datasets. If a dataset contains K well separated topics and

when LDA is run with $K' > K$ topics, the K' word distributions over topics obtained are also orthogonal to each other. This can happen when topics 'split' which means that a set of words assigned (with high probability) to a particular topic gets assigned to the new topic(s). Thus the new set of topics will still remain orthogonal to each other i.e the new set of topic vectors will form an orthogonal basis in the bigger topic space as well. This becomes an issue if we wish to compare the singular value distributions with the L_2 row norms as they will remain close even after the topics get nearly orthogonal to each other. Of course, we could look at the first topic number where the topics get nearly orthogonal to each other, but this is difficult to judge on real world datasets. So, we require a measure which will increase once the 'right' number of topics is identified.

3.4 Norms in Higher Dimension

As mentioned in section 3.1, the sums of rows of matrix $M1$ is equal to the sums of columns of matrix $M2$. This of course will not hold once the matrices are made stochastic. Let $M1$ and $M2$ be row-stochastic. Now consider the product of vector L which contains the length of each document with matrix $M2$. We get a vector of length T with components indicating the fraction of each topic present in the corpus. Let the normalized version of this vector be called C_{M1} . (to indicate distribution of topics in the corpus C got from the matrix $M1$). If the lengths of all the documents were the same, this would be equal to the L_1

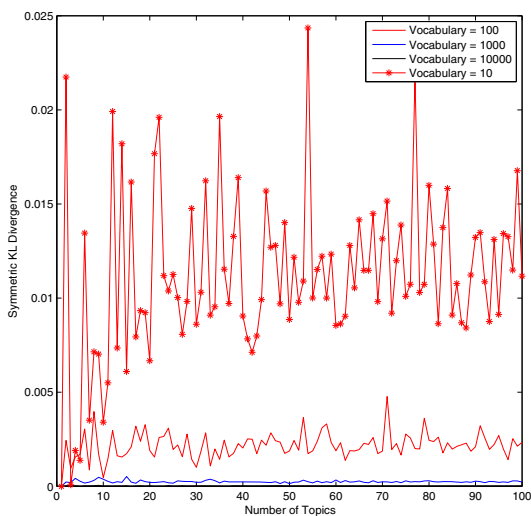


Fig. 1. Plot showing how divergence between L_1 and L_2 norm distributions vary with topics for different vocabulary sizes. As seen, for high vocabulary, the divergence is almost zero.

norm of the row vectors in $M2$. Also let C_{M2} be the distribution over singular values of matrix $M1$.

As the best use of topic models such as LDA arise when the dimension (i.e. vocabulary size) is large, we shall assume that the datasets that we deal with are vectors in high dimension. In such cases, we observe that for a random matrix R of order $T * W$, the vector R_{l1} which is the distribution of row L_1 norms and the vector R_{l2} , the distribution over row L_2 norms look very similar component-wise when W is large enough. An example is given in Figure 1 where, as the topics are varied from $T = 1$ to 100, the Symmetric Kullback-Leibler (KL) divergence [9] is calculated for four different values of W . It can be seen that as W becomes large enough, the Symmetric KL divergence goes towards zero. This will happen when the components of both the vectors are very close to each other so that every term in the Symmetric KL divergence of R_{l1} and R_{l2} which is defined as $KL(R_{l1}||R_{l2}) = \sum_{i=1}^T R_{l1}(i)*log(R_{l1}(i)/R_{l2}(i)) + \sum_{i=1}^T R_{l2}(i)*log(R_{l2}(i)/R_{l1}(i))$ goes to zero.

However, this behavior need not hold true when we consider divergence between distributions generated from non-random matrices such as $M1$ and $M2$. In fact what we observe empirically is that this divergence between the L_1 distribution and the singular value distribution (which is close to the L_2 norm distribution if topics are orthogonal) start to increase once the right number of topics is reached. The reason we believe is the fact that once topic-splitting happens after topics become nearly orthogonal, the L_1 norm acts as a penalty term in the sense that more and more noise gets added in terms of low probability values for words not belonging to a topic which contribute to the increase in divergence value.

3.5 Divergence Measure

We summarize the proposed divergence measure here. For a given corpus C and a given topic T , LDA outputs two stochastic matrices $M1$ and $M2$. The proposed measure is the following:

$$ProposedMeasure(M1,M2) = KL(C_{M1}||C_{M2}) + KL(C_{M2}||C_{M1})$$

where ,

C_{M1} is the distribution of singular values of Topic-Word matrix $M1$,
 C_{M2} is the distribution obtained by normalizing the vector $L * M2$ (where L is $1 * D$ vector of lengths of each document in the corpus and $M2$ is the Document-Topic matrix).

A point to be mentioned here is that both the distributions C_{M1} and C_{M2} are in sorted order so that the corresponding topic components are expected to match.

In the next section, we give results for various experiments conducted on both image and text data on toy as well as real world datasets that illustrate the efficacy of the proposed measure in finding the right number of topics.

4 Experiments

4.1 Image Data

Color Palette. We conducted a simple toy experiment on an palette image containing various colors as shown in the left panel of Figure 2. A document in this case was a row of pixel values in the image. Each pixel had three component values for Red, Green and Blue, in the range 0-255. These three component values were concatenated together to form a single value. (For example RGB values of 220,245,230 was made into a single number 220245230). The image was in *jpg* format and hence the pixels values are not all the same even in a palette with a single perceived color. The number of dimensions (or unique pixel values) for this image is 27777. But as we can see, the number of latent topics is smaller by a huge magnitude. The values of the proposed measure is plotted against various topic values. As we observe from the right pane of Figure 2, the measure dips down close to zero in the range 30 - 40 which in fact is the number of perceived colors ('latent topics') in the palette image.

This simple experiment where the intuitive number of topics is the number of palettes (36 in this case) clearly demonstrates the efficacy of the proposed measure.

Gray-Scale Palette. The immediate idea was to check the effect of dimensionality on the same image. To this end, we converted the image to a gray scale and hence reduced the number of dimensions from 27777 to 255. The same measure for the gray scale image is plotted in Figure 3. As we see, the dip and rise though not wrong, is not as indicative as it was in the previous case. This suggests that

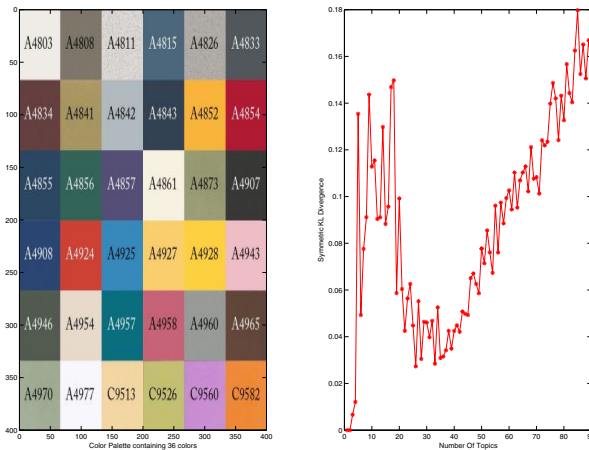


Fig. 2. Number of Topics Vs Symmetric KL Divergence for Color Palette. The dip in divergence is obtained when the number of topics is in the range of 30-40 which is same as the perceived number of 36. (to be viewed in color).

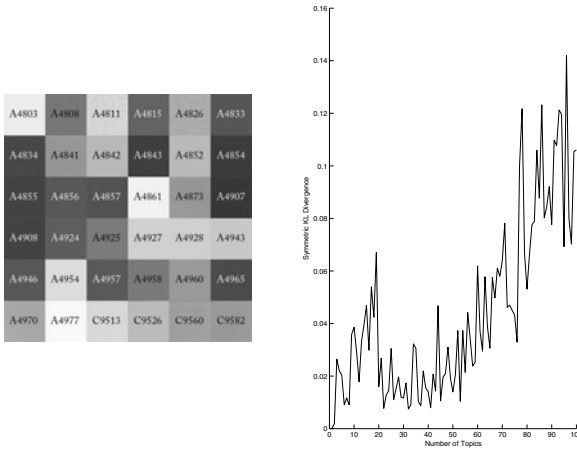


Fig. 3. Number of Topics Vs Symmetric KL Divergence for Gray scale Palette. Note that the range for right number of topics is correct, but not as indicative as in the Color Palette case.

the proposed measure is a good indicator of right number of topics for datasets involving high dimensions.

4.2 Text Data

We conducted several experiments on toy as well as real world text data sets. Each of these is explained below.

Toy Dataset. In this experiment, 12 documents of average length of 500 words were considered. The number of dimensions (vocabulary) was 1525. The documents were wikipedia articles on 3 broad topics (Science , Dance and Dostoyevsky) with 4 documents in each. Each of these topics had sub-topics (such as Quantum Mechanics, Probability, Electromagnetism etc under Science - Salsa, Mambo, Tango etc under Dance - Crime and Punishment, Brothers of Karamazov, Prison Experiences etc under Dostoyevsky). The Measure values are plotted in the left pane of Figure 4. We see a dip starting at around 20 , which is roughly a reasonable number of low level noise free topics to expect from this dataset.

Authorship Dataset. Usually, the top few words in every topic is indicative whether the topics are well split or not. If the right number of topics is reached, the words belonging to every topic are expected to be semantically close to each other. While this might help in deciding the right number of topics for small datasets, it becomes tough to decide the same on abstract datasets. To verify this, we built a dataset containing literary works of 12 authors. Each work was broken up into 5000 word per document and the total number of documents was 834. The details of the dataset are given in table 1. We then stripped off

Table 1. Statistics of the Authorship dataset

| Author | Genre ¹ | Timeline | Stop-words | Content Words |
|------------------------|--------------------|-----------|------------|---------------|
| Daniel Defoe | A,F,Hi | 1808-1894 | 477201 | 210550 |
| Jane Austen | F,Hu,Ps | 1811-1818 | 474305 | 248676 |
| Allen Grant | B,F,Sc,Ph | 1848-1899 | 215001 | 148670 |
| George Elliot | F,Ps | 1859-1871 | 520661 | 311030 |
| Harold Bindloss | Ro,F | 1866-1945 | 525724 | 321902 |
| James Otis | A,C,F,Hi | 1883-1899 | 252518 | 136089 |
| George Bernard Shaw | D,F,Hi,Hu,W | 1885-1912 | 145385 | 85476 |
| Hamlin Garland | A,F,Ps,Ro,Sp,T | 1897-1921 | 349296 | 229164 |
| Captain Ralph Bonehill | A | 1902 | 175659 | 105169 |
| Phillips Oppenheim | F,M,Po | 1902-1920 | 415892 | 243386 |
| G K Chesterton | M,Ph,Ps,Re | 1905-1916 | 186409 | 111290 |
| Baroness Orczy | A,Hi,Po,Ro | 1905-1921 | 392127 | 261019 |
| Total | 18 | 1808-1921 | 4130178 | 2307252 |

the content words and retained words which fell in a small set of 555 stop-words (words such as 'and', 'the', 'it' etc). We ran LDA on this dataset to test its efficacy in classifying the authors. As the vocabulary contains only stop words, it is not easy to identify the cohesion in a topic by looking at the top few words. Though, the implementation details and other stylometric properties studied from this dataset are explained elsewhere, we just mention here that the accuracy on a held out testset was highest when the number of topics was 15-25. The plot in the right pane of Figure 4 shows the variation of the proposed measure with topics. As we see, the graph seems to linearly increase with a very small dip at around the right number of topics. The reason for the dip not being significant can again be attributed to the number of dimensions being not very large (555 in this case). However, it is easy to infer a broad range for the right number of topics from the plot.

NIPS Dataset/ AP corpus. We now present our results on two real world text datasets. The first one is a standard collection of bag-of-words from NIPS corpus [15] containing 1500 documents with a total of 1932365 words and 12419 dimensions (vocabulary). The plot of our measure for this dataset is shown in left pane of Figure 5 has a dip in the range 100 - 120. [13] compares LDA performance on the same dataset using different number of processors. They consider topics till 80 and observe that irrespective of the processor count, LDA perplexity value goes down from 20 topics to 80 topics on a held out dataset.

¹ A - Adventure, Au - Autobiography, B - Biography, C - Children, D - Drama, F - Fiction Hi - History, Hu - Humour, M - Mystery, Ph - Philosophy, Po - Politics, Ps - Psychology R - Religion, Ro - Romance, Sc - Science, Sp - Spirituality, T - Travel, W - War.

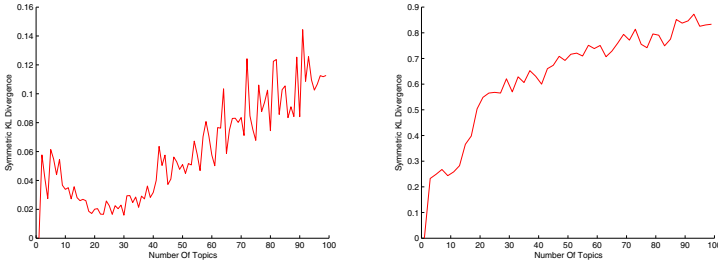


Fig. 4. Plot showing how the proposed measure varies with number of topics for Toy Dataset (Left) and Authorship Dataset(Right)

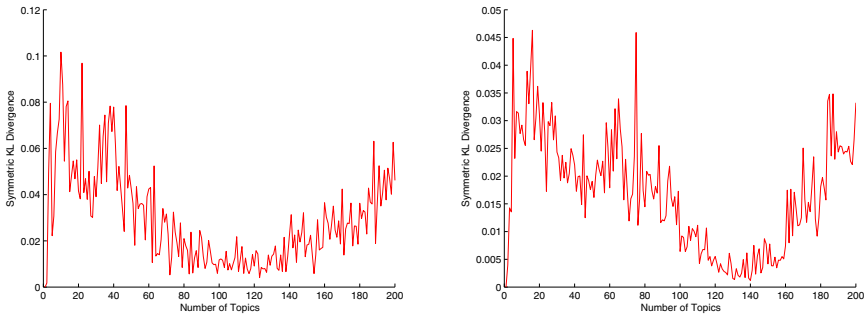


Fig. 5. Plot showing how the proposed measure varies with number of topics for NIPS abstract Dataset (Left) and Associated Press corpus Dataset(Right). The dip is seen the right number of topics for which lowest perplexity is reported.

The second is a collection of articles from the Associated Press dataset [15] containing 2246 documents with 435839 words and 10473 dimensions. The proposed measure plot is shown in the right pane of Figure 5. The best number of topics is seen at around 140. [3] reports that the perplexity reduces until the number of topics is 100 and then there seems to be very little change in the range 100 to 200. But in our case, we observe a steady increase after 130 to 140 topics which is a smaller range for fixing the topic number than what the perplexity values indicate.

5 Discussions and Conclusion

While it might seem that in using the proposed measure, LDA has to be run once for every topic to get to the right number of topics. But this need not be the case always. In most real world datasets, we observe that the variance between divergence values decreases significantly when the right number of topics is reached. This could be taken as a cue to jump appropriately to get nearer to the correct topic number.

Also we mention two points for clarification and emphasis. The first is that comparing the Singular value distribution with the row L_2 norm distribution is not very useful as the measure will not increase once the right number of topics is reached and so it becomes tough to guess the right range of topics. The second point is that Singular value distribution cannot be compared directly with the row L_1 norm of the $M1$ matrix because of its stochasticity. Hence we need to reconstruct the corpus topic distribution from the matrix $M2$. Also it is not correct to compare the singular value distribution (or the row L_2 norm) with the column L_2 norm of $M2$ as the components in a column vector of $M2$ represent distribution over individual documents which will not be same as the distribution over words.

To summarize, we have proposed a new measure for identifying the right number of topics in a give corpus by looking at distributions generated from Topic-Word and Document-Topic matrix outputs of LDA. We showed that the distribution over singular values is close to the distribution over row L_2 norm when the topics become orthogonal . Further, in high dimension the distribution over L_1 and L_2 norms tend to converge for random matrices and hence become candidates for comparison. The measure proposed combines these two facts and compares the singular value distribution of Topic-Word matrix with the row L_1 norm of the Document-Topic matrix. to We illustrated the efficacy of the measure by testing it on several real world and synthetic datasets and on both text and images.

In the future, We hope to explore further in the direction of arriving at more robust theoretical justifications and possible worst case bounds for the proposed measure.

References

1. Deerwester, S.C., Dumais, S.T., Landauer, T.K., Furnas, G.W., Harshman, R.A.: Indexing by Latent Semantic Analysis. *JASIS* 41(6), 391–407 (1990)
2. Hofmann, T.: Probabilistic Latent Semantic Indexing. In: *SIGIR 1999*, pp. 50–57 (1999)
3. Blei, D.M., Ng, A.Y., Jordan, M.I.: Jordan: Latent Dirichlet Allocation. *Journal of Machine Learning Research* 3, 993–1022 (2003)
4. Lee, D.D., Sebastian Seung, H.: Learning the parts of objects by non-negative matrix factorization. *Nature* 401(6755), 788–791 (1999)
5. Cao, J., Xia, T., Li, J., Zhang, Y., Tang, S.: A density-based method for adaptive LDA model selection. *Neurocomputing* 72(7-9), 1775–1781 (2009)
6. Aggarwal, C.C., Hinneburg, A., Keim, D.A.: On the Surprising Behavior of Distance Metrics in High Dimensional Spaces. In: Van den Bussche, J., Vianu, V. (eds.) *ICDT 2001*. LNCS, vol. 1973, pp. 420–434. Springer, Heidelberg (2000)
7. Gaussier, E., Goutte, C.: Relation between PLSA and NMF and Implications. In: *Proc. 28th international ACM SIGIR conference on Research and development in information retrieval (SIGIR 2005)*, pp. 601–602 (2005)
8. Eckart, C., Young, G.: The approximation of one matrix by another of lower rank. *Psychometrika* 1(3), 211–218 (1936)

9. Kullback, S., Leibler, R.A.: On Information and Sufficiency. *Annals of Mathematical Statistics* 22(1), 79–86 (1951)
10. Zavitsanos, E., Petridis, S., Paliouras, G., Vouros, G.A.: Determining Automatically the Size of Learned Ontologies. In: *ECAI 2008*, pp. 775–776 (2008)
11. Teh, Y.W., Jordan, M.I., Beal, M.J., Blei, D.M.: Sharing Clusters among Related Groups: Hierarchical Dirichlet Processes. In: *NIPS 2004* (2004)
12. Blei, D.M., Lafferty, J.D.: Correlated Topic Models. In: *NIPS 2005* (2005)
13. Smyth, P., Welling, M.: Asynchronous Distributed Learning of Topic Models. In: *NIPS 2008*, pp. 81–88 (2008) (bibliographical record in XML Arthur Asuncion)
14. Li, W., McCallum, A.: Pachinko allocation: DAG-structured mixture models of topic correlations. In: *ICML 2006*, pp. 577–584 (2007)
15. <http://archive.ics.uci.edu/ml/datasets/Bag+of+Words>,
<http://www.cs.princeton.edu/~blei/lda-c/>

Supervising Latent Topic Model for Maximum-Margin Text Classification and Regression

Wanhong Xu

Carnegie Mellon University
5000 Forbes Ave., Pittsburgh PA 15213
wanhong@cmu.edu

Abstract. In this paper, we investigate the text classification and regression problems: given a corpus of text documents as training, each of which has a response label, the task is to train a predictor for predicting its response of any given document. In previous work, many researchers decompose this task into two separate steps: they first use a generative latent topic model to learn low-dimensional semantic representations of documents; and then train a max-margin predictor using them as features. In this work we demonstrate that it is beneficial to combine both steps of learning low-dimensional representations and training a predictor into one step of minimizing a single learning objective. We present a novel step-wise convex optimization algorithm which solves this objective properly via a tight variational upper bound. We conduct an extensive experimental study on public available movie review and 20 Newsgroups datasets. Experimental results show that compared with state of art results in the literature, our one step approach can train noticeably better predictors and discover much lower-dimensional representations: a 2% relative accuracy improvement and a 95% relative number of dimensions reduction in the classification task on the Newsgroups dataset; and a 5.7% relative predictive R^2 improvement and a 55% relative number of dimensions reduction in the regression task on the movie review dataset.

1 Introduction

With tremendous text information made available online, there is a growing demand to analyze and manage large corpuses of electronic text. Learning low-dimensional semantic representations of text documents is a common and often necessary step for various applications and text analyses. For example, this low-dimensional semantic representation has been used for structurally browsing a text corpus and categorizing and clustering text documents in information retrieval domain.

A recent trend in learning low-dimensional semantic representations focuses on generative latent probabilistic models based on so-called *topics*. The belief behind those latent topic models is that a document consisting of a large number of words might be concisely described as a smaller number of semantic themes. A topic is a probability distribution over words of a vocabulary, and is used to statistically describe a semantic theme. Then, a document is semantically represented as a mixture of topics.

In the literature, most popular latent topic models are Latent Dirichlet Allocation (LDA) [2] and Probabilistic Latent Semantic Analysis (pLSA) [5]. Particularly, LDA

is a Bayesian version of pLSA. Both models are unsupervised to simultaneously discover topics and low-dimensional topic representations of documents, and have been successfully used in various applications [7,3].

Unfortunately, semantic representations produced in this unsupervised manner may not necessarily be good features for classification and regression tasks. The reason is that topics are learned without considering document labels to be predicted, for example the categories of news postings. Those unsupervised learned topics describe semantic themes that generally happen in all documents and don't describe semantic themes discriminative across document categories. Therefore, semantic representations of documents based on general topics are not well distinctive against document categories and those two-step approaches of building predictors subsequently on them would result in sub-optimal performance. We believe that topics should contain as much discriminative information as possible from document labels such that semantic representations based on them are suitable for prediction.

In this paper, we propose an approach to integrate the latent topic model for learning low-dimensional semantic representations and support vector machine or regression for training max-margin predictors into one single learning objective. By coupling them together, we are able to supervise the latent topic model with benefits of the maximum margin principle, and guide it to discover topics describing discriminative information and generate semantic representations more suitable for prediction tasks. Due to the optimization hardness of the single learning objective, we propose a tight variational upper bound for the single learning objective and develop a novel step-wise convex algorithm for optimizing the upper bound. For both classification and regression tasks, we present experiments showing that our approach can achieve better predictive power and is able to discover much lower dimensional representations than two-step approaches and also three state of art methods.

2 Preliminaries

We first introduce notations that will be used throughout the paper; then review latent topic models and max-margin classifier and regressor.

A text document d is a sequence of N words $\langle w_1 w_2 \dots w_N \rangle$, where each word is from a fixed vocabulary with totally V words. Following a common bag-of-words assumption, we represent this document as a bag of words. The document could have a response label y , which is either a categorical class, or a continuous real number. Let D be a corpus of M labeled documents. The problem in this work is to learn a good predictor using D as training data for predicting the response label of a new document.

2.1 Latent Topic Model

To represent a document by semantic topics, we define a K -topic vocabulary T . Each topic t of T is a multinomial distribution of all words in the vocabulary, i.e. $\{p(w|t)\}_{w=1..V}$, simply denoted as $\beta_{t,:}$. We also let β be the set of all topics, i.e., $\{\beta_{t,:}\}_{t=1..K}$. Then, we can represent the document by a topic mixture proportion $\theta = \{p(t)\}_{t=1..K}$.

This topic representation implies a generative process to documents. For each word w_n in a document d , we

1. draw a topic assignment $z_n|\theta \sim Mult(\theta)$, where z_n has the 1-of-K representation;
2. draw the word $w_m|z_n, \beta \sim Mult(\beta_{z_n, \cdot})$.

Both LDA and pLSA are based on the above generative process. The difference is that LDA introduces a Dirichlet prior to θ for alleviating the potential overfitting problem of pLSA. For classification and regression tasks, both models show no difference in prediction performance. Therefore, to keep our approach simple, we recruit pLSA as the topic model in our approach.

Given a corpus D with M documents, pLSA minimizes the following negative log likelihood to learn topics β and also estimate topic proportion θ for each document:

$$\min_{\Theta, \beta} -L(D; \Theta, \beta) = - \sum_{m=1}^M \sum_{n=1}^{N_m} \log \sum_{z_{m,n}=1}^K p(z_{m,n}|\theta_m)p(w_{m,n}|\beta_{z_{m,n}, \cdot}),$$

where $\Theta = \{\theta_m\}_{m=1..M}$ denotes the collection of all topic proportions. We let $\bar{z}_m = \frac{1}{N_m} \sum_{n=1}^{N_m} z_{m,n}$, which is the empirical topic proportion of document m , and let $\bar{Z} = \{\bar{z}_m\}_{m=1..M}$ denote the set of all empirical topic proportions of D .

2.2 Max-margin Classification and Regression

The empirical topic proportion \bar{z}_m from the latent topic model and the document label y_m are used to build max-margin predictors, for example support vector machine (SVM) [4] for classification and support vector regressor (SVR) [10] for regression.

For example, if $y_m \in \{-1, 1\}$ is a binary categorical label, we can learn a SVM $\langle \omega, b \rangle$ from the corpus D for classification by minimizing the following loss function:

$$\min_{\omega, b} C(\omega, b, \bar{Z}) = \frac{1}{M} \sum_{m=1}^M \max\{0, 1 - y_m(\omega^T \bar{z}_m + b)\} + \lambda \|\omega\|_2,$$

where the first term measures the classification error of $\langle \omega, b \rangle$ and the second is a penalty term on ω to avoid over-fitting. Similarly, if y_m is continuous, we can learn a SVR $\langle \omega, b \rangle$ with a pre-defined precision ϵ from Corpus D for regression by minimizing the following loss function:

$$\min_{\omega, b} R(\omega, b, \bar{Z}) = \frac{1}{M} \sum_{m=1}^M \{\max\{0, y_m - \epsilon - \omega^T \bar{z}_m - b\} + \max\{0, \omega^T \bar{z}_m + b - y_m - \epsilon\}\} + \lambda \|\omega\|_2,$$

where the first term measures the prediction error of $\langle \omega, b \rangle$ on the ϵ precision, and the second is a penalty term on ω too.

Because \bar{z}_m is a hidden variable, we need to minimize the expected two loss functions $E_{\bar{Z}}(C(\omega, b, \bar{Z}))$ and $E_{\bar{Z}}(R(\omega, b, \bar{Z}))$. But, it is hard to compute them because of the non-integrable max function in two loss functions. To circumvent this difficulty, two-step approaches minimize $C(\omega, b, E(\bar{Z}))$ and $R(\omega, b, E(\bar{Z}))$ instead. Because $E(\bar{Z})$ has a closed form Θ , this alternative makes possible using standard SVM and SVR solver.

However, we find that both $R(\omega, b, \bar{Z})$ and $C(\omega, b, \bar{Z})$ are convex on \bar{Z} . Therefore, $C(\omega, b, E(\bar{Z}))$ and $R(\omega, b, E(\bar{Z}))$ are lower bounds of $E_{\bar{Z}}(C(\omega, b, \bar{Z}))$ and $E_{\bar{Z}}(R(\omega, b, \bar{Z}))$ respectively according to Jensen's inequality. It is obvious that minimizing lower bounds of $E_{\bar{Z}}(C(\omega, b, \bar{Z}))$ and $E_{\bar{Z}}(R(\omega, b, \bar{Z}))$ in two-step approaches are problematic and doesn't guarantee $E_{\bar{Z}}(C(\omega, b, \bar{Z}))$ and $E_{\bar{Z}}(R(\omega, b, \bar{Z}))$ themselves to be minimized too.

3 Framework of Supervising Latent Topic Model

As discussed in the introduction, the goal is to learn semantic topics β describing discriminative themes among documents in the corpus D such that representations of documents by those discriminative topics could be suitable for prediction purpose.

To achieve this goal, two requirements should be satisfied together. On one hand, topics β should be common to make possible a to-be-predicted document be well described by them too. This suggests that the negative log likelihood of the latent topic model $-L(D; \Theta, \beta)$, which measures how data fits those topics β , should be minimized. On the other hand, the empirical topic proportions \bar{Z} can be used to build good predictors. It implies that the expected loss functions $E_{\bar{Z}}(C(\omega, b, \bar{Z}))$ and $E_{\bar{Z}}(R(\omega, b, \bar{Z}))$ of SVM and SVR, which measure how well they are used for prediction, should be minimized too.

To satisfy both requirements together, we intuitively couple them linearly by a positive tradeoff η to a single learning objective. For classification and regression tasks, we need to minimize the following single learning objectives respectively:

$$\min_{\Theta, \beta, \omega, b} -L(D; \Theta, \beta) + \eta E_{\bar{Z}}(C(\omega, b, \bar{Z})) \quad (1)$$

$$\min_{\Theta, \beta, \omega, b} -L(D; \Theta, \beta) + \eta E_{\bar{Z}}(R(\omega, b, \bar{Z})) \quad (2)$$

Therefore, in the single learning objectives, the expected losses of max-margin predictors are used to penalize or supervise the latent topic model to generate discriminative topics suitable for prediction. We name this framework maximum margin latent topic model, shortly denoted as MMpLSA.

To solve those two learning problems, we have to remove or integrate out hidden variables $z_{m,n}$ so that convex optimization could be applied to them.

In the log likelihood function, it is easy to remove hidden variables $z_{m,n}$ by utilizing the multinomial distribution of $z_{m,n}$. We have the following alternative form of the log likelihood function:

$$-L(D; \Theta, \beta) = -\sum_{m=1}^M \sum_{n=1}^{N_m} \log \sum_{z_{m,n}=1}^K p(z_{m,n} | \theta_m) p(w_{m,n} | \beta_{z_{m,n}, :}) = -\sum_{m=1}^M \sum_{n=1}^{N_m} \log \theta_m^T \beta_{:, w_{m,n}},$$

where $\beta_{:, w_{m,n}} = \{\beta_{k, w_{m,n}}\}_{k=1..K}$ is the vector of probabilities of word $w_{m,n}$ in all topics respectively.

However, no closed forms exist for $E_{\bar{Z}}(C(\omega, b, \bar{Z}))$ and $E_{\bar{Z}}(R(\omega, b, \bar{Z}))$ because of non-integrable max functions involved in them. One way to circumvent this closed form trouble as two-step approaches is using $C(\omega, b, E(\bar{Z}))$ and $R(\omega, b, E(\bar{Z}))$ to replace $E_{\bar{Z}}(C(\omega, b, \bar{Z}))$ and $E_{\bar{Z}}(R(\omega, b, \bar{Z}))$ in the learning objectives. But, this way will have the same problem as two step approaches: minimizing lower bounds of single learning objectives doesn't guarantee they will be small too.

Instead, in this work, we propose tight closed form upper bounds of $E_{\bar{Z}}(C(\omega, b, \bar{Z}))$ and $E_{\bar{Z}}(R(\omega, b, \bar{Z}))$ such that when tight upper bounds are minimized, single learning objectives are approximately minimized too.

3.1 Variational Upper Bounds for Expected Max-margin Loss Functions

We first propose a variational upper bound for the max function $\max\{0, x\}$, and then give upper bounds of $E_{\bar{Z}}(C(\omega, b, \bar{Z}))$ and $E_{\bar{Z}}(R(\omega, b, \bar{Z}))$.

The function $\max\{0, x\}$ has an upper bound $g_\gamma(x)$ [12], which is defined as below:

$$g_\gamma(x) = \frac{x}{2} + \frac{1}{\gamma} \log(\exp(-\frac{\gamma x}{2}) + \exp(\frac{\gamma x}{2})). \quad (3)$$

It is easy to show that $\max\{0, x\} < g_\gamma(x)$ when $\gamma > 0$. This upper bound is tight because for any x , $\lim_{\gamma \rightarrow \infty} (g_\gamma(x) - \max\{0, x\}) = 0$. Let $f_\gamma(x) = \frac{1}{\gamma} \log((\exp(-\frac{\gamma x}{2}) + \exp(\frac{\gamma x}{2})))$. It is a concave function to the variable x^2 [6], and thus its first order Taylor expansion at the variable x^2 is a global upper bound,

$$f_\gamma(x) \leq f_\gamma(\psi) + \frac{1}{4\gamma\psi} \tanh(\frac{\gamma\psi}{2})(x^2 - \psi^2). \quad (4)$$

Note that this upper bound is exact whenever $\psi^2 = x^2$. Combining Eq 3 and 4 yields the desired variational upper bound of $\max\{0, x\}$,

$$\max\{0, x\} < \frac{x}{2} + f_\gamma(\psi) + \frac{1}{4\gamma\psi} \tanh(\frac{\gamma\psi}{2})(x^2 - \psi^2), \quad (5)$$

where ψ is a variational variable which gives the upper bound one degree of freedom to tightly approximate the max function.

Based on the variational upper bound of max function, we then give the variational upper bound of $E_{\bar{Z}}(C(\omega, b, \bar{Z}))$. For a document m and its empirical topic proportion \bar{z}_m , we have $E(\bar{z}_m) = \theta_m$ and $E(\bar{z}_m \bar{z}_m^T) = (N_m - 1)/N_m \cdot \theta_m \theta_m^T + 1/N_m \text{diag}(\theta_m)$, denoted as Ω_m . Putting them with Eq. 5 and knowledge of $y_m^2 = 1$ together leads to an expected upper bound of $\max\{0, 1 - y_m(\omega^T \bar{z}_m + b)\}$:

$$\begin{aligned} & E(\max\{0, 1 - y_m(\omega^T \bar{z}_m + b)\}) \\ & < E\left\{ \frac{1 - y_m(\omega^T \bar{z}_m + b)}{2} + f_\gamma(\psi_m) + \frac{\tanh(\gamma\psi_m/2)}{4\gamma\psi_m} [(1 - y_m(\omega^T \bar{z}_m + b))^2 - \psi_m^2] \right\} \\ & = \frac{1 - y_m(\omega^T \theta_m + b)}{2} + f_\gamma(\psi_m) + \frac{\tanh(\gamma\psi_m/2)}{4\gamma\psi_m} [\omega^T \Omega_m \omega + 2\omega^T \theta_m (b - y_m) + (b - y_m)^2 - \psi_m^2]. \end{aligned}$$

We define this upper bound as $B_\gamma(\theta_m, \omega, b, \psi_m)$, and then the variational upper bound of $E_{\bar{Z}}(C(\omega, b, \bar{Z}))$ is $\frac{1}{M} \sum_{m=1}^M B_\gamma(\theta_m, \omega, b, \psi_m) + \lambda \|\omega\|_2$.

Similarly, for regression, we have an expected upper bound $\max\{0, y_m - \epsilon - \omega^T \bar{z}_m - b\} + \max\{0, \omega^T \bar{z}_m + b - y_m - \epsilon\}$:

$$\begin{aligned} & E(\max\{0, y_m - \epsilon - \omega^T \bar{z}_m - b\} + \max\{0, \omega^T \bar{z}_m + b - y_m - \epsilon\}) \\ & < -\epsilon + f_\gamma(\psi_m) + \frac{\tanh(\gamma\psi_m/2)}{4\gamma\psi_m} [\omega^T \Omega_m \omega + 2\omega^T \theta_m (b - y_m + \epsilon) + (b - y_m + \epsilon)^2 - \psi_m^2] \\ & \quad + f_\gamma(\psi_m^*) + \frac{\tanh(\gamma\psi_m^*/2)}{4\gamma\psi_m^*} [\omega^T \Omega_m \omega + 2\omega^T \theta_m (b - y_m - \epsilon) + (b - y_m - \epsilon)^2 - \psi_m^{*2}]. \end{aligned}$$

We define this upper bound as $U_\gamma(\theta_m, \omega, b, \psi_m, \psi_m^*)$, and then the variational upper bound of $E_{\bar{Z}}(R(\omega, b, \bar{Z}))$ is $\frac{1}{M} \sum_{m=1}^M U_\gamma(\theta_m, \omega, b, \psi_m, \psi_m^*) + \lambda \|\omega\|_2$.

4 Optimization Procedure for Classification

Armed with the variational upper bound of the expected classification loss function proposed in the previous section, we can approximately minimize the single learning objective for classification (Eq. 1) by minimizing its following upper bound:

$$\begin{aligned}
 \min_{\Theta, \beta, \omega, b, \Psi} & - \sum_{m=1}^M \sum_{n=1}^{N_m} \log \theta_m^T \beta_{:,w_{m,n}} + \lambda_1 \sum_{m=1}^M B_\gamma(\theta_m, \omega, b, \psi_m) + \lambda_2 \|\omega\|_2 \\
 \text{subject to: } & \theta_{m,k} > 0, & m = 1 \dots M, k = 1 \dots K; \\
 & \beta_{k,v} > 0, & k = 1 \dots K, v = 1 \dots V; \\
 & \sum_{k=1}^K \theta_{m,k} = 1, & m = 1 \dots M; \\
 & \sum_{v=1}^V \beta_{k,v} = 1, & k = 1 \dots K;
 \end{aligned}$$

where λ_1 and λ_2 are absorbed terms of the objective tradeoff η and constant $1/M$ and parameter λ in the variational upper bound of the expected classification loss. Because Θ and β are parameters of multinomial distributions, self-explained constraints as above must be satisfied in this minimization.

It could be proved that this objective upper bound is variable-wise convex¹. Therefore, we could iteratively minimize it with respect to one of variables with the rest of variables fixed. Because every iteration reduces its overall value, this iterative minimization procedure will cause the value of objective upper bound to converge to a local minimum. Next, we describe this iterative procedure below starting from the simplest iterating step:

OPTIMIZE Ψ : Because variational variables are uncoupled to each other in the objective upper bound, we can divide the optimization for Ψ into M subproblems, one per variational variable. There is only one term involving the variational variables in the objective upper bound. Therefore, the objective upper bound is simplified to the below for optimizing each variational variable:

$$\min_{\psi_m} B_\gamma(\theta_m, \omega, b, \psi_m),$$

which turns out to have a closed form solution $\psi_m = \sqrt{\omega^T \Omega_m \omega + 2\omega^T \theta_m (b - y_m) + (b - y_m)^2}$.

OPTIMIZE ω AND b : The first term of the objective upper bound doesn't involve ω and b and also all constraints don't. With them dropped, the optimization for $\langle \omega, b \rangle$ is simplified to the following unconstrained optimization problem:

$$\min_{\omega, b} \lambda_1 \sum_{m=1}^M B_\gamma(\theta_m, \omega, b, \psi_m) + \lambda_2 \|\omega\|_2,$$

which tries to choose $\langle \omega, b \rangle$ for good prediction. The Hessian matrix of $\langle \omega, b \rangle$ is:

$$H(\langle \omega, b \rangle) = \lambda_1 \sum_{m=1}^M \left\{ \frac{\tanh(\gamma \psi_m / 2)}{2\gamma \psi_m} \begin{bmatrix} \Omega_m & \theta_m \\ \theta_m^T & 1 \end{bmatrix} \right\} + 2\lambda_2 \begin{bmatrix} I_{K \times K} & 0_{K \times 1} \\ 0_{1 \times K} & 0 \end{bmatrix},$$

¹ Due to the space limitation, we skip the proof in this writing. Please refer to [11] for details.

where $I_{K \times K}$ is the identity matrix and $0_{K \times 1}$ and $0_{1 \times K}$ are vectors with only 0s. This Hessian matrix involves γ , which is supposed to be large enough for well approximating the max function as shown in Sec 3.1. But when γ is big, the Hessian matrix could be ill-conditioned, which will lead to the instability of our algorithm solving this optimization problem. Our stable solution is that we first solve the minimization problem for a small γ to get optimal ω and b , and based on them we solve this problem again for a bigger γ to update optimal ω and b , and so on. In implementation, we start γ from 10 and increase it by 20 until it reaches 200.

OPTIMIZE Θ : Topic proportion θ_m s are uncoupled to each other. Therefore, we can divide the optimization for Θ into M subproblems, one per topic proportion. By dropping the third term of objective upper bound without involving Θ and constraints on β , the optimization for Θ is simplified to the following constrained optimization problem:

$$\begin{aligned} \min_{\theta_m} \quad & - \sum_{n=1}^{N_m} \log \theta_m^T \beta_{:,w_m,n} + \lambda_1 B_\gamma(\theta_m, \omega, b, \psi_m) \\ \text{subject to:} \quad & \theta_{m,k} > 0, \quad k = 1 \dots K; \\ & \sum_{k=1}^K \theta_{m,k} = 1. \end{aligned}$$

The Hessian of θ_m is:

$$H(\theta_m) = \sum_{n=1}^{N_m} \frac{\beta_{:,w_m,n} \beta_{:,w_m,n}^T}{(\theta_m^T \beta_{:,w_m,n})^2} + \lambda_1 \frac{\tanh(\gamma \psi_m / 2)}{2\gamma \psi_m} \cdot \frac{N_m - 1}{N_m} \cdot \omega \omega^T,$$

which involves γ too and has the same ill-conditioned problem when γ is large as the Hessian Matrix of $\langle \omega, b \rangle$. We use the same solution for stability as optimizing $\langle \omega, b \rangle$.

OPTIMIZE β : Only the first term of objective upper bound involves β . By keeping it and also constraints on β , we simplify the optimization for β to the following constrained optimization problem:

$$\begin{aligned} \min_{\beta} \quad & - \sum_{m=1}^M \sum_{n=1}^{N_m} \log \theta_m^T \beta_{:,w_m,n} \\ \text{subject to:} \quad & \beta_{k,v} > 0, \quad k = 1 \dots K, v = 1 \dots V \\ & \sum_{v=1}^V \beta_{k,v} = 1, \quad k = 1 \dots K. \end{aligned}$$

Based on those optimization steps, the iterative optimization procedure for classification is given in Algorithm 1. We discuss the implementation detail in the next section.

4.1 Implementation

In the beginning of the optimization procedure, for each topic t , we initialize its multinomial word distribution $\beta_{t,:}$: by sampling a dirichlet distribution with parameter $(1, \dots, 1)$; we also initialize each value of ω and b by sampling a standard normal distribution.

Every time when the optimization procedure optimizes β and ω and b , we do cross validation to check whether currently learned β are good topics for prediction and $\langle \omega, b \rangle$ is a good classifier. In the cross validation of β , topic proportion θ_c of a document c

Alg1: Optimization Procedure for Classification

Input: corpus D , model parameters λ_1 and λ_2 , and topic number K .

Output: β , ω and b .

- 1: Initialize β , ω and b ;
- 2: **repeat**
- 3: **for** $\gamma = 10; \gamma < 200; \gamma = \gamma + 20$ **do**
- 4: **repeat**
- 5: Optimize Θ ; Optimize Ψ ;
- 6: **until** convergence
- 7: **end for**
- 8: Optimize β ;
- 9: **for** $\gamma = 10; \gamma < 200; \gamma = \gamma + 20$ **do**
- 10: **repeat**
- 11: Optimize ω and b ; Optimize Ψ ;
- 12: **until** convergence
- 13: **end for**
- 14: Cross Validation on β .
- 15: **until** convergence

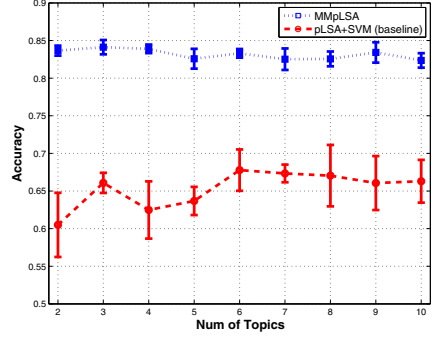


Fig. 1. Experimental results of text classification: Mean and variance of accuracies of the proposed approach MMpLSA and the two-step approach pLSA+SVM. MMpLSA performs significantly better than pLSA+SVM.

preselected for cross validation is estimated by minimizing the negative log likelihood with β fixed:

$$\begin{aligned} \min_{\theta_c} & - \sum_{n=1}^{N_c} \log \theta_c^T \beta_{:,w_{c,n}} \\ \text{subject to: } & \theta_{c,k} > 0, \quad k = 1 \dots K; \\ & \sum_{k=1}^K \theta_{c,k} = 1. \end{aligned}$$

Let \bar{z}_c be its empirical topic proportion. We predict its label by $\text{sign}(E(\omega^T \bar{z}_c + b))$, i.e., $\text{sign}(\omega^T \theta_c + b)$.

5 Optimization Procedure for Text Regression

Similar to classification, by utilizing the variational upper bound of the expected regression loss function, we can approximately minimize the single learning objective for regression (Eq. 2) by minimizing its following upper bound:

$$\begin{aligned} \min_{\Theta, \beta, \omega, b, \Psi, \Psi^*} & - \sum_{m=1}^M \sum_{n=1}^{N_m} \log \theta_m^T \beta_{:,w_{m,n}} + \lambda_1 \sum_{m=1}^M U_\gamma(\theta_m, \omega, b, \psi_m, \psi_m^*) + \lambda_2 \|\omega\|_2 \\ \text{subject to: } & \theta_{m,k} > 0, \quad m = 1 \dots M, k = 1 \dots K; \\ & \beta_{k,v} > 0, \quad k = 1 \dots K, v = 1 \dots V; \\ & \sum_{k=1}^K \theta_{m,k} = 1, \quad m = 1 \dots M; \\ & \sum_{v=1}^V \beta_{k,v} = 1, \quad k = 1 \dots K. \end{aligned}$$

Due to space limitation, please refer to [11] for the detail of the optimization procedure, which shares many commons with the optimization procedure for classification.

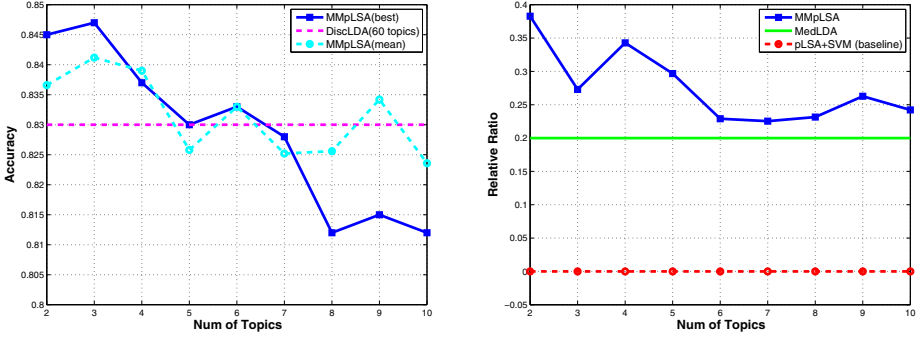


Fig. 2. Experimental results of text classification: (Left) Accuracy mean of five runs of MMpLSA and the accuracy of the run with best cross validation at different numbers of topics v.s. the best result 0.830 of state of art model DiscLDA achieved at 60 topics; MMpLSA achieved the best accuracy 84.7% with 3 topics, a 2% percent relative accuracy improvement and a 95% relative number of topics reduction; (Right) Relative improvement ratio of accuracy mean of MMpLSA against pLSA+SVM v.s. best relative ratio achieved by MedLDA

6 Experiments

In this section, we conducted experiments to evaluate our approach MMpLSA with state of art methods and also a baseline from two-step approaches; we report extensive performance results on both text classification and regression. Our experiments are able to demonstrate the advantages of applying the max-margin principle to supervise latent topic models. MMpLSA can learn from data a compacter latent representation that contains more plentiful information for prediction.

6.1 State of Art Approaches

There have been moderate efforts on supervising latent topic models for classification and regression in the literature. The most earliest work is sLDA [11], which supervises LDA for regression by assuming a normal distribution of the response y of a document and also assuming y linearly dependent on its empirical topic proportion \bar{z} , i.e., $y \sim N(\mu^T \bar{z}, \sigma^2)$. But this normality assumption doesn't hold for many real datasets. DiscLDA [8] is a discriminative variant for classification. It assumes that topic proportions of each class after a linear transformation should be nearby to each other. Parameters of linear transformations are learned by maximizing the conditional likelihood of the response classes. But DiscLDA can't guarantee that topic proportions of different classes after linear transformations are well separated, which is critical for classification.

Applying the max-margin principle to supervise latent topic models could avoid drawbacks of sLDA and DiscLDA. For example, SVR doesn't require document labels to be normally distributed and SVM could help forcing topic proportions of different classes to be well separated by a good margin. The most recent MedLDA [13] is an approach utilizing the max-margin principle. However, MedLDA uses the lower

bounds of expected SVM and SVR loss functions to supervise latent topic models. It extremely simplifies inference algorithms, but it is problematic as we discuss in Section 3. Our approach MMpLSA also recruits the max-margin principle to supervise latent topic models. But different to MedLDA, we propose tight variational upper bounds of expected loss functions. Based on upper bounds, we develop a stepwise convex algorithm for optimization, totally different to EM algorithms used by those existing approaches.

6.2 Text Classification

To be able to compare MMpLSA with DiscLDA and MedLDA, we also evaluated MMpLSA on the *20 Newsgroups* dataset containing postings to Usenet newsgroups. As DiscLDA and MedLDA, we formulated the same classification problem for distinguishing postings from two newsgroups: *alt.atheism* and *talk.religion.misc*, a hard task due to the content similarity between them. We also used the training/testing split provided in the *20 Newsgroups* dataset to make possible a fair comparison among them.

To obtain a baseline from two-step approaches, we first fit all the data to pLSA model, and then used empirical topic proportions as features to train a linear SVM for prediction. This baseline is denoted as pLSA+SVM for the rest of section.

For both pLSA+SVM and MMpLSA, 30% of training postings were randomly chosen for cross validation. For the number of topics from 2 to 10, we ran the experiment five times and report accuracies in the Fig. 1. We can observe that MMpLSA performs much better than unsupervised pLSA+SVM. In other words, supervising the latent topic model can discover discriminative topics for better classification.

We further compared MMpLSA with MedLDA and DiscLDA. Lacoste-Julien et al. [8] reported that DiscLDA achieves best accuracy 83.0% at 60 topics. Zhu et al. [13] didn't report the accuracy of MedLDA, but reported the relative improvement ratio of MedLDA against a two-step approach. The best relative improvement ratio is around 0.2, achieved at 20 topics. We show results of comparison between MMpLSA and them in the Fig. 2.

Fig. 2 (Left) reports both the accuracy mean of five runs of MMpLSA and the accuracy of the run with best cross validation against the best accuracy of DiscLDA. We can see that when the number of topics is small, MMpLSA is noticeably better than DiscLDA. MMpLSA achieved the best accuracy 84.7% with 3 topics, a 2% relative accuracy improvement and a 95% relative number of topics reduction. Therefore, compared with DiscLDA, the max-margin principle used by MMpLSA helps in discovering much fewer topics but with more discriminative information. However, when the number of topics increased, the performance of MMpLSA downgraded. The possible reason is that discriminative information is limited and using more than necessary topics to describe it could cause over-fitting.

Fig. 2 (Right) illustrates relative improvement ratio of accuracy mean of MMpLSA against pLSA+SVM v.s. best relative improvement ratio of MedLDA achieved at 20 topics. MMpLSA is better than MedLDA in all cases. It suggests that MMpLSA has advantages of learning more discriminative topics by using the upper bound of expected classification loss in optimization not the lower bound as MedLDA.

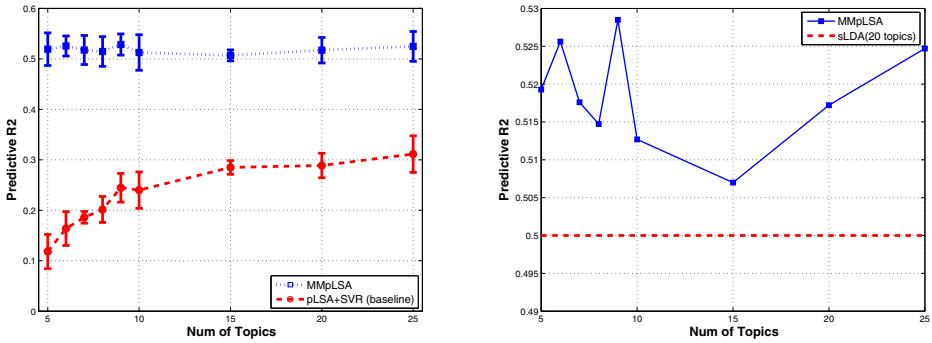


Fig. 3. Experimental results of text regression: (Left) Mean and variance of Predictive R^2 's from an 5-fold experiment of the proposed approach MMpLSA and the two-step approach pLSA+SVR; (Right) Predictive R^2 mean of MMpLSA at different numbers of topics v.s. the best result 0.50 of state of art approach sLDA achieved at 20 topics. MMpLSA achieved the best pR^2 0.5285 with 9 topics, a 5.7% relative pR^2 improvement and a 55% relative number of topics reduction.

6.3 Text Regression

To compare MMpLSA with sLDA and MedLDA on regression, we evaluated MMpLSA on the public available movie review dataset [9], in which each review is paired with a rating within $[0, 1]$. The regression task is to predict the rating of a movie review.

To obtain a baseline from two-step approaches, we first fit training reviews to pLSA model, and then used empirical topic proportions as features to train a linear SVR. We denote this baseline as pLSA+SVR.

Following sLDA and MedLDA, we also ran an 5-fold experiment on the same dataset to evaluate pLSA+SVR and MMpLSA, and assessed the quality of predictions by Predictive R^2 (pR^2) as sLDA and MedLDA. In this 5-fold experiment, when one fold was for test, the rest were for training with 25% of reviews randomly chosen for tuning parameters.

Fig. 3 (Left) shows the results. We can see that the supervised MMpLSA can get much better results than the unsupervised two-step approach pLSA+SVR. Moreover, the performance of MMpLSA is consistent for numbers of topics ranging from 5 to 25. It suggests that MMpLSA can discover most discriminative information with few topics and simply increasing number of topics won't improve performance.

We further compared MMpLSA with sLDA and MedLDA. Zhu et al. [13] showed that sLDA and MedLDA have similar performance and MedLDA is only better than sLDA when the number of topics is small. For the 5-fold experiment, the best pR^2 mean was 0.50, achieved by sLDA with 20 topics. Fig. 3 (Right) compares the pR^2 mean of MMpLSA to this best result in the literature. MMpLSA is noticeably better than this best result for all numbers of topics. MMpLSA achieved the best pR^2 mean 0.5285 with 9 topics, a 5.7% relative pR^2 improvement and a 55% relative number of topics reduction.

The experimental result shows again that applying the max-margin principle to supervise latent topic models helps in discovering a much compacter semantic representation with more discriminative information for prediction than state of art approaches.

7 Conclusions and Future Work

We have proposed MMpLSA that applies the max-margin principle to supervise latent topic models for both classification and regression. MMpLSA integrates learning latent topic representations and training a max-margin predictor into one single learning objective. This integration generates topics describing discriminative themes in the corpus so that topic representations of documents are more suitable for prediction. Due to the optimization hardness of single learning objectives, we proposed tight variational upper bounds for them and developed step-wise convex procedures for optimizing those upper bounds. We studied the predictive power of MMpLSA on movie review and 20 News-groups data sets, and found that MMpLSA performed noticeably better in prediction with significantly fewer topics than state of art models. These results illustrate the benefits of the max-margin supervised latent topic model when dimension reduction and prediction are the ultimate goals. However, discriminative information in documents is always limited. MMpLSA could possibly over-fit documents if it is asked to discover more discriminative topics than real discriminative topics existing in documents. Therefore, one of future work could be introducing priors to topics in the MMpLSA for alleviating possible over-fitting.

References

1. Blei, D.M., McAuliffe, J.D.: Supervised topic models. In: NIPS, pp. 121–128 (2007)
2. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. *Journal of Machine Learning Research* 3, 993–1022 (2003)
3. Bosch, A., Zisserman, A., Munoz, X.: Scene classification via pls. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3954, pp. 517–530. Springer, Heidelberg (2006)
4. Burges, C.J.C.: A tutorial on support vector machine for pattern recognition. *Data Mining and Knowledge Discovery* 2, 121–167 (1998)
5. Hofmann, T.: Probabilistic latent semantic indexing. In: Proceedings of SIGIR, pp. 491–501 (1999)
6. Jaakkola, T., Jordan, M.: A variational approach to bayesian logistic regression models and their extensions. In: Proceedings of the 1997 Conference on Artificial Intelligence and Statistics (1997)
7. Klie, S.: An application of latent topic document analysis to large-scale proteomics databases. In: German Bioinformatics Conference (2007)
8. Lacoste-Julien, S., Sha, F., Jordan, M.I.: Disclda: Discriminative learning for dimensionality reduction and classification. In: NIPS, pp. 897–904 (2008)
9. Pang, B., Lee, L.: Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In: ACL (2005)
10. Smola, A., Scholkopf, B.: A tutorial on support vector regression. *Statistics and Computing*, 199–222 (2003)
11. Xu, W.: Supervising latent topic model for maximum-margin text classification and regression. CMU Technical Report (2009)
12. Zhang, T., Oles, F.: Text categorization based on regularized linear classification methods. *Information Retrieval*, 5–31 (2001)
13. Zhu, J., Ahmed, A., Xing, E.P.: Medlda: Maximum margin supervised topic models for regression and classification. In: ICML, pp. 1257–1264 (2009)

Resource-Bounded Information Extraction: Acquiring Missing Feature Values on Demand

Pallika Kanani, Andrew McCallum, and Shaohan Hu

¹ University of Massachusetts, Amherst USA
{pallika,mccallum}@cs.umass.edu

² Dartmouth College USA
shaohan.hu@dartmouth.edu

Abstract. We present a general framework for the task of extracting specific information “on demand” from a large corpus such as the Web under resource-constraints. Given a database with missing or uncertain information, the proposed system automatically formulates queries, issues them to a search interface, selects a subset of the documents, extracts the required information from them, and fills the missing values in the original database. We also exploit inherent dependency within the data to obtain useful information with fewer computational resources. We build such a system in the citation database domain that extracts the missing publication years using limited resources from the Web. We discuss a probabilistic approach for this task and present first results. The main contribution of this paper is to propose a general, comprehensive architecture for designing a system adaptable to different domains.

1 Introduction

The goal of traditional information extraction is to accurately extract as many fields or records as possible from a collection of unstructured or semi-structured text documents. In many scenarios, however, we already have a partial database and we need only fill in its holes. This paper proposes methods for finding such information in a large collection of external documents, and doing so efficiently with limited computational resources. For instance, this small piece of information may be a missing record, or a missing field in a database that would be acquired by searching a very large collection of documents, such as the Web. Using traditional information extraction models for this task is wasteful, and in most cases computationally intractable. A more feasible approach for obtaining the required information is to automatically issue appropriate queries to the external source, select a subset of the retrieved documents for processing and then extract the specified field in a focussed and efficient manner. We can further enhance our system’s efficiency by exploiting the inherent relational nature of the database. We call this process of searching and extracting for specific pieces of information, on demand, *Resource-bounded Information Extraction* (RBIE). In this paper, we present the design of a general framework for Resource-bounded Information Extraction, discuss various design choices involved and present experimental results.

1.1 Example

Consider a database of scientific publication citations, such as Rexa, Citeseer or Google Scholar. The database is created by crawling the web, downloading papers, extracting citations from the bibliographies and then processing them by tagging and normalizing. In addition, the information from the paper header is also extracted. In order to make these citations and papers useful to the users, it is important to have the year of publication information available. Even after integrating the citation information with other publicly available databases, such as DBLP, a large fraction of the papers do not have a year of publication associated with them. This is because, often, the headers or the full text of the papers do not contain the date and venue of publication (especially for preprints available on the web). Approximately one third of the papers in Rexa are missing the year of publication field. Our goal is to fill in the missing years by extracting them from the web.

We chose this particular example task, since it demonstrates the relational aspect of RBIE. Along with extracting headers, the bibliographic information is often extracted, creating a citation network. This network information can be further exploited by noting that in almost all cases, a paper is published before (or the same year) as other papers that cite it. Using these temporal constraints, we obtain 87.7% of the original F1 by using only 13.2% of computational resources such as queries and documents.

1.2 Motivation

The knowledge discovery and data mining community has long struggled with the problem of missing information. Most real-world databases come with holes in the form of missing records or missing feature values. In some cases, the values exist, but there is considerable uncertainty about their correctness. Incompleteness in the database provides incomplete responses to user queries, as well as leads to less accurate data mining models and decision support systems. In order to make the best use of the existing information, it is desirable to acquire missing information from an external source in an efficient manner. The external source can be another database that is purchased, or a large collection of free documents, such as the web. In the latter case, we may run information extraction to obtain the missing values in our database. However, the traditional models of information extraction can not be directly applied in this “on demand” setting.

Note that, in the setting described above, we are often not interested in obtaining the complete records in the database, but just filling in the missing values. Also, the corpus of documents, such as the web, is extremely large. Moreover, in most real scenarios, we must work under pre-specified resource constraints. The resource constraints may be computational, such as processors, network bandwidth, or related to time and money. Any method that aims to extract required information in the described setting must be designed to work under the given resource constraints.

Many of these databases are relational in nature, e.g. obtaining the value of one field may provide useful information about the remaining fields. Similarly, if the records are part of a network structure with uncertain or missing values, as in the case of our example task, then information obtained for one node can reduce uncertainty in the entire network. We show that exploiting these kinds of dependencies can significantly reduce the amount of resources required to complete the task.

In this paper, we propose a general framework for resource-bounded information extraction, along with the design of a prototype system used to address the task of finding missing years of publication in citation records. We also present first results on this task.

2 Related Work

Resource-bounded Information Extraction encompasses several different types of problems. It deals with extracting information from a large corpus, such as the web; it actively acquires this information under resource constraints; and it exploits the interdependency within the data for best performance. Here we discuss various related tasks and how RBIE is uniquely positioned between them.

2.1 Traditional Information Extraction

In the traditional information extraction settings, we are given a database schema, and a set of unstructured or semi-structured documents. The goal is to automatically extract records from these documents, and fill in the values in the given database. These databases are then used for search, decision support and data mining. In recent years, there has been much work in developing sophisticated methods for performing information extraction over a closed collection of documents, e.g. [3]. Several approaches have been proposed for different phases of information extraction task, such as segmentation, classification, association and coreference. Most of these proposed approaches make extensive use of statistical machine learning algorithms, which have improved significantly over the years. However, only some of these methods remain computationally tractable as the size of the document corpus grows. In fact, very few systems are designed to scale over a corpus as large as, say, the Web [2,15].

2.2 Information Extraction from the Web

There are some large scale systems that extract information from the web. Among these are KnowItAll [2], InfoSleuth [11] and Kylin [14]. The goal of the KnowItAll system is a related, but different task called, “Open Information Extraction”. In Open IE, the relations of interest are not known in advance, and the emphasis is on discovering new relations and new records through extensive web access. In contrast, in our task, what we are looking for is very specific and the corresponding schema is known. The emphasis is mostly on filling the missing fields in known records, using resource-bounded web querying. Hence,

KnowItAll and RBIE frameworks have very different application domains. InfoSleuth focuses on gathering information from given sources, and Kylin focuses only on Wikipedia articles. These systems also do not aim to exploit the inherent dependency within the database for maximum utilization of resources.

The Information Retrieval community is rich with work in document relevance (TREC). However, traditional information retrieval solutions can not directly be used, since we first need to automate the query formulation for our task. Also, most search engine APIs return full documents or text snippets, rather than specific feature values.

A family of methods closely related to RBIE, is question answering systems [8]. These systems do retrieve a subset of relevant documents from the web, along with extracting a specific piece of information. However, they target a single piece of information requested by the user, whereas we target multiple, interdependent fields of a relational database. They formulate queries by interpreting a natural language question, whereas we formulate and rank them based on the utility of information within the database. They do not address the problem of selecting and prioritizing instances or a subset of fields to query. This is why, even though some of the components in our system may appear similar to that of QA systems, their functionalities differ. The semantic web community has been working on similar problems, but their focus is not targeted information extraction.

2.3 Active Information Acquisition

Learning and acquiring information under resource constraints has been studied in various forms. Consider these different scenarios at training time: *active learning* selects the best instances to label from a set of unlabeled instances; *active feature acquisition* [10] explores the problem of learning models from incomplete instances by acquiring additional features; *budgeted learning* [9] identifies the best set of acquisitions, given a fixed cost for acquisitions. More recent work [12] deals with learning models using noisy labels. At test time, the two common scenarios are selecting a subset of features to acquire [13,17], and selecting the subset of instances for which to acquire features [6,5].

The interdependency within the data set is often conveniently modeled using graphs, but it poses interesting questions about selection of instances to query and propagating uncertainty through the graph [4]. In [5], the test instances are not independent of each other, and the impact of acquisition in the context of graph partitioning is studied. The general RBIE framework described in this paper aims to leverage these methods for both train and test time for optimization of query and instance selection.

In summary, RBIE requires a comprehensive architecture for efficiently integrating multiple functionalities, such as instance and query selection, automatic query formulation, and targeted information extraction by exploiting inherent data dependency under limited resources. This leads us to the new framework presented in this paper.

3 A General Framework for Resource-Bounded Information Extraction

As described in the previous section, we need a new framework for performing information extraction to automatically acquire specific pieces of information from a very large corpus of unstructured documents. Fig. 1 shows a top-level architecture of our proposed framework. In this section, we discuss the general ideas for designing a resource-bounded information extraction system. Each of these modules may be adapted to suit the needs of a specific application, as we shall see for our example task.

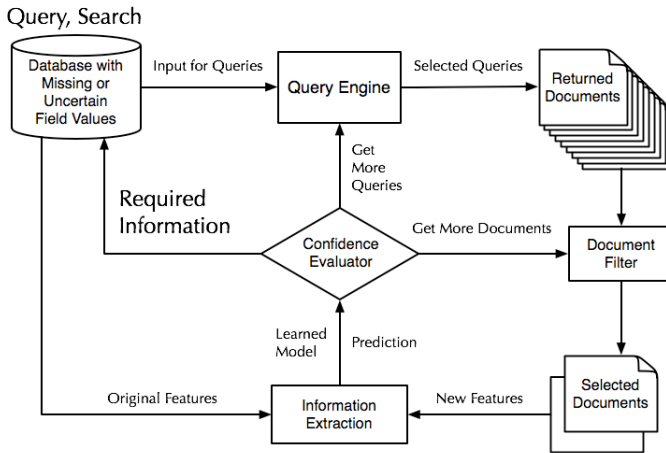


Fig. 1. General Framework for Resource-bounded Information Extraction

3.1 Overview of the Architecture

We start with a database containing missing values. In general, the missing information can either be a complete record, or values of a subset of the features for all records, or a subset of the records. We may also have uncertainty over the existing feature values that can be reduced by integrating external information. We assume that the external corpus provides a search interface that can be accessed automatically, such as a search engine API.

The information already available in the database is used as an input to the *Query Engine*. The basic function of the query engine is to automatically formulate queries, prioritize them optimally, and issue them to a search interface. The documents returned by the search interface are then passed on to the *Document Filter*. Document Filter removes documents that are not relevant to the original database and ranks the remaining documents according to the usefulness of each document in extracting the required information.

A machine learning based information extraction system extracts relevant features from the documents obtained from the Document Filter, and combines

them with the features obtained from the original database. Hence, information from the original database and the external source is now merged, to build a new model that predicts the values of missing fields. In general, we may have resource constraints at both training and test times. In the training phase, the learned model is passed to the *Confidence Evaluation System*, which evaluates the effectiveness of the model learned so far and recommends obtaining more documents through Document Filter, or issuing more queries through the Query Engine in order to improve the model. In the test phase, the prediction made by the learned model is tested by the Confidence Evaluation System. If the model's confidence in the predicted value crosses a threshold, then it is used to fill (or to replace a less certain value) in the original database. Otherwise, the Confidence Evaluation System requests a new document or a new query to improve the current prediction. This loop is continued until either all the required information is satisfactorily obtained, or we run out of a required resource. Additionally, feedback loops can be designed to help improve performance of Query Engine and Document Filter.

This gives a general overview of the proposed architecture. We now turn to a more detailed description for each module, along with the many design choices involved while designing a system for our specific task.

3.2 Task Example: Finding Paper's Year of Publication

We present a concrete resource-bounded information extraction task and a probabilistic approach to instantiate the framework described above: We are given a set of citations with fields, such as, paper title, author names, contact information available, but missing year of publication. The goal is to search the web and extract this information from web documents to fill in the missing year values. We evaluate the performance of our system by measuring the precision, recall and F1 values at different confidence levels. The following sections describe the architecture of our prototype system, along with possible future extensions.

3.3 Query Engine

The basic function of query engine is to automatically formulate queries, prioritize them optimally, and issue them to a search interface. There are three modules of query engine. The available resources may allow us to acquire the values for only a subset of the fields, for a subset of the records. Input selection module decides which feature values should be acquired from the external source to optimize the overall utility of the database. The query formulation module combines input values selected from the database with some domain knowledge, and automatically formulates queries. For instance, a subset of the available fields in the record, combined with a few keywords provided by the user, can form useful queries. Out of these queries, some queries are more successful than others in obtaining the required information. Query ranking module ranks the queries in an optimal order, requiring fewer queries to obtain the missing values. In the future, we would like to explore sophisticated query ranking methods, based on the feedback from other components of the system.

In our system, we use existing fields of the citation, such as paper title and names of author, and combine them with keywords such as “cv”, “publication list”, etc. to formulate the queries. We experiment with the order in which we select citations to query. In one method, the nodes with most incoming and outgoing citation links are queried first. We issue these queries to a search API and the top n hits (where n depends on the available resources) are obtained.

3.4 Document Filter

The primary function of the document filter is to remove irrelevant documents and prioritize the remaining documents for processing. Following are the two main components of the Document Filter. Even though queries are formed using the fields in the database, some documents may be irrelevant. This may be due to the ambiguities in the data (e.g. person name coreference), or simply imperfections in retrieval engine. Initial filter removes documents which are irrelevant to the original database. The remaining documents are then ranked by document ranker, based on their relevance to the original database. Remember that the relevance used by the search interface is with respect to the queries, which may not necessarily be the same as the relevance with respect to the original database. In the future, we would like to learn a ranking model, based on the feedback from the information extraction module (via *Confidence Evaluation System*) about how useful the document was in making the actual prediction.

In our system, many of the returned documents are not relevant to the original citation record. For example, a query with an author name and keyword “resume” may return resumes of different people sharing a name with the paper author. Hence, even though these documents are relevant to an otherwise useful query, they are irrelevant to the original citation. Sometimes, the returned document does not contain any year information. The document filter recognizes these cases by looking for year information and soft matching the title with body of the document.

3.5 Information Extraction

The design of this module differs from traditional information extraction, posing interesting challenges. We need a good integration scheme to merge features from the original database with the features obtained from the external source. As new information (documents) arrives, the parameters of the model need to be updated incrementally (at train time), and the confidence in the prediction made by the system must be updated efficiently (at test time).

Probabilistic Prediction Model. In our task, the field with missing values can take one of a finite number of possible values (i.e. a given range of years). Hence, we can view this extraction task as a multi-class classification problem. Features from the original citation and web documents are combined to make the prediction using a maximum entropy classifier.

Let c_i be a citation ($i = 1, \dots, n$), q_{ij} be a query formed using input from citation c_i and d_{ijk} be a document obtained as a result of q_{ij} . Assuming that we

use all the queries, we drop the index j . Let y_i be a random variable that assigns a label to the citation c_i . We also define a variable y_{ik} to assign a label to the document d_{ik} . If Y is the set of all years in the given range, then $y_i, y_{ik} \in Y$. For each c_i , we define a set of m feature functions $f_m(c_i, y_i)$. For each d_{ik} , we define a set of l feature functions $f_l(c_i, d_{ik}, y_{ik})$. For our model, we assume that $f_m(c_i, y_i)$ is empty. This is because the information from the citation by itself is not useful in predicting the year of publication. In the future, we would like to design a more general model that takes these features into account. We can now construct a model given by

$$P(y_{ik}|c_i, d_{ik}) = \frac{1}{Z_d} \sum_l \exp(\lambda_l f_l(c_i, d_{ik}, y_{ik})), \quad (1)$$

where $Z_d = \sum_y \exp(\lambda_l f_l(c_i, d_{ik}, y_{ik}))$

Combining Evidence in Feature Space vs. Output Space. The above model outputs y_{ik} instead of the required y_i . We have two options to model what we want. We can either merge all the features $f_l(c_i, d_{ik}, y_{ik})$ from d_{ik} 's to form a single feature function. This is equivalent to combining all the evidence for a single citation in the feature space. Alternatively, we can combine the evidence from different d_{ik} 's in the output space. Following are two possible schemes for combining the evidence in the output space. In the first scheme, we take a *majority vote*, i.e., the class with the highest number of y_{ik} is predicted as the winning class and assigned to y_i . In the second scheme, *highest confidence* scheme, we take the most confident vote, i.e., $y_i = \operatorname{argmax}_{y_{ik}} P(y_{ik}|c_i, d_{ik})$

3.6 Uncertainty Propagation in Citation Graph

The inherent dependency within the given data set can be exploited for better resource utilization. In our case, the citation link structure can be used for inferring temporal constraints. For example, if paper A cites paper B, then assuming that papers from future can't be cited, we infer that B must have been published in the same or earlier year than A. Initially, we have no information about the publication year for a citation. As information from the web arrives, this uncertainty is reduced. If we propagate this reduction in uncertainty (or belief) for one of the nodes through the entire graph, we may need fewer documents (or fewer queries) to predict the publication year of the remaining nodes. Selecting the citations to query in an effective order may further improve efficiency.

Notation. Let $c \in C$ be the citation which is currently being queried. Let $a \rightarrow b$ denote that citation a cites citation b . Let $C_B = \{c_b|c_b \rightarrow c\}$ and $C_A = \{c_a|c \rightarrow c_a\}$. Let X be the random variable that represents year of publication of c ; $P_c(X = x)$ be the probability that it takes one of finite values in the given range, and $P'(X = x)$ be the posterior probability from the *Document Classifier*.

Propagation Methods. The method *Best Index* passes the uncertainty message to the neighbors of c as follows:

$$\forall c_b \in C_B P_{c_b}(X = x) = P(X = x|x \geq y) \quad (2)$$

$$\forall c_a \in C_A P_{c_a}(X = x) = P(X = x|x < y) \quad (3)$$

Where $y = \operatorname{argmax}_y P'_c(X = y)$. $P(X = x|x \geq y)$ and $P(X = x|x < y)$ are given by one of the update methods described below. The method *Weighted Average* takes a weighted average over all possible y 's:

$$\forall c_b \in C_B P_{c_b}(X = x) = P'_c(X = y) \sum_y P(X = x|x \geq y) \quad (4)$$

$$\forall c_a \in C_A P_{c_a}(X = x) = P'_c(X = y) \sum_y P(X = x|x < y) \quad (5)$$

Update Methods. If we know that the given paper was published after a certain year, then we can set the probability mass from before the corresponding index to zero and redistribute it to the years after the index. We only show update in one direction here for brevity. The first update method, *Uniform Update*, simply redistributes the probability mass, $P(x \geq y)$ uniformly to the remaining years. The second update method, *Scale Update*, uses conditional probability.

$$P(X = x|x \geq y) = 0, x < y \quad (6)$$

$$= P(X = x) + \frac{1}{P(x \geq y)}, x \geq y \quad (7)$$

$$P(X = x|x \geq y) = 0, x < y \quad (8)$$

$$= \frac{P(X = x)}{P(x \geq y)}, x \geq y \quad (9)$$

Combination Methods. Along with passing a message to its neighbors, the node updates itself by combining information from the *Document Classifier* and the graph structure.

$$P_c(X = x) = P'_c(X = y) \sum_y P(X = x|x = y) \quad (10)$$

The following options can be used for computing $P_c(X = x)$. *Basic*, $P(X = x|x = y)$ *Product* $P_c(X = x) * P'_c(X = x)$ and *Sum* $P_c(X = x) + P'_c(X = x)$

3.7 Confidence Evaluation System

At train time, after adding each new training document, the Confidence Evaluation System can measure the ‘goodness’ of the model by evaluating it on a validation set. At test time, confidence in the prediction improves as more information is obtained. It sets a threshold on the confidence, to either return the required information to the database, or to request more information from external source. It also makes the choice between obtaining a new document or

to issue a new query at each iteration, by taking into account the cost and utility factors. Finally, it keeps track of the effectiveness of queries and documents in making a correct prediction. This information is useful for learning better ranking models for Query Engine and Document Filter.

In our system, we train our model using all available resources, and focus on evaluating test time confidence. For merging evidence in the output space, we employ two schemes. In *max votes*, we make a prediction if the percentage of documents in the winning class crosses a threshold. In *highest confidence*, we make a prediction if $P(y_{ik}|c_i, d_{ik})$ value of the document with the highest P in the winning class passes a threshold. These schemes help determine if we have completed the task satisfactorily. For combining evidence in feature space, we use the *Entropy Method*, in which we compute the value $H = -\sum_i p_i \log p_i$ of the current distribution, and compare it against the confidence threshold.

4 Experimental Description and Results

4.1 Dataset and Setup

Our data set consists of five citation graphs (462 citations), with years of publication ranging from 1989 to 2008. The sampling process is parameterized by size of the network (20-100 citations per graph) and density (min in-degree = 3 and min out-degree = 6). We use five-fold cross validation on these data sets for all our experiments. We use the Mallet infrastructure for training and testing, and the Google search API to issue queries. The queries formed using the information from input citations include the raw title, title in quotes, and author names combined with keywords like “publication list”, “resume”, “cv”, “year” and “year of publication”. We issue queries in a random order, and obtain top 10 hits from google. We use around 7K queries and obtain around 15K documents after filtering. The documents are tokenized and tokens are tagged to be possible years using a regular expression. The document filter discards a document if there is no year information found on the webpage. It also uses a soft match between the title and all n-grams in the body of the page, where n equals the title length. The selected documents are passed on in a random order to the MaxEnt model, which uses the following features for classification: Occurrence of a year on the webpage; the number of unique years on the webpage; years on the webpage found in any particular order; the years that immediately follow or precede the title matches; the distance between a ‘surrounding’ year and its corresponding title match and occurrence of the same ‘following’ and ‘preceding’ year for a title match.

4.2 Results and Discussion

We first run our RBIE system without exploiting the citation network information. We first present the results for combining evidence in the feature space. We measure Precision, Recall and F1 based on using a confidence threshold, where F1 is the harmonic mean of precision and recall. As seen in table 1, as we increase

Table 1. Baseline results

| Entropy Threshold | Precision | Recall | F1 | #Queries | #Docs |
|-------------------|-----------|--------|--------|----------|-------|
| 0.1 | 0.9357 | 0.7358 | 0.8204 | 4497 | 9564 |
| 0.3 | 0.9183 | 0.8220 | 0.8666 | 3752 | 8010 |
| 0.5 | 0.9013 | 0.8718 | 0.8854 | 3309 | 7158 |
| 0.7 | 0.8809 | 0.9041 | 0.8909 | 2987 | 6535 |
| 0.9 | 0.8625 | 0.9171 | 0.8871 | 2768 | 6088 |

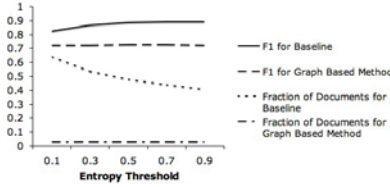


Fig. 2. The change in F1 v.s. the change in use of resources

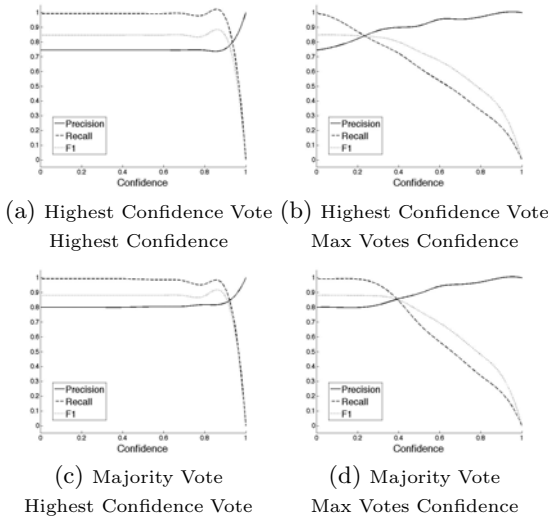


Fig. 3. Different combinations of voting and confidence evaluation schemes

the entropy threshold, precision drops, as expected. F1 peaks at threshold 0.7. Note that the number of documents is proportional to the number of queries, because in our experiments, we stop obtaining more documents or issuing queries when the threshold is reached.

Next, we present the results of exploiting citation network information for better resource utilization. Fig. 2 shows F1 as well as the fraction of the total documents used for the baseline method, and for one of the graph based method (*Weighted Avg* propagation, *Scaling* update, and *Basic* combination). The F1

Table 2. Comparison of Uncertainty Propagation Methods

| Update | Combination | F1 for Best Index | F1 for Weighted Avg |
|---------|-------------|-------------------|---------------------|
| Uniform | Basic | 0.7192 | 0.7249 |
| Uniform | Sum | 0.7273 | 0.5827 |
| Uniform | Product | 0.6475 | 0.3460 |
| Scaling | Basic | 0.7249 | 0.7249 |
| Scaling | Sum | 0.6875 | 0.5365 |
| Scaling | Product | 0.6295 | 0.4306 |

values are smaller compared to the baseline because we use far fewer resources, and the uncertainty propagation methods are not perfect. Using this method, we are able to achieve 87.7% of the baseline F1, by using only 13.2% of the documents. This demonstrates the effectiveness of exploiting relational nature of the data. Table 2 shows the results of different uncertainty propagation methods at entropy threshold 0.7.

We also experiment with combining evidence in the output space using the two schemes described in section 3.5, and the confidence evaluation schemes described in section 3.7. Fig. 3 shows the four precision-recall curves. We see that for *High Confidence Confidence* evaluation scheme (fig. 3(a),(c)), we obtain high values of precision and recall for reasonable values of confidence. That is, in the confidence region below 0.9, we obtain a good F1 value. Especially, the *Majority Vote - High Confidence* scheme (fig. 3(c)) performs exceptionally well in making predictions. However, in the confidence region between 0.9 to 1.0, the *Max Vote* scheme (fig. 3(b),(d)) gives a better degradation performance.

5 Conclusion and Future Work

We propose a new framework for targeted information extraction under resource constraints to fill missing values in a database. We present first results on an example task of extracting missing year of publication of scientific papers, along with exploiting the underlying citation network for better resource utilization. The overall framework is flexible, and can be applied to a variety of problem domains and individual system components can be adapted to the task. The specific methods recommended here can also be generalized in many different relational domains, especially when the dataset has an underlying network structure. In future, we would like to explore more sophisticated uncertainty propagation methods, such as belief-propagation. We would also like to develop individual components like *Query Engine* and *Document Filter*, by using good ranking procedures. Finally, it would be interesting to see how these methods extend to extracting multiple interdependent fields.

Acknowledgments

We thank Andrew McGregor for useful discussions. This work was supported in part by the Center for Intelligent Information Retrieval and in part by The CIA,

the NSA and NSF under NSF grant #IIS-0326249. 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. Bilgic, M., Getoor, L.: Voila: Efficient feature-value acquisition for classification. In: AAAI, pp. 1225–1230. AAAI Press, Menlo Park (2007)
2. Etzioni, O., Cafarella, M., Downey, D., Kok, S., Popescu, A., Shaked, T., Soderland, S., Weld, D., Yates, A.: Web-scale information extraction in knowitall. In: WWW 2004, May 2004, ACM, New York (2004)
3. Grishman, R., Sundheim, B.: Message understanding conference-6: a brief history. In: Proceedings of the 16th conference on Computational linguistics (1996)
4. Kanani, P., McCallum, A.: Resource-bounded information gathering for correlation clustering. In: Bshouty, N.H., Gentile, C. (eds.) COLT. LNCS (LNAI), vol. 4539, pp. 625–627. Springer, Heidelberg (2007)
5. Kanani, P., McCallum, A., Pal, C.: Improving author coreference by resource-bounded information gathering from the web. In: Proceedings of IJCAI (2007)
6. Kanani, P., Melville, P.: Prediction-time active feature-value acquisition for customer targeting. In: Workshop on Cost Sensitive Learning, NIPS 2008 (2008)
7. Krause, A., Guestrin, C.: Near-optimal nonmyopic value of information in graphical models. In: UAI 2005, p. 5 (2005)
8. Lin, J., Fernandes, A., Katz, B., Marton, G., Tellex, S.: Extracting answers from the web using knowledge annotation and knowledge mining techniques (2002)
9. Lizotte, D., Madani, O., Greiner, R.: Budgeted learning of naive-Bayes classifiers. In: UAI 2003, Acapulco, Mexico (2003)
10. Melville, P., Saar-Tsechansky, M., Provost, F., Mooney, R.: An expected utility approach to active feature-value acquisition. In: ICDM 2005, pp. 745–748 (2005)
11. Nodine, M.H., Fowler, J., Ksiezzyk, T., Perry, B., Taylor, M., Unruh, A.: Active information gathering in infosleuth. IJCIS 9(1-2), 3–28 (2000)
12. Sheng, V., Provost, F., Ipeirotis, P.G.: Get another label? improving data quality and data mining using multiple, noisy labelers. In: SIGKDD (2008)
13. Sheng, V.S., Ling, C.X.: Feature value acquisition in testing: a sequential batch test algorithm. In: ICML 2006, pp. 809–816. ACM, New York (2006)
14. Wu, F., Hoffmann, R., Weld, D.S.: Information extraction from wikipedia: moving down the long tail. In: 14th ACM SIGKDD, pp. 731–739 (2008)
15. Zhao, S., Betz, J.: Corroborate and learn facts from the web. In: KDD, pp. 995–1003 (2007)

Efficient Deep Web Crawling Using Reinforcement Learning

Lu Jiang, Zhaohui Wu, Qian Feng, Jun Liu, and Qinghua Zheng

MOE KLINNS Lab and SKLMS Lab, Xi'an Jiaotong University

No.28, Xianning West Road, Xi'an 710049, P.R.China

roadjiang@yahoo.com, wzh@stu.xjtu.edu.cn, qfeng@stu.xjtu.edu.cn,

liukeen@mail.xjtu.edu.cn, qhzheng@mail.xjtu.edu.cn

Abstract. Deep web refers to the hidden part of the Web that remains unavailable for standard Web crawlers. To obtain content of Deep Web is challenging and has been acknowledged as a significant gap in the coverage of search engines. To this end, the paper proposes a novel deep web crawling framework based on reinforcement learning, in which the crawler is regarded as an agent and deep web database as the environment. The agent perceives its current state and selects an action (query) to submit to the environment according to Q-value. The framework not only enables crawlers to learn a promising crawling strategy from its own experience, but also allows for utilizing diverse features of query keywords. Experimental results show that the method outperforms the state of art methods in terms of crawling capability and breaks through the assumption of full-text search implied by existing methods.

Keywords: Hidden Web, Deep Web Crawling, Reinforcement Learning.

1 Introduction

Deep web or hidden web refers to World Wide Web content that is not part of the surface Web, which is directly indexed by search engines. Studies [1] show deep web content is particularly important. Not only its size is estimated as hundreds of times larger than the so-called surface Web, but also it provides users with high quality information. However, to obtain such content of deep web is challenging and has been acknowledged as a significant gap in the coverage of search engines [2]. Surfacing is a common solution to provide users deep web content search service¹, in which the crawler pre-computes the submissions for deep web forms and exhaustively indexes the response results off-line as other static HTML pages. The approach enables leveraging the existing search engine infrastructure hence adopted by most of crawlers, such as HiWE (Hidden Web Exposer) [3], Hidden Web crawler [4] and Google's Deep Web crawler [2].

One critical challenge in surfacing approach is how a crawler can automatically generate promising queries so that it can carry out efficient surfacing. The challenge has been studied by several researches such as [2], [4], [5], [6], [7]. In these methods,

¹ We may use crawl and surface interchangeably in the rest of the paper.

candidate query keywords are generated from the obtained records, and then their harvest rates, i.e. the promise to obtain new records, are calculated according to their local statistics, such as DF (Document Frequency) and TF (Term Frequency). The one with the maximum expected harvest rate will be selected for the next query. Their basic idea is similar while the difference is that they choose different strategies to derive the estimated harvest rate of each query candidate.

However, to the best of our knowledge, existing methods suffer from the following three deficiencies. Firstly, the future reward of each query is ignored, which is also known as “myopia problem” [8]. The next query is selected according to the harvest rate defined as the immediate reward at current step. This inherent deficiency makes the existing methods fail to look ahead to future steps thus cannot make a decision of long-term interest. Secondly, the existing methods solely utilize the statistic of acquired data records while ignoring the experience gained from previous queries, which usually results in reducing the efficiency of crawling. For example when a crawler issues an unpromising keyword which brings few or even no response records, as the acquired data record hardly accumulates, the statistic of the data will remain the same. Therefore, it is likely that the crawler will make the same mistakes in its future decisions. Finally, existing methods relied on a critical assumption that full-text search are provided by deep web databases, in which all of the words in every document are indexed. However, this assumption is too strong to hold in the cases of databases providing non full-text search interfaces, in which some words appearing on the pages e.g. noisy words and template words are excluded from the index. The disagreement leads to that the existing estimation techniques for full-text databases, e.g. Zipf’ Law [4], [9] can hardly be applied to non full-text databases. E.g. Fig. 1 illustrates the keywords distribution on AbeBooks (www.abebooks.com), in which x-axis represents document frequency ranks of keywords in the data corpus and y-axis denotes the percent of response records brought by the keywords. As one can see, the Zipf curve fails to simulate the distribution of keyword response.

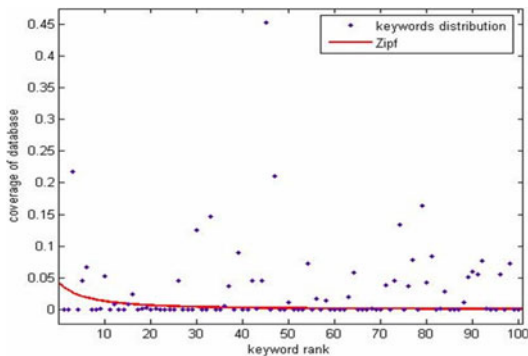


Fig. 1. Keywords distribution on AbeBooks

In this paper, we present a formal framework based on the RL (Reinforcement Learning) [10] for deep web crawling. In the framework, a crawler is regarded as an agent and deep web database as the environment. The agent perceives its current state

and selects an action (query) to submit to the environment according to long-term reward. The environment responds by giving the agent some reward (new records) and changing it into the next state. Each action is encoded as a tuple using its linguistic, statistic and HTML features. The rewards of unexecuted actions are evaluated by their executed neighbors. Because of the learning policy, a crawler can avoid using unpromising queries, as long as some of them have been issued. The experimental results on 5 real world deep web sites show that the reinforcement learning crawling method relaxes the assumption of full-text search and outperforms existing methods. To sum up, the main contributions of our work are:

- 1) We introduce a formal framework for the deep web surfacing problem. To the best of our knowledge, ours is the first work that introduces machine learning approaches to the deep web surfacing problem.
- 2) We formalize the problem in the framework and propose an efficient and applicable surfacing algorithm working well on both full-text and non full-text databases.
- 3) We develop a Q-value approximation algorithm allows for a crawler selecting a query according to the long-term reward, which overcomes the myopia problem to some extent.

The rest of this paper is organized as follows: Section 2 gives a brief introduction of related work. Section 3 presents the formal reinforcement learning framework. Section 4 discusses the crawling algorithm and key issues in it. The experimental results are discussed in Section 5 whereas conclusions and future work are presented in the final section.

2 Related Work

The state-of-the-art deep web crawling approaches generate new keywords by analyzing statistic of current acquired records returned from previous queries. Barbosa L. et al. first introduced the ideas, and presented a query selection method which generated the next query using the most frequent keywords in the acquired records [5]. However, queries with the most frequent keywords in hand do not ensure that more new records are returned from the deep web database. Ntoulas A. et al. proposed a greedy query selection method based on the expected harvest rate [4]. In the method, the one with the maximum expected harvest rate will be selected for the next query. Ping W. et al. modeled each web database as a distinct attribute-value graph and a greedy link-based query selection method was proposed to approximate the optimal solution [8]. Lu J. et al. resolved the problem using set-covering sampling method [7]. Liu J. et al. extended the Ntoulas's method to entire form by introducing a novel concept MEP (Minimum Executable Pattern). In the method, a MEP set is build and then promising keywords are selected by joint harvest rate of a keyword and its pattern. By selecting among multiple MEPs, the crawler achieves better results [6]. Jayant M. et al. improved the keyword selection algorithm by ranking keywords by their TFIDF (Term Frequency Inverse Document Frequency) [2]. Jiang L. et al. present a method to evaluate a keyword by its HTML features besides TF and DF using supervised learning [11].

3 Reinforcement Learning Framework

In this section, we propose a formal framework for the deep web crawling based on RL and formalize the crawling problem under the framework. First of all we give an overview of the RL framework.

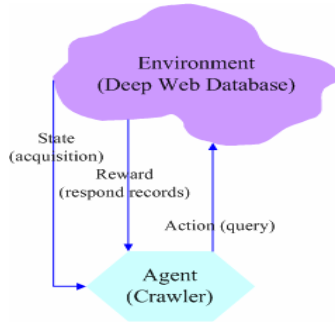


Fig. 2. Overview of the reinforcement learning framework

The relation between a crawler and a deep web database is illustrated in Fig. 2. From the figure, one can conclude that at any given step, an agent (crawler) perceives its state and selects an action (query). The environment responds by giving the agent some (possibly zero) reward (new records) and changing the agent into the successor state. More formally we have

Definition 1. Suppose S and A are two sets of states and actions respectively. A state $s_t \in S$ represents the acquired portion of the deep web database records at the step t . An action $a(k) \in A$ (a for short) denotes a query to the deep web database with the keyword k , which causes a transition from state s_t to some successor state s_{t+1} with the probability $p(s_{t+1}|a, s_t)$.

Definition 2. The process of deep web crawling is defined as a discrete Decision Process (S, A, P) consisting of a set of states S , a set of actions A and transition probabilities distribution P . A crawling process follows a specific issue policy $\pi : S \rightarrow A$, which is a mapping from the set of states to the set of actions.

In the paper, we assume that the decision process is a deterministic process i.e. P is subjected to a uniform distribution. During the process, after execution of an action the agent is responded by giving a collection of data records by the environment. The response record can be defined as:

Definition 3. Suppose D is the collection of all data records residing in deep web database. After execution of action a at state s_t , the response record set $R(s_t, a) \subseteq D$ represents the collection of data records responded by the environment. Likewise the portion of the new records in the response record set retrieved by action a at state s_t is denoted as $R_{NEW}(s_t, a)$ ($R_{NEW}(s_t, a) \subseteq R(s_t, a)$).

Suppose a crawling process follows an issue policy π , the portion of new records in the response records of action a at state s_t can be formulated as

$$R_{new}(s_t, a) = R(s_t, a) \setminus \bigcup_{i=1}^{t-1} R(s_i, \pi(s_i)) \tag{1}$$

Note that the response record set of an action is irrelevant to the state hence $\forall i, j, R(s_i, a) = R(s_j, a)$.

There are two important functions in the process. Transition function $\delta : S \times A \rightarrow S$ denotes the successor state of the given state and action. Reward function $r(s_t, a)$ is the reward received at the transition from state s_t to state s_{t+1} by executing action a , i.e. the portion of new records brought by executing a_t , computed from equation

$$r(s_t, a) = |R_{new}(s_t, a)|/|D| \tag{2}$$

Though in some cases $|D|$ is either unknown or cannot be obtained beforehand, the absence of the value does not influence the calculation of the reward as they are relative values to rank actions in the same baseline.

The transition of actions causes a cost. In the paper, the cost is measured in terms of time consumed, i.e. $cost(s_t, a) = t_a + t_r + |R_{new}(s_t, a)|$. t_a is the cost of issuing an action and t_r is proportional to the average time of handling a response record.

The expectation conditioned on the current state s and the policy π is called state-value function $V^\pi(s)$ of state s , computed from

$$V^\pi(s_t) = \sum_{i=0}^h \gamma^i r(s_{t+i}, \pi(s_{t+i})) \tag{3}$$

in which h is referred as the step length and γ is the discount factor. Among all policies, there must exist an optimal policy, noted π^* defined as $V^{\pi^*}(s) \geq V^\pi(s) (\forall s \in S, \forall \pi)$. To simplify notations, we write $V^{\pi^*} = V^*$.

Based on the presentations above, the formal definition of deep web crawling problem can be defined as:

Problem. Under the constraint $\sum_{i=0} cost(s_i, a) \leq cost_{MAX}, \forall s_i \in S$ find such policy $\pi^* \equiv \arg \max_{\pi} V^\pi(s_i)$ that maximizes the accumulative reward value. Here $cost_{MAX}$ is the maximum cost constraint.

4 Algorithm

In this section we discuss how to resolve the deep web crawling problem defined in Section 3. There are two crucial factors in solving the problem i.e. the reward of each action r and the reward of an issue policy Q-value. Section 4.1 and 4.2 introduces the methods for the action reward calculation and Q-value approximation respectively. Finally an adaptive algorithm for surfacing deep web is presented in the end of Section 4.2.

4.1 Reward Calculation

Before specifying the method for the action reward calculation, we need the definition of the document frequency.

Definition 4. Suppose on the current state s_t and the issue policy π , the document frequency of action $a(k) \in A$ denoted by $DF(s_t, a(k))$ ($DF(s_t, a)$ for short) is the number of documents containing keyword k in acquired record set $\bigcup_{i=1}^{t-1} R(s_i, \pi(s_i))$.

Note that the document frequency of each action is a known statistic. Since records of $\bigcup_{i=1}^{t-1} R(s_i, \pi(s_i))$ having been retrieved at the step t , the number of documents containing keyword k can be counted up in the acquired record set. Relying on Def. 4, the following theorem can be established.

Theorem 1. At state s_t , the reward of each action a in A can be calculated from

$$r(s_t, a) = \frac{|R(s_t, a)| - DF(s_t, a)}{|D|}. \tag{4}$$

Proof: By incorporating Eq. (1) into Eq. (2) we have

$$r(s_t, a) = |R(s_t, a) \setminus \bigcup_{i=1}^{t-1} R(s_i, \pi(s_i))|/|D|. \tag{5}$$

Eq. (5) can be further rewritten as

$$r(s_t, a) = \frac{|R(s_t, a)| - |R(s_t, a) \cap \bigcup_{i=1}^{t-1} R(s_i, \pi(s_i))|}{|D|}. \tag{6}$$

The intersection part in Eq. (6) denotes the collection of documents containing the keyword of action a in the data set $\bigcup_{i=1}^{t-1} R(s_i, \pi(s_i))$. According to the Def. 4 the value equals to the document frequency of the action, i.e.

$$|R(s_t, a) \cap \bigcup_{i=1}^{t-1} R(s_i, \pi(s_i))| = DF(s_t, a) \tag{7}$$

Consequently the Eq. (4) could be proved by incorporating Eq. (7) into Eq. (6).

The absence of D in Eq. (4) does not affect the final result for the same reason described in Section 3. According to Eq. (4) for an executed action, as response record set $R(s_t, a)$ is acquired, the reward can be calculated. In contrast, the reward calculation for an unexecuted action directly through Eq. (4) is infeasible. Nevertheless, the response record set of an unexecuted action can be estimated by generalizing from those executed. Before proceeding any further, we define the action training and candidate set.

Definition 5. Suppose at state s_t , training set Tr is a set of executed actions, $|Tr|=t$. Similarly, candidate set C is a set of available action candidates for submission in the current state. Each action in either Tr or C is encoded in the same vector space.

Based on Def. 4, for an action a_i in C , its reward can be estimated as:

$$|\tilde{R}(s_t, a_i)| = \sum_{a_j \in Tr} \kappa(a_i, a_j) |R(s_t, a_j)|. \tag{8}$$

in which $\kappa(a_i, a_j)$ is a kernel function used to evaluate the distance between the given two actions. Since the response record set $R(s_x, a)$ of an action is irrelevant to the state s_x , the response record set can be rewritten to the current state s_t i.e. $R(s_x, a) = R(s_t, a)$. Accordingly, the intuition behind the Eq. (8) is to estimate the

reward for an action in C by evaluating those in Tr . As all response record sets of executed action are at the current state, the size of response record set of an action in C can be learnt from those sharing the similar features in Tr . Once the size of response record set of an unexecuted is calculated by Eq. (8), the value can then be applied in Eq. (4) to calculate its reward.

Now the action rewards for both executed and unexecuted actions can be calculated from Eq. (4). In the rest of the subsection, we will discuss how to calculate kernel function $\kappa(a_i, a_j)$ in Eq. (8). Calculating the similarity of actions requires encoding them in a feature space. We incorporate three types of features i.e. linguistic features, statistical features and HTML features to establish the feature space [11].

Linguistic features consist of POS (Part of Speech), length and language of a keyword (action). Length is the number of characters in the keyword. Language represents the language that a keyword falls into. It takes effect in multilingual deep web database.

Statistical features include TF (Term Frequency), DF (Document Frequency) and RIDF (Residual Inverse Document Frequency) of a keyword in the acquired records. The value of RIDF is computed as:

$$RIDF = \log(1 - e^{-TF/|D|}) - \log(DF/|D|) \quad (9)$$

RIDF tends to highlight technical terminology, names, and good keywords and to exhibit nonrandom distributions over documents [12].

The HTML format usually plays an important role in indicating the semantics of the presented data. This brings us to consider the HTML information of keywords. We propose two HTML features tag-attribute and location. Tag-attribute feature encodes HTML tag and attribute information of a keyword, and location represents the depth of the keyword's node in the DOM tree derived from the HTML document. The features may imply the semantic information of a keyword hence is useful in distinguishing unpromising keywords.

For linguistic and HTML features whose values are discrete, the liner kernel is assigned. Considering that value of statistical feature tends to a Gaussian distribution over documents, the Gaussian kernel is adopted to evaluate similarity upon the statistical features, which is formulated as

$$\kappa_s(a_i, a_j) = \exp(-\|a_i - a_j\|^2 / 2\delta^2). \quad (10)$$

The final kernel function is hybrid of these kernels. Suppose λ_l , λ_h and λ_s ($\lambda_l + \lambda_h + \lambda_s = 1$) are weights for linguistic, HTML and statistical kernel respectively, the kernel function to evaluate similarity of two actions is

$$\kappa = \lambda_l \kappa_l + \lambda_h \kappa_h + \lambda_s \kappa_s. \quad (11)$$

In experiments the weight of statistical features usually accounts for a larger part.

4.2 Q-Value Approximation and Surfacing Algorithm

Once the reward of each action is obtained, given the problem definition in Section 3, the agent can find an optimal policy V^* if the V^π of each state can be calculated. The calculation of V^π could be well solved when the agent uses Q-function [13], [14]:

$$V^\pi(s_t) = Q(s_t, \pi(s_t)) = r(s_t, \pi(s_t)) + \gamma \max_\pi [V^\pi(\delta(s_t, \pi(s_t)))]. \quad (12)$$

Here Q-function $Q(s, a)$ represents the reward received immediately upon executing action a from state s , plus the value discounted by γ thereafter. Using Eq. (3), we can rewrite Q-function as

$$Q(s_t, a) = r(s_t, a) + \max[\sum_{i=1}^h \gamma^i \times r(s_{t+i}, \pi(s_{t+i}))]. \quad (13)$$

To simplify the notion, here we let $\gamma = 1$ i.e. the reward for the future steps are regarded as important as those for the present. h is a critical parameter denoting the step length looking ahead to the future reward. If $h = 0$, the future reward is ignored and Q-value equals to the immediate reward i.e. $Q(s_t, a) = r(s_t, a)$. When $h > 1$, Q-value represents the long-term reward. However, as the action reward at state s_{t+1} is unavailable at state s_t , the Q-value has to be approximated. To estimate the Q-value, we make the following assumption: assume at the current state, the action set A will not enlarge in the next $h+1$ steps ($h \ll |A|$). When h is not very large the assumption is reasonable. Under the assumptions, Theorem 2 could be established.

Theorem 2. At state s_t when $h+1$ the Q-value of an action a_i ($a_i, a_j \in C, a_i \neq a_j$) can be estimated as:

$$Q(s_t, a_i) \propto r(s_t, a_i) + \max_j [r(s_t, a_j) + \frac{DF(s_t, a_j)}{|D|} - \frac{|\bigcup_{i=1}^t R(s_i, \pi(s_i))| |R(s_t, a_j)|}{|D|^2}] \quad (14)$$

Proof: To simplify the notion, let $\bigcup_{i=1}^{t-1} R(s_i, \pi(s_i)) = R_{t-}$, $R(s_t, a_i) = R_t$, $R(s_{t+1}, a_j) = R_{t+1}$. First of all, because the action set will not enlarge, the optimal Q-value can be searched in the action set at the current state. According to the Eq. (13), when $h = 1$ the Q-value can be formulated as

$$Q(s_t, a_i) = r(s_t, a_i) + \max_j [r(s_{t+1}, a_j)]. \quad (15)$$

Following the method described in Section 4.1, $r(s_t, a_i)$ can be calculated; whereas $r(s_{t+1}, a_j)$ is unknown at state s_t . Therefore rewrite the Eq. (15) as

$$Q(s_t, a_i) = \max[|(R_t \cup R_{t+1}) \setminus R_{t-}| / |D|]. \quad (16)$$

Because the response records are independent with each others, the capture-mark-recapture [15] method can be applied to estimate the overlaps records:

$$|(R_{t-} \cup R_t) \cap R_{t+1}| / |R_{t+1}| \propto |R_{t-} \cup R_t| / |D|. \quad (17)$$

Further Eq. (17) can be transformed into

$$|R_{t+1} \cap R_t| - |R_{t+1} \cap R_t \cap R_{t-}| \propto |R_t \cup R_{t-}| |R_{t+1}| / |D| - |R_{t+1} \cap R_{t-}|. \quad (18)$$

By incorporating Eq. (18) into Eq. (16) we have

$$Q(s_t, a_i) = 1/|D| \times \max[|R_t \setminus R_{t-}| + |R_{t+1} \setminus R_{t-}| + |R_{t+1} \cap R_{t-}| - |R_t \cup R_{t-}| |R_{t+1}| / |D|] \quad (19)$$

Note that according to the characteristics of response record set in Def. 3 and Eq. (5):

$$|R(s_{t+1}, a_j) \setminus R_{t-}| = |R(s_t, a_j) \setminus R_{t-}| = r(s_t, a_j) \times |D|. \quad (20)$$

Following Eq. (5) and Eq. (20), Eq. (19) can be reformulated as

$$Q(s_t, a_i) = r(s_t, a_i) + \max_j [|R_{t+1} \cap R_{t-}| / |D| + r(s_t, a_j) - |R_t \cup R_{t-}| \times |R_{t+1}| / |D|^2] \quad (21)$$

Then the Theorem 2 can be derived by incorporating Eq. (7) into Eq. (21).

As all the factors in the Eq. (14) are either calculated or can be statistically numerated, the Q-value of each action can be approximated based on the acquired data set. Now, we can approximate Q-value with a given step length by iteratively applying Eq. (14). Due to the lack of space, we cannot present the details here. Note if h goes too big, the assumption may not hold and the future state may diverge from the experienced state rendering the approximation for future reward imprecise.

We develop an adaptive algorithm for deep web surfacing based on the framework, as shown in Algorithm 1. The algorithm takes the current state and last executed action as input and outputs the next optimal action.

Algorithm 1: Adaptive RL surfacing algorithm

Input: s_t, a_m **Output:** $\pi(s_{t+1})$

- 1: calculate the reward of action a_m following Eq.(4);
- 2: for each document $d_i \in R(s_{t-1}, a_m)$
- 3: for each keyword k in d_i do
- 4: if action $a(k) \notin A$ then $A = A \cup a(k)$;
- 5: else then update TF and DF of action $a(k)$;
- 6: end for
- 7: end for
- 8: change the current state to s_{t+1} ;
- 9: $Tr = Tr \cup \{a_m\}$; update candidate set C ; $C = C \setminus \{a_m\}$;
- 10: for each $a_i \in C$ update its reward using Eq. (8) and Eq. (4);
- 11: for each $a_i \in C$ calculate its Q-value using Eq. (14);
- 12: return $\arg \max_a [Q(s_t, a)]$;

Specifically, the surfacing algorithm first calculates the reward of the last executed action and then updates the action set through Step 2 to Step 7, which causes the agent to transit from its state s_t to the successor state s_{t+1} . Then the training and candidate set are updated in accord with the new action set in Step 9. After that the algorithm estimates the reward and Q-value for each action in candidate set in Step 10 and Step 11 respectively. The action that maximizes Q-value will be returned as the next to be executed action.

5 Experiments

To demonstrate the efficiency of our proposed approach for deep web crawling, we execute our algorithm on five real world deep web databases with different scales and domains. The detailed information about these databases is listed in Tab.1. In the case

of AbeBooks, as its large scale, the agent restricted to crawling the “historical fictions” category to accelerate the experiment. Queries to the site are applied to the textbox “keywords”. Regarding Wikicfp, Yahoo movie, which are the typical medium Deep Web databases, we utilize the only generic search textboxes as their query interfaces. As for Baidu Baike and Google Music, the sites are multilingual sites consisting of both English and Chinese. On the sites we select the rewarding textbox “Keyword Tag” and “Singer” as their query interfaces.

To compare our RL method with existing ones, we choose following three methods as baseline methods: (Suppose at state s_t , the agent is to evaluate an action a)

- Random [4], [7], [8]: the reward of an action is assigned to a random float i.e. $Q(s_t, a) = random(1.0)$.
- GF (Generic Frequency) [5], [7], [8]: the reward of an action is evaluated by the generic DF of the action at current state, i.e. $Q(s_t, a) = DF(s_t, a)$.
- Zipf [4], [6], [9]: The size of response record set of each action is estimated by Zipf-Mandelbrot’s Law [16]: $|R(s_t, a)| = \alpha(r + \beta)^{-\gamma}$, where α , β and γ are parameters and r is the DF rank of the action.

All methods above (including RL) share the same candidate set generation policy which selects the top 500 actions with highest RIDF. It is interesting to note that RL is more general compared with the baseline methods. If future reward of an action is ignored i.e. $h=0$ and the reward of an action is determined by a presumed distribution, the RL degenerates to Zipf, i.e. $Q(s_t, a_i) = r(s_t, a_i)$. Further if the acquired portion of an action is ignored too i.e. $R(s_t, a_i) = \Phi$, the RL degenerates to the GF, i.e. $Q(s_t, a_i) = DF(s_t, a_i)$.

Table 1. Experiment web sites and their experimental results

| DB Name | URL | Domain | Harvest | Estimated Database | #Queries |
|--------------|----------------------------|------------|---------|--------------------|----------|
| AbeBooks | www.abebooks.com | Book | 90,224 | 110,000 | 322 |
| Wikicfp | www.wikicfp.com | Conference | 4,125 | 5,200 | 499 |
| Yahoo movie | movies.yahoo.com/mv/search | Movie | 126,710 | 128,000 | 367 |
| Google music | www.google.cn/music/ | Music | 85,378 | 110,000 | 592 |
| Baidu Baike | Baike.baidu.com | Wikipedia | 820,180 | 1,000,000 | 1,950 |

5.1 Effectiveness of RL Method

Our interest is to discover their records as many as possible with affordable cost. To make the results more intelligible, we roughly use *harvest* (Fourth column) i.e. the number of actual retrieved records and *number of queries* (Sixth column) to evaluate the crawling effects. Tab. 1 shows that our method is quite efficient. In the first four cases the agent achieves more than 80% coverage by issuing around 500 hundred queries.

5.2 Performance Comparison with Baseline Method

We performed our method as well as the baseline methods on the experiment sites. The experimental results are displayed in Fig. 3 in which the y-axis denotes the

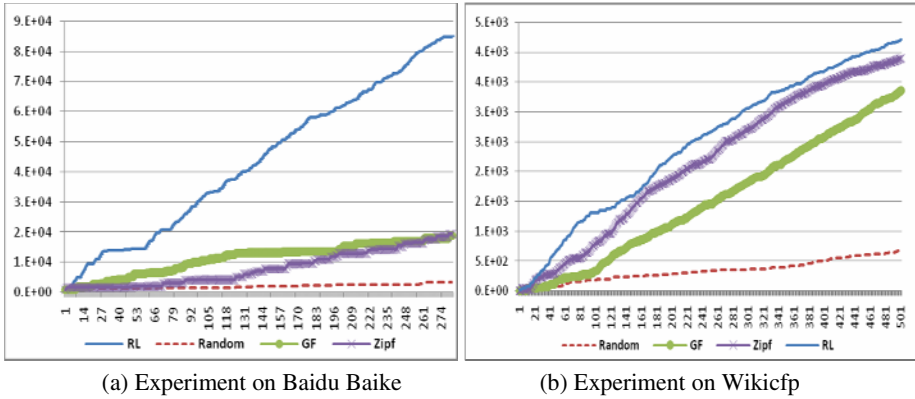


Fig. 3. Performance Comparisons with baseline methods

database coverage, while the x-axis represents the query number (due to the lack of space we only present some results here). In experiments step length was set to 1. As can be seen, the result shows that RL method is more efficient than baseline methods on the experiment websites. We analyzed the queries logs and summarized two reasons accounting for excellence of RL method. Firstly because the RL method selects a keyword according to the long-term rather than immediate reward, it is able to acquire a better awareness of the environment leading to more accurate estimation for succeeding rewards. As we found in experiments the rewarding keywords are issued earlier in RL than other methods. Secondly the keywords issued in RL are more relevant to the pattern selected, e.g. “keywords” on Baidu Baike. This suggests that the agent using RL learns the experience from its previous queries and hence sticks on the keywords matching against more records; whereas the agent using other methods do not make any adjustment when the presumed assumption is not applied.

6 Conclusion and Future Work

In this paper we tackle the problem of deep web surfacing. The paper first presents a formal reinforcement learning framework to study the problem and then introduce an adaptive surfacing algorithm based on the framework and its related methods for reward calculation and Q-value approximation. The framework enables a crawler to learn an optimal crawling strategy from its experienced queries and allows for it making decisions on long-term rewards. Experimental evaluation on 5 real deep web sites suggests that the method is efficient and applicable. It excels the baseline method and works well on both full-text and non full-text databases. In general, it retrieves more than 80% of the total records by issuing a few hundreds of queries.

We are studying the issues of deep web crawling in practical and developing an open source platform for Deep Web crawling: DWIM (Deep Web Intelligent Miner). DWIM is the first open source software in the respect Deep Web crawling and integrates many crawling policies and keywords selection criteria which can be used as an experimental platform for researches and a crawler engine for developers.

Acknowledgement

The research was supported by the National High-Tech R&D Program of China under Grant No.2008AA01Z131, the National Science Foundation of China under Grant Nos.60825202, 60803079, 60633020, the National Key Technologies R&D Program of China under Grant Nos.2008BAH26B02, 2009BAH51B00, the Open Project Program of the Key Laboratory of Complex Systems and Intelligence Science, Institute of Automation, Chinese Academy of Sciences under Grant No. 20080101, Cheung Kong Scholar's Program.

References

1. Lawrence, S., Giles, C.L.: Searching the World Wide Web. *Science* 280, 98–100 (1998)
2. Madhavan, J., Ko, D., Kot, L., Ganapathy, V., Rasmussen, A., Halevy, A.: Google's Deep-Web Crawl. In: Proceedings of VLDB 2008, Auckland, New Zealand, pp. 1241–1252 (2008)
3. Raghavan, S., Garcia-Molina, H.: Crawling the Hidden Web. In: Proceedings of VLDB 2001, Rome, Italy, pp. 129–138 (2001)
4. Ntoulas, A., Zerkos, P., Cho, J.: Downloading Textual Hidden Web Content through Keyword Queries. In: Proceedings of JCDL 2005, Denver, USA, pp. 100–109 (2005)
5. Barbosa, L., Freire, J.: Siphoning Hidden-Web Data through Keyword-Based Interfaces. In: Proceedings of SBBD 2004, Brasilia, Brazil, pp. 309–321 (2004)
6. Liu, J., Wu, Z.H., Jiang, L., Zheng, Q.H., Liu, X.: Crawling Deep Web Content Through Query Forms. In: Proceedings of WEBIST 2009, Lisbon, Portugal, pp. 634–642 (2009)
7. Lu, J., Wang, Y., Liang, J., Chen, J., Liu, J.: An Approach to Deep Web Crawling by Sampling. In: Proceedings of IEEE/WIC/ACM Web Intelligence, Sydney, Australia, pp. 718–724 (2008)
8. Wu, P., Wen, J.R., Liu, H., Ma, W.Y.: Query Selection Techniques for Efficient Crawling of Structured Web Source. In: Proceedings of ICDE 2006, Atlanta, GA, pp. 47–56 (2006)
9. Ipeirotis, P., Gravano, L.: Distributed search over the hidden web: Hierarchical database sampling and selection. In: VLDB 2002, Hong Kong, China, pp.394–405 (2002)
10. Kaelbling, L.P., Littman, M.L., Moore, A.W.: Reinforcement learning: A survey. *Journal of Artificial Intelligence Research* 4, 237–285 (1996)
11. Jiang, L., Wu, Z.H., Zheng, Q.H., Liu, J.: Learning Deep Web Crawling with Diverse Features. In: Proceedings of IEEE/WIC/ACM Web Intelligence, Milan, Italy, pp. 572–575 (2009)
12. Yamamoto, M., Church, K.W.: Using suffix arrays to compute term frequency and document frequency for all substrings in a corpus. *Computational Linguistics* 27(1), 1–30 (2001)
13. Watkins, C.J., Dayan, P.: Q-learning. *Machine Learning* 8, 279–292 (1992)
14. Ratsaby, J.: Incremental Learning with Sample Queries. *IEEE Trans. on PAMI* 20(8), 883–888 (1998)
15. Amstrup, S.C., McDonald, T.L., Manly, B.F.J.: Handbook of capture–recapture analysis. Princeton University Press, Princeton (2005)
16. Mandelbrot, B.B.: Fractal Geometry of Nature. W. H. Freeman and Company, New York (1988)
17. Sutton, R.C., Barto, A.G.: Reinforcement learning: An Introduction. The MIT Press, Cambridge (1998)

Topic Decomposition and Summarization

Wei Chen*, Can Wang, Chun Chen, Lijun Zhang, and Jiajun Bu

College of Computer Science, Zhejiang University
Hangzhou, 310027, China
{chenw,wcan,chenc,zljzju,bjj}@zju.edu.cn

Abstract. In this paper, we study topic decomposition and summarization for a temporal-sequenced text corpus of a specific topic. The task is to discover different topic aspects (i.e., sub-topics) and incidents related to each sub-topic of the text corpus, and generate summaries for them. We present a solution with the following steps: (1) deriving sub-topics by applying Non-negative Matrix Factorization (NMF) to terms-by-sentences matrix of the text corpus; (2) detecting incidents of each sub-topic and generating summaries for both sub-topic and its incidents by examining the constitution of its encoding vector generated by NMF; (3) ranking each sentences based on the encoding matrix and selecting top ranked sentences of each sub-topic as the text corpus' summary. Experimental results show that the proposed topic decomposition method can effectively detect various aspects of original documents. Besides, the topic summarization method achieves better results than some well-studied methods.

Keywords: Non-negative Matrix Factorization, Topic Decomposition, Topic Summarization, Singular Value Decomposition.

1 Introduction

Users nowadays are overwhelmed by the vast amount of information on the Web. Although they can find information for a specific topic easily using search engines, they still have difficulty in finding more detailed aspects of a topic before reading dozens of Web documents returned. For example, it is a non-trivial task to make a comprehensive survey of a topic such as “9/11 attacks”. Related reports may cover various aspects (i.e., sub-topics) including “attackers and their motivation”, “the rescue attempts”, “9/11 investigations”, etc. Each sub-topic may further contain a set of related incidents, e.g., “9/11 investigations” has a series of related incidents along the timeline, such as “the NSA intercepted communications that pointed to bin Laden on Sep.11, 2001”, “FBI released photos of the 19 hijackers on Sep.27, 2001”, etc. Thus, discovering sub-topics and related incidents for a specific topic in a text corpus and summarizing them will greatly facilitate user’s navigation in the corpus space.

The above problems can be partially solved by topic decomposition and text summarization, which was first proposed systematically by Chen and Chen[1]. Their solution is called TSCAN (Topic Summarization and Content ANatomy). TSCAN equals to latent semantic analysis (LSA) based on the singular value decomposition (SVD)[2]. We

* This work is supported by China National Key Technology R&D Program (2008BAH26B00).

also study this problem in this paper. However, our solution is based on Non-negative Matrix Factorization (NMF)[3]. NMF has been demonstrated advantages over SVD in latent semantic analysis, document clustering [4]. In our work, we model the documents of a specific topic as a terms-by-sentences matrix. NMF is used to factorize the matrix into a non-negative sub-topic matrix and a non-negative encoding matrix. Each row of the encoding matrix is examined to extract incidents and their summaries. Summary for each sub-topic is generated by composing its incidents' summaries. We rank sentences by analyzing the encoding matrix, and the top ranked sentences of each sub-topic are selected as the summary for the text corpus.

2 Related Work

For a given temporal documents of a specific topic, TSCAN has following steps: Firstly, the documents are decomposed into a set of blocks. Then, a $m \times n$ terms-by-blocks matrix \mathbf{A} is constructed. $A_{i,j}$ is the weight of term i in block j , which is computed by TF-IDF weighting scheme. The block association matrix $\mathbf{B} = \mathbf{A}^T \mathbf{A}$, it is factorized as follow:

$$\mathbf{B} \approx \mathbf{T}_r \mathbf{D}_r \mathbf{T}_r^T \quad \mathbf{T}_r \in \mathbb{R}^{n \times r}, \mathbf{D}_r \in \mathbb{R}^{r \times r}, \tag{1}$$

where \mathbf{D}_r is a $r \times r$ diagonal matrix where the diagonal entries are the top r eigenvalues of \mathbf{B} . And \mathbf{T}_r is a $n \times r$ matrix in which each of the r columns represents a sub-topic. By examining the constitution of each columns of \mathbf{T}_r , the significant incidents of each topic aspect are detected and their summaries are generated. Then, the summary of the topic documents is obtained by combining all detected incident's summary.

We assume the SVD of the terms-by-blocks matrix \mathbf{A} as follow:

$$\mathbf{A} = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^T \quad \mathbf{U} \in \mathbb{R}^{m \times m}, \mathbf{\Sigma} \in \mathbb{R}^{m \times n}, \mathbf{V} \in \mathbb{R}^{n \times n}, \tag{2}$$

where both \mathbf{U} and \mathbf{V} are orthogonal matrices, and $\mathbf{\Sigma}$ is a diagonal matrix. The diagonal entries of $\mathbf{\Sigma}$ are the singular values of the matrix \mathbf{A} . Each column of matrices \mathbf{U} and \mathbf{V} are called left-singular and right-singular vectors. Then,

$$\mathbf{B} = \mathbf{A}^T \mathbf{A} = (\mathbf{U} \mathbf{\Sigma} \mathbf{V}^T)^T (\mathbf{U} \mathbf{\Sigma} \mathbf{V}^T) = \mathbf{V} \mathbf{\Sigma}^T \mathbf{U}^T \mathbf{U} \mathbf{\Sigma} \mathbf{V}^T = \mathbf{V} (\mathbf{\Sigma}^T \mathbf{\Sigma}) \mathbf{V}^T. \tag{3}$$

The squares of singular values of the matrix \mathbf{A} (i.e., $\mathbf{\Sigma}^T \mathbf{\Sigma}$) are equal to the eigenvalues of the matrix $\mathbf{A}^T \mathbf{A}$ (i.e., \mathbf{B}) [5]. In LSA, the r largest singular values with corresponding singular vectors from \mathbf{U} and \mathbf{V} are used to approximation the matrix \mathbf{A} [2], i.e.,

$$\mathbf{A} \approx \mathbf{U}_r \mathbf{\Sigma}_r \mathbf{V}_r^T \quad \mathbf{U}_r \in \mathbb{R}^{m \times r}, \mathbf{\Sigma}_r \in \mathbb{R}^{r \times r}, \mathbf{V}_r^T \in \mathbb{R}^{r \times n}, \tag{4}$$

Then, \mathbf{B} can be approximated as follow:

$$\mathbf{B} = \mathbf{A}^T \mathbf{A} \approx \mathbf{V}_r (\mathbf{\Sigma}_r^T \mathbf{\Sigma}_r) \mathbf{V}_r^T. \tag{5}$$

Because $\mathbf{\Sigma}_r^T \mathbf{\Sigma}_r$ are the top r eigenvalues of the matrix \mathbf{B} , the matrix \mathbf{V}_r is equal to the sub-topic matrix \mathbf{T}_r derived by TSCAN. That is, the sub-topics derived by TSCAN corresponds to the right singular vectors with most significant singular values of \mathbf{A} . In this paper, we focus on extractive multi-document summarization which are widely studied[6,7,8,9]. It extracts top significant sentences calculated by a set of ranking methods from the documents set.

3 The Proposed Solution Based on NMF

Given a pre-specified topic t , it is represented as $D = \{d_1, d_2, \dots, d_i, \dots\}$, where d_i is a document at time point i . We call various topic aspects as sub-topics and define them as $ST = \{st_1, st_2, \dots, st_k, \dots, st_r\}$. The series of incidents corresponding to sub-topic st_k is defined as $STI_k = \{sti_{k,1}, sti_{k,2}, \dots, sti_{k,i}, \dots, sti_{k,l}\}$. Our task of topic decomposition is to find out sub-topics ST , the incidents STI_k related to sub-topic st_k . Besides, we generate summaries for the text corpus D , sub-topics ST and incidents STI . Each document in D is decomposed into a sequence of sentences using sentence separation software provided by DUC [10]. 425 Rijsbergen’s stopwords are removed and stemming is performed. The documents in D is represented by a $m \times n$ terms-by-sentences matrix \mathbf{M} , where m is the number of terms and n is the number of sentences respectively. $M_{i,j}$ is computed by TF-IDF weighting scheme. The terms set is defined as $T = \{t_1, t_2, \dots, t_i, \dots, t_m\}$ while the sentences set is $S = \{s_1, s_2, \dots, s_j, \dots, s_n\}$.

3.1 Topic Decomposition Based on NMF

Non-negative Matrix Factorization (NMF) is a matrix factorization method that generates positive factorization of a given positive matrix [3]. It represents object as a non-negative linear combination of part information extracted from plenty of objects and is able to learn parts of semantic features from text. Given the matrix \mathbf{M} , NMF decomposes \mathbf{M} into a non-negative matrix \mathbf{B} and a non-negative matrix \mathbf{E} so that

$$\mathbf{M} \approx \mathbf{BE} \quad \mathbf{B} \in \mathbb{R}^{m \times r}, \mathbf{E} \in \mathbb{R}^{r \times n}. \tag{6}$$

We can find out \mathbf{B} and \mathbf{E} by minimizing the following cost function:

$$\arg \min_{\mathbf{B}, \mathbf{E}} \|\mathbf{M} - \mathbf{BE}\|_F^2, \tag{7}$$

where $\|\cdot\|_F$ denotes the Frobenius norm. The above constrained optimization problem can be solved by continuously updating \mathbf{B} and \mathbf{E} until the cost function converges under the predefined threshold or exceeds the number of repetitions [3,4]. The update rules are as follows ($1 \leq i \leq m, 1 \leq j \leq n$ and $1 \leq k \leq r$):

$$B_{i,k} \leftarrow B_{i,k} \frac{(\mathbf{ME}^T)_{i,k}}{(\mathbf{BEE}^T)_{i,k}} \quad E_{k,j} \leftarrow E_{k,j} \frac{(\mathbf{B}^T \mathbf{M})_{k,j}}{(\mathbf{B}^T \mathbf{BE})_{k,j}}. \tag{8}$$

The r columns of \mathbf{B} embed the so called sub-topics and each column of \mathbf{E} is the encoding. We refer \mathbf{B} as the sub-topic matrix and \mathbf{E} as the encoding matrix. Each sentence s_j can be represented by a linear combination of sub-topics. i.e.,

$$\mathbf{m}_j = \mathbf{Be}_j, \tag{9}$$

where \mathbf{m}_j is j -th sentence (i.e., j -th column of \mathbf{M}) and \mathbf{e}_j represents the j -th column of matrix \mathbf{E} . The entry $B_{i,k}$ indicates that the degree of term t_i belongs to sub-topic k , while $E_{k,j}$ represents that the degree of sentence s_j associates with sub-topic k .

Because the sentences set $S = \{s_1, s_2, \dots, s_j, \dots, s_n\}$ is indexed chronologically, the row k of encoding matrix \mathbf{E} (i.e., $e_{k,j}, 1 \leq j \leq n$, we refer it as sub-topic encoding

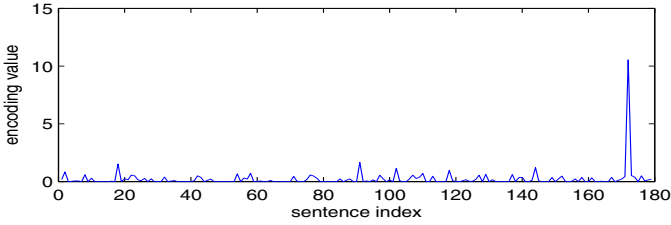


Fig. 1. A sub-topic’s encoding vector of the document cluster ‘d30001t’ in DUC 2004

vector) also denotes the relative strength of sub-topic st_k along the timeline. Herein, a list of continuous elements of $e_{k,j}(1 \leq j \leq n)$ with high bursty values can be regarded as an incident related to sub-topic k . Fig. 1 shows a sub-topic encoding vector of document cluster ‘d30001t’ in DUC 2004 [10] after applying NMF with $r = 10$. In Fig. 1 the encoding value is bursty around the sentence 170. It means that the sub-topic has a significant development around the sentence 170 (i.e., an incident breaks out).

The bursty detection problem is well studied in stream mining community [11]. Formally, given an aggregate function G (here is sum), a sliding window of size w and corresponding thresholds γ , the problem is to discover all these sub-sequences such that the function G applied to $e_{k,j:j+w-1}(1 \leq j \leq n - w + 1)$ exceeds threshold γ , i.e., check if

$$G(e_{k,j:j+w-1}) = \sum_{i=0}^{i=w-1} e_{k,j+i} \geq \gamma. \tag{10}$$

The threshold is set as $\gamma = mean(G(e_{k,j:j+w-1})) + \epsilon \times std(G(e_{k,j:j+w-1}))$, ($1 \leq j \leq n - w + 1$), where $mean()$ and $std()$ are the average and standard deviation function respectively. We set ϵ as 3 and w as 7 in our experiments. Finally, the detected bursty sequences are recognized as incidents.

3.2 Topic, Sub-topics and Incidents Summarization

An interesting by-product of topic decomposition is that the produced information can also be used to generate summary. Lee et al. [9] also use NMF to do generic document summarization. In their work, the rank of each sentence is calculated as follows:

$$rank(s_j) = \sum_{k=1}^r (E_{k,j} \times weight(e_{k,1:n})), \tag{11}$$

$$weight(e_{k,1:n}) = \frac{\sum_{y=1}^n E_{k,y}}{\sum_{x=1}^r \sum_{y=1}^n E_{x,y}}. \tag{12}$$

The $weight(e_{k,1:n})$ is the relative relevance of k -th sub-topic among all sub-topics. Finally, the top- x sentences with highest rank are chosen as summaries. We refer this method as NMF in our experiments. As point out by [8], a good summary should contain as few redundant sentences as possible while contain every important aspects of the documents. However, the solution of Lee et al. [9] doesn’t satisfy above requirements

sometimes. The top- x sentences with highest rank may belong to the same sub-topics and contain some overlapping information. In fact, most of the traditional summarization methods select sentences from different sub-topics [6,7]. We design a generic multi-document summarization method based on NMF (We refer it as INMF). Before going on, we give the definition of a sentence's main topic:

$$main_topic(s_j) = \arg \max_k (E_{k,j}), \quad (13)$$

That is, the main topic of sentence s_j is the sub-topic with the maximum encoding value in column j of encoding matrix \mathbf{E} [4]. The function $topic()$ returns the union of each sentence's main topic of a sentences set, e.g.,

$$topic(S) = main_topic(s_1) \cup main_topic(s_2) \cup \dots \cup main_topic(s_n). \quad (14)$$

The proposed multi-document summarization method INMF is described in Algorithm 1. Most of the summarization evaluations require the generated summaries in limited size or limited sentences number. In Algorithm 1, we limit the number of sentences of the summary. It can be easily revised to control the size of final summary. Different from [9], the INMF algorithm selects sentences with most significant ranking scores from different sub-topics in order to ensure the coverage and diversity of the summary. For each incident $sti_{k,i}$, we can straightforwardly choose the sentences with the largest or top- x encoding values of $sti_{k,i}$ as the summary. Then, the summary for sub-topic st_k can be generated by composing all the summaries of STI_k .

Algorithm 1. The proposed multi-document summarization method based on NMF

Input: $S = \{s_1, s_2, \dots, s_n\}$; ns, the limitation of sentences number in summary
Output: a string array $SS = \{ss_1, ss_2, \dots, ss_{ns}\}$, sentences set of summary
1: Sort sentences in S using equation [1] $rank(s_1) \geq rank(s_2) \geq \dots \geq rank(s_n)$
2: $k = 1$; $TS = \emptyset$; // TS is the sub-topics set
3: for $k \leq ns$ do
4: $i = 1$;
5: for $i \leq size(S)$ do // size() returns the number of elements in set S
6: if $main_topic(s_i) \notin TS$ then
7: $ss_k \leftarrow s_i$; $S = S - s_i$; $TS = TS \cup main_topic(s_i)$; $k++$; break;
8: end if
9: $i++$;
10: end for
11: if $size(TS) == r$ then // r is total sub-topics number
12: $TS = \emptyset$;
13: end if
14: end for

4 Experimental Studies

In the following, we first evaluate the proposed topic summarization method. And then, we give a case study of topic decomposition. The dataset of multi-document summarization task in DUC 2004 [10] is used to evaluate the proposed methods.

4.1 Summarization Evaluations

We implement four baseline summarization systems: FORWARD(extracts the initial sentences of all documents of a topic); BACKWARD(generates summaries by selecting the end sentences of a topic); TSCAN; NMF(method of Lee et al., [9]). The number of sentences of summary generated by TSCAN is indeterminate. To ensure the comparison is fair, the evaluation procedure is as follows [1]: For each r , we firstly apply TSCAN to each document cluster to select a set of sentences as summary. Then, we use other methods to extract the same number of sentences for each r and document cluster. Both ROUGE-1 [12] and summary-to-document content coverage [1] metrics are used.

Table 1. Overall performance comparison of ROUGE-1 on DUC 2004

| r | INMF | NMF | TSCAN | FORWARD | BACKWARD |
|-----|---------|---------|---------|---------|----------|
| 2 | 0.31161 | 0.29707 | 0.23983 | 0.23875 | 0.18211 |
| 3 | 0.33475 | 0.32156 | 0.25342 | 0.25383 | 0.19766 |
| 4 | 0.36529 | 0.35522 | 0.27096 | 0.27092 | 0.22081 |
| 5 | 0.39042 | 0.38238 | 0.29288 | 0.29061 | 0.24342 |
| 6 | 0.39739 | 0.40410 | 0.30370 | 0.31152 | 0.25867 |
| 7 | 0.43594 | 0.42632 | 0.31636 | 0.32451 | 0.28100 |
| 8 | 0.46620 | 0.45518 | 0.33862 | 0.34409 | 0.29974 |
| 9 | 0.47680 | 0.47117 | 0.35014 | 0.35653 | 0.31159 |
| 10 | 0.48975 | 0.48382 | 0.36947 | 0.36348 | 0.32110 |

The overall performance comparison of ROUGE-1 on DUC 2004 is listed in Table 1. It shows that the two NMF based summarization methods get much better results than other methods for all r . This is because both the two NMF based methods try to cover all content as much as possible. However, TSCAN may not consider sub-topics successfully, FORWARD extracts beginning sentences and BACKWARD takes the end sentences. As r increase, TSCAN extracts more sentences as the summary. Because ROUGE is recall-oriented, the ROUGE-1 scores of all methods increase with the increasing of summary size as showed in Table 1. The proposed INMF method increase summary coverage by selecting sentences from different sub-topics explicitly. As a result, INMF outperforms NMF in most cases except $r = 6$.

A good summary should contain important aspects of original topic documents as much as possible [8]. Herein, we apply summary-to-document content similarity to measure the coverage of summary according to [1]. That is, given a document cluster of a specific topic and its summary which are represented by TF-IDF term vectors. It computes the average cosine similarity between each of the document clusters and its summary. The higher the similarity, the better the summary represents document cluster. We show the summary-to-documents similarity corresponding to table 1 in table 2.

In table 2, both INMF and NMF achieve much better results than TSCAN, FORWARD, BACKWARD for all r . It is easy to understand that all the three latter methods lose some information and the coverage is poor. However, Non-negative Matrix Factorization decomposes all of topic's information into r sub-topics, and the two NMF based summarization method extract the sentences with as much information as possible.

Table 2. Summary-to-document content similarity corresponding to table 1

| r | INMF | NMF | TSCAN | FORWARD | BACKWARD |
|-----|----------|----------|----------|----------|----------|
| 2 | 0.126703 | 0.123421 | 0.105427 | 0.103786 | 0.102524 |
| 3 | 0.128219 | 0.122003 | 0.107548 | 0.103958 | 0.104369 |
| 4 | 0.126229 | 0.121232 | 0.105550 | 0.104454 | 0.100763 |
| 5 | 0.127125 | 0.122199 | 0.108460 | 0.107260 | 0.102478 |
| 6 | 0.127987 | 0.122781 | 0.103569 | 0.104365 | 0.101695 |
| 7 | 0.128505 | 0.124848 | 0.102935 | 0.101614 | 0.102715 |
| 8 | 0.130945 | 0.126131 | 0.108045 | 0.105535 | 0.101850 |
| 9 | 0.127965 | 0.123313 | 0.106552 | 0.105038 | 0.099187 |
| 10 | 0.128130 | 0.124492 | 0.111100 | 0.109034 | 0.107870 |

Table 3. Sub-topic's description and the sentence id of each sub-topic's summary

| ST id | sub-topic description | sentence id |
|-------|--|----------------|
| 1 | Hun Sen and Ranariddh often clashed over power-sharing and the integration of guerrilla fighters from the crumbling Khmer Rouge. | 74,119 |
| 2 | King Norodom Sihanouk called Ranariddh and Sam Rainsy to return to Cambodia and wanted to preside over a summit meeting of the three party leaders. | 20,46,56,57 |
| 3 | Norodom Ranariddh and Sam Rainsy, citing Hun Sen's threats to arrest opposition figures, said they could not negotiate freely in Cambodia. | 3,30,141,163 |
| 4 | In July election, Hun Sen's party collected 64 of the 122 parliamentary seats, but was short of the two-thirds majority needed to set up a new government. | 8,25,44,85,111 |
| 5 | The violent crackdown in Cambodia, at least four demonstrators were killed. | 40,64,69 |
| 6 | Hun Sen and Ranariddh agreed to form a coalition that leave Hun Sen as sole prime minister and make Ranariddh president of the National Assembly. | 83,123,152,175 |
| 7 | People's Party criticized the resolution passed earlier this month by the U.S. House of Representatives. | 59 |
| 8 | King Norodom Sihanouk praised agreements by Cambodia's top two political parties previously bitter rivals to form a coalition government. | 172 |
| 9 | Sam Rainsy wanted to attend the first session of the new National Assembly on Nov. 25, but complained that his party members' safety. | 160 |
| 10 | The Cambodian People's Party gave a statement about supporting the police action of violent crackdown to protesters. | 70 |

Besides, the proposed INMF summarization method explicitly tries to select sentences belong to different topic aspects. That's why INMF outperforms NMF in all cases.

4.2 Topic Decomposition

The documents set 'd30001t' of DUC 2004 is used as a case study for topic decomposition. It includes 10 documents and 179 sentences about "political crisis in Cambodia in October 1998". The detailed description about each sub-topic and sentence id of its summary is showed in table 3. We manually compared each sub-topics' summaries with reference summaries('D30001.M.100.T.A', 'D30001.M.100.T.B', 'D30001.M.100.T.C' and 'D30001.M.100.T.D' with size 658, 661, 647 and 656 bytes respectively) created

by DUC assessors. For ‘D30001.M.100.T.C’ and ‘D30001.M.100.T.D’, the information coverage is 100%. The text “the opposition tried to cut off his access to loans” (total 51 bytes) in ‘D30001.M.100.T.A’ and “Opposition parties ask the Asian Development Bank to stop loans to Hun Sen’s government”(total 87 bytes) in ‘D30001.M.100.T.B’ are lost in the generated summaries. Then, the average information coverage of the generated sub-topics’ summary to the reference summaries is $((658 - 51)/658 + (661 - 87)/661 + 100 + 100)/4 = 94.75\%$.

Some sub-topics contain a series of incidents while others contain only one. For example, sub-topic 6 is about the process of forming a coalition government which contains several incidents. Sub-topic 8 has only one incident, which is about King Norodom Sihanouk’s praise about the agreements by Cambodia’s top two political parties. We also compare our results with TSCAN for topic decomposition with $r = 10$. TSCAN detects total 23 incidents. 5 incidents are the same (some incidents duplicate more than 2 times), with only 15 sentences left as the incidents’ summary. The 15 sentences are from 7 documents while the incidents’ summaries of our method are from all 10 documents. Besides, our method covers more aspects than TSCAN, e.g. sub-topic 5 and sub-topic 10 are not included in the result of TSCAN.

5 Conclusion

In this paper, we study the problem of topic decomposition and summarization for a temporal-sequenced text corpus of a specific topic. We represent the text corpus as a terms-by-sentences matrix and derive sub-topics by factorize the matrix using Non-negative Matrix Factorization. By analyzing the encoding matrix, we can detect incidents of each sub-topic and generate summaries for both sub-topics and their related incidents. The summary for the text corpus is generated by firstly ranking each sentences based on the encoding matrix, and then selecting most significant sentences from each sub-topics. Experimental results show that our method can effectively find out different topic aspects of a documents set and generate promising results in summarization.

References

1. Chen, C.C., Chen, M.C.: TSCAN: A Novel Method for Topic Summarization and Content Anatomy. In: Proc. of the 31st ACM SIGIR conference, pp. 579–586. ACM, USA (2008)
2. Deerwester, S., Dumais, S.T., Harshman, R.: Indexing by latent semantic analysis. *Journal of the American Society for Information Science* 41(6), 391–407 (1990)
3. Lee, D.D., Seung, H.S.: Learning the parts of objects by non-negative matrix factorization. *Nature* 401, 788–791 (1999)
4. Xu, W., Liu, X., Gong, Y.H.: Document Clustering Based on Non-negative Matrix Factorization. In: Proc. of the 26th ACM SIGIR conference, pp. 267–273. ACM, USA (2003)
5. Strang, G.: *Introduction to Linear Algebra*. Wellesley Cambridge Press, Wellesley (2003)
6. Gong, Y.H., Liu, X.: Generic Text Summarization Using Relevance Measure and Latent Semantic Analysis. In: Proc. of the 24th ACM SIGIR conference, pp. 19–25. ACM, USA (2001)

7. Zha, H.Y.: Generic Summarization and Keyphrase Extraction Using Mutual Reinforcement Principle and Sentence Clustering. In: Proc. of 25th ACM SIGIR, pp. 113–120 (2002)
8. Wan, X.J., Yang, J.W., Xiao, J.G.: Manifold-Ranking Based Topic-Focused Multi-Document Summarization. In: Proc. of IJCAI, pp. 2903–2908. ACM, USA (2007)
9. Lee, J.H., Park, S., Ahn, C.M., Kim, D.: Automatic generic document summarization based on non-negative matrix factorization. *Info. Processing and Management* 45, 20–34 (2009)
10. Document Understanding Conferences (2004),
<http://www-nlpir.nist.gov/projects/duc/index.html>
11. Vlachos, M., Meek, C., Vagena, Z., Gunopulos, D.: Identifying Similarities, Periodicities and Bursts for Search Queries. In: Proc. of ACM SIGMOD, pp. 131–142. ACM, USA (2004)
12. Lin, C.Y.: ROUGE: a Package for Automatic Evaluation of Summaries. In: Proc. of the Workshop on Text Summarization Branches Out, Barcelona, Spain, pp. 74–81 (2004)

UNN: A Neural Network for Uncertain Data Classification

Jiaqi Ge, Yuni Xia, and Chandima Nadungodage

Department of Computer and Information Science, Indiana
University – Purdue University, Indianapolis, USA
{jiaqge, yxia, chewanad}@cs.iupui.edu

Abstract. This paper proposes a new neural network method for classifying uncertain data (UNN). Uncertainty is widely spread in real-world data. Numerous factors lead to data uncertainty including data acquisition device error, approximate measurement, sampling fault, transmission latency, data integration error and so on. The performance and quality of data mining results are largely dependent on whether data uncertainty are properly modeled and processed. In this paper, we focus on one commonly encountered type of data uncertainty - the exact data value is unavailable and we only know the probability distribution of the data. An intuitive method of handling this type of uncertainty is to represent the uncertain range by its expectation value, and then process it as certain data. This method, although simple and straightforward, may cause valuable information loss. In this paper, we extend the conventional neural networks classifier so that it can take not only certain data but also uncertain probability distribution as the input. We start with designing uncertain perceptron in linear classification, and analyze how neurons use the new activation function to process data distribution as inputs. We then illustrate how perceptron generates classification principles upon the knowledge learned from uncertain training data. We also construct a multilayer neural network as a general classifier, and propose an optimization technique to accelerate the training process. Experiment shows that UNN performs well even for highly uncertain data and it significantly outperformed the naïve neural network algorithm. Furthermore, the optimization approach we proposed can greatly improve the training efficiency.

Keywords: Uncertainty, classification, neural network.

1 Introduction

Data tends to be uncertain in many applications [1], [2], [3], [4], [5]. Uncertainty can originate from diverse sources such as data collection error, measurement precision limitation, data sampling error, obsolete source, network latency and transmission error. The error or uncertainty in data is commonly treated as a random variable with probability distribution. Thus, uncertain attribute value is often represented by an interval with a probability distribution function over the interval [6], [7]. It is important to cautiously handle the uncertainty in various data mining applications, as the

data uncertainty is useful information which can be leveraged in order to improve the quality of the underlying results [17]. However, many traditional data mining problems become particularly challenging for the uncertain case. For example, in a classification application, the class which a data point belongs to may be changing as a result of the vibration of its uncertain attributes' values. Furthermore, the uncertainty over the whole dataset may blur the boundaries among different classes, which brings extra difficulties to classify uncertain datasets. Thus, data mining algorithms for classification, clustering and frequent pattern mining may need to integrate data uncertainty models to achieve satisfactory performance.

Classification is one of the key processes in machine learning and data mining. Classification is the process of building a model that can describe and predict the class label of data based on the feature vector [8]. An intuitive way of handling uncertainty in classification is to represent the uncertain value by its expectation value and treat it as a certain data. Thus, conventional classification algorithms can be directly applied. However, this approach does not effectively utilize important information such as probability function or distribution intervals. We extend data mining techniques so that they can take uncertain data such as data interval and probability distribution as the input. In this paper, we design and develop a new classifier named uncertain neural network (UNN), which employs new activation function in neurons to handle uncertain values. We also propose a new approach to improve the training efficiency of UNN. We prove through experiments that the new algorithm has satisfactory classification performance even when the training data is highly uncertain. Comparing with the traditional algorithm, the classification accuracy of UNN is significantly higher. Furthermore, with the new optimization method, the training efficiency can be largely improved.

The paper is organized as follows. In section 2, we discuss related work. Section 3 defines the classification problem for uncertain data. In section 4, we first analyze the principle of uncertain perceptron in linear classification, and then construct the multilayer uncertain neural network, and discuss the training approach. Section 5 introduces an optimized activation function to improve the efficiency. The experiments results are shown in section 6, and section 7 makes a conclusion for the paper.

2 Related Works

There has been a growing interest in uncertain data mining. A number of data mining algorithms have been extended to process uncertain dataset. For example, UK-Means [9], uncertain support vector machine [10], and uncertain decision tree [11]. The key idea in [10] is to provide a geometric algorithm which optimizes the probabilistic separation between the two classes on both sides of the boundary [7]. And [11] extends the decision tree to handle interval inputs and takes probability cardinality to select the best splitting attribute. However, both these two uncertain classifiers use a simple bounded uncertain model, and in our work, we use Gaussian noise instead to model the uncertainty, which is more common in realistic world. Artificial neural network has been used in model-based clustering with a probability gained from expectation-maximization algorithm for classification-likelihood learning [12]. We adopt the concept to estimate the probability of membership when the uncertain data

are covered by multiples classes. However, probability estimation presented here is unprecedented.

In fuzzy neural network models for classification, either attributes or class labels can be fuzzy and are presented in fuzzy terms [13]. Given a fuzzy attribute of a data tuple, a degree (called membership) is assigned to each possible class, showing the extent to which the tuple belongs to a particular class. Some other fuzzy systems [18] build reasoning mechanisms based on rules, and try to simulate the fuzzy cases inside the network. But they do not take the detailed uncertainty information as probability distribution into account in neuron level. Our work differs from previous work in that we revise the activation functions to compute the membership based on uncertain data distribution information, instead of using Fuzzy logic for tuning neural network training parameters. Our approach can work on both certain and uncertain data.

3 Problem Definition

In our model, a dataset D consists of d training tuples, $\{t_1, t_2, \dots, t_d\}$, and k numerical attributes, A_1, \dots, A_k . Each tuple t_i is associated with a feature vector $V_i = (f_{i,1}, f_{i,2}, \dots, f_{i,k})$, and a class label $c_i \in C$. Here, each $f_{i,j}$ is a pdf modeling the uncertain value of attribute A_j in tuple t_i . Table. 1 shows an example of an uncertain dataset. The first attribute is uncertain. The exact value of this attribute is unavailable, and we only know the expectation and variance of each data tuple. This type of data uncertainty widely exists in practice [1], [2], [5], [6], [7].

Table. 1. An example of uncertain dataset

| ID | Class Type | Attribute #1 (expectation, standard variance) |
|----|------------|--|
| 1 | Yes | (105, 5) |
| 2 | NO | (110,10) |
| 3 | No | (70,10) |
| 4 | Yes | (120,18) |
| 5 | No | (105,10) |
| 6 | No | (60,20) |
| 7 | Yes | (210,20) |
| 8 | No | (90,10) |
| 9 | No | (85,5) |
| 10 | No | (120,15) |

The classification problem is to construct a relationship M that maps each feature vector $(f_{x,1}, f_{x,2}, \dots, f_{x,k})$ to the membership P_x on class label C , so that given a test tuple $t_0=(f_{0,1}, f_{0,2}, \dots, f_{0,k})$, $M(f_{0,1}, f_{0,2}, \dots, f_{0,k})$ predict the membership to each class. If the test instance has positive probability to be in different classes, then it will be predicted to be in the class which has the highest probability. The work in this paper is to build a neural network when only uncertain training data tuples are available, and the goal is to find the model with the highest accuracy despite of the uncertainty.

4 Algorithm

4.1 Uncertain Perceptron

We start with perceptron, which is a simple type of artificial neural network. Perceptron is a classical model which constructs linear classifier as:

$$y = F\left(\sum_{i=1}^n x_i \omega_i + \theta\right), \tag{4.1}$$

$$F(s) = \begin{cases} 1, & s \geq 0 \\ -1, & s < 0 \end{cases}.$$

Where $x = (x_1, \dots, x_n)$ is the input vector, $\omega = (\omega_1, \dots, \omega_n)$ is the weight vector, F is the activation function, and y is the perceptron's output.

For data sets with uncertain attributes, we need revise the functions and develop an uncertain perceptron for linear classification. We will illustrate our approach through a simple 2-dimensional dataset. Assume dataset has two attributes $X = (x_1, x_2)$ and one class type y , and assume each uncertain attribute has a distribution as $x_i \sim N(\mu_i, \sigma_i)$, and the class type can be +1 or -1, Fig. 1 is a geometric representation of linear classification for a 2-dimensional uncertain dataset. In this figure, each data instance is represented by an area instead of a single point because each dimension/attribute is an uncertain distribution, not an accurate value.

The straight line L in Fig 1 represents the equation:

$$\omega_1 x_1 + \omega_2 x_2 + \theta = 0. \tag{4.2}$$

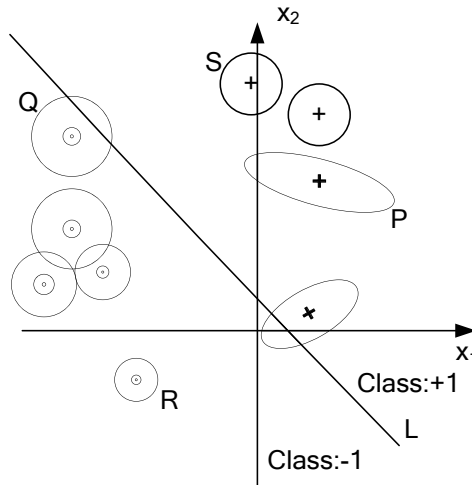


Fig. 1. Geometric representation of uncertain Perceptron

where x_1, x_2 are uncertain attributes. We define a parameter t as

$$t = \omega_1 x_1 + \omega_2 x_2 + \theta . \tag{4.3}$$

As mentioned earlier, attributes (x_1, x_2) follow the distribution $x_i \sim N(\mu_i, \sigma_i^2)$. Since these attributes are independent, t will have a distribution as:

$$f(t) \sim N(\omega_1 \mu_1 + \omega_2 \mu_2 + \theta, \omega_1^2 \sigma_1^2 + \omega_2^2 \sigma_2^2) . \tag{4.4}$$

Let $s = P(t>0)$ represent the probability of t larger than 0. If $P(t>0) = 1$, t is definitely larger than 0, which means this tuple is in class +1, and locates above the line L in Fig. 1, for example, like Point P . If $P(t>0) = 0$, t is less than or equal to 0, which means this tuple is in class -1, and it is below line L such as Point R . For uncertain data, it is possible that the uncertain range of a data instance may cover the linear classification line L , for example, Point Q is one such instance. In this case, Q has positive probability to belong to both classes, and the membership of class will be determined by which class has a high probability. Therefore, we construct an activation function as equation (4.5).

$$F(s) = \begin{cases} 1, & s \geq 0.5 \\ -1, & s < 0.5 \end{cases} . \tag{4.5}$$

Where, $s = P(t>0)$. Fig. 2 is structure of the uncertain perceptron model. In Fig.2, (μ_i, σ_i) is the expectation and standard deviation of uncertain attributes, as inputs. When the distribution is Gaussian, s can be calculated as:

$$s = \int_0^\infty \frac{1}{\sqrt{2\pi}\sigma_t} \exp\left(-\frac{(t-u_t)^2}{2\sigma_t^2}\right) dt . \tag{4.6}$$

Based on the single uncertain neurons, we can develop a multilayer neural network.

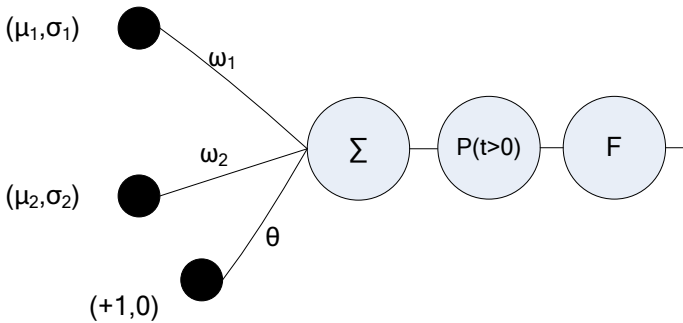


Fig. 2. Uncertain perceptron structure

4.2 Uncertain Neural Network

An uncertain multilayer feed-forward neural network is constructed by adding a hidden layer which contains the uncertain neurons between input and output layers. We call this algorithm as UNN (for uncertain neural network). Fig. 3 is an instance of the layer structure of neural network. Here, the hidden layer has a transfer function as

$$F(\mu, \sigma) = P(t > 0) . \tag{4.7}$$

Where,

$$t = \sum_{i=1}^2 \omega_i x_i + \theta .$$

$$x_i \sim N (\mu_i, \omega_i) .$$

$P(t > 0)$ will be computed based on uncertain data distribution function, For example, if the data follows Gaussian distribution, then

$$P(t > 0) = \int_0^\infty \frac{1}{\sqrt{2\pi}\sigma_t} \exp\left(-\frac{(t - u_t)^2}{2\sigma_t^2}\right) dt .$$

The output layer can have an activation function as Sigmoid, since the output values fall in the range (0,1), to represent the membership of every class.

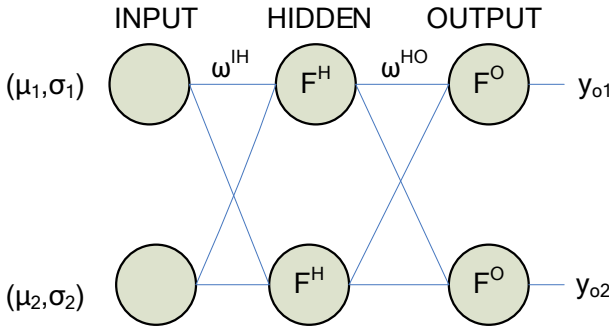


Fig. 3. Multilayer neural network structure

4.3 Algorithm Analysis

A straight-forward way to deal with the uncertain information is to replace the probability distribution function with its expected value. Then the uncertain data can be treated as certain data and the traditional neural network can be used for classification. We call this approach AVG (for Averaging). This approach, as mentioned earlier, does not utilize valuable uncertain information and may result in loss of accuracy. We illustrate the reason with the following example. Fig. 4 is an example of

classifying an uncertain dataset. Line $L1$ and $L2$ reflect the training result of the hidden layers of a neural network. Suppose P is a test data instance and we need predict the class type of P . Because the expectation of P locates in area II, it will be assigned to class II if using AVG algorithm. However, from Fig. 4, it is obvious that if we consider the distribution of P , it has a larger probability to be in area I than in area II. Therefore, it should be classified to class I. UNN will perform the classification correctly since it computes the probability of P belonging to both classes I and II according to the probability distribution information and predicts it to be in the class which has a larger probability. In this sense, the uncertain neural network can achieve higher classification accuracy.

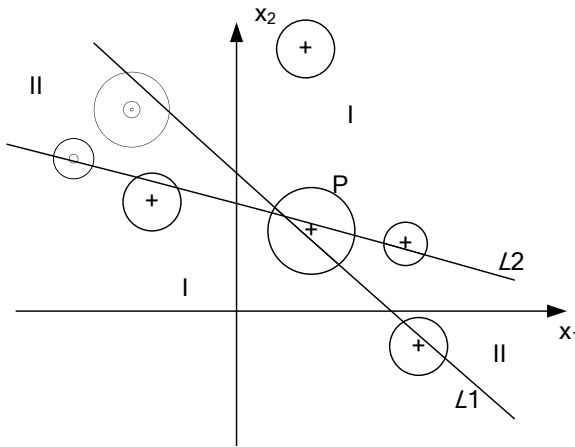


Fig. 4. Classifying a test tuple P

4.4 Network Training

We adopt a Levenberg-Marquardt back propagation algorithm [14], to train this supervised feed-forward neural network. It requires all the activation function has a derivative. Suppose Equation (4.7) is the hidden layer activation function of the uncertain neural network, then its derivative is like:

$$\frac{dF}{d(\mu, \sigma^2)} = \left(\frac{\partial F}{\partial \mu}, \frac{\partial F}{\partial \sigma^2} \right) = \left(\frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{\mu^2}{2\sigma^2}}, -\frac{\mu}{2\sqrt{2\pi}\sigma^3} e^{-\frac{\mu^2}{2\sigma^2}} \right). \tag{4.8}$$

And,

$$\frac{d\mu_i}{d\omega_i} = \mu_i, \frac{d\sigma_i^2}{d\omega_i} = 2\sigma_i^2 * \omega_i. \tag{4.9}$$

Therefore, by substituting Equation (4.8) (4.9) into Equation (4.10), we can get the activation function's derivatives.

$$\frac{dF}{d\omega_i} = \frac{\partial F}{\partial \mu} \frac{d\mu}{d\omega_i} + \frac{\partial F}{\partial \sigma} \frac{d\sigma}{d\omega_i} . \tag{4.10}$$

When we have the derivatives of these activation functions, it is intuitive to train the network based on traditional method such as gradient decent. After training, we can then use the model for prediction for uncertain data.

5 Improve on Activate Function

The hidden layer’s activate function, in Equation (4.7), has an output ranging between 0 and 1. When we consider two different data instances that are absolutely in the same class, their function output will both be 1. This may cause the network training to be time consuming in some scenarios. In order to improve the training efficiency, we can design new hidden layer activate functions. For example, when the uncertainty is represent by Gaussian distribution, we devise a new hidden layer activate function, as Equation (5.1) to accelerate the training process.

$$F_2(\mu, \sigma) = \begin{cases} u_i * P(t > 0), & \text{if } u_i > 0; \\ 0, & \text{if } u_i = 0; \\ u_i * P(t < 0), & \text{if } u_i < 0; \end{cases} \tag{5.1}$$

$$f(t) \sim N \left(\sum_i \omega_i \mu_i + \theta, \sum_i \omega_i^2 \sigma_i^2 \right) .$$

Here $F_2(\mu, \sigma)$ is continuous at $u_i = 0$, since

$$\lim_{\mu \rightarrow 0^+} F_2(\mu, \sigma) = \lim_{\mu \rightarrow 0^-} F_2(\mu, \sigma) = F_2(0, \sigma) = 0 .$$

$F_2(\mu, \sigma)$ also has a derivative:

$$\frac{dF_2}{d(\mu, \sigma^2)} = \left(\frac{\partial F_2}{\partial \mu}, \frac{\partial F_2}{\partial \sigma^2} \right) . \tag{5.2}$$

$$\frac{\partial F_2}{\partial \mu} = \begin{cases} \eta + \frac{\mu}{\sqrt{2\pi}\sigma} e^{-\frac{\mu^2}{2\sigma^2}}, & \text{if } \mu_i > 0 \\ 1/2, & \text{if } \mu_i = 0 \\ 1 - \eta - \frac{\mu}{\sqrt{2\pi}\sigma} e^{-\frac{\mu^2}{2\sigma^2}}, & \text{if } \mu_i < 0 \end{cases} . \tag{5.3}$$

$$\frac{\partial F_2}{\partial \sigma^2} = -\frac{\mu^2}{2\sqrt{2\pi}\sigma^3} e^{-\frac{\mu^2}{2\sigma^2}} . \tag{5.4}$$

Thus, substitute Equation (5.2) (5.3) (5.4) into Equation (5.5), we get the derivative of F_2 .

$$\frac{dF_2}{d\omega_i} = \frac{\partial F_2}{\partial \mu} \frac{d\mu}{d\omega_i} + \frac{\partial F_2}{\partial \sigma} \frac{d\sigma}{d\omega_i} \quad (5.5)$$

Equation (5.5) then can be used in Levenberg-Marquardt back propagation training algorithm.

6 Experiments

6.1 Experiment on Accuracy

We have implemented the UNN approach using Matlab6.5[15], and applied them to 5 real data sets taken from the UCI Machine Learning Repository [16]. The results are shown in Table. 2. For the datasets except “Japanese Vowel”, the data uncertainty is modeled with a Gaussian distribution with a controllable parameter ω , which is a percentage of the standard deviation to the value of expectation. In our experiments, we vary the ω value to be 0.1, 0.3 and 0.5. For “Japanese Vowel” data set, we use the uncertainty given by the original data to estimate its Gaussian distribution.

Table 2. Accuracy experiment results

| Japanese Vowel | Uncertainty | Train | Test |
|-----------------|-----------------------------|--------|--------|
| UNN | Distribution based raw data | 98.50% | 94.95% |
| AVG | | 99.17% | 94.31% |
| Iris | Uncertainty | Train | Test |
| UNN | $\omega=0.1$ | 98.05% | 99.93% |
| | $\omega=0.2$ | 98.33% | 99.93% |
| | $\omega=0.5$ | 97.78% | 99.38% |
| AVG | | 99.17% | 98.89% |
| Ionosphere | Uncertainty | Train | Test |
| UNN | $\omega=0.1$ | 92.75% | 93.71% |
| | $\omega=0.2$ | 94.50% | 90.73% |
| | $\omega=0.5$ | 99.13% | 92.05% |
| AVG | | 97.17% | 87.86% |
| Magic Telescope | Uncertainty | Train | Test |
| UNN | $\omega=0.1$ | 96.93% | 80.01% |
| | $\omega=0.2$ | 97.50% | 76.58% |
| | $\omega=0.5$ | 97.50% | 80.56% |
| AVG | | 99.67% | 73.17% |
| Glass | Uncertainty | Train | Test |
| UNN | $\omega=0.1$ | 77.05% | 65.75% |
| | $\omega=0.2$ | 76.00% | 69.59% |
| | $\omega=0.5$ | 79.02% | 65.57% |
| AVG | | 74.02% | 65.22% |

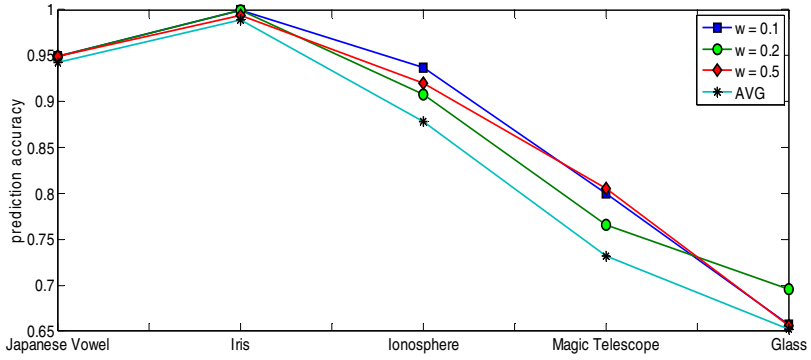
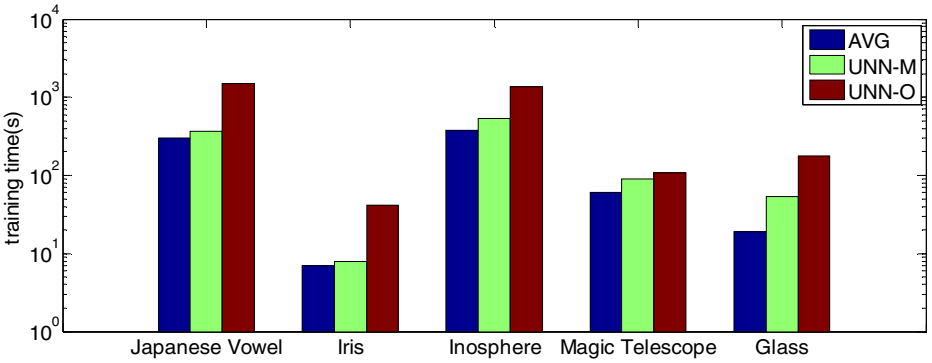
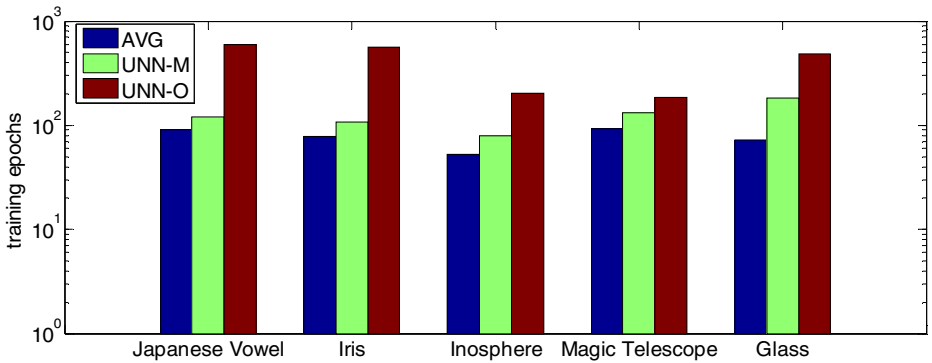


Fig. 5. Accuracy Comparison of UNN and AVG



(a) Training time



(b) Training epochs

Fig. 6. Performance comparison

In our experiments, we compare UNN with the AVG (Averaging) approach, which process uncertain data by simply using the expected value. The results are shown in Fig 5. From the figure, we can see that UNN outperforms AVG in accuracy almost all the time. For some datasets, for example, Ionosphere and Magic Telescope datasets, UNN improves the classification accuracy by over 6% to 7%. The reason is that UNN utilizes the uncertain data distribution information and computes the probability of data being in all different classes. Therefore, the classification and prediction process is more sophisticated and comprehensive than AVG, and has the potential to achieve higher accuracy.

6.2 Experiment on Efficiency

In section 5, we have discussed an alternative activate function for improving the efficiency of network training process. Here, we present an experiment which compares the efficiency of two networks with different hidden layer activate functions. In this experiment, we name the network using the original function (Equation 4.7) as UNN-O, and the network using activate function (5.1) as UNN-M.

The training time of UNN-O and UNN-M is shown in Fig. 6 (a) and the training epochs of UNN-O and UNN-M is shown in Fig 6. (b). Because of the more complex calculations in handling uncertainty, UNNs generally require more training time and epochs than AVG. However, the figures also indicate that efficiency of UNN-M is highly improved, compared with UNN-O. The training of UNN-M requires much fewer epochs than UNN-O, and is significantly faster.

7 Conclusion

In this paper, we propose a new neural network (UNN) model for classifying and predicting uncertain data. We employ the probability distribution which represent the uncertain data attribute, and redesign the neural network functions so that they can directly work on uncertain data distributions. Experiments show that UNN has higher classification accuracy than the traditional approach. The usage of probability distribution can increases the computational complexity, and we propose new activation function for improved efficiency. We plan to explore more classification approaches for various uncertainty models and find more efficient training algorithms in the future.

References

1. Aggarwal, C.C., Yu, P.: A framework for clustering uncertain data streams. In: IEEE International Conference on Data Engineering, ICDE (2008)
2. Cormode, G., McGregor, A.: Approximation algorithms for clustering uncertain data. In: Principle of Data base System, PODS (2008)
3. Kriegel, H., Pfeifle, M.: Density-based clustering of uncertain data. In: ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD), pp. 672–677 (2005)
4. Singh, S., Mayfield, C., Prabhakar, S., Shah, R., Hambrusch, S.: Indexing categorical data with uncertainty. In: IEEE International Conference on Data Engineering (ICDE), pp. 616–625 (2007)

5. Kriegel, H., Pfeifle, M.: Hierarchical density-based clustering of uncertain data. In: IEEE International Conference on Data Mining (ICDM), pp. 689–692 (2005)
6. Aggarwal, C.C.: On Density Based Transforms for uncertain Data Mining. In: IEEE International Conference on Data Engineering, ICDE (2007)
7. Aggarwal, C.C.: A Survey of Uncertain Data Algorithms and Applications. *IEEE Transactions on Knowledge and Data Engineering* 21(5) (2009)
8. Agrawal, R., Imielinski, T., Swami, A.N.: Database mining: A performance perspective. *IEEE Transactions on Knowledge & Data Engineering* (1993)
9. Chau, M., Cheng, R., Kao, B., Ng, J.: Uncertain data mining: An example in clustering location data. In: Ng, W.-K., Kitsuregawa, M., Li, J., Chang, K. (eds.) PAKDD 2006. LNCS (LNAI), vol. 3918, pp. 199–204. Springer, Heidelberg (2006)
10. Bi, J., Zhang, T.: Support Vector Machines with Input Data Uncertainty. In: *Proceeding of Advances in Neural Information Processing Systems* (2004)
11. Qin, B., Xia, Y., Li, F.: DTU: A Decision Tree for Classifying Uncertain Data. In: Theeramunkong, T., Kijssirikul, B., Cercone, N., Ho, T.-B. (eds.) PAKDD 2009. LNCS, vol. 5476, pp. 4–15. Springer, Heidelberg (2009)
12. Cheng, S.-S., Fu, H.-C., Wang, H.-M.: Model-Based Clustering by Probabilistic Self-Organizing Maps. *IEEE Transactions on Neural Networks* 20(5) (2009)
13. Kulkarni, A.D., Muniganti, V.K.: Fuzzy Neural Network Models For Clustering. In: *ACM Symposium on Applied Computing* (1996)
14. Stan, O., Kamen, E.W.: New block recursive MLP training algorithms using the Levenberg-Marquardt algorithm. *Neural Networks* 3 (1999)
15. Matlab: <http://www.mathworks.com/>
16. Asuncion, A., Newman, D.: UCI machine learning repository (2007), <http://www.ics.uci.edu/mllearn/MLRepository.html>
17. Aggarwal, C.C.: *Managing and Mining Uncertain Data*. Springer, Heidelberg (2009)
18. Jang, J.S.R.: ANFIS: Adaptive-network-based fuzzy inference system. *IEEE transactions on systems, man, and cybernetics* 23, 665–685 (1993)

SkyDist: Data Mining on Skyline Objects

Christian Böhm¹, Annahita Oswald¹, Claudia Plant²,
Michael Plavinski¹, and Bianca Wackersreuther¹

¹ University of Munich

² Technische Universität München

{boehm,oswald,plavinski,wackersreuther}@dbs.ifi.lmu.de,
plant@lrz.tum.de

Abstract. The skyline operator is a well established database primitive which is traditionally applied in a way that only a single skyline is computed. In this paper we use multiple skylines themselves as objects for data exploration and data mining. We define a novel similarity measure for comparing different skylines, called SkyDist. SkyDist can be used for complex analysis tasks such as clustering, classification, outlier detection, etc. We propose two different algorithms for computing SkyDist, based on Monte-Carlo sampling and on the plane sweep paradigm. In an extensive experimental evaluation, we demonstrate the efficiency and usefulness of SkyDist for a number of applications and data mining methods.

1 Introduction

Skyline queries are an important area of current database research, and have gained increasing interest in recent years [1,2,3,4,5]. Most papers focus on efficient algorithms for the construction of a *single* skyline which is the answer of a user's query. This paper extends the idea of skylines in such a way that multiple skylines are treated as objects for data exploration and data mining.

One of the most prominent applications of the skyline operator is to support complex decisions. As an example consider an online marketplace for used cars, where the user wants to find out offers which optimize more than one property of the car such as p (price) and m (mileage), with an unknown weighting of the single conditions. The result of such a query has to contain all offers which may be of interest: not only the cheapest offer and that with lowest mileage but also all offers providing an outstanding combination of p and m .

However, not all of these offers are equally attractive to the user. For instance, consider an offer A that has both a lower price *and* a lower mileage than another offer G and many other offers. Therefore, we say that A *dominates* G (in symbols $A \prec G$), because A is better than G w.r.t. any possible weighting of the criteria price and mileage. The *skyline* contains all objects which are *not dominated* by any other object in the database.

For the used car market, the skyline of each car model has a particular meaning: Many arbitrarily bad offers may be present in the database but only the offers in (or close to) the skyline have a high potential to find a customer. The skyline of the offers marks to some degree the fair value of a car for each mileage in the market. Therefore,

the skyline characterizes a car model (or higher-order objects of other applications) in a highly expressive way.

This high expressiveness leads us to the idea of treating the skylines themselves as a measure of similarity. Figure 1 illustrates the skylines of four different car models derived from real data of an online automotive market. Car models which exhibit similar skylines (like Audi A3 1.6 and Ford Focus 1.6) may be considered as similar: A recommender system might find out that the Focus is a perfect alternative for the Audi A3. The different car models may be subject to clustering, classification, outlier detection, or other supervised or unsupervised data mining tasks, using a similarity measure which is built upon the skyline.

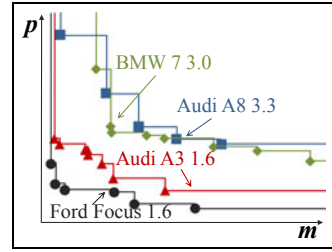


Fig. 1. Skylines of different car models

The remainder is organized as follows. We review related work in Section 2. Our novel distance measure for skylines is described in Section 3. We present an experimental evaluation in Section 4 and conclude the paper in Section 5.

The remainder is organized as follows. We review related work in Section 2. Our novel distance measure for skylines is described in Section 3. We present an experimental evaluation in Section 4 and conclude the paper in Section 5.

2 Related Work

Besides introducing studies on skyline computation we briefly survey clustering algorithms which are applied to demonstrate the potential of data mining on skylines for knowledge discovery in Section 4.

Skyline computation. Many different methods for skyline computation have been emerged in recent years. Brzsznyi *et al.* [1] proposed a SQL syntax for skyline queries and developed the Block-Nested-Loops (BNL) algorithm and the Extended Divide-Conquer algorithm. The Sort-Filter-Skyline algorithm by Chomicki *et al.* [6] improves BNL by pre-sorting the entire dataset. Tan *et al.* [2] presented Bitmap and Index. Kossmann *et al.* [3] developed a nearest neighbor search technique by browsing the data set indexed by an *R*-tree.

Clustering. In iterative partitioning clustering *k*-medoid methods such as PAM [7] or CLARANS [8] aim at finding a set of *k* representatives among all objects that characterize the objects in the data set best. Clusters are created by assigning each object to the cluster of the medoid that is closest to that object. One of the most wide spread approaches to hierarchical clustering is the Single Link algorithm [9]. Starting with singleton clusters for each object, the algorithm merges in each step the closest pair of clusters until it ends up with the root which is one large cluster containing all objects. The hierarchy obtained by the merging order is visualized as a tree which is called dendrogram. In density-based clustering, clusters are regarded as areas of high object density which are separated by areas of lower object density. The algorithm DBSCAN [10] formalizes this idea by two parameters: *MinPts* specifying a number of objects and ϵ specifying a volume. An object is called core object if it has at least *MinPts* objects within its ϵ -neighborhood. DBSCAN determines a non-hierarchical, disjoint partitioning of the data set into clusters.

3 SkyDist

In this paper we describe SkyDist, a novel distance function for skyline objects. At the beginning of this section we define the essential concepts underlying SkyDist in general. Then we motivate the idea behind assigning SkyDist for 2-dimensional skylines and conclude with a generalization according n -dimensional skylines, where $n > 2$.

3.1 Theoretical Background

Consider a set of database objects, each of which is already associated with an individual skyline. For instance, each car type is associated with the skyline of the offers posted in a used car market. An effective distance measure for a pair of skyline objects should be useful in the sense that the characteristic properties of the skyline concept are suitably reflected. Whenever two skylines are similar in an intuitive sense, then SkyDist should yield a small value. In order to define such a reasonable distance measure, we recall here the central concept of the classical skyline operator, the *dominance relation* which can be built on the preferences on attributes $\mathcal{D}_1, \dots, \mathcal{D}_n$ in a n -dimensional numeric space \mathcal{D} .

Definition 1 (Dominance). For two data points u and v , u is said to **dominate** v , denoted by $u \prec v$, if the following two conditions hold:

\forall dimensions $\mathcal{D}_{i \in \{1, \dots, n\}}$: $u.\mathcal{D}_i \leq v.\mathcal{D}_i$

\exists dimension $\mathcal{D}_{j \in \{1, \dots, n\}}$: $u.\mathcal{D}_j < v.\mathcal{D}_j$.

Definition 2 (Skyline Point / Skyline). Given a set of data points DB , a data point u is a **skyline point** if there exists no other data point $v \in DB$ such that v dominates u . The **skyline** on DB is the set of all skyline points.

Two skylines are obviously *equal* when they consist of identical points. Intuitively, one can say that two skylines are similar, whenever they consist of similar points. But in addition, two skylines may also be considered similar if they dominate approximately the same points in the data space. This can be grasped in a simple and efficient way by requiring that the one of the two skylines should not change much whenever the points of the other skyline are inserted into the first one and vice versa. This leads us to the idea to base SkyDist on the set-theoretic difference among the parts of the data space which are dominated by the two skylines. For a more formal view let us define the terms *dominance region* and *non-dominance region* of a skyline.

Definition 3 (Dominance Region of a Skyline Point). Given a skyline point $x_i \in \{1, \dots, n\}$ of a skyline $X = \{x_1, \dots, x_n\}$. The **dominance region** of x_i , denoted by DOM_{x_i} , is the data space, where every data point $u \in DOM_{x_i}$ complies the condition $x_i \prec u$.

Definition 4 (Dominance Region of a Skyline). Given the set of all skyline points of a skyline X . The **dominance region of the skyline** X , denoted by DOM_X , is defined over the union of the dominance regions of all skyline points $x_i \in \{1, \dots, n\}$.

Figure 2 illustrates this notion for two given skylines X and Y . The green and blue areas show the dominance regions of skylines X and Y , respectively.

Definition 5 (Non-Dominance Region of a Skyline). Given a numeric space $\mathcal{D} = (\mathcal{D}_1, \dots, \mathcal{D}_n)$ and the dominance region DOM_X of a skyline X . The **non-dominance region of X** , denoted by \overline{DOM}_X , is $\mathcal{D} \setminus DOM_X$.

The basic idea behind SkyDist is to determine the data space that is represented by all possible data points that are located in the dominance region of one skyline and, at the same time, the non-dominance region of the other skyline. More formally we can say that SkyDist is the volume of the distance area between two skylines which can be specified by the following equation.

$$SkyDist(X, Y) = Vol((DOM_X \setminus DOM_Y) \cup (DOM_Y \setminus DOM_X)) \quad (1)$$

In order to determine the value of the distance of two skylines X and Y based on the concept described above, we have to limit the corresponding regions in each dimension. Therefore we introduce the notion of a bounding skyline point.

Definition 6 (Bounding Skyline Point). Given a skyline X in a n -dimensional numeric space $\mathcal{D} = (\mathcal{D}_1, \dots, \mathcal{D}_n)$, where $x_i \in [0, 1]$. The **bounding skyline point of skyline X in dimension i** , denoted by x_{Bound_i} , is defined as follows.

$$x_{Bound_i} \cdot \mathcal{D}_j = \begin{cases} 1, & \text{if } j = i \\ 0, & \text{otherwise} \end{cases}$$

The bounding skyline points for two 2-dimensional skyline objects X and Y are marked in Figure 2 in red color. We remark that this concept is also applicable if the domain of one or more dimensions is not bounded. In this case the affected dimensions are bounded by the highest value that occurs in either skyline object X or Y in the according dimension. The coordinates of the remaining skyline points are then scaled respectively.

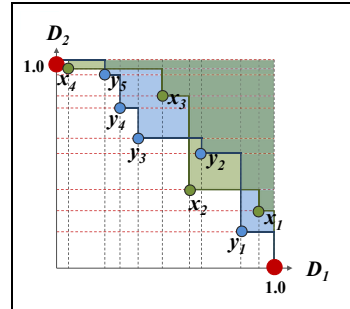


Fig. 2. Dominance regions of two skylines X and Y

3.2 SkyDist by Monte-Carlo Sampling

First we want to give an approximation of the distance of two skylines (cf. Equation 1) by a Monte-Carlo Sampling approach (MCSkyDist). SkyDist is approximated by randomly sampling points and computing the ratio between samples that fall into the region defined by Equation 1 and the ones that do not. Let us consider Figure 3(a). The region marked in red illustrates the region underlying SkyDist of two Skylines X and Y . This region is the dominance region of skyline X and simultaneously the non-dominance region of skyline Y , thus the distance of skyline X to Y . A user defined number of points is randomly sampled and the amount of samples that fall into the SkyDist region is determined. The ratio between the samples located in the SkyDist region and the ones that do not give an approximation of the distance of the two skylines. We use this technique in our experiments as a baseline for comparing a more sophisticated approach.

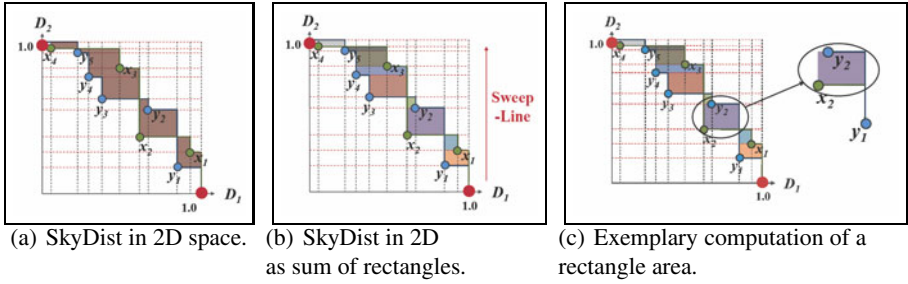


Fig. 3. SkyDist Computation in 2-dimensional space

3.3 SkyDist for 2-Dimensional Skylines

Now we describe the method of computing the exact distance of two skylines X and Y in 2-dimensional space. Let the skyline points of both skylines X and Y be ordered by one dimension, e.g. \mathcal{D}_2 . Note that the dominance region of a skyline X is composed by the dominance regions of all of its skyline points. For the computation of SkyDist, we consider only the dominance regions that are exclusive for each skyline point. Meaning that when we look at a particular skyline point x_i , we assign the dominance region of x_i and discard all dominance regions of skyline points x_j , where $x_j \cdot \mathcal{D}_2 > x_i \cdot \mathcal{D}_2$.

Figure 3(a) illustrates in red the region underlying SkyDist according to skyline objects X and Y . This region can be considered as a sum of rectangles as illustrated in Figure 3(b). To calculate the region between the skylines X and Y and thus their distance we use the concept of a sweep-line approach. For this purpose we store the skyline points of both skylines X and Y in a heap structure called event point schedule (EPS), ordered by one of the two dimensions (e.g. \mathcal{D}_2) ascending. The general idea behind the sweep-line algorithm is that a line traverses across the plane, stopping at every point that is stored in the EPS. In our example, the sweep line moves along the axis of \mathcal{D}_2 . Due to the ordering of the skyline points in the EPS we can determine the area of the rectangle at every stop of the sweep-line and calculate SkyDist in an incremental way. Figure 3(c) demonstrates for example the calculation process when the sweep-line holds on the skyline point y_2 . Now we can calculate the area of the rectangle horizontal limited by the skyline points y_2 and x_2 as $(x_2 \cdot \mathcal{D}_1 - y_1 \cdot \mathcal{D}_1)(y_2 \cdot \mathcal{D}_2 - x_2 \cdot \mathcal{D}_2)$.

3.4 A Sweep-Plane Approach for the High-Dimensional Case

In this section we describe how to exactly determine the SkyDist between skylines based on a sweep-plane paradigm referred to as SPSkyDist. We consider a d -dimensional skyline as a sequence of skylines with dimensionality $(d - 1)$ (see Figure 4): Here, we use \mathcal{D}_3 (in decreasing order) as the ordering dimension and build a sequence of 2-dimensional skylines (each in the $\mathcal{D}_1/\mathcal{D}_2$ -plane).

Traversal. The event point schedule (EPS) contains all points, ordered by the last coordinate (decreasingly). In each step of the sweeping plane, we want to obtain a valid skyline in the space spanned by the remaining coordinates. This sub-skyline is stored in

the sweep-plane status (SPS). In Figure 4, this refers to the four sub-skylines X_1, \dots, X_4 . More precisely, in each step of the sweep-plane, this $(d - 1)$ -dimensional sub-skyline is updated by (1) projecting the current point of the EPS into the $(d - 1)$ -dimensional space, (2) inserting the projected point into the skyline in the SPS, (3) deleting those points from the skyline in the SPS which are dominated by the new point in the $(d - 1)$ -dimensional space, and (4) calling the traversal-algorithm recursively for the obtained $(d - 1)$ -dimensional skyline. In our example, the EPS has the order (A, B, C, D, E) . We start with an empty SPS, and at the first stopping point, the $\mathcal{D}_1/\mathcal{D}_2$ -projection of A is inserted into the SPS to obtain X_1 . No point is dominated. Hence, we call the traversal algorithm for the obtained 2-dimensional skyline (A) . At the next stop, the projection of B is inserted. Since the projection of B dominates the projection of A (in our figure this fact is symbolized by the canceled-out copy of A), X_2 contains only point B . After the recursive call, in the next stop, the projection of C is inserted which does not dominate object B in the skyline, and, therefore, the next skyline $X_3 = (B, C)$. Finally, D and E are inserted into the skyline in the SPS (which can be done in one single stop of the sweep-plane or in two separate stops in any arbitrary order), to obtain $X_4 = (D, C, E)$, since B is dominated by D in the $(d - 1)$ -dimensional projection.

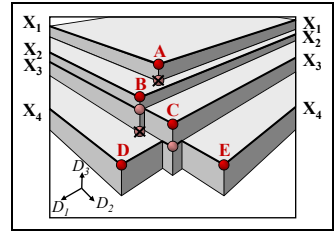


Fig. 4. A sequence of 2-dimensional skylines, forming a 3-dimensional skyline

Simultaneous Traversal for SkyDist. The computation of $\text{SkyDist}(X, Y)$ requires a simultaneous traversal of the two skylines X and Y to be compared. That means that the EPS contains the points of both X and Y , simultaneously ordered by the last coordinate. Each of the stops of the sweep-plane corresponds either to a point of X or a point of Y , and the corresponding sub-skyline in the SPS must be updated accordingly, as defined in the last paragraph by (1) projecting the point, (2) inserting the point in the corresponding sub-skyline for either X_i or Y_i , (3) cancelling out the dominated points of the sub-skyline, and finally making the recursive call for $(d - 1)$. Having developed this algorithmic template, we can easily obtain $\text{SkyDist}(X, Y)$, because each stop of the sweep-plane defines a disklet, the thickness of which is given by the difference between the last and the current event point (taking only the coordinate which is used to order the EPS). This thickness must be multiplied with the $(d - 1)$ -dimensional volume which is returned by the recursive call that computes SkyDist of the $(d - 1)$ -dimensional sub-skylines X_i and Y_i . All volumes of all disklets of the obtained sub-skylines have to be added. This works no matter whether current and the previous event points belong to the same or different skylines. Also in the case of tie situations where both skylines have identical values in the ordering coordinates or where some points in the same skyline have identical values, our algorithm works correctly. In this case, some disklets with thickness 0 are added to the overall volume, and, therefore, the order in which the corresponding sub-skylines are updated, does not change anything.

4 Experimental Evaluation

We present an extensive experimental evaluation on synthetic and real world data. We demonstrate that SkyDist is highly effective and efficient and that data mining on skylines provides novel insights in various application domains. For skyline construction, the approach of [11] is applied.

All experiments are performed on an Asus Z92J which is equipped with an Intel Dual-Core T2050 Processor and has 2 GByte of RAM.

4.1 Efficiency

To evaluate the stability of the baseline approach MCSkyDist and the scalability of SkyDist, we generate synthetic data of various number of objects and dimensions. Unless otherwise specified, the skyline is constructed from 1,000 uniformly distributed 2-dimensional data objects and for MCSkyDist we use a sample rate of 1,000.

Accuracy of MCSkyDist w.r.t. Sample Rate. We vary the sample rate in a range of 1 to 50,000 and quantify the accuracy of SkyDist in each run. These experiments indicate that the accuracy of MCSkyDist is very robust w.r.t. the number of samples. Its results achieve a constant value even with a small sample rate. Actually with 1,000 samples the same result as using SPSkyDist can be achieved.

Runtime w.r.t. Number of Objects and Dimensionality

We use data sets with varying number of points and dimensionality and generate skylines for each data set. In most real world applications, we are interested in the skyline w.r.t a few selected dimensions. Hence, in this experiments, we focus on skylines up to dimensionality $d = 4$. All results are summarized in Table 1. In the 2-dimensional case the runtime of SkyDist remains constant even with increasing data size. This is due to the fact, that despite increasing database size the number of skyline points remains relatively constant. It has been shown in [5] that for independent distributed data the number of skyline objects is $O(\log_2 n)$.

Also in the 3- and 4-dimensional case it is evident that considering the skyline points instead of all data points is very efficient. It takes 78 and 266 ms for SPSkyDist and MCSkyDist respectively to return the result when comparing two skylines X and Y each determined out of 10,000 data points. MCSkyDist and SPSkyDist both scale linear with increasing dimensionality. The sweep-plane approach outperforms the baseline by a factor of two which confirms the effectivity and scalability of an exact computation of SkyDist even for large data sets with more than two dimensions.

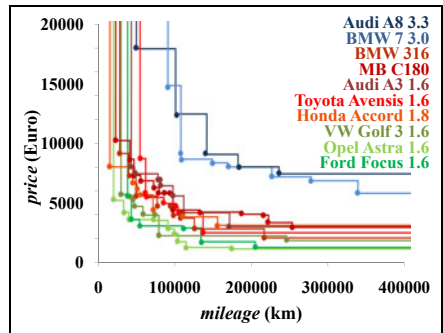


Fig. 5. Clustering of car models represented by their skyline

Table 1. Runtime analysis for SPSkyDist and MCSkyDist

| | # data points | # skyline points of skyline X | # skyline points of skyline Y | SPSkyDist | MCSkyDist |
|---------|---------------|------------------------------------|------------------------------------|-----------|-----------|
| 2D data | 10 | 4 | 3 | 15 ms | 47 ms |
| | 100 | 5 | 6 | 16 ms | 47 ms |
| | 1000 | 9 | 5 | 15 ms | 63 ms |
| | 10000 | 7 | 12 | 15 ms | 78 ms |
| 3D data | 10 | 5 | 5 | 31 ms | 62 ms |
| | 100 | 18 | 16 | 47 ms | 109 ms |
| | 1000 | 29 | 19 | 63 ms | 125 ms |
| | 10000 | 68 | 39 | 78 ms | 266 ms |
| 4D data | 10 | 5 | 8 | 47 ms | 78 ms |
| | 100 | 25 | 36 | 172 ms | 141 ms |
| | 1000 | 80 | 61 | 203 ms | 297 ms |
| | 10000 | 187 | 151 | 609 ms | 1078 ms |

4.2 Clustering Skylines of Real World Data

In addition to synthetic data we used real world data to demonstrate the potential of SkyDist for data mining. We demonstrate that interesting reasonable knowledge can be obtained by clustering skylines with SkyDist. In particular, we focus on two case studies from different applications and we apply three different clustering algorithms (PAM, Single Link and DBSCAN) with SkyDist.

Case Study 1: Automotive Market. The data used in this experiment is obtained from the online automotive market place (<http://www.autoscout24.de>). The resulting data set comprises in total 1,519 used cars constructed in the year 2000. Thus, each data point represents a specific offer of a used car which are labeled to one of three classes *compact*, *medium-sized* and *luxury*, respectively. This information allows for an evaluation of the results. Each car model is represented by one 2-dimensional skyline using the attributes *mileage* and *price*. Hence each skyline point represents a specific offer that provides an outstanding combination of these attributes, and therefore it is interesting for the user. PAM, DBSCAN and Single Link combined with SkyDist create an identical clustering result (cf. Figure 5) using the following parameterization: PAM ($K = 3$), DBSCAN ($\epsilon = 10$, $MinPts = 2$) and the dendrogram Single Link with a vertical cut at $maxDistance = 90$.

All algorithms produce 100% class-pure clusters w.r.t the labelling provided by the online market place. As expected the Audi A8 and BMW 7 of class *luxury* are clustered together in the blue cluster with a very low distance. Also the Mercedes Benz C180 (MB C180) and the Audi A3 belong to a common cluster (marked in red) and show a larger distance to the Toyota Avensis or the Honda Accord. These two car models are clustered together in the green cluster, whose members usually have a cheaper price. The clustering result with SkyDist is very informative for a user interested in outstanding combinations of *mileage* and *price* but not fixed on a specific car model. By clustering, groups of models with similar skylines become evident.

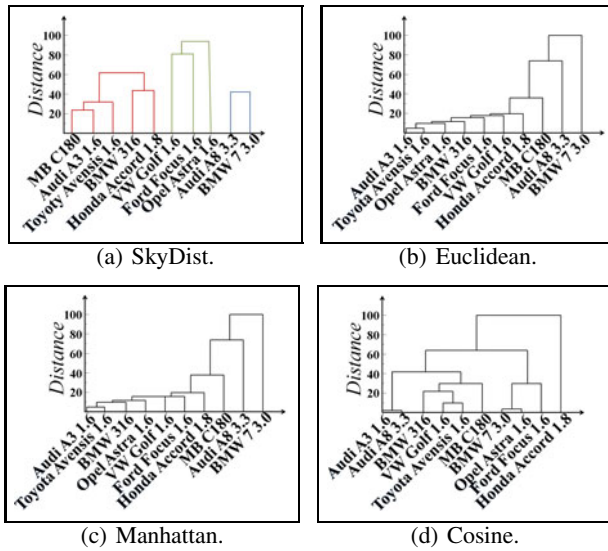


Fig. 6. The dendrogram of Single Link using SkyDist in comparison to conventional metrics

SkyDist vs. Conventional Metrics. Conventional metrics can in principle be applied for clustering skylines. For this purpose we represent each car model as a vector by calculating the average of the skyline points of the respective car model in each dimension. Then we cluster the resulting vectors with Single Link using the Euclidean, Manhattan and Cosine distance. In contrast to clustering skylines using SkyDist (cf. Figure 6(a)), Figures 6(b), 6(c) and 6(d) demonstrates that no clear clusters are identifiable in the dendrogram and the result is not very comprehensible. In Figure 6(b) and 6(c) it can easily be seen that Euclidean and Manhattan distance lead to similar results. Both show the so called Single Link effect, where no clear clusters can be identified. Using the Cosine distance avoids this effect but does not produce meaningful clusters either. For example, the luxury car model BMW 7 has minimum distance to the Opel Astra of class *compact* but has an unexpected high distance to the luxury Audi A8.

Case Study 2: Performance Statistics of Basketball Players. The NBA game-by-game technical statistics are available via <http://www.NBA.com>. We focus on the years 1991 to 2005. To facilitate demonstration and interpretation, we select players who have played at minimum 500 games and are noted for their skills and got various awards. The players are labeled with the three different basketball positions *guard* (G), *forward* (F) and *center* (C). The individual performance skyline of each player represents the number of assists and points. We cluster the skylines using SkyDist. Single Link with a vertical cut of the dendrogram at $maxDistance = 96$ and DBSCAN ($\epsilon = 4$, $MinPts = 2$) result in the same clustering. Figure 7(b) shows that the players cluster very well in three clusters that refer to the labels G, F and C. With PAM ($K = 3$) (cf. Figure 7(a)) only Steve Nash (G) clusters into the red *forward* cluster. This can be explained by the fact, that this player has performance statistics as a player in the position forward concerning number of points and assists.

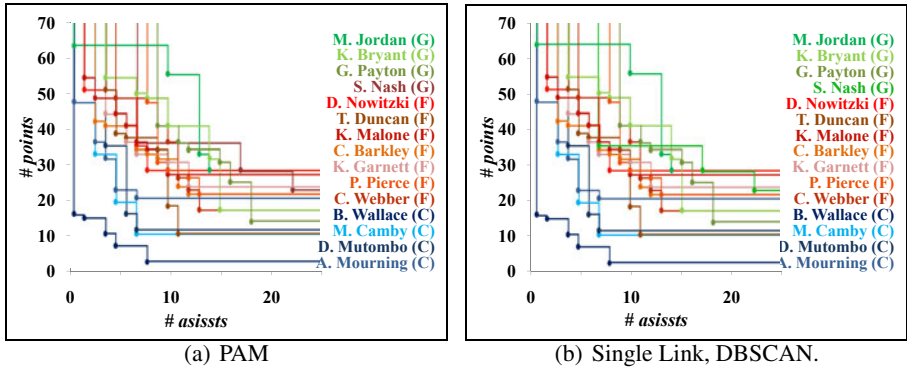


Fig. 7. Clustering of NBA Players represented as skylines

5 Conclusions

This is the first approach to data mining on skyline objects. Inspired by the success of the skyline operator for multi-criteria decision making, we demonstrated that the skyline of a data set is a very useful representation capturing the most interesting characteristics. Hence, data mining on skylines is very promising for knowledge discovery. As an essential building block, we proposed SkyDist, which is a novel distance function for skylines. We presented a baseline approach that approximates the distance and an exact plane sweep based computation. SkyDist can easily be integrated into many data mining techniques. Real world case studies on clustering skylines demonstrated that data mining on skylines enabled by SkyDist yields interesting novel knowledge. Moreover, SkyDist is efficient and thus scalable to large data sets.

References

1. Börzsönyi, S., Kossman, D., Stocker, K.: The skyline operator. In: ICDE, pp. 421–430 (2001)
2. Tan, K.L., Eng, P.K., Ooi, B.C.: Efficient progressive skyline computation. In: VLDB (2001)
3. Kossman, D., Ramsak, F., Rost, S.: Shooting stars in the sky: An online algorithm for skyline queries. In: VLDB, pp. 275–286 (2002)
4. Papadias, D., Tao, Y., Fu, G., Seeger, B.: An optimal and progressive algorithm for skyline queries. In: SIGMOD Conference, pp. 467–478 (2003)
5. Lin, X., Yuan, Y., Wang, W., Lu, H.: Stabbing the sky: Efficient skyline computation over sliding windows. In: ICDE, pp. 502–513 (2005)
6. Chomicki, J., Godfrey, P., Gryz, J., Liang, D.: Skyline with presorting: Theory and optimizations. In: Intelligent Information Systems, pp. 595–604 (2005)
7. Kaufman, L., Rousseeuw, P.J.: Finding Groups in Data: An Introduction to Cluster Analysis. John Wiley, Chichester (1990)
8. Ng, R.T., Han, J.: Efficient and effective clustering methods for spatial data mining. In: VLDB, pp. 144–155 (1994)
9. Jain, A.K., Dubes, R.C.: Algorithms for Clustering Data. Prentice-Hall, Englewood Cliffs (1988)
10. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: KDD, pp. 226–231 (1996)

Multi-Source Skyline Queries Processing in Multi-Dimensional Space*

Cuiping Li^{1,2}, Wenlin He¹, and Hong Chen^{1,2}

¹ Key Labs of Data and Knowledge Engineering, MOE, China 100872

² Information School, Renmin University of China, Beijing, China 100872
{licp, hwenlin, chong}@ruc.edu.cn

Abstract. This paper studies the problem of optimizing skyline queries with respect to multiple sources in the multidimensional space (MDMS skyline). It is challenging to process such kinds of queries efficiently due to the difficulties arising from both multi-source preferences and multi-dimensional analysis. We propose a new query evaluation model, called BitStructure, to answer MDMS skyline queries efficiently. Based on the BitStructure, we develop efficient query algorithms. The main intuition and novelty behind our approaches is that we exploit the unified BitStructure structure to seamlessly integrate multi-dimensional selection and multi-source skyline analysis. Our experimental evaluation using various synthetic datasets demonstrates that the proposed algorithms are efficient and scalable.

Keywords: Multi-Source, Skyline Query Processing, Multi-Dimension.

1 Introduction

There has been fruitful research work on efficiently processing skyline queries [2,4,6,12,11,8] in database systems. However, current methods have only considered so-called min/max attributes like price and quality which a user wants to minimize or maximize. Actually, in addition to the min/max attributes, objects can also have spatial attributes like x , y coordinates that can be used to represent relevant constraints on the query results. Paper [7] distinguished these two types of attributes (attributes such as *quality* and *price* are called **min/max** attributes and attributes such as x and y are called **spatial** attributes) and proposed a new family of query types, neighborhood dominant queries (NHDQs), which consider the relationship between min/max and spatial attributes. One limitation of the paper is that they only consider to process queries with respect to one reference point. However, there are many applications that demand to consider multiple reference points at the same time. Moreover, in most of scenarios, the desired decisions are described with some qualifying boolean constraints, which specify what subsets of data should be considered valid.

This paper studies the problem of MDMS skyline query (Multi-Dimensional Multi-Source skyline query), which optimizes skyline query with respect to multiple sources

* The work was supported by China National 863 grant 2008AA01Z120, NSFC grants 60673138 and 60603046, MOE New Century Talent Support Plan, and MingDe Scholar Plan of Renmin University of China.

in the multi-dimensional space. We propose a new query evaluation model, called *BitStructure*, to answer MDMS skyline queries efficiently. We store compressed bit vector in the structure to capture our goal of responding to multi-source and multi-dimensional analysis. Based on the *BitStructure*, we develop efficient query algorithms.

The rest of the paper is organized as follows. We give the problem statement and present challenges for processing it in Section 2. In Section 3, we present an overview of our solution, introduce the *BitStructure* data structure, and provide our query processing framework based on *BitStructure*. The experimental results are discussed in section 4. Section 5 surveys the related work and Section 6 concludes the paper.

2 Problem Statement

Let B be a set of *boolean* dimensions, D be a set of *min/max* dimensions, and S be a set of *spatial* dimensions of the dataset P . The three sets of dimensions are not necessarily exclusive. To ensure easier discussion later on in the paper, a min/max or spatial dimension is also called a preference dimension, and a tuple is also called a point. For the min/max attributes, we assume, without loss of generality, that users prefer *minimal* values.

Definition 1. Given a point $p \in P$, and a set of query points $Q = \{q_1, q_2, \dots, q_n\}$ in the space of S . The **multi-source distance** between p and Q is defined as:

$$MSDist(p, Q) = \sum_{q \in Q} dist(p, q)$$

where $dist(p, q)$ means the Euclidean distance between points p and q .

Definition 2. Given two points $p \in P$, $p' \in P$, and a set of query points $Q = \{q_1, q_2, \dots, q_n\}$ in the space of S . Let $p[i]$ be the value of p on dimension i . we say p **dominates** p' with respect to Q if:

1. $p[i] \leq p'[i]$ for all dimensions D_i , $D_i \in D$, $1 \leq i \leq |D|$.
2. $MSDist(p, Q) \leq MSDist(p', Q)$.
3. $p[i] < p'[i]$ for at least one dimension D_i , $1 \leq i \leq |D|$ or $MSDist(p, Q) < MSDist(p', Q)$.

Problem Statement: Given a set of boolean predicates g_i over dimensions in B in which $1 \leq i \leq u$ (if not empty) and u is a positive integer, a set of query points $Q = \{q_1, q_2, \dots, q_n\}$ in the space of S , a MDMS skyline query with respect to Q returns the set *Sky* of those points in P which satisfy g_i , for all $1 \leq i \leq u$, and are not dominated by any other point of P .

A straightforward approach to address MDMS skyline query is to, (1)materialize MSDists for each point in P ; (2)evaluate the skyline query using an existing method; (3)check for the boolean conditions and return the result. However, such an approach can be very expensive due to the fact that we need to compute MSDist for all points although we only need the skyline ones.

3 Our Approach

In this section, we first specify the data structure of *BitStructure*, its block constructing and filtering, and then introduce our query processing framework based on it.

3.1 BitStructure

Given a data set P , we assume that its first $|B|$ attributes are the boolean attributes, the next $|D|$ attributes are the min/max attributes and the remaining $|S|$ attributes are the spatial attributes.

Definition 3. Given a data set P , a set of boolean attributes $B = \{B_1, \dots, B_{|B|}\}$, and a set of preference attributes $R = \{R_1, \dots, R_{|R|}\}$, a **BitStructure** consists of $|B|$ materialized selection cuboids $C^S(B_i) (i = 1, \dots, |B|)$ and $|R|$ materialized bin cuboids $bin^{R_i} (i=1, \dots, |R|)$.

The BitStructure is efficient if the data set P is small. However, when P has a huge number of points, the BitStructure would be very large. The common method to the problem is to use a compact tuple ID list as an alternative. We choose to compress the bit vectors using the WAH compression method [17] to reduce the BitStructure size instead.

3.2 Block Constructing and Filtering

We propose to generate a grid to speed up the processing of our queries. Instead of intersecting bins on all attributes and generating one unified grid, we intersect bins on min/max attributes and spatial attributes respectively and generate an asymmetrical grid: $G = \{b_1^D, \dots, b_{y^D}^D; b_1^S, \dots, b_{y^S}^S\}$, where $b_i^D (i = 1, \dots, y^D)$ is the block formed by intersecting bins over min/max attributes and $b_j^S (j = 1, \dots, y^S)$ over spatial attributes.

Note that in implementation, there is **no need** to generate a grid and materialize it. Since most of the blocks can be pruned during query processing even before they are constructed, we adopt the policy to construct blocks on-the-fly when needed. Once constructed, a block is assigned a $block_{id}$. In addition, a block b in a N attributes space can also be represented by two points $b_l = \{b_{l1}, \dots, b_{lN}\}$ and $b_u = \{b_{u1}, \dots, b_{uN}\}$ where b_{li} and b_{ui} is the lower bound and upper bound value for the block along attribute i respectively.

To perform boolean filtering on b , we only need to intersect the bit vector of b with that of corresponding selection cuboids.

3.3 Query Processing

Given a data set P , a set of boolean selections B , and a set of query points Q , our algorithm for processing MDMS skyline query consists of two phases: min/max search phase and spatial search phase. The intuition behind this two-phase query processing idea is that the intermediate result obtained in the first phase can be used to prune the search space of the second phase.

Min/max Search phase: This phase searches the min/max space and returns a subspace skyline Sky^D . The algorithm follows the branch-and-bound principle to progressively retrieve data blocks [12]. It consists three steps: pre-process, search, and evaluate (Algorithm 1). We briefly explain each step as follows. Having initialized the result set Sky^D (line 1), the pre-process step computes the first candidate block and inserts it

Algorithm 1. Min/max Searching Algorithm for MDMS skyline query

```

INPUT: Boolean selections set  $B = \{v_1, \dots, v_{|B|}\}$ ,
OUTPUT: Skyline point set  $Sky^D$ 
ALGORITHM:
    /* I. PRE-PROCESS */
01:  $Sky^D = \{\}$ ;
02: Compute the first candidate block and insert it into heap;
    /* II. SEARCH */
03: while exist blocks in heap do
04:     Extract a block  $b$  from heap;
05:     if DominatePrune( $b$ ) then
06:         continue;
07:     else
08:          $tempset = BooleanFilter(b)$ ;
09:         Insert neighboring blocks of  $b$  into heap;
    /* III. EVALUATE */
10:     if  $tempset$  is not empty then
11:          $Sky' = ComputeBlockSkyline(tempset)$ ;
12:          $Sky^D = Sky^D \cup Sky'$ ;
13: Output  $Sky^D$ ;

```

into the heap (line 2). The first candidate block corresponds to the block which has the smallest MinDist to the origin point o in the min/max space.

The search step finds the next candidate block for query processing. The blocks in the heap are ordered in ascending order of MinDist to the original point o . Initially, the heap only contains the first candidate block found in the pre-process step. Iteratively, the algorithm picks the top block b of the heap as the next candidate block and removes it from the heap (line 4). b is examined by the procedure *DominatePrune()* against the already computed skyline points (line 5). If the lower bound of block b in the min/max space is dominated by some point $s \in Sky^D$, all points in b can be dominated by s and b can be safely pruned.

Line 8 checks whether b is pruned by the boolean predicate. As discussed in Section 3.2, the only operation we need to do here is to intersect the bit vector of boolean condition and that of the block b . A temporal variable $tempset$ is used to record those points that pass the dominate and boolean checking. The algorithm further retrieves all neighboring blocks of candidate block b and inserts them into the heap (line 9). Since each block can be neighboring with multiple blocks, a hash-table is maintained to ensure each block will only be inserted once.

Given the $tempset$ computed in the search step, procedure *ComputeBlockSkyline()* in the evaluate step performs the task of computing the skyline for all points in $tempset$. If $tempset$ is not empty, the algorithm first performs similar dominate pruning on all points in it against the already computed skyline points, and then does pair-wise Comparison to compute the skyline Sky' . Sky' is further merged into the final result Sky^D (line 12). The algorithm terminates when there is no more blocks in the heap.

Lemma 1. *Given any point $p \in Sky^D$, if p is non-duplicate in subspace D , p is a MDMS skyline point in the full space $(B \cup D \cup S)$.*

PROOF: Omitted for space limitation. □

Based on the lemma, we know that any non-duplicate skyline point in subspace D is a MDMS skyline point in the full space. For those duplicate points in Sky^D , we further compare their MSDist to Q and then delete those points who have larger MSDists. From now on, we assume Sky^D only including non-duplicate points.

Spatial Search Phase: This phase searches the spatial space and computes the final MDMS skyline Sky . Given the intermediate result Sky^D returned in the first phase, we define two bound variables $dist_{lb}$ and $dist_{ub}$ as the minimal and maximal MSDist over all points in Sky^D . Based on this, we have the following lemmas.

Lemma 2. *Any block b can not contain a MDMS skyline point if:*

$$MinMinDist(b, R^Q) > \frac{dist_{ub}}{|Q|}$$

where R^Q represents the minimum bounding rectangle (MBR) formed by points in Q .

PROOF: Omitted for space limitation. □

According to lemma 2, if the minimal distance between block b and R^Q is greater than $\frac{dist_{ub}}{|Q|}$, then b can be safely pruned. Lemma 2 incurs minimum overhead, since for every block it requires a single distance computation. However, as mentioned in [11], the condition is not very tight. It leads to unnecessary block access. For example, assuming $dist_{ub} = 7$, since $MinMinDist(b, R^Q) = 6 < dist_{ub} = 7$, according to lemma 2, block b in Fig. ?? needs to be visited although it cannot contain MDMS skyline points. We presents a tighter bound to avoid such visits.

Lemma 3. *Any block b can not contain a MDMS skyline point if:*

$$\sum_{q \in Q} MinDist(b, q) > dist_{ub}.$$

PROOF: Straightforward. Omitted. □

We will now explain how to employ lemma 2 and 3 to prune the spatial space, and how to get the final MDMS skyline. The pseudo code of Algorithm 2 is used to perform this task. The main differences between Algorithm 2 and Algorithm 1 lie in: (1)how to locate the first candidate block; (2)how to order candidate blocks in the heap; (3)how to do the dominant pruning; (4)how to compute the block skyline.

At the very beginning of Algorithm 2, we set the values for two bound variables, $dist_{lb}$ and $dist_{ub}$. At the same time, we initialize another two variables Sky_1^S and Sky_2^S to be empty. Sky_1^S and Sky_2^S are used for computing the block skyline and will be explained later.

Recall in min/max space, the first candidate block is corresponding to the one that is closest to the origin. In contrast, in spatial space, the block who has the most powerful pruning ability is the one that is nearest to Q . Therefore, we locate the first candidate block to be that whose centroid is nearest to all points in Q . For the same reason, in the heap, candidate blocks with nearer centroids to Q are ordered before blocks with farther centroids.

Algorithm 2. Spatial Searching Algorithm for MDMS skyline queryINPUT: Sky^D , B , Q

OUTPUT: Skyline point set

ALGORITHM:

- 01: Let $dist_{lb}$ be the minimal MSDist over all points in Sky^D ;
 Let $dist_{ub}$ be the maximal MSDist over all points in Sky^D ;
 $Sky_1^S = Sky_2^S = \emptyset$;
 02-11: The same as in Algorithm 1
 12: Merge Sky_1^S , Sky_2^S , Sky^D , and Output;

Procedure DominatePrune(b)

- 13: **if** $MinMinDist(b, R^Q) > \frac{dist_{ub}}{|Q|}$ **then**
 14: Return 1; /* LEMMA 2 */
 15: **else if** $\sum_{q \in Q} MinDist(b, q) > dist_{ub}$ **then**
 16: Return 1; /* LEMMA 3 */
 17: Return 0;

Procedure ComputeBlockSkyline($tempset$)

- 18: **for** each point $p \in tempset$ **do**
 19: **if** $p.MSDist < dist_{lb}$ **then**
 20: **if** p is not dominated by Sky_1^S **then**
 21: Add p to Sky_1^S
 Remove any dominated points from $Sky_1^S \cup Sky_2^S$;
 22: **else**
 23: **if** p is not dominated by $Sky^D \cup Sky_2^S$ **then**
 24: Add p to Sky_2^S
 Remove any dominated points from $Sky_1^S \cup Sky_2^S$;

Now we explain how the procedure *DominatePrune()* works in the spatial space. If lemma 2 is satisfied (line 13-14), *DominatePrune()* returns value 1 which indicates that block b can be pruned safely. Otherwise, we need further check if lemma 3 is satisfied (15-16). Please note that once a block b is pruned, we need not consider the neighboring blocks of b any more. The reason is, for those neighboring blocks of b that are farther to Q than b is, they must satisfy the pruning condition also and can be pruned safely; for those neighboring blocks of b that are nearer to Q than b is, they must have been visited before since our block visiting order is from near to far.

If both lemma 2 and 3 fail, *DominatePrune()* returns value 0 which means that b cannot be pruned and need to be processed further. Procedure *ComputeBlockSkyline()* is called to process those blocks that pass the dominate pruning and boolean filtering. For each point $p \in b$, it can either be dominated by at least one point in Sky , and therefore fails to affect Sky , or can not be dominated by *any point* in Sky and should become part of Sky itself. In that case, p might also dominate points currently in Sky , which must of course be removed. Therefore, Procedure *ComputeBlockSkyline()* in the spatial space must efficiently support two operations: first, checking whether a point p is dominated by the current Sky ; second, removing the points in Sky that are dominated

by p . The ability of any technique to perform these two tasks can be augmented by utilizing the following observations.

Lemma 4. Given any point $p \in P$, if:

$$MSDist(p, Q) < dist_{lb},$$

then, p can not be dominated by any point $p' \in P$ if p' satisfies $MSDist(p', Q) \geq dist_{lb}$.

PROOF: Straightforward. Omitted. \square

Lemma 5. Given any point $p \in P$, p cannot dominate any point $p' \in Sky^D$.

PROOF: Straightforward. Omitted. \square

These lemmas imply that a number of points in Sky is irrelevant for the checking tasks. We divide all points in Sky into three separate subsets: Sky^D , Sky_1^S , and Sky_2^S . The last two subsets maintain skyline points generated in the spatial search phase; Sky_1^S represents those points whose MSDists are smaller than $dist_{lb}$, and Sky_2^S represents those points whose MSDists are equal to or larger than $dist_{lb}$. Thus, to do dominant checking, we only need to consider points in the corresponding subsets instead of the entire Sky . line 18-24 of Algorithm 2 summarizes the high level strategy that is employed to keep Sky_1^S and Sky_2^S up to date. Please note that Sky^D is only used for pruning here, and it will not change when the algorithm is proceeding.

Finally, Sky^D , Sky_1^S and Sky_2^S are merged together to form the final skyline result.

4 Performance Study

To evaluate the efficiency and scalability of our query processing algorithms, extensive experiments are conducted. In this section, we report only part of our results due to space limitation. All our experiments are conducted on a PC with an Intel Pentium IV 1.6GHz CPU and 1G main memory, running Microsoft Windows XP. Experiment results are reported on synthetic datasets. All run time reported here includes I/O time.

The default values of dimensionality is 6, data size is 100k, the number of query points in $|Q|$ is 5. For simplicity, we used the same number of boolean dimensions, min/max dimensions and spatial dimensions. As an example, for 6 dimensions, we chose 2 dimensions as boolean dimensions, 2 dimensions as min/max dimensions, and the remaining 2 dimensions as spatial dimensions.

We compare our proposed query processing algorithm based on an asymmetrical grid (referred as *ASYM_MDMS*) against the following alternative methods: (1) Straight-forward method (referred as *STR_MDMS*) implemented using the method discussed in Section 2; (2) Query processing algorithm based on a symmetrical grid (referred as *SYM_MDMS*).

We randomly generate 100 different MDMS skyline queries. Each query has a number n of points, distributed uniformly in the space of S . Figure 1(a) shows the average run time of the three algorithms for answering a MDMS skyline query when n is increased from 2 to 7. Obviously, from the results, we can see that *SYM_MDMS* outperforms *STR_MDMS*. Among the algorithms, *ASYM_MDMS* performs best as expected

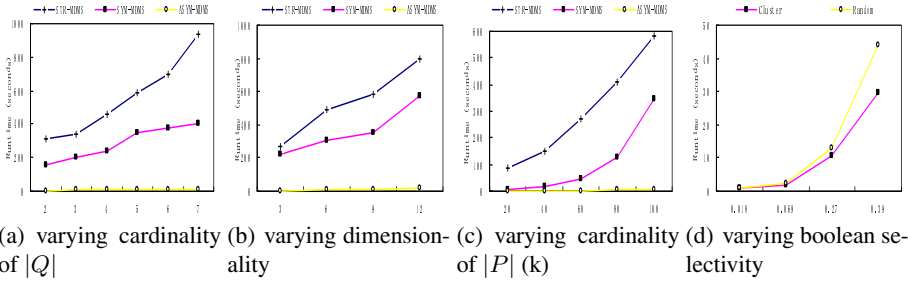


Fig. 1. Performance for MDMS Skyline Query

due to the asymmetrical partition for different types of attributes and the early pruning strategies applied in the spatial search phase.

Next, we look at the run time of three algorithms as the number of dimension increases from 3 to 12. Figure 1(b) depicts the experiment result. We observe that with increasing number of dimension, the runtime of STR_MDMS and SYM_MDMS increases more significantly than that of ASYM_MDMS. This is again due to the fact that the pruning is more effective in the ASYM_MDMS approach.

Figure 1(c) shows run time of the three algorithms as the number of points increases from 20,000 to 100,000. From the results, we can see that the run time of both STR_MDMS and SYM_MDMS algorithms increases dramatically with respect to the size of data sets. However, the ASYM_MDMS method increases modestly.

To evaluate possible impacts by different distributions of query points, we generate 100 different MDMS skyline queries with clustered query points. Figure 1(d) depicts the experiment result over two different query distributions (Cluster and Random) with different boolean selectivity. Since the trends are the same for all three algorithms, we only show the results of ASYM_MDMS as it is the most efficient. Boolean selectivity determines the filtering power by boolean predicates and is defined as the number of points that pass the boolean checking as a proportion of $|P|$. We observe that Cluster outperforms Random. The difference between them increases with selectivity. This is because cluster query distribution has high possibility to generate a small $|R^Q|$ which results in effective pruning in spatial space.

5 Conclusions

In this paper, we have introduced a novel type of skyline queries, so-called MDMS skyline queries, which optimizes skyline query with respect to multiple sources in the multi-dimensional space. Such queries support a micro-economic approach to decision making, considering not only min/max attributes but also spatial attributes. We propose a new query evaluation model, called *BitStructure*, to answer MDMS skyline queries efficiently. Based on the *BitStructure*, we develop efficient query algorithms. Our experimental evaluation demonstrates that the proposed algorithms are efficient and scalable.

References

1. Balke, W.-T., Guntzer, U., Zheng, J.X.: Efficient distributed skylining for web information systems. In: EBDT (2004)
2. Borzsonyi, S., Kossmann, D., Stocker, K.: The skyline operator. In: ICDE (2001)
3. Deng, K., Zhou, X., Shen, H.T.: Multi-source skyline query processing in road networks. In: ICDE (2007)
4. Tan, K., et al.: Efficient progressive skyline computation. In: VLDB (2001)
5. Godfrey, P., Shipley, R., Gryz, J.: Maximal vector computation in large data sets. In: VLDB (2005)
6. Kossmann, D., Ramsak, F., Rost, S.: Shooting stars in the sky: An online algorithm for skyline queries. In: VLDB (2002)
7. Li, C., Tung, A.K.H., Jin, W., Ester, M.: On dominating your neighborhood profitably. In: VLDB Conference (2007)
8. Lin, X., Yuan, Y., Wang, W., Lu, H.: Stabbing the sky: efficient skyline computation over sliding windows. In: ICDE (2005)
9. Lin, X., Yuan, Y., Zhang, Q., Zhang, Y.: Selecting stars: The k most representative skyline operator. In: ICDE (2007)
10. Morse, M., Patel, J., Jagadish, H.V.: Efficient skyline computation over low-cardinality domains. In: VLDB (2007)
11. Papadias, D., Shen, Q., Tao, Y., Mouratids, K.: Group nearest neighbor queries. In: ICDE Conference (2004)
12. Papadias, D., Tao, Y., Fu, G., Seeger, B.: An optimal and progressive algorithm for skyline queries. In: SIGMOD (2003)
13. Pei, J., Jin, W., Ester, M., Tao, Y.: Catching the best views of skyline: a semantic approach based on decisive subspaces. In: VLDB (2005)
14. Pei, J., Jiang, B., Lin, X., Yuan, Y.: Probabilistic skylines on uncertain data. In: VLDB (2007)
15. Sharifzadeh, M., Shahabi, C.: The spatial skyline queries. In: VLDB Conference (2006)
16. Wang, S., Ooi, B.C., Tung, A.K.H., Xu, L.: Efficient skyline query processing on peer-to-peer networks. In: ICDE (2007)
17. Wu, K., Otoo, E.J., Shoshani, A.: Optimizing bitmap indices with efficient compression. *ACM Transactions on Database Systems* (2006)
18. Yiu, M., Mamoulis, N., Papadias, D.: Aggregate nearest neighbor queries in road networks. In: TKDE (2005)
19. Yuan, Y., Lin, X., Liu, Q., Wang, W., Yu, J.X., Zhang, Q.: Efficient computation of the skyline cube. In: VLDB (2005)

Efficient Pattern Mining of Uncertain Data with Sampling

Toon Calders¹, Calin Garboni², and Bart Goethals²

¹ TU Eindhoven, The Netherlands
t.calders@tue.nl

² University of Antwerp, Belgium
{calin.garboni,bart.goethals}@ua.ac.be

Abstract. Mining frequent itemsets from transactional datasets is a well known problem with good algorithmic solutions. In the case of uncertain data, however, several new techniques have been proposed. Unfortunately, these proposals often suffer when a lot of items occur with many different probabilities. Here we propose an approach based on sampling by instantiating “possible worlds” of the uncertain data, on which we subsequently run optimized frequent itemset mining algorithms. As such we gain efficiency at a surprisingly low loss in accuracy. This is confirmed by a statistical and an empirical evaluation on real and synthetic data.

1 Introduction

In frequent itemset mining, the transaction dataset is typically represented as a binary matrix where each line represents a transaction and every column corresponds to an item. An element M_{ij} represents the presence or the absence of the item j in transaction i by the value 1 or 0 respectively. For this basic traditional model, where an item is either present or absent in a transaction, many algorithms have been proposed for mining frequent itemsets; i.e., sets of columns of M that have all ones in at least a given number of transactions (see e.g. [5] for an overview on frequent itemset mining).

In many applications, however, an item is not present or absent in a transaction, but rather an existence probability of being in the transaction is given. This is the case, for example, for data collected from experimental measurements or from noisy sensors. Mining frequent patterns from this kind of data is more difficult than mining from traditional transaction datasets. After all, computing the support of an itemset now has to rely on the existence probabilities of the items, which leads to an *expected support* as introduced by Chui et al. [4].

If the binary matrix is transformed into a probabilistic matrix, where each element takes values in the interval $[0, 1]$, we have the so called uncertain data model. Under the assumption of statistical independence of the items in all transactions in the dataset, the support of an itemset in this model, as defined by Chui et al. [4], is based on the possible world interpretation of uncertain data. Basically, for every item x and every transaction t there exist two sets of

possible worlds, one with the worlds in which x is present in t and one with the worlds where x is not present in t . The probability of the first set of worlds is given by the existence probability of x in t ($P(x, t)$) and the probability of the second set of worlds by $1 - P(x, t)$. The probability of a single world is then obtained by multiplying the probabilities for all its individual items; i.e., $P(W) = \sum_{t \in W} \prod_{x \in t} P(x, t) \prod_{x \notin t} (1 - P(x, t))$. The expected support of an itemset can be obtained by summing the support of that itemset over all possible worlds, while taking into consideration the probability of each world. There exist $2^{|D| \times |I|}$ worlds, where $|D|$ is the total number of transactions in the probabilistic dataset and $|I|$ is the total number of items. This rather complicated formula can be reduced to:

$$expSup(X) = \sum_{t \in D} \prod_{x \in X} P(x, t)$$

Every transaction thus supports an itemset with the probability given by the product of existence probabilities of all items in the itemset and in that transaction. The expected support of an itemset over the entire dataset is the sum of the existence probabilities of that itemset in every transaction of the dataset.

In the remainder of this paper we revisit the related work, then we present our proposed method based on sampling, followed by theoretical and empirical analysis of the quality of the results.

2 Related Work

The efficient data structures and techniques used in frequent itemset mining such as TID-lists [2], FP-tree, which adopts a prefix tree structure as used in FP-growth [6], and the hyper-linked array based structure as used in H-mine [8] can no longer be used as such directly on the uncertain data. Therefore, recent work on frequent itemset mining in uncertain data that inherits the breadth-first and depth-first approaches from traditional frequent itemset mining adapts the data structures to the probabilistic model.

U-Apriori [4] is based on a level wise algorithm and represents a baseline algorithm for mining frequent itemsets from uncertain datasets. Because of the generate and test strategy, level by level, the method does not scale well.

UCP-Apriori [3] is based on the decremental pruning technique which consists in maintaining an upper bound of the support and decrementing it while scanning the dataset. The itemset is pruned as soon as its most optimistic value falls below the threshold. This approach represents the state of the art for mining frequent patterns from uncertain data with a generate-and-prune strategy.

UF-growth [7] extends the FP-Growth algorithm [6]. It is based on a UF-tree data structure (similar to FP-tree). The difference with the construction of a FP-tree is that a transaction is merged with a child only if the same item and the same expected support exist in the transaction and in the child node, leading to a far lower compression ratio as in the original FP-tree. The improvements consist in discretization of the expected support to avoid the huge number of different

values and in adapting the idea of co-occurrence frequent itemset tree (COFI-tree). The UF-trees are built only for the first two levels. It then enumerates the frequent patterns by traversing the tree and decreasing the occurrence counts.

Aggarwal et al. [1] extended several existing classical frequent itemset mining algorithms for deterministic data sets, and compared their relative performance in terms of efficiency and memory usage. The *UH-mine* algorithm, proposed in their paper, provides the best trade-offs. The algorithm is based on the pattern growth paradigm. The main difference with UF-growth is the data structure used which is a hyperlinked array.

The limitations of these existing methods are the ones inherited from the original methods. The size of the data for the level-wise generate-and-test techniques affects their scalability and the pattern-growth techniques require a lot of memory for accommodating the dataset in the data structures, such as the FP-tree, especially when the transactions do not share many items. In the case of uncertain data, not only the items have to be shared for a better compression but also the existence probabilities, which is often not the case.

3 Sampling the Uncertain Dataset

The first method we propose, called *Concatenating the Samples*, takes the uncertain dataset and samples according to the given existential probabilities. For every transaction t and every item i in transaction t we generate an independent random number $0 \leq r \leq 1$ (coin flip) and we compare it with the probability p associated with the item i . If $p \geq r$ then item i will appear in the currently sampled transaction. For every transaction in the uncertain dataset we repeat the step above n times, for a given n . The result is a dataset which can be mined with any traditional frequent itemset mining algorithm. To obtain the estimated support of an itemset in the uncertain dataset, its support in the sampled dataset still needs to be divided by n .

The difficulty of this method resides in the fact that we physically instantiate and store the sampled “certain” dataset which can be up to n times larger than the original uncertain dataset. Fortunately, for most efficient itemset mining algorithms, we do not actually have to materialize this samples database. After all, most efficient techniques read the database from disk only once, after which their advanced data structures contain the database in the main memory. Therefore, the sample can be generated immediately in memory when the database is being read from disk for the first time. We call this method *Inline Sampling*.

To this end, we made minor modifications of the frequent itemset mining algorithms. We will briefly describe **U-Eclat** and **UFP-growth**, the modified versions of the **ECLAT** and **FP-growth** algorithms.

U-Eclat is an adaptation of the **ECLAT** algorithm [11] with an improvement based on diffsets as described in [10]. In only one scan of the dataset the relevant items are stored into memory together with the list of transactions where the items appear, called tid-list. The candidates are then generated using a depth-first search strategy and their support is computed by intersecting the tid-list of

the subsets. The only adaptation for **U-Eclat** consists in reading the uncertain transactions and instantiating them as described above. More specifically, given the number of iterations n , for every transaction t and every item i in transaction t we generate n independent random numbers r_1, \dots, r_n between 0 and 1 and we compare them with the probability p associated with the item i . If $p \geq r_j$, for $1 \leq j \leq n$, then $n \cdot t + j$ will appear in the tid-list of item i . From there on, the standard Eclat algorithm is being executed.

UFP-growth extends the initial **FP-growth** algorithm [6]. The FP-tree construction needs two scans of the dataset. The first scan collects the frequent items and their support and in the second scan every transaction is accommodated in the FP-tree structure. The frequent itemsets are then generated recursively from the FP-tree. In order to adapt this algorithm to our method, the first scan computes the expected support of every itemset exactly by computing their support as the sum of existential probabilities in every transaction where it occurs. In the second scan, every transaction is instantiated n times, according to the existential probability of the items in the transaction and then it is inserted in FP-tree structure. The algorithm then extracts the frequent itemsets the same way as the **FP-growth** algorithm.

4 Statistical Bounds on the Quality of the Approximation

As before, D denotes the set of transactions, and I the set of items. $P(x, t)$ denotes the probability assigned to item x by transaction t . We extend this notation to itemsets X ; i.e., $P(X, t)$ will denote $\prod_{x \in X} P(x, t)$. Our whole analysis will be based on the numbers $P(X, t)$ only and hence, will not depend on the assumption of independence between the items. Notice that this implies that our sampling-based method, in contrast to the other existing proposals, could also be applied when a more involved probabilistic model is assumed. We first start our analysis for a single itemset X and will extend it later on for the complete collection of itemsets.

Suppose that, for every transaction $t \in D$, we sample n deterministic versions of this tuple, t_1, \dots, t_n . Let X_t^i be the stochastic variable denoting if $X \subseteq t_i$; i.e., $X_t^i = 1$ if $X \subseteq t_i$, and $X_t^i = 0$ otherwise. Notice that the variables X_t^i are statistically independent as they are sampled using independent coin flips. X_t^i follows a Bernoulli distribution with mean $P(X, t)$. It is easy to see that the stochastic variable $X_t = \sum_{i=1}^n X_t^i$ follows a binomial distribution with mean $nP(X, t)$ and variance $nP(X, t)(1 - P(X, t))$. Consider now the sum: $S(X) := \frac{\sum_{t \in D} X_t}{n}$. The expected value and variance of this sum are as follows:

$$E[S] = expSup(X)$$

$$V[S] = V \left[\frac{\sum_{t \in D} X_t}{n} \right] = \frac{\sum_{t \in D} V[X_t]}{n^2} = \frac{\sum_{t \in D} nP(X, t)(1 - P(X, t))}{n^2} \leq \frac{|D|}{4n}.$$

Hence, not surprisingly, the sum S we use to approximate the expected support is an unbiased estimator with a variance that decreases linearly with n . For the relative version, $rS = \frac{S}{|D|}$, we get $V[rS] = V \left[\frac{S}{|D|} \right] = \frac{V[S]}{|D|^2} \leq \frac{1}{4n|D|}$.

We now apply Hoeffding’s inequality. This inequality is as follows: given independent (but not necessarily identically distributed) stochastic variables X_1, \dots, X_m such that for all $i = 1 \dots n$, $P(a_i \leq X_i - E[X_i] \leq b_i) = 1$, then

$$p \left[\left| \sum_i X_i - E \left[\sum_i X_i \right] \right| \geq m\epsilon \right] \leq 2 \exp \left(- \frac{2m^2\epsilon^2}{\sum_{i=1}^n (b_i - a_i)^2} \right).$$

In our case, for all X_t^i , $X_t^i - E(X_t^i)$ is in the interval $[-1, 1]$, and hence we get:

$$\begin{aligned} p \left[\left| \sum_{t \in D} \sum_{i=1}^n X_t^i - E \left[\sum_{t \in D} \sum_{i=1}^n X_t^i \right] \right| \geq n|D|\epsilon \right] &\leq 2 \exp \left(- \frac{2(n|D|)^2\epsilon^2}{\sum_{i=1}^n |D| 2^2} \right) \\ &= 2 \exp \left(- \frac{n|D|\epsilon^2}{2} \right). \end{aligned}$$

If we now rewrite in function of $rS(X)$ and $rsupp(X) := \frac{expSup(X)}{|D|}$, we get:

$$p[|rS(X) - rsupp(X)| \geq \epsilon] \leq 2 \exp \left(- \frac{n|D|\epsilon^2}{2} \right).$$

Hence, for given $\epsilon, \delta > 0$, we have: If $\delta \geq 2 \exp \left(- \frac{n|D|\epsilon^2}{2} \right)$, i.e., $n \geq - \frac{2 \ln(\delta/2)}{|D|\epsilon^2}$, then $p[|rS(X) - rsupp(X)| \leq \epsilon] \geq 1 - \delta$.

The significance of this result can best be illustrated by an example. Suppose D contains 100 000 probabilistic transactions and X is an itemset. In order to guarantee that the support of X is approximated with 99% probability with less than 1% error, we need to have $n \geq - \frac{2 \ln(0.01/2)}{100\,000(0.01)^2} \approx 1$. Hence, we need approximately 1 sample per transaction in D to achieve this result. Furthermore, suppose that we have a collection of 1 000 000 frequent itemsets. In order to guarantee that all these itemsets have less than 1% error with 99% probability, we need to have (using the union rule) $n \geq - \frac{2 \ln(1/200\,000\,000)}{100\,000(0.01)^2} \approx 3.8$; i.e., less than 4 samples per transaction.

As a side note, even tighter bounds can be gotten by approximating the distribution of rS with a normal distribution, using a weaker form of the Central Limit Theorem, called *Lyapunov’s central limit theorem*. That is, $\frac{S - supp(X)}{nV[S]}$ converges in probability to $N(0, 1)$.

5 Experiments

The experiments were conducted on a GNU/Linux machine with a 2.1GHz CPU with 2 Gb of main memory. We used the datasets and the executables for comparison from [1]. Kosarak contains anonymized click-stream data. It is a very sparse dataset, with a density of less than 0.02%, about 1 million transactions, 42170 distinct items and an average of 8 item per transaction. The dataset T40I10D100K was generated using the IBM synthetic data generator, having

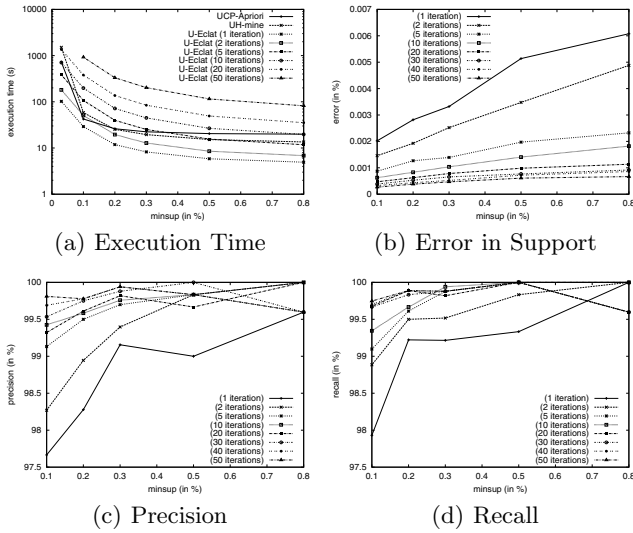


Fig. 1. kosarak Dataset

100K transactions, 942 distinct items and a density of 4.2%. The original datasets were transformed by Aggarwal et al. [1] into uncertain datasets by assigning to every item in every transaction existential probabilities according to the normal distribution $N(\mu, \sigma^2)$, where μ and σ were randomly and independently generated with values between $[0.87, 0.99]$ and $[1/21, 1/12]$ respectively.

For different values of the minimum support, we ran our implementations of **U-Eclat** and **UFP-growth**. The number of times we instantiate the uncertain dataset varies between 1 and 50. The higher the number of instantiations, the better the accuracy of the results becomes, at the cost of an increase in execution time. We also experimented with the original **ECLAT** and **FP-growth** algorithms after materializing the sampled datasets. Obviously the size of these datasets become very large for multiple iterations, and thus, those experiments always resulted in a decrease in performance as compared to their inline versions. Experimentally we show that for relatively low number of instantiations we reach highly accurate results. The gain in time motivates the use of our method which outperforms in execution time the existing state of the art methods mentioned in [1]. For every dataset, we plot the execution times we obtained for different values of the minimum support and for some different numbers of iterations. It turns out that **U-Eclat** always outperformed **UFP-growth**. In many cases, the FP-tree simply became too large to handle [5]. In the experiments, for clarity, we thus only show the results for **U-Eclat**. For a fair comparison we also only show the best performing implementations of the algorithms mentioned in [1], being UCP-Apriori and UH-mine. The execution times are depicted in Figures 1(a) and 2(a).

For low support threshold, our **U-Eclat** outperforms UCP-Apriori and UH-mine for up to 5 sampling iterations of the dataset. Note that more efficient

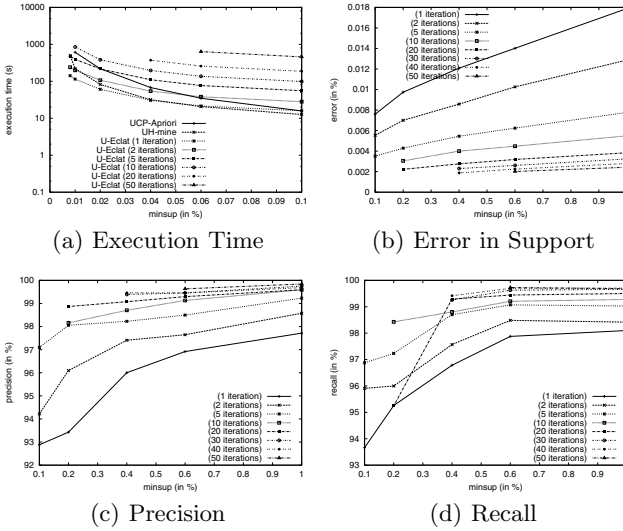


Fig. 2. t40 Dataset

| kosarak | | | | | | |
|---------|-----------|-------|-----|--------|-------|-----|
| | precision | | | recall | | |
| Iter | min | avg | max | min | avg | max |
| 1 | 97.67 | 98.95 | 100 | 97.93 | 99.28 | 100 |
| 2 | 98.27 | 99.41 | 100 | 98.88 | 99.62 | 100 |
| 5 | 99.13 | 99.63 | 100 | 99.10 | 99.70 | 100 |
| 10 | 99.43 | 99.77 | 100 | 99.34 | 99.74 | 100 |
| 20 | 99.32 | 99.74 | 100 | 99.60 | 99.83 | 100 |
| 30 | 99.53 | 99.79 | 100 | 99.60 | 99.83 | 100 |
| 40 | 99.69 | 99.87 | 100 | 99.60 | 99.84 | 100 |
| 50 | 99.60 | 99.83 | 100 | 99.75 | 99.92 | 100 |

| t40 | | | | | | |
|------|-----------|-------|-------|--------|-------|-----|
| | precision | | | recall | | |
| Iter | min | avg | max | min | avg | max |
| 1 | 92.88 | 96.16 | 100 | 93.66 | 96.95 | 100 |
| 2 | 94.22 | 97.25 | 99.54 | 95.91 | 97.73 | 100 |
| 5 | 97.10 | 98.52 | 100 | 96.88 | 98.48 | 100 |
| 10 | 98.16 | 99.12 | 100 | 98.43 | 99.14 | 100 |
| 20 | 98.87 | 99.37 | 100 | 95.25 | 98.70 | 100 |
| 30 | 99.39 | 99.65 | 100 | 99.26 | 99.64 | 100 |
| 40 | 99.46 | 99.65 | 100 | 99.42 | 99.71 | 100 |
| 50 | 99.63 | 99.82 | 100 | 99.68 | 99.80 | 100 |

Fig. 3. Summary of Precision and Recall

frequent set mining algorithms as can be found in the FIMI repository will perform even better, also for a higher number of iterations. As our theoretical results already indicated that 2 iterations already result in a very accurate approximation of the expected supports of all itemsets. This also shows in practice.

To this end, we compare the collections of frequent patterns and their support obtained using the exact method and our sampling method. First, the collection of frequent itemsets is generated using UCP-Apriori [1]. Based on this, we evaluate the errors in support computed with the sampling method.

In terms of support error, we compute the average of the absolute difference between the support of itemsets found by both methods. The error is depicted in Figures 1(b) and 2(b). It can be seen that, as expected and predicted by the statistical evaluation, the higher the number of iterations grows, the lower the error becomes. But even for relatively low number (5 or 10 iterations), the average error in support estimation drops below 1%.

For itemsets having the support close to the minimum support threshold, small variations of support can introduce false positives when the real support is overestimated or false negatives when the real support is underestimated. To evaluate the impact of this, we report Precision and Recall of our method w.r.t. the true collection in terms of patterns found as frequent. We plot in Figures 1(c), 1(d), 2(c) and 2(d) the values of precision and recall for different number of iterations. A summary of these values is reported in Figure 3 as the overall minimum, average and maximum for each dataset and different numbers of iterations. The values confirm the quality of the approximation.

Acknowledgements. We thank to the authors of Aggarwal et al. [1] for making available the executables and datasets. This research was partially funded by FWO project “Foundations for Inductive Databases”.

References

1. Aggarwal, C.C., Li, Y., Wang, J., Wang, J.: Frequent pattern mining with uncertain data. In: Proc. of KDD 2009, pp. 29–38. ACM, New York (2009)
2. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: Proc. of VLDB 1994, pp. 487–499. Morgan Kaufmann Publishers Inc, San Francisco (1994)
3. Chui, C.K., Kao, B.: A decremental approach for mining frequent itemsets from uncertain data. In: Washio, et al [9], pp. 64–75
4. Chui, C.K., Kao, B., Hung, E.: Mining frequent itemsets from uncertain data. In: Zhou, Z.-H., Li, H., Yang, Q. (eds.) PAKDD 2007. LNCS (LNAI), vol. 4426, pp. 47–58. Springer, Heidelberg (2007)
5. Goethals, B.: Frequent set mining. In: The Data Mining and Knowledge Discovery Handbook, ch. 17, pp. 377–397. Springer, Heidelberg (2005)
6. 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)
7. Leung, C.K.-S., Mateo, M.A.F., Brajczuk, D.A.: A tree-based approach for frequent pattern mining from uncertain data. In: Washio, et al [9], pp. 653–661
8. Pei, J., Han, J., Lu, H., Nishio, S., Tang, S., Yang, D.: H-mine: Hyper-structure mining of frequent patterns in large databases. In: Proc. of ICDM 2001, Washington, DC, USA, pp. 441–448. IEEE Computer Society, Los Alamitos (2001)
9. Washio, T., Suzuki, E., Ting, K.M., Inokuchi, A. (eds.): PAKDD 2008. LNCS (LNAI), vol. 5012. Springer, Heidelberg (2008)
10. Zaki, M.J., Gouda, K.: Fast vertical mining using diffsets. In: Getoor, L., Senator, T.E., Domingos, P., Faloutsos, C. (eds.) Proc. of KDD 2003, pp. 326–335. ACM, New York (2003)
11. Zaki, M.J., Parthasarathy, S., Ogihara, M., Li, W.: New algorithms for fast discovery of association rules. In: Proc. of KDD 1997, pp. 283–286 (1997)

Classifier Ensemble for Uncertain Data Stream Classification*

Shirui Pan¹, Kuan Wu², Yang Zhang^{1,**}, and Xue Li³

¹ College of Information Engineering, Northwest A&F University, China
{panshirui, zhangyang}@nwsuaf.edu.cn

² School of Information Science and Technology, Northwest University, China
wukuan1988@163.com

³ School of Information Technology and Electrical Engineering,
The University of Queensland, Brisbane, Australia
xueli@itee.uq.edu.au

Abstract. Currently available algorithms for data stream classification are all designed to handle precise data, while data with uncertainty or imperfection is quite natural and widely seen in real-life applications. Uncertainty can arise in attribute values as well as in class values. In this paper, we focus on the classification of streaming data that has different degrees of uncertainty within class values. We propose two types of ensemble based algorithms, Static Classifier Ensemble (SCE) and Dynamic Classifier Ensemble (DCE) for mining uncertain data streams. Experiments on both synthetic and real-life data set are made to compare and contrast our proposed algorithms. The experimental results reveal that DCE algorithm outperforms SCE algorithm.

1 Introduction

Mining on streaming data draws more and more attentions in research community in recent years. One of the most challenge issues for data stream mining is classification analysis with concept drift [4,5,6,7,8]. The currently available algorithms for data stream classification are all dedicated to handle precise data, while data with uncertainty is quite common in real-life applications. Uncertainty can appear in attributes in some applications. In a sensor network system, the information such as humidity, temperature and weather usually contains massive uncertainty during the processes of data collecting and transmitting [2]. Uncertainty can also appear in class values. For example, in medical diagnosis, it's hard for the doctor to decide the exact disease of a patient. It's wise to predict all the possibilities of each candidate diseases rather than give a crisp result.

In this paper, we study the data stream classification tasks with uncertain class values. Firstly, we extend the NS-PDT algorithm [1], a decision tree algorithm dealing with categorical attributes and uncertainty in class values, to handle continuous attributes following the scheme of C4.5 [10]. Secondly, we propose two ensemble based algorithms,

* This research is supported by the National Natural Science Foundation of China (60873196) and Chinese Universities Scientific Fund (QN2009092).

** Corresponding author.

Static Classifier Ensemble (SCE) and Dynamic Classifier Ensemble (DCE), to classify uncertain data streams with concept drift. Experiments on both synthetic data and real-life data set are made to validate our proposed methods. The experimental results show that DCE outperforms SCE.

This paper is organized as following: Section 2 reviews the related work on data streams and uncertainty. Section 3 describes NS-PDT algorithm for data with uncertainty in class values. Section 4 presents our ensemble based algorithms for classifying uncertain data streams. Section 5 shows our experimental results on both synthetic and real-life data set. And Section 6 concludes this paper and gives our future work.

2 Related Work

To the best of our knowledge, there is no work so far on classifying uncertain data streams, while both studies on mining of data stream and uncertain data have been well investigated in recent years.

At present, the classification algorithms for the data stream scenario are all dedicated to precise data only. A classical approach for data stream classification follows ensemble based scheme [4,5,6,7,8], which usually learns an ensemble of classifiers from the streaming data, and then uses all the classifiers to predict the newly coming instances. The way to integrate classifiers can be distinguished into two categories:

- **Static classifier ensemble.** The weight of each base classifier is decided before the testing phase.
- **Dynamic classifier ensemble.** The weight of each base classifier is decided by the testing instance.

It is concluded in [7] that dynamic methods outperform the static methods [4].

For the classification task on uncertain data, a few decision tree algorithms have been proposed [1,2,3]. Both DTU algorithm [2] and UDT algorithm [3] model the uncertainty in attribute values by probability distribution functions (pdf). This is different from our algorithm, for our goal is to deal with data uncertainty in class values. NS-PDT [1] is a decision tree algorithm to handle data with uncertain class values. However, it can only handle categorical attributes. Hence, in this paper, we firstly extend the initial NS-PDT algorithm [1] to handle continuous attributes and then use it as a base classifier for ensemble classification of data streams.

3 NS-PDT for Uncertain Data

We briefly describe NS-PDT algorithm [1] for uncertainty here. Given an uncertain data set with M classes, the possibility distribution $\pi = \{\pi(L_1), \pi(L_2), \dots, \pi(L_M)\}$ represents how likely an instance belongs to each class. More specifically, $\pi(L_i)$ accesses the possibility that an instance belongs to class $L_i, i \in [1, M]$.

Compared with the state-of-the-art tree building algorithm C4.5 [10], NS-PDT replaces the Information Gain by the Non-Specificity Gain (*NSG*), and uses the maximum Non-Specificity Gain Ratio (*NSGr*) to decide which attribute to be selected as the next splitting attribute. *NSG* is defined as [1]:

$$NSG(T, A_k) = U(\pi_{Rep}^T) - U_{A_k}(\pi_{Rep}^T) \quad (1)$$

Here, $NSG(T, A_k)$ represents the amount of *information precision* obtained after splitting the data set T according to attribute A_k ; $U(\cdot)$ accesses the *information precision* of the data set. Please refer to [11] for more details.

As NS-PDT can only deal with categorical attributes, here we simply follow the schemes utilized in C4.5 algorithm [10] to extend NS-PDT to handle continuous attributes. The detailed algorithm is omitted here for lack of space.

4 Mining Uncertain Data Streams

In this paper, we follow the assumption that in data stream scenario, data arrives as batches with variable length [12]:

$$\begin{aligned}
 & d_{1,1}, d_{1,2}, \dots, d_{1,m_1}; \\
 & d_{2,1}, d_{2,2}, \dots, d_{2,m_2}; \\
 & \dots; \\
 & d_{n,1}, d_{n,2}, \dots, d_{n,m_n}; \\
 & \dots;
 \end{aligned} \tag{2}$$

Here, $d_{i,j}$ represents the j -th instance in the i -th batch. In our paper, we learn a NS-PDT classifier from each batch of uncertain data, and then the base classifiers are combined to form an ensemble. To keep the population capacity of the ensemble, following [6], we delete the oldest classifier when the number of classifiers in the ensemble exceeds a predefined parameter *EnsembleSize*. In the testing phase, our goal is to output a possibility distribution of test instance x over all possible classes.

4.1 Static Classifier Ensemble

We proposed a static classifier ensemble algorithm to classify the uncertain data streams, which is illustrated in algorithm 1.

In algorithm 1, the function $C_i.dis(x)$ in step 3 returns a possibility distribution over different classes of x . In steps 4-6, the possibility for each class is accumulated. In step 8, we normalize the possibility by formula $\pi(L_i) = \frac{\pi(L_i)}{\max_{i=1}^{|M|} \{\pi(L_i)\}}$ [13], so that the maximum possibility in π is 1.

Note that in algorithm 1 wei_i represents the weight of classifier C_i , and it is decided in the most up-to-date data batch. Let's write $|NUM|$ for the total number of testing instances; $D(x_i)$ for the accuracy while predicting instance x_i ; $\pi^{res}(L_j(x_i))$ for the real possibility distribution of instance x_i on category j ; $\pi(L_j(x_i))$ for the predicted resulting possibility distribution of instance x_i on category j . Then the weight wei_i is computed by the *PCC_dist* metric given by [14]:

$$PCC_dist = \frac{\sum_{i=1}^{|NUM|} D(x_i)}{|NUM|} \tag{3}$$

$$D(x_i) = 1 - \frac{\sum_{j=1}^{|M|} (\pi^{res}(L_j(x_i)) - \pi(L_j(x_i)))^2}{|M|} \tag{4}$$

Algorithm 1. Classifying Uncertain Data Streams by Static Classifier Ensemble**Input:**

E_n : Ensemble of classifiers;
 x : Testing instance ;
 wei : An array of each classifier's weight;

Output:

$\pi = \{\pi(L_1), \pi(L_2) \dots, \pi(L_M)\}$: The possibility distribution of x ;
1: Initialize $\pi(L_i) = 0, i \in [1, M]$;
2: **for** each $C_i \in E_n$ **do**
3: $pre[] = C_i.dis(x)$;
4: **for** $j = 1$ to M **do**
5: $\pi(L_j) = \pi(L_j) + wei_i \times pre_j$;
6: **end for**
7: **end for**
8: Normalize π ;
9: **return** π ;

Here, PCC_dist criterion takes into account the average of the distances between the resulting possibility distribution and the real possibility distribution. Note that high value of the PCC_dist criterion implies not only that the algorithm is accurate, but also that the possibility distributions outputted by the algorithm is of high quality and faithful to the original possibility distribution [1].

4.2 Dynamic Classifier Ensemble

In our DCE algorithm, the weight of each base classifier is decided by test instance x . The weighted Euclidean distance between two instances with m attributes is given by :

$$d(x_1, x_2) = \sqrt{\sum_{A_i=1}^m w_{A_i} \cdot diff(x_1^{A_i}, x_2^{A_i})^2} \quad (5)$$

For a continuous attribute A_i , we have $diff(x_1^{A_i}, x_2^{A_i}) = \frac{x_1^{A_i} - x_2^{A_i}}{range_{A_i}}$. And for a categorical attribute A_i , we have $diff(x_1^{A_i}, x_2^{A_i}) = \begin{cases} 0 & \text{if } x_1^{A_i} = x_2^{A_i} \\ 1 & \text{if } x_1^{A_i} \neq x_2^{A_i} \end{cases}$. Here, w_{A_i} represents the weight of attribute A_i , and it is decided by the non-specificity gain (NSG) [1] of the attribute A_i . Algorithm 2 gives the details of our DCE algorithm.

In algorithm 2, in step 2 we retrieve K_{nei} neighbors of x from validation data set V according to formula (5). And the most up-to-date data batch from the stream is used as V . In steps 3-9, the weight of each base classifier is determined. The function $C_i.Acc(nei_j)$ in step 6 returns the accuracy of classifier C_i while predicting the instance nei_j , and the accuracy is decided according to formula (4). The function $d(nei_j, x)$ in step 7 returns the distance following formula (5). After predicting an instance, the weight of the corresponding base classifier is accumulated together. Accuracy and distance are the main factors that affect wei_i . In steps 10-14, some base classifiers with lower weight are ignored following [6]. The function $predict(E, x, wei)$ in step 14

Algorithm 2. Classifying Uncertain Data Streams by Dynamic Classifier Ensemble**Input:**

E_n : Ensemble of classifiers;
 x : Testing instance ;
 V : Validation Set;

Output:

$\pi = \{\pi(L_1), \pi(L_2) \dots, \pi(L_M)\}$: The possibility distribution of x ;
1: Initialize $\pi(L_i) = 0, i \in [1, M]$;
2: Retrieve K_{nei} neighbors from V to form data set Nei ;
3: **for** each $C_i \in E_n$ **do**
4: $wei_i = 0$;
5: **for** each $nei_j \in Nei$ **do**
6: $acc = C_i.Acc(nei_j)$;
7: $wei_i = wei_i + acc/d(nei_j, x)$;
8: **end for**
9: **end for**
10: $m_1 = \max\{wei_i\}, i \in [1, |E_n|]$;
11: $m_2 = \min\{wei_i\}, i \in [1, |E_n|]$;
12: $mid = (m_1 + m_2)/2$;
13: $E = \{e_i | wei_i \geq mid, e_i \in E_n\}$;
14: $\pi = predict(E, x, wei)$;
15: **return** π ;

returns the predicted result of x by a weighted voting scheme. The weighted voting scheme is the same as in algorithm 1.

5 Experiments

In this section, we report our experimental results. We deal with the data streams with uncertain class information and concept drift, which is regarded as an important issue in mining data streams [4,5,6,7]. Since no benchmark uncertain data set can be found in the literature, we simulate the uncertain data on synthetic data and real-life data set. Here, moving hyperplane concept [4,7] was used as the synthetic data set with continuous attributes, and RCV1-v2¹ was used as the real-life data set with categorical attributes in our experiments.

The uncertain data set was generated by using the approach described in [1]. We simulated various levels of uncertainty ($X\%$) when generating possibilistic training data. Firstly, we randomly selected $X\%$ of instances in each data batch as uncertain set. Secondly, for each instance in the uncertain set, if the instance belongs to the i -th class, we set $\pi(\omega_i) = 1$, and $\pi(\omega_j) = \varphi, \forall j \neq i$. Here, φ is a random value distributed uniformly in range of [0, 1]. Thirdly, for each instance of the remaining $(100 - X)\%$ instances, we assigned a certainly sure possibility distribution (if the instance belongs to the i -th class, we set $\pi(\omega_i) = 1$, and $\pi(\omega_j) = 0, \forall j \neq i$). The data set used for test in our experiments was also with a certainly sure possibility distribution, following [1].

¹ http://jmlr.csail.mit.edu/papers/volume5/lewis04a/lyrl2004_rcv1v2_README.htm

PCC_dist is used as the evaluation metric in our experiments for measuring the classification performance of classifiers on uncertain data set. Note that high value of PCC_dist implies not only that the algorithm is accurate, but also that the distribution possibility is faithful to the original possibility distribution [1].

In our experiments, we set $EnsembleSize=25$, following [7]; and we set $K_{nei}=5$. Meanwhile, we set $ChunkSize$, the number of instances in each batch, to 500.

Our experiments were made on a PC with Pentium 4 3.0 GHz CPU and 1G memory. All of the algorithms were implemented in Java with help of WEKA² software package.

5.1 Moving Hyperplane Concept with Gradual Concept Drift

Moving hyperplanes have been widely used to simulate time-changing concepts [4]. In this subsection, we report our experiments on moving hyperplane concept. A moving hyperplane in d -dimensional space is denoted by: $\sum_{i=1}^d a_i x_i = a_0$.

We followed the same procedure in [4] to simulate concept drift. We used K for the total number of dimensions whose weights are changing; S for the magnitude of the change (every N instances) for weights a_1, \dots, a_k . For more detailed descriptions of moving hyperplanes, please refer to [4].

We set $d=10$, $N=1000$ in our experiments. For each parameter setting, we generated 100 batches of data. The averaged results on the 100 batches are reported in Table 1.

From Table 1 it could be concluded that, DCE outperforms SCE in all scenarios, with averaged improvement being about 2%. It could be also seen that, with the increasing of X , the performances of both SCE and DCE decline. It is shown that the higher the level of uncertainty in the training set is, the more imprecise the training set will be, and therefore the more difficult for the classifier to learn an accurate model.

We also study the impact of the parameter K_{nei} . Here, we set $S=0.1$, $X=20$ and K was selected from [1,10] randomly. We conducted 10 trails and the averaged result is shown in Fig. 1. It can be shown in Fig. 1 that K_{nei} can not be neither too small nor too large. We set $K_{nei}=5$ in other experiments as it usually gives us good results.

Table 1. Experiments on Moving Hyperplane Concept

| X% | S | K=2 | | K=5 | | K=8 | |
|----|-----|-------|--------------|-------|--------------|-------|--------------|
| | | SCE | DCE | SCE | DCE | SCE | DCE |
| 20 | 0.1 | 0.770 | 0.787 | 0.768 | 0.785 | 0.768 | 0.783 |
| | 0.5 | 0.773 | 0.789 | 0.764 | 0.781 | 0.763 | 0.778 |
| | 1.0 | 0.771 | 0.790 | 0.761 | 0.778 | 0.762 | 0.777 |
| 40 | 0.1 | 0.749 | 0.767 | 0.748 | 0.766 | 0.748 | 0.764 |
| | 0.5 | 0.749 | 0.768 | 0.743 | 0.760 | 0.743 | 0.759 |
| | 1.0 | 0.752 | 0.770 | 0.738 | 0.756 | 0.741 | 0.758 |
| 60 | 0.1 | 0.725 | 0.744 | 0.723 | 0.741 | 0.720 | 0.738 |
| | 0.5 | 0.725 | 0.744 | 0.715 | 0.733 | 0.717 | 0.735 |
| | 1.0 | 0.726 | 0.746 | 0.712 | 0.732 | 0.717 | 0.735 |

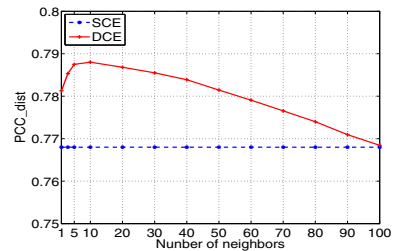


Fig. 1. Experiment with K_{nei}

² <http://www.cs.waikato.ac.nz/ml/weka/>

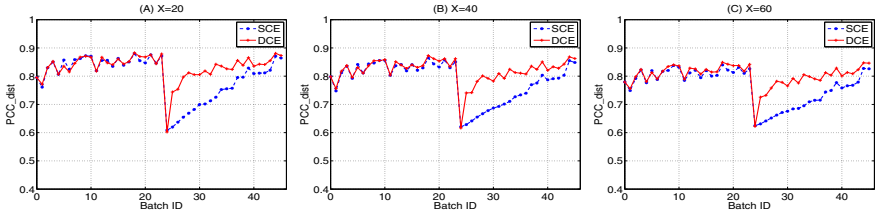


Fig. 2. Experiments on RCV1_v2 Set for Abrupt Concept Drift

5.2 Experiments on RCV1-v2 Text Data Set

In this section, we report our experimental results on a real-life text dataset. RCV1-v2 is a new benchmark collection for text categorization [9]. The news stories in RCV1-v2 were divided into two datasets, training set and testing set. The training dataset was used and four largest categories, CCAT, ECAT, GCAT, and MCAT were considered as the main topics in our experiments to simulate concept drift. After preprocessing, each document was presented by a binary vector, and information gain algorithm was used to select 100 most predictive features.

The data set in our experiments was decomposed into 46 batches. We vary uncertainty $X\%$ as follow: 20, 40, 60. In each scenario of our experiments, we selected one category as *TopicA* and others as *TopicB*. Concept drift occurs between *TopicA* and *TopicB*. Therefore, there are 12 possible combination for 4 categories. We experiment 12 trails and the averaged result is reported.

We simulate abrupt drift in our experiments as follows: In the batches 0-24, all the instances of *TopicA* were labeled as positive, others were labeled as negative. In batches 25-46, all the instances of *TopicB* were labeled as positive, others were labeled as negative. The experimental results for abrupt concept drift are given in Fig. 2.

As expected, from Fig. 2, it could be seem that in batch 25 when abrupt drift occurs, both DCE and SCE have a dramatic decline in PCC_{dist} . However, DCE recovers much faster than SCE in the later batches. In all levels of uncertainty, DCE outperforms SCE. It reveals that DCE outperforms SCE in handling abrupt concept drift with uncertainty.

5.3 Time Analysis

Both SCE and DCE consume the same time in training, because the ways to construct the ensemble is the same. Here we compare the testing time. We generated data streams with varied *ChunkSize* on hyperplane concept ($K=10, S=1$). Consider 100 batches of data with $X=20$. Fig. 3 shows that large *ChunkSize* offers better performances. The DCE also performs better than SCE. However, the testing time of the two methods on the whole streams is significantly different. From Fig. 4 we can see that with a large *ChunkSize*, DCE consumes much more testing time than SCE. It is because when predicting an instance, DCE needs to traverse the whole validation set to get some nearest neighbors. So it's a tradeoff between good performance and faster responding action.

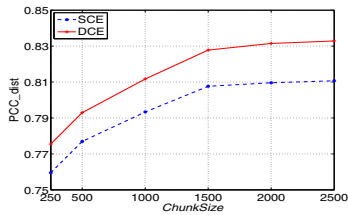


Fig. 3. *PCC_dist* on varying *ChunkSize*

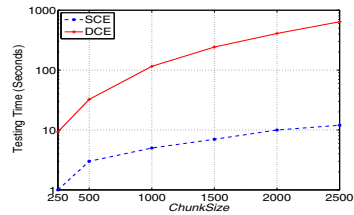


Fig. 4. Testing time on varying *ChunkSize*

6 Conclusion and Future Work

In this paper, we propose two types of ensemble based approaches to classify data streams with uncertainty in class values. We also extend the decision tree (NS-PDT) approach to handle data with continuous attributes. Experiments on both synthetic and real-life datasets are made to validate our proposed methods. Our experimental results show that the dynamic classifier ensemble (DCE) for uncertain data stream outperforms the static classifier ensemble (SCE).

Attribute uncertainty is also natural and prevalent in many real-life applications, we plan to study this kind of uncertain data streams in future work.

References

1. Jenhani, I., Amor, N.B., Eluouedi, Z.: Decision trees as possibilistic classifiers. *International Journal of Approximate Reasoning* 48, 784–807 (2008)
2. Qin, B., Xia, Y., Li, F.: DTU: A Decision Tree for Uncertain Data. In: Theeramunkong, T., Kijssirikul, B., Cercone, N., Ho, T.-B. (eds.) *PAKDD 2009*. LNCS, vol. 5476, pp. 4–15. Springer, Heidelberg (2009)
3. Tsang, S., Kao, B., Yip, K.Y., Ho, W., Lee, S.D.: Decision Trees for Uncertain Data. In: *Proc. of ICDE 2009*, pp. 441–444 (2009)
4. Wang, H., Fan, W., Yu, P., Han, J.: Mining concept-drifting data streams using ensemble classifiers. In: *Proc. of KDD 2003*, pp. 226–235 (2003)
5. Zhu, X., Wu, X., Yang, Y.: Dynamic classifier selection for effective mining from noisy data streams. In: *ICDM 2004*, pp. 305–312 (2004)
6. Zhang, Y., Jin, X.: An automatic construction and organization strategy for ensemble learning on data streams. *ACM SIGMOD Record* 35(3), 28–33 (2006)
7. Tsymbal, A., Pechenizkiy, M., Cunningham, P., Puuronen, S.: Dynamic integration of classifiers for handling concept drift. *Information fusion* 9, 56–68 (2008)
8. Kolter, J., Maloof, M.: Dynamic weighted majority: a new ensemble method for tracking concept drift. In: *Proc. of ICDM 2003*, pp. 123–130 (2003)
9. Lewis, D.D., Yang, Y., Rose, T., Li, F., Zhu, X., Wu, X., Yang, Y.: RCV1: A New Benchmark Collection for Text Categorization Research. *Journal of Machine Learning Research* 1(5), 361–397 (2004)
10. Quinlan, J.R.: *C4.5: Programs for Machine Learning*. Morgan Kaufman Publishers, San Francisco (1993)
11. Klinkenberg, R., Joachims, T.: Detecting concept drift with support vector machines. In: *Proc. of ICML 2000*, pp. 487–494 (2000)

Author Index

- Abulaish, Muhammad II-238
Achananuparp, Palakorn I-375
Adams, Brett II-283
Agrawal, R.K. II-81
Ahmad, Tanvir II-238
Ah-Pine, Julien II-362
Akoglu, Leman II-410
Ang, Hock Hee II-63
An, Yuan I-375
Arun, R. I-391
Aziz, Mohammad S. I-28
- Bailey, James I-107
Bala, Rajni II-81
Balby Marinho, Leandro I-348
Barnathan, Michael I-246
Basu, Dipak Kumar II-101
Berger-Wolf, Tanya Y. I-91
Berlingerio, Michele I-81
Bezdek, James C. I-16
Bhardwaj, Vikas I-328
Bhattacharya, Arnab I-319
Bifet, Albert II-299
Blanton, Marina I-198
Böhm, Christian I-38, I-461
Bu, Jiajun I-440
- Calders, Toon I-480
Cao, Longbing I-262, II-55, II-89
Carrasco-Ochoa, Jesús Ariel I-150
Chandrashekar, Shruti II-160
Chawla, Nitesh V. I-198, II-488
Chawla, Sanjay II-422
Chen, Arbee L.P. I-56
Chen, Chun I-440
Chen, Daoxu I-99
Cheng, Hong I-310
Cheng, Michael W.K. I-158
Chen, Hong I-471
Chen, Ming-Syan II-27, II-354
Chen, Wei I-440
Cheung, William Kwok Wai I-158
Cho, Hyuk I-48
Choi, Byron Koon Kau I-158
- Chowdhury, Shiladitya II-101
Chua, Tat-Seng I-383
Clérot, Fabrice II-343
Coscia, Michele I-81
Crestani, Fabio I-286
Cule, Boris I-300
- Dai, Bo I-134, II-507
Dash, Manoranjan II-35
Datta, Anwitaman II-63
Denker, Stefan II-325
Devarajan, Ramaswamy I-328
Di Caro, Luigi II-125
Doja, Mohammad Najmud II-238
Dou, Dejing II-43
Du, Haohua I-254
Dutta, Sourav I-319
- Eick, Christoph F. I-216
Eliassi-Rad, Tina II-449
- Faloutsos, Christos I-246, II-410,
II-435, II-449
Faro, Scott I-246
Feng, Mengling I-121
Feng, Qian I-428
Frank, Eibe II-299
Frank, Robert II-43
Frias-Martinez, Enrique II-125
Frias-Martinez, Vanessa II-125
Frikken, Keith I-198
Frishkoff, Gwen II-43
Fukuzaki, Mutsumi II-147
- Gabsi, Nesrine II-343
Gao, Jing II-311
Gao, Junbin II-113
Garboni, Calin I-480
García-Borroto, Milton I-150
Gedeon, Tom II-291
Ge, Jiaqi I-449
Geva, Shlomo I-340
Giannotti, Fosca I-81
Goebel, Sebastian I-38
Goethals, Bart I-300, I-480

- Gopalkrishnan, Vivekanand II-63
 Günemann, Stephan II-133
 Guo, Lifan I-375
 Gu, Qing I-99
 Gwadera, Robert I-286
- Han, Jiawei II-222, II-311
 Harwood, Aaron II-272
 Hassan, Md. Rafiul I-107
 Hébrail, Georges II-343
 He, Dan II-201
 He, Tingting I-375
 He, Wenlin I-471
 Hilario, Melanie II-374
 Hintsanen, Petteri II-168
 Hoens, T. Ryan II-488
 Hoi, Steven C.H. II-63
 Holmes, Geoff II-299
 Hossain, M. Maruf I-107
 Houle, Michael E. I-4
 Hsu, Kuo-Wei II-500
 Huang, Jen-Wei II-27
 Huang, Joshua Zhexue I-228
 Huang, YaLou I-358
 Hu, Shaohan I-415
 Hu, Xiaohua I-375
- Inokuchi, Akihiro II-178
- Jahiruddin II-238
 Jiang, Lu I-428
 Jiang, Peng II-249
 Jiang, Xinwei II-113
 Jia, Peifa II-261
 Ji, Yangsheng I-134
- Kalousis, Alexandros II-374, II-386
 Kanani, Pallika I-415
 Karmakar, Chandan I-107
 Kasari, Melissa II-168
 Kashima, Hisashi II-147
 Kecman, Vojislav II-55
 Khan, Latifur II-311
 Koh, Jia-Ling I-56
 Koh, Yun Sing I-274
 Kranen, Philipp II-325
 Krieger, Ralph II-325
 Kumar, Vipin I-2
 Kwan, Paul W. II-113
- Laurent, Anne II-335
 Leckie, Christopher A. I-16
 Liao, Yang II-272
 Li, Chunping I-367
 Li, Cuiping I-471
 Li, Jiuyong I-181
 Li, Jun II-261
 Lim, Ee-Peng I-68
 Lin, Chen-Yi I-56
 Lin, Cindy Xide II-222
 Ling, Charles X. II-476
 Lin, Su-Chen II-27, II-354
 Lin, Yi I-238
 Liu, Bing II-222
 Liu, Guimei I-121
 Liu, Haishan II-43
 Liu, Hongyan I-310
 Liu, Jie I-358
 Liu, Jun I-428
 Liu, Junqiang I-171
 Li, Xue I-488
 Li, Yidong I-208
 Li, Zhao II-214
 Lo, David I-68
 Lu, Min I-358
 Luo, Jun I-228
 Lu, Zhenyu II-214
 Lu Dang Khoa, Nguyen II-422
- Machiraju, Sridhar II-435
 Magdalinos, Panagis II-14
 Ma, Huifang II-189
 Maiya, Arun S. I-91
 Manna, Sukanya II-291
 Martínez-Trinidad, José Francisco I-150
 Masud, Mohammad M. II-311
 Ma, Wei-Ying I-1
 McCallum, Andrew I-415
 McCool, Michael D. I-238
 McGlohon, Mary II-410
 Megalooikonomou, Vasileios I-246
 Mendis, B. Sumudu U. II-291
 Miao, Yajie I-367
 Ming, Zhao-Yan I-383
 Mohamed, Feroze B. I-246
 Monreale, Anna I-81
- Nadungodage, Chandima I-449
 Nakagawa, Hiroshi I-189, II-230
 Narahari, Y. I-3

- Narasimha Murthy, M.N. I-391
 Nasipuri, Mita II-101
 Ngo, Thanh-Son I-121
 Nguyen, Thin II-283
 Nguyen, Uyen T.V. I-16
 Ng, Wee Keong II-63
 Ni, Eileen A. II-476
 Niu, Gang I-134, II-507
 Niu, Zhendong II-249

 Oswald, Annahita I-38, I-461

 Pan, Shirui I-488
 Papadimitriou, Spiros II-449
 Parthasarathy, Srinivasan II-160
 Pears, Russel I-274
 Pedreschi, Dino I-81
 Pfahringer, Bernhard II-299
 Phung, Dinh II-283
 Pitarch, Yoann II-335
 Plant, Claudia I-38, I-461
 Plavinski, Michael I-38, I-461
 Poncelet, Pascal II-335
 Prakash, B. Aditya II-435
 Preisach, Christine I-348

 Raeder, Troy I-198
 Ramamohanarao, Kotagiri I-16,
 I-107, II-272
 Reddy, Chandan K. I-28
 Rinsurongkawong, Vadeerat I-216

 Sardana, Manju II-81
 Sato, Issei II-230
 Schmidt-Thieme, Lars I-348
 Seidl, Thomas II-133, II-325
 Seki, Mio II-147
 Sese, Jun II-147
 Seshadri, Mukund II-435
 Shalom, S.A. Arul II-35
 Shang, Lin I-134
 Sharan, Richa II-71
 Shaw, Gavin I-340
 Shen, Hong I-208
 Shim, Kyong Jin II-71
 Shi, Zhongzhi II-189
 Sing, Jamuna Kanta II-101
 Squire, David McG. I-142

 Sridharan, Ashwin II-435
 Srihari, Sriganesh II-160
 Srivastava, Jaideep II-71, II-500
 Sun, Xiaoxun I-181
 Suresh, V. I-391
 Suter, David I-142
 Suzuki, Einoshin II-1

 Tan, Qing II-189
 Terada, Akira II-230
 Thuraisingham, Bhavani II-311
 Toivonen, Hannu II-168
 Tong, Bin II-1
 Tong, Hanghang II-449
 Tran, Truyen II-283
 Tue, Minh II-35

 Valsamou, Dialecti II-14
 Vazirgiannis, Michalis II-14
 Veni Madhavan, C.E. I-391
 Venkatesh, Svetha II-283
 Vinh, Nguyen Xuan I-4

 Wackersreuther, Bianca I-38, I-461
 Wang, Can I-440
 Wang, Hua I-181
 Wang, Jiayao I-254
 Wang, Kai I-383
 Wang, Ke I-171
 Wang, Liang I-16
 Wang, Qing II-464
 Wang, Tianjiang II-113
 Wang, Yang I-358
 Washio, Takashi II-178
 Wong, Limsoon I-121
 Woźnica, Adam II-374, II-386
 Wu, Kuan I-488
 Wu, Xindong II-201, II-214
 Wu, Zhaohui I-428

 Xia, Yuni I-449
 Xie, MaoQiang I-358
 Xu, Hua II-261
 Xu, Wanhong I-403
 Xu, Yue I-340

 Yang, Bin I-189
 Yang, Christopher C. I-375
 Yang, Qing II-249
 Yang, Tao II-55, II-89

- Yang, Weidong II-398
Yeap, Wai I-274
Yeh, Mi-Yen II-354
Yin, Hang I-228
Yoshida, Minoru II-230
Yu, Philip S. II-449
Yu, Yintao II-222
- Zaidi, Nayyar Abbas I-142
Zhai, Zhongwu II-261
Zhan, F. Benjamin I-228
Zhang, Chengqi II-55, II-89
Zhang, Chunxia II-249
Zhang, Kuan I-68
Zhang, Liang II-464
- Zhang, Lijun I-440
Zhang, Xueping I-254
Zhang, Yang I-488
Zhang, Yao I-99
Zhao, Weizhong II-189
Zhao, Yanchang I-262
Zheng, Jun I-99
Zheng, Qinghua I-428
Zheng, Zhigang I-262
Zhou, Wenzhi I-310
Zhou, Xiaotao I-228
Zhu, Hao II-398
Zhu, Haohan I-228
Zhu, Xingquan II-201
Zuo, Ziyi I-262