Jian Pei Vincent S. Tseng
Longbing Cao Hiroshi Motoda
Guandong Xu (Eds.)

LNAI 7819

Advances in Knowledge Discovery and Data Mining

17th Pacific-Asia Conference, PAKDD 2013
Gold Coast, Australia, April 2013
Proceedings, Part II

2 Part II

Springer

Lecture Notes in Artificial Intelligence 7819

Subseries of Lecture Notes in Computer Science

Jian Pei   Vincent S. Tseng   Longbing Cao
Hiroshi Motoda   Guandong Xu (Eds.)

Advances in Knowledge Discovery and Data Mining

17th Pacific-Asia Conference, PAKDD 2013
Gold Coast, Australia, April 14-17, 2013
Proceedings, Part II

Springer

Volume Editors

Jian Pei
Simon Fraser University, Burnaby, BC, Canada
E-mail: jpei@cs.sfu.ca

Vincent S. Tseng
National Cheng Kung University, Tainan, Taiwan
E-mail: tsengsm@mail.ncku.edu.tw

Longbing Cao
Guandong Xu
University of Technology Sydney, NSW, Australia
E-mail: {longbing.cao, guandong.xu}@uts.edu.au

Hiroshi Motoda
Osaka University, Japan
E-mail: motoda@ar.sanken.osaka-u.ac.jp

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-37455-5 e-ISBN 978-3-642-37456-2
DOI 10.1007/978-3-642-37456-2
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2013934551

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

LNCS Sublibrary: SL 7 – Artificial Intelligence

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

Printed on acid-free paper

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

Preface

As the Program Committee Co-chairs, we welcome you to the proceedings of the 17th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2013), held at Gold Coast, Australia, during April 14-17, 2013.

The PAKDD conference series, since its inception in 1997, has been a leading international conference in the areas of data mining and knowledge discovery (KDD). It provides an inviting and inspiring forum for researchers and practitioners, from both academia and industry, to share new ideas, original research results, and practical experience. The 17th edition continued the great tradition, and had three world-class keynote speeches, a wonderful technical program, a handful of high-quality tutorials and workshops, as well as an interesting invited talk from industry.

The PAKDD 2013 conference received 363 submissions to the technical program, involving more than 1,000 authors in total. In the rigorous double-blind review process, each submission was reviewed by one senior Program Committee member and at least three Program Committee members. Many submissions were extensively and thoroughly discussed by the reviewers. Based on the detailed and critical discussion and reviews, the senior Program Committee members made recommendations. Overall, 98 papers from 341 authors were accepted in the technical program, yielding a 27% acceptance rate. Of these, 39 (10.7%) had long presentations (30 minutes) and 59 (16.3%) had short presentations (15 minutes). The technical program consisted of 22 sessions, covering the general fields of data mining and KDD extensively, including pattern mining, classification, graph mining, applications, machine learning, feature selection and dimensionality reduction, multiple information sources mining, social networks, clustering, text mining, text classification, imbalanced data, privacy-preserving data mining, recommendation, multimedia data mining, stream data mining, data preprocessing and representation.

We were lucky to have three world-class keynote speakers this year. Usama Fayyad, a renowned pioneer in big data entrepreneurship, addressed us on the big picture of big data. Huan Liu, a world-wide leader in social media mining, discussed this exciting new frontier of data mining. Qiang Yang, a famous expert on artificial intelligence and machine learning, talked on how machine learning can address the big data challenge. We were also pleased to have Alexandros Batsakis as an invited speaker from industry. He shared with us the latest developments on big data analytics infrastructure and enterprise applications.

The conference also included six workshops, covering a few exciting and fast-growing hot topics. We also had five very timely and educational tutorials, covering the hot topics of social networks, transfer learning, stream mining, outlier detection, and feature discovery.

In addition to the intellectually inspiring keynote speeches, technical program, workshops and tutorials, we had several dynamic social events to facilitate communication and informal interaction, including a welcome reception, a banquet, and an excursion.

Putting together a conference like PAKDD is never easy. It becomes possible only with tremendous contributions from the organizing team and many volunteers. We thank Jiuyong Li, Kay Chen Tan, and Bo Liu for organizing the workshop program. We also thank Tu Bao Ho and Mengjie Zhang for organizing the tutorial program. We are grateful to Chengqi Zhang for his leadership in the award selection. We owe a big thank-you to the 39 senior Program Committee members, 151 Program Committee members, and the external reviewers for their great contributions and collaboration. We thank Guandong Xu for assembling the proceedings. We also thank the General Chairs, Hiroshi Motoda and Longbing Cao, and the local organization team for their great support. Without the dedicated hard work of so many people, PAKDD 2013 would simply have been mission impossible.

February 2013

Jian Pei
Vincent S. Tseng

Organization

Honorary Co-chairs

Jiawei Han	University of Illinois at Urbana-Champaign, USA
Ramamohanarao Kotagiri	University of Melbourne, Australia
Graham Williams	Australia Taxation Office, Australia

Conference Co-chairs

Hiroshi Motoda	Osaka University, Japan
Longbing Cao	University of Technology, Sydney, Australia

Program Committee Co-chairs

Jian Pei	Simon Fraser University, Canada
Vincent S. Tseng	National Cheng Kung University, Taiwan

Local Arrangements Co-chairs

Vladimir Estivill-Castro	Griffith University (Gold Coast), Australia
Xue Li	University of Queensland, Australia
Richi Nayak	Queensland University of Technology, Australia
Xinhua Zhu	University of Technology, Sydney, Australia

Workshop Co-chairs

Jiuyong Li	University of South Australia, Australia
Kay Chen Tan	National University of Singapore, Singapore
Bo Liu	Guangdong University of Technology, China

Tutorial Co-chairs

Tu Bao Ho	Japan Advanced Institute of Science and Technology, Japan
Mengjie Zhang	Victoria University of Wellington, New Zealand

Awards Chair

Chengqi Zhang	University of Technology, Sydney, Australia

Sponsorship Co-chairs

Yue Xu	Queensland University of Technology, Australia
Eugene Dubossarsky	Analyst First
Suresh Sood	University of Technology, Sydney, Australia

Publicity Co-chairs

P.Krishna Reddy	The International Institute of Information Technology, Hyderabad, India
Yifeng Zeng	Aalborg University, Denmark
Xin Wang	University of Calgary, Canada
Zhihong Deng	Peking University, China

Proceedings Chair

Guandong Xu	Unverisity of Technology, Sydney, Australia

Registration Co-chairs

Qiang Wu	University of Technology, Sydney, Australia
Can Wang	University of Technology, Sydney, Australia

Big Data School Co-chairs

Jinyan Li	University of Technology, Sydney, Australia
Xinhua Zhu	University of Technology, Sydney, Australia

Steering Committee

Graham Williams	Australian Taxation Office, Australia
Tu Bao Ho	Japan Advanced Institute of Science and Technology, Japan
Hiroshi Motoda	Osaka University, Japan (Since 1997)
Rao Kotagiri	University of Melbourne, Australia (Since 1997)
Ning Zhong	Maebashi Institute of Technology, Japan (Since 1999)
Masaru Kitsuregawa	Tokyo University, Japan (Since 2000)
David Cheung	University of Hong Kong, China (Since 2001)
Graham Williams (Treasurer)	Australian National University, Australia (Since 2001)
Ming-Syan Chen	National Taiwan University, Taiwan, ROC (Since 2002)

Kyu-Young Whang	Korea Advanced Institute of Science & Technology, Korea (Since 2003)
Huan Liu	Arizona State University, USA (Since 1998)
Chengqi Zhang	University of Technology Sydney, Australia (Since 2004)
Tu Bao Ho	Japan Advanced Institute of Science and Technology, Japan(Since 2005)
Ee-Peng Lim	Singapore Management University, Singapore (Since 2006)
Jaideep Srivastava	University of Minnesota, USA (Since 2006)
Zhi-Hua Zhou	Nanjing University, China (Since 2007)
Takashi Washio	Institute of Scientific and Industrial Research, Osaka University, Japan (Since 2008)
Thanaruk Theeramunkong	Thammasat University, Thailand (Since 2009)
P. Krishna Reddy	International Institute of Information Technology, Hyderabad (IIIT-H), India (Since 2010)
Joshua Z. Huang	Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China (Since 2011)

Senior Program Committee Members

James Bailey	University of Melbourne, Australia
Michael Berthold	University of Konstanz, Germany
Nitesh Chawla	University of Notre Dame, USA
Sanjay Chawla	University of Sydney, Australia
Ming-Syan Chen	National Taiwan University, Taiwan
Arbee L. P. Chen	National Chengchi University, Taiwan
Peter Christen	The Australian National University, Australia
Diane Cook	Washington State University, USA
Ian Davidson	UC Davis, USA
Bart Goethals	University of Antwerp, Belgium
Tu-Bao Ho	Japan Advanced Institute of Science and Technology, Japan
Xiaohua Hu	Drexel University, USA
Ming Hua	Facebook, USA
Joshua Huang	Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China
Hisashi Kashima	University of Tokyo, Japan
Jiuyong Li	University of South Australia, Australia
Ee-Peng Lim	Singapore Management University, Singapore
Chih-Jen Lin	National Taiwan University, Taiwan
Huan Liu	Arizona State University, USA
Nikos Mamoulis	University of Hong Kong, Hong Kong

Wee Keong Ng	Nanyang Technological University, Singapore
Wen-Chih Peng	National Chiao Tung University, Taiwan
P. Reddy	International Institute of Information Technology, Hyderabad (IIIT-H), India
Kyuseok Shim	Seoul National University, Korea
Masashi Sugiyama	Tokyo Institute of Technology, Japan
Pang-Ning Tan	Michigan State University, USA
Hanghang Tong	IBM T. J. Watson Research, USA
Shusaku Tsumoto	Shimane University, Japan
Wei Wang	University of North Carolina at Chapel Hill, USA
Haixun Wang	Microsoft Research Asia, China
Jianyong Wang	Tsinghua University, China
Ji-Rong Wen	Microsoft Research Asia, China
Xing Xie	Microsoft Research Asia, China
Hui Xiong	Rutgers University, USA
Xifeng Yan	UC Santa Barbara, USA
Jieping Ye	Arizona State University
Jeffrey Yu	The Chinese University of Hong Kong, Hong Kong
Chengqi Zhang	University of Technology, Australia
Zhi-Hua Zhou	Nanjing University, China

Program Committee Members

Hideo Bannai	Kyshu University, Japan
Marut Buranarach	National Electronics and Computer Technology Center, Thailand
Rui Camacho	Universidade do Porto, Portugal
Tru Cao	Ho Chi Minh City University of Technology, Vietnam
Keith Chan	The Hong Kong Polytechnic University, Hong Kong
Muhammad Cheema	University of New South Wales, Australia
Enhong Chen	University of Science and Technology of China, China
Jake Chen	Indiana University-Purdue University Indianapolis, USA
Jian Chen	Southern China University of Technology, China
Ling Chen	University of Technology Sydney, Australia
Shu-Ching Chen	Florida International University, USA
Songcan Chen	Nanjing University of Aeronautics and Astronautics, China
Yi-Ping Phoebe Chen	La Trobe University, Australia
Zheng Chen	Microsoft Research Asia, China

Hong Cheng	The Chinese University of Hong Kong, Hong Kong
Yiu-ming Cheung	Hong Kong Baptist University, Hong Kong
Bruno Cremilleux	Université de Caen, France
Bin Cui	Peking University, China
Alfredo Cuzzocrea	ICAR-CNR and University of Calabria, Italy
Dao-Qing Dai	Sun Yat-Sen University, China
Bolin Ding	Microsoft Research, USA
Wei Ding	University of Massachusetts Boston, USA
Guozhu Dong	Wright State University, USA
Wei Fan	IBM T. J. Watson Research Center, USA
Eibe Frank	University of Waikato, New Zealand
Joao Gama	Universidade do Porto, Portugal
Dragan Gamberger	Rudjer Boskovic Institute, Croatia
Cong Gao	Nanyang Technological University, Singapore
Jun Gao	Peking University, China
Junbin Gao	Charles Sturt University, Australia
Yong Guan	Iowa State University, USA
Ravi Gupta	Anna University, India
Sung-Ho Ha	Kyungpook National University, Korea
Yi Han	National Defence Technology University, China
Choochart Haruechaiy	National Electronics and Computer Technology Center, Thailand
Jingrui He	IBM Research, USA
Qi He	IBM Research, USA
Chin-Kuan Ho	Multimedia University, Malaysia
Kuo-Wei Hsu	National Chengchi University, Taiwan
Jun Huan	University of Kansas, USA
Jin Huang	Southern China Normal University, China
Daisuke Ikeda	Kyshu University, Japan
Akihiro Inokuchi	Osaka University, Japan
Sanjay Jain	National University of Singapore, Singapore
Shuiwang Ji	Old Dominion University, USA
Ruoming Jin	Kent State University, USA
Toshihiro Kamishima	National Institute of Advanced Industrial Science and Technology, Japan
Hung-Yu Kao	National Cheng Kung University, Taiwan
Panagiotis Karras	Rutgers University, USA
George Karypis	University of Minnesota, USA
Hiroyuki Kawano	Nanzan University, Japan
Latifur Khan	University of Texas at Dallas, USA
Hiroyuki Kitagawa	University of Tsukuba, Japan
Irena Koprinska	University of Sydney, Australia
Marzena Kryszkiewicz	Warsaw University of Technology, Poland
Wai Lam	The Chinese University of Hong Kong, Hong Kong

Yue-Shi Lee	Ming Chuan University, Taiwan
Carson K. Leung	University of Manitoba, Canada
Chengkai Li	The University of Texas at Arlington, USA
Chun-hung Li	Hong Kong Baptist University, Hong Kong
Feifei Li	Florida State University, USA
Jinyan Li	University of Technology Sydney, Australia
Ming Li	Nanjing University, China
Xiaoli Li	Institute for Infocomm Research, Singapore
Xue Li	The University of Queensland, Australia
Xuelong Li	University of London, UK
Zhenhui Li	Pennsylvania State University, USA
Kawuu W. Lin	National Kaohsiung University of Applied Sciences, Taiwan
Shou-de Lin	National Taiwan University, Taiwan
Fei Liu	Bosch Research, USA
Qingshan Liu	NLPR Institute of Automation Chinese Academy of Science, China
Shixia Liu	Microsoft Research Asia, China
Tie-Yan Liu	Microsoft Research Asia, China
David Lo	Singapore Management University, Singapore
Woong-Kee Loh	Sungkyul University, South Korea
Chang-Tien Lu	Virginia Polytechnic Institute and State University, USA
Hua Lu	Aalborg University, Denmark
Shuai Ma	Beihang University, China
Marco Maggini	Università degli Studi di Siena, Italy
Tao Mei	Microsoft Research Asia, China
Wagner Meira	Universidade Federal de Minas Gerais, Brazil
Toshiro Minami	Kyushu University Library, Japan
Pabitra Mitra	Indian Institute of Technology Kharagpur, India
Yang-Sae Moon	Kangwon National University, Korea
Yasuhiko Morimoto	Hiroshima University, Japan
Tsuyoshi Murata	Tokyo Institute of Technology, Japan
Richi Nayak	Queensland University of Technologies, Australia
See-Kiong Ng	Institute for Infocomm Research, A*STAR, Singapore
Wilfred Ng	Hong Kong University of Science and Technology, Hong Kong
Tadashi Nomoto	National Institute of Japanese Literature, Japan
Masayuki Numao	Osaka University, Japan
Manabu Okumura	Japan Advanced Institute of Science and Technology, Japan

Salvatore Orlando	University of Venice, Italy
Yonghong Peng	University of Bradford, UK
Jean-Marc Petit	Universite de Lyon, France
Vincenzo Piuri	Università degli Studi di Milano, Italy
Weining Qian	East China Normal University, China
Chotirat Ratanamatan	Chulalongkorn University, Thailand
Chandan Reddy	Wayne State University, USA
Patricia Riddle	University of Auckland, New Zealand
C. Sekhar	Indian Insitute of Technology, India
Hong Shen	Adelaide University, Australia
Jialie Shen	Singapore Management University, Singapore
Yi-Dong Shen	Chinese Academy of Sciences, China
Andrzej Skowron	University of Warsaw, Poland
Mingli Song	Zhejiang University, China
Yizhou Sun	Northeastern University, USA
Thepchai Supnithi	National Electronics and Computer Technology Center, Thailand
David Taniar	Monash University, Australia
Tamir Tassa	The Open University, Israel
Ivor Tsang	Nanyang Technological University, Singapore
Jeffrey Ullman	Stanford University, USA
Marian Vajtersic	University of Salzburg, Austria
Hui Wang	University of Ulster, UK
Jason Wang	New Jersey Science and Technology University, USA
Lipo Wang	Nanyang Technological University, Singapore
Xin Wang	University of Calgary, Canada
Raymond Wong	Hong Kong University of Science and Technology, Hong Kong
Jian Wu	Zhejiang University, China
Junjie Wu	Beihang University, China
Xindong Wu	University of Vermont, USA
Xintao Wu	University of North Carolina at Charlotte, USA
Xiaofeng Meng	Renmin University of China, China
Seiji Yamada	National Institute of Informatics, Japan
Min Yao	Zhejiang University, China
Mi-Yen Yeh	Academia Sinica, Taiwan
Dit-Yan Yeung	Hong Kong University of Science and Technology, China
Jian Yin	Sun Yat-Sen University, China
Jin Soung Yoo	IUPU, USA
Kennichi Yoshida	University of Tsukuba, Japan
Tetsuya Yoshida	Hokkaido University, Japan
Hwanjo Yu	Pohang University of Science and Technology, South Korea

Yifeng Zeng	Teesside University, UK
Bo Zhang	Tsinghua University, China
Daoqiang Zhang	Nanjing University of Aeronautics and Astronautics, China
Du Zhang	California State University, USA
Harry Zhang	University of New Brunswick, Canada
Junping Zhang	Fudan University, China
Mengjie Zhang	Victoria University of Wellington, New Zealand
Shichao Zhang	University of Technology, Australia
Wenjie Zhang	University of New South Wales, Australia
Ying Zhang	University of New South Wales, Australia
Zhongfei Zhang	Binghamton University, USA
Zili Zhang	Deakin University, Australia
Peixiang Zhao	Florida State University, USA
Yu Zheng	Microsoft Research Asia, China
Bin Zhou	University of Maryland Baltimore County, USA
Shuigeng Zhou	Fudan University, China
Wenjun Zhou	University of Tennessse - Knoxville, USA
Feida Zhu	Singapore Management University, Singapore
Xingquan Zhu	University of Technology Sydney, Australia

Table of Contents – Part II

Erratum

Table of Contents – Part I

ProCF: Probabilistic Collaborative Filtering for Reciprocal Recommendation

Xiongcai Cai, Michael Bain, Alfred Krzywicki, Wayne Wobcke,
Yang Sok Kim, Paul Compton, and Ashesh Mahidadia

School of Computer Science and Engineering,
The University of New South Wales, Sydney NSW 2052, Australia
{xcai,mike,alfredk,wobcke,yskim,compton,ashesh}@cse.unsw.edu.au

Abstract. Similarity in people to people (P2P) recommendation in social networks is not symmetric, where both entities of a relationship are involved in the reciprocal process of determining the success of the relationship. The widely used memory-based collaborative filtering (CF) has advantages of effectiveness and efficiency in traditional item to people recommendation. However, the critical step of computation of similarity between the subjects or objects of recommendation in memory-based CF is typically based on a heuristically symmetric relationship, which may be flawed in P2P recommendation. In this paper, we show that memory-based CF can be significantly improved by using a novel asymmetric model of similarity that considers the probabilities of both positive and negative behaviours, for example, in accepting or rejecting a recommended relationship. We present also a unified model of the fundamental principles of collaborative recommender systems that subsumes both user-based and item-based CF. Our experiments evaluate the proposed approach in P2P recommendation in the real world online dating application, showing significantly improved performance over traditional memory-based methods.

Keywords: Social Network Mining, Recommender Systems.

1 Introduction

Memory-based collaborative filtering (CF) is the basis of many commercial recommender systems, of which Amazon's [13] item-based approach is probably the best known. In this work we present a unified framework incorporating both item-based and user-based CF and within it develop a novel probabilistic method of similarity that overcomes some of the limitations of previous approaches.

Conventional *recommender systems* attempt to discover user preferences over items by modelling the relation between users and items. The aim is to recommend items that match the *taste* (likes or dislikes) of users in order to assist the active user, i.e., the user who will receive recommendations, to select items from an overwhelming set of choices. It is used to 1) predict whether a particular user will like a particular item (a prediction problem), or 2) identify a set of N items that will be of interest to a certain user (a Top-N recommendation

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 1–12, 2013.

problem). Recently, recommender systems have also been extended to *people to people (P2P)* recommendation to model the relation between the active user and other users by finding user preferences over other users.

Assuming that users with similar tastes would rate items (or other users) similarly, memory-based collaborative filtering (CF) methods recommend items based on heuristic aggregated user preferences for items, independent of the availability of item descriptions. In this paper we formalise memory-based CFs in a uniform way that allows the derivation of a *probabilistic* method, ProCF, that is shown to improve performance in a P2P recommendation application.

Section 2 discusses related work. Section 3 defines the problems. Section 4 develops a probabilistic approach for both recommendation and ranking. Experimental evaluation is in Section 5 and we conclude in Section 6.

2 Related Work

CF algorithms fall into two categories: model-based and memory-based approaches. Model-based CF [1,2,10,16] uses the collection of ratings to learn a model, which is then used to make rating predictions. Although model-based methods have reported higher accuracy of recommendation than memory-based approaches, there are some limitations. These methods are computationally expensive since they usually require all users and items to be used in creating models, and the number of users and items is typically large. Memory-based CF is popular in many commercial recommender systems, being effective and easy to implement. Memory-based approaches [2,11,13,18] make rating predictions based on the entire set or a sample of items previously rated by users. The unknown rating value $r_{c,s}$ of the active user c for an item s is typically computed as an aggregate of the ratings of users similar to c for the same item s. This aggregate can be an average or a weighted sum, where the weight is a distance that measures the similarity between users c_1 and c_2. By using similarity as a weight, more similar users make a greater contribution to a predicted rating.

In memory-based CF, similarity computation between items or users is essential. The definition of similarity measure varies depending on the recommendation application. Often the similarity between two users is based on the ratings of items both users have rated. Two of the most popular approaches are correlation [11,18] and cosine-based [2,17]. Extensions to these include default voting, inverse user frequency, case amplification, and weighted-majority prediction [2,7]. Usually these use heuristics to model the weights and are not able to handle the different rating scales of different users. Solutions to this problem include the adjusted weighted sum and preference-based filtering [14], which focuses on predicting the relative preferences of users instead of absolute rating values.

Memory-based probabilistic CF is an alternative. Yu *et al.* [20] use a mixture model for user preferences. Deshpande and Karypis [8] proposed conditional probability based similarity in item-based CF. These models only consider common purchase information, which causes the problem that frequently purchased items tend to have high conditional probabilities, leading to reduced diversity in recommendation [9]. Adding a scaling parameter to control for the effect of popular items in the model may help, but finding a suitable parameter value becomes challenging. Also, these methods are uni-directional, relying only on users' *taste*, so they are not applicable to P2P recommendation, which is reciprocal.

People recommenders deal with the problem of finding meaningful relationships among people or organisations. In online social networks, relationships can be friends [19] e.g., on Facebook, professional contacts [3] e.g., on LinkedIn, online dating [5,12], or jobs on employment websites [15]. The nature of these domains makes P2P recommender systems significantly different from traditional item to people (I2P) recommenders. The basic difference in the people recommender domain is the characteristic of *reciprocal* relationships.

3 Problem Statement

Recommender systems can be classified into two general classes: classical item to people recommender systems (I2PRec) and people to people recommender systems (P2PRec). In classical I2PRec, there are two types of entities, buyers (e.g., customers) and items (e.g., books, movies, songs). In recent P2PRec [5], there only exists a uniform entity type: users (e.g., online dating service subscribers, job seekers and employers). To distinguish the different roles in a recommendation, we use *subject*, $S = \{s_1, ..., s_{|S|}\}$, to refer to the recommendation recipient (e.g., customers in I2PRec and active partner seekers for P2PRec) and *object*, $O = \{o_1, ..., o_{|O|}\}$, to refer to the recommendation candidate (e.g., books in I2PRec and partner seekers in P2PRec). Recommender systems using CF methods rely on collaborative information. There are several types of collaborative information. One important distinction is between *explicit* (i.e., ratings, up and down votes) and *implicit* (i.e., clicks, purchases, contacts, replies) expressions of user preferences. Depending on the type of system, implicit information may be positive-only, i.e., no recorded negative preference observations, or positive-and-negative, i.e., both positive and negative preference observations are available.

In I2PRec, collaborative information used in traditional CF is merely based on the behaviours of subjects, i.e., the preference of buyers determines the transactions that represent the collaborative information. However, in P2PRec, collaborative information usually depends on behaviours of *both* subject and object, since the relationship between the subject and the object can only be established when both parties agree on it, denoted by a successful interaction (i.e., the subject makes contact to express interest and gets positive feedback from the object). We use so^+ to refer to this, representing the establishment of a successful interaction. Similarly, so^- refers to an unsuccessful interaction. This requires P2PRec to consider collaborative information based on behaviours of both subject and object rather than only those of subject, which consequently prevents traditional I2PRec from solving the P2P recommendation problem [4].

The task of P2P recommendation from implicit, positive and negative, preferences is to rank the objects from a candidate set O^c for a subject according to the probability of establishing a relationship between them based on the collaborative information. This task is related to, but distinct from, rating prediction,

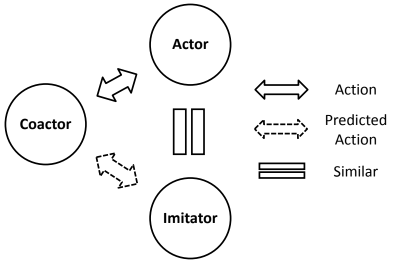


Fig. 1. A general framework for CF. In user-based CF (UBCF) (resp. item-based CF (IBCF)), Coactor is the candidate entity (active entity), Actor the entity similar to the active entity (candidate entity), Imitator the active entity (candidate entity). (Section 3)

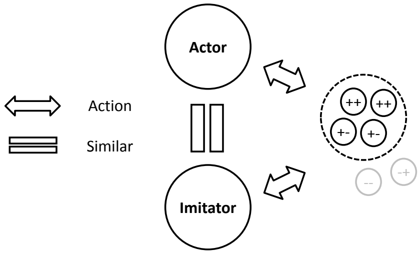


Fig. 2. Construction of probabilistic similarity. Actor and Imitator both interact with Coactor entities, with four possible outcomes, e.g., the label (++) means that both interactions have a positive outcome.

where the task is to predict how much a subject will like an object. Therefore, a ranked list of candidate objects for each subject, rather than an explicit rating, is the output of a typical P2P recommender.

Similarity. Memory-based CF approaches to recommender systems differ according to the method used to compute the similarity between the various subjects/objects, which consequently determines the overall performance of the recommender. In general, the similarity between two entities should be high if they have interacted with a large set of others in common and low if few in common. In this case, the similarity between two entities is *symmetric* since the common set is shared with both entities and no other information is considered. But this may not be sufficient, since other subjects/objects that were not interacted with in common by the similar pairs, but only by one in the pair, can have impact on the similarity and thus the recommendation. When using information from such entities not interacted with in common by the similar pairs, similarity becomes naturally *asymmetric*, since each entity in the pair can have a different set contacted by it alone. For example, in P2P recommendation, Bob may prefer Alice since she is younger than 30 years old and likes music, while Charlie could also prefer Alice, but only because she has age less than 30. Thus, Bob is similar to Charlie since they have a common preference and vice versa. In symmetric similarity, similarity to each other is the same since they prefer the same woman. However, Diana, who is 20 years old but hates music could be preferred by Charlie but not by Bob, since Bob prefers women who like music. This shows that Bob and Charlie's preferences are not the same, and moreover, that Bob's similarity to Charlie is different from Charlie's to Bob, and thus *asymmetric*.

Similarity can be represented by similar pairs. There are two roles within a similar pair: *actor* and *imitator*, as illustrated in Fig. 1. To define those two roles, we first define the *coactor*:

Definition 1. *A* coactor *is the entity who interacts with both entities in a similar pair, i.e., either the candidate of recommendation in user-based CF or the active entity in item-based CF.*

In the example, Bob is similar to Charlie. If Emma is preferred by Bob and likes music, we can recommend Emma to Charlie by user-based CF. Or if Emma prefers Bob, we can recommend Charlie to Emma by item-based CF. In both cases, we call Emma a *coactor*, the person who either receives recommendations or is recommended. The two roles within a similar pair are defined as:

Definition 2. *An* actor *is the entity within a similar pair that provides collaborative information, i.e., the entity who actually interacted with the coactor.*

Bob is a *actor* in our example since he interacted with the coactor Emma.

Definition 3. *An* imitator *is the other entity in the similar pair with similar behaviour to the actor, i.e., either the active entity in user-based CF or candidate in item-based CF.*

Charlie is an *imitator* in our example for his similarity to the actor Bob. Given the above definitions, we formally define symmetric and asymmetric similarity:

Definition 4. *A* symmetric similarity *is a measurement of the amount of commonality in the behaviour of an actor and its imitator.*

Definition 5. *An* asymmetric similarity *is a measurement of the* combined *(e.g., summed) amount of commonality and difference in the behaviour of an actor and its imitator.*

These definitions subsume both user-based and item-based CF into a novel unified model, which reveals the fundamental principles that drive the formation of collaborative recommender systems by discovering the relation among the entities involved in the collaborative recommendation (Fig. 1).

4 Methods

In this section, we describe the probabilistic CF framework and apply it to the task of P2P recommendation from implicit, positive and negative, feedback.

4.1 Conditional Probabilistic Similarity

We define probabilistic similarity in the context of item-based CF as the probability that an action succeeds given that the subject of recommendation had a positive interaction with one of the similar objects and the subject attempted the same action with another similar object. This definition is sufficiently general to cover both item-based and user-based CF (Fig. 2). It also covers people to people recommendation, in which there are two types of actions: acceptance, i.e., positive feedback, and rejection, i.e., negative feedback. Depending on the type of action we can then define the probability of acceptance or rejection:

Definition 6. *The acceptance probability $P(i|j)$ is the probability a positive action will occur with entity pair i given a positive action occurred with pair j:*

$$P(ic^+|ac^+) = P(ic^+|ac^+, ic) = \frac{P(ac^+, ic^+)}{P(ac^+, ic)} = \frac{|ae^+ \cap ie^+|_{e \in E}}{|ae^+ \cap ie|_{e \in E}} \tag{1}$$

where P is the probability, E the set of entities in the training set, i the imitator in Definition 3, c the coactor in Definition 1 and a the actor in Definition 2.

Definition 7. *The rejection probability $P(x|y)$ is the probability a negative action will occur with entity pair x given a positive action occurred with pair y:*

$$P(ic^-|ac^+) = \frac{|ae^+ \cap ie^-|_{e \in E}}{|ae^+ \cap ie|_{e \in E}} \tag{2}$$

Note that the conditional probability-based similarity of [8] occurs as a special case of this framework when there is only a single type of action. For example, in item-based CF the similarity between two items is the conditional probability that one item will be purchased given that the other has been purchased. This is equivalent to $P(ic|ac)$, which is obtained by dropping the sign of the interactions and simplifying in either of Definitions 1 or 2, when they become identical.

To compute the acceptance and rejection probability, as shown in Fig. 2, we first find the number (C) of common entities each of which has a positive action with the actor, and either a positive or negative action with the imitator. Here the common entities with negative actions with respect to the actor ($-+$ or $--$), shown greyed-out in Fig. 2, are not taken into account. Secondly, we count how many of those have positive actions in common with the imitator (A), and how many of them have negative actions with the imitator (B). Then, the acceptance probability is (A) divided by (C) and the rejection probability is (B) divided by (C). For the example in Fig. 2, the number of such common entities (in the dashed circle) is 4. Out of these the number of entities having positive actions with the imitator ($++$) is 2 and those having negative actions ($+-$) is 2. Therefore, the acceptance probability is $2/4 = 0.5$ and the rejection probability is $2/4 = 0.5$. In this case the actor and imitator as similar entities have equivalent acceptance and rejection probability with respect to the coactors.

4.2 Probability Residue

We consider both acceptance probability and rejection probability in estimating the impact of the preferences of similar entities. If we have a similar pair for which the acceptance probability (P^+, for short) is greater than the rejection probability (P^-), we would naturally be inclined to decide that the similar pair will contribute to the acceptance of the recommendation of a similar entity. Conversely, if P^- is greater than P^+, we would be inclined to consider rejection. To justify this decision procedure, we can calculate the probability of error. Whenever we observe a particular similar pair i, the probability of an error is:

$$P(error|i) = \begin{cases} P^- \text{ if we decide acceptance} \\ P^+ \text{ if we decide rejection} \end{cases} \tag{3}$$

Given i we can minimise the probability of error by deciding acceptance if $P^+ > P^-$ and rejection otherwise. Thus, the *probability residue* is the contribution of each similar entity to the ranking by minimising the decision error:

Definition 8. *The probability residue is the difference between the acceptance probability and rejection probability:*

$$\omega_{ai} = P(ic^+|ac^+) - P(ic^-|ac^+) = P^+ - P^- \tag{4}$$

The probability residue reflects the balance of the acceptance and rejection probability of an interaction between a pair of entities. It measures the degree to which the interaction departs from random, i.e., increases the possibility of either success or failure. Thus, a similar entity will contribute towards a positive rating and thus success if the probability residue $\omega_{ai} > 0$, contribute towards a negative rating and thus failure if the probability residue $\omega_{ai} < 0$, or not contribute to recommendation at all if the probability residue $\omega_{ai} = 0$.

For the example in Fig. 2 the probability residue is 0 and thus will be ignored in recommendation, since the actor as a similar entity to the imitator, as an active entity or candidate, has equivalent acceptance and rejection probability.

4.3 Rating

Rating of candidates as active entities is then based on probability residues. It is the sum of probability residues of all the entities similar to the current active entity corresponding to a candidate:

$$r = \sum_{k \in S} \omega_k \tag{5}$$

where S is the set of similar entities and ω_k the probability residue of Definition 8.

Theorem 1. *If ω is a probability residue and $|S|$ is the number of similar entities, then the candidate rating function r of Equation 5 is a non-monotonically increasing function on the number of similar entities $|S|$.*

Proof. Since ω_i could be negative, for all $|S_1|$ and $|S_2|$ such that $|S_1| \leq |S_2|$, one could have $r(|S_1|) \geq r(|S_2|)$ according to Equation 5. Thus function r does not preserve the ordering and is non-monotonic.

Notice that this non-monotonic characteristic of the rating function is desired in recommender systems. In conventional recommender systems [11,13,8], the rating functions are usually monotonically increasing on the number of similar entities. This is a problem since it will cause popular (i.e., preferred by a large number of entities) or active (i.e., preferring a large number of entities) entities to have higher rating than others since popular or active entities have usually more similar entities than other entities.

More specifically, popular objects are preferred by a large number of subjects and thus have more chance to be co-preferred with other objects, which makes popular objects have more similar objects than non-popular ones. Similarly, active subjects prefer a large number of objects and thus have more chance to co-prefer with other subjects, which makes active subjects have more similar subjects than non-active ones. However, since the rating function defined in Equation 5 is non-monotonic it is *not* necessarily increasing for popular or active entities. Therefore, by using the probability residue defined above, we are able to avoid the common problem of favouring popular entities in recommendation, and therefore increase the diversity as shown in Section 5. Promoting novel recommendation generates global diversity and improves user experience [6].

A ranked candidate list is then generated by descending sort of all candidates on rating. For tied ratings, we have two steps: (i) favour the candidate with a greater total number of interactions used in calculating similarity; and if this is also tied (ii) favour the candidate with more contributed similar pairs. In these rare cases the increased support means favouring more reliable ratings.

4.4 Summary of the ProCF Algorithm

The proposed framework is realised in the ProCF algorithm. It constructs a similarity table and then generates a recommended candidate list for each subject. Specifically, to construct the similarity table, ProCF collects all similar pairs, each of which has at least one common coactor, and then assigns each pair a probability residue value according to Equation 4. To generate recommendations for an active entity, ProCF finds each imitator in the similarity table for which all pairs of their corresponding actors were interacted with by the active entity. It then computes a rating for the imitator according to Equation 5 and adds the imitator to the recommendation list.

The complexity of ProCF is approximately $O(N)$, and $O(N^2)$ in the worst case, where N is the number of entities in the training set, since it examines N entities and up to $N - 1$ other entities for each entity. However, because the average entity interaction vector is extremely sparse, the performance of the algorithm tends to be closer to $O(N)$ in practice. Scanning every entity is approximately $O(N)$ rather than $O(N^2)$ because almost all entity interaction vectors contain only a small number of interactions with other entities. Although there are a few entities who interact with a significant percentage of all other entities, each still only requires $O(N)$ processing time.

5 Experiments

In these experiments, we evaluate the proposed approach on people to people recommendation (Top-N recommendation) in a demanding real-world social network data set. We compare the proposed probabilistic CF method to several conventional recommendation strategies based on a set of common evaluation metrics. Owing to space restrictions, we can only summarize our results, which will be detailed in an extended version of the paper.

Datasets. Data was collected from a commercial online dating site. In online dating, people are looking for potential partners. A user contacts people they like by sending messages. Receivers of messages then have options to reply, positively if they like the sender, negatively if they do not like the sender or are not sure, or they may just not reply. Specifically, the data contains interaction records, each of which represents a contact by a tuple containing the identities of the sender and receiver and whether the contact was accepted (positive response from receiver to sender) or not. The former case is denoted a successful or positive interaction, otherwise it is unsuccessful or negative.

The training set covered a four week period in February, 2010 and the test set a one week period from the first of March, 2010 (test results from a three week period from the same date were essentially identical and are omitted due to lack of space). Training and test sets contained all users with at least one contact in the respective periods. The training (resp. test) sets contained 166699 (95814) users with a total of 1710332 (436128) interaction tuples. Of these 264142 (66482) were positive interactions and 1446190 (369646) were negative (including non-replies). The *default success rate* (DSR), the proportion of interactions that were positive, was 15.4% (15.2%).

Methodologies. We compare the proposed algorithm with the P2P *Best 2CF+* method [12] on their evaluation metric and a new metric defined below. As far as we are aware this is the best-performing published CF method for recommendation in online dating. Model-based methods such as matrix factorisation methods were also tested but were not able to handle such a large dataset.

We trained ProCF and Best 2CF+ using the above training set. The learned model for each approach was then tested by generating the Top-N recommendations for each user in the test set. Finally, recommendations from ProCF and Best 2CF+ were evaluated as described below and the results are compared.

Evaluation Metrics. Evaluation in this domain is more complex than standard CF applications. Metrics used to capture key aspects of system performance are defined as follows. *Precision/Success Rate(SR)*: proportion of interactions predicted to be successful that were actually successful to all predicted successful interactions (PSI). *Default Success Rate(DSR)*: proportion of actual successful interactions (SI) to all interactions in the dataset. *SRI*: ratio of SR to DSR. *Recall and F Value*: recall is proportion of true PSI to all true SI. F Value is defined by $F = \frac{2*Precision*Recall}{Precision+Recall}$. *Accept Rate and ARI*: accept rate (AR) is proportion of true PSI to all PSI with either positive or negative reply. ARI is the ratio of AR to default accept rate without recommendation. *Reject Rate and RRI*: reject rate (RR) is proportion of false PSI with negative reply to all PSI with any reply. RRI is the ratio of RR to default reject rate without recommendation.

We use SRI to test how likely recommendation is to help the active user to have a positive interaction. Recall and F value tests how different user behaviour based on recommendation is to default. Finally, ARI and RRI test responses of the recommended user when the active user follows the recommendation.

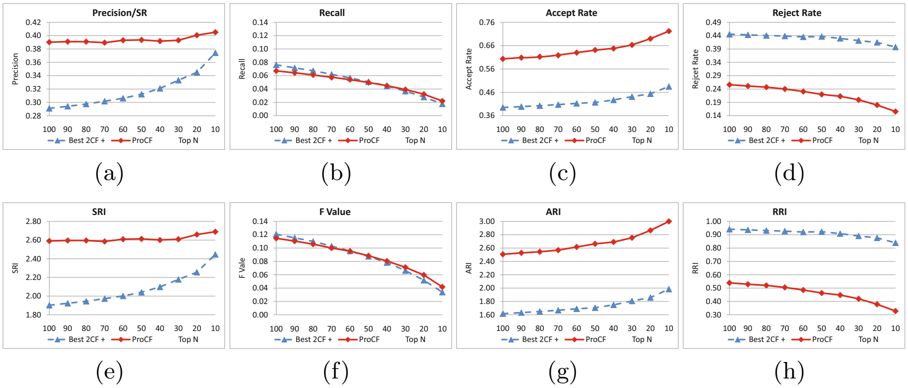
Table 1. Average Positive Reply Rate per Receiver

	baseline	all	100	90	80	70	60	50	40	30	20	10
B2CF	0.372	0.370	0.370	0.370	0.370	0.371	0.371	0.371	0.371	0.372	0.374	0.378
ProCF	0.372	0.369	0.370	0.370	0.370	0.370	0.370	0.370	0.370	0.370	0.372	0.379

Results. A comparison of ProCF and Best2CF+ on the test set in terms of the evaluation metrics is shown in Fig. 3. Clearly ProCF outperforms Best 2CF+ on precision for all Top-N recommendations; the most significant comparative improvement is on Top 100 where ProCF outperforms Best 2CF+ by 34%. SRI shows that although Best 2CF+ improves the baseline performance of the system for all Top-N, ProCF achieves greater improvement. Since [12] show that Best 2CF+ outperforms traditional CFs on P2P recommendation, ProCF has a clear advantage. This suggests that considering both positive and negative collaborative information in creating similar pairs using probability residue leads to recommending users with higher probabilities of successful interaction with the active user. Recall and F Value improvements for ProCF indicates greater reliability in recommendation. Also, ProCF shows increased accept rate and reduced reject rate for the most highly ranked users. This supports the hypothesis that by looking at the difference between positive and negative information, ProCF can down-rank candidates with higher reject rate while up-ranking those with higher accept rate.

In Table 1, we show the average positive reply rate (APRR) per receiver over all recommended users compared to the those over all users in datasets. We also compared ProCF to Best 2CF+ on APRR with ProCF shown in bold in the figure. We can see from the results that both Best 2CF+ and ProCF have a similar APRR to the baseline over all Top-N. Best 2CF+ achieved smaller APRR than the baseline except from Top 20 and 10 while ProCF achieved even smaller APRR than Best 2CF+ except only Top 10. This indicates that ProCF does not recommend users who have higher positive reply rate. In contrast, it recommends users with lower positive reply rate in general. A recommender system that prefers recommending users with higher positive reply could improve the success rate. However, ProCF does not have a concentration bias on those users and thus does not commit the problem of recommending very frequent items as in [8]. Diversity in P2P recommendation is important; for example, recommending people who always say “yes” to any contact is poorly personalized and leads to poor recommendation performance. ProCF’s achievement of high success rate while maintaining diversity is a significant result.

Discussion. The experiments have shown that ProCF implementing our probabilistic similarity function significantly outperforms Best 2CF+ on all evaluation metrics used. This proves that ProCF also outperforms all standard CF methods and combined CF methods evaluated in [12]. Note that ProCF achieved its performance by a single improved CF method not by any combination of CFs or profile-based methods. The characteristics of ProCF and its good performance suggest it could be used as an improved standard CF method to be integrated



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 3. Results on Test Set (1-7 March, 2010). ProCF significantly outperformed Baseline and *Best 2CF+* in precision, accept rate and reject rate on all Top-N evaluations and improved the recall/F value from Top 50 to Top 10). (For reject rate/RRI, the lower the better. For others, the higher the better.)

into existing common recommendation frameworks for increased performance. Beside good performance, ProCF has the advantage of being simple, easy to implement and fast. All presented algorithms were implemented using SQL in Oracle 11. ProCF required about 1 hour for training and several minutes for testing on a workstation with 64-bit Windows 7 Professional, 2 processors of Intel(R) Xeon(R) CPU x5660@2.80GHz and 32GB RAM.

6 Conclusion

We presented a general and straightforward framework, ProCF, for recommender systems. Although ProCF is in general applicable to both I2P and P2P recommendation, this paper focuses on P2P recommendation only. We demonstrated the usefulness of ProCF in a set of extensive experiments. The experiments were conducted on demanding real world datasets collected from a commercial social network site. The experimental evaluation of ProCF shows that it is suitable for P2P recommendation. The comparative evaluation to two of the best CF methods on this task shows that ProCF outperforms the best CF-based method for P2P recommendation. We also showed that ProCF retains diversity of recommendation while providing higher accuracy recommendation. It does not only recommend a small group of users with high positive reply rate.

An appealing property of our framework is its simplicity and modularity. Because it follows a standard CF framework with its improved similarity and ranking functions, it can be applied or integrated into existing CF recommender systems to improve system performance.

In the future, we will extend this work to test ProCF using other probabilistic functions for similarity and other distribution functions for calculating probability residue. We will also investigate the integration of profile based approach into ProCF for even better recommendation performance.

Acknowlegement. This work was funded by Smart Services Cooperative Research Centre. We would also like to thank our industry partners for providing the datasets.

References

1. Billsus, D., Pazzani, M.J.: Learning collaborative information filters. In: Proc. ICML 1998, pp. 46–54 (1998)
2. Breese, J., Heckerman, D., Kadie, C.: Empirical analysis of predictive algorithms for collaborative filtering. In: Proc. UAI 1998, pp. 43–52 (1998)
3. Bull, S., Greer, J.E., McCalla, G.I., Kettel, L., Bowes, J.: User modelling in i-help: What, why, when and how. In: Bauer, M., Gmytrasiewicz, P.J., Vassileva, J. (eds.) UM 2001. LNCS (LNAI), vol. 2109, pp. 117–126. Springer, Heidelberg (2001)
4. Cai, X., Bain, M., Krzywicki, A., Wobcke, W., Kim, Y., Compton, P., Mahidadia, A.: Collaborative Filtering for People to People Recommendation in Social Networks. In: Li, J. (ed.) AI 2010. LNCS, vol. 6464, pp. 476–485. Springer, Heidelberg (2010)
5. Cai, X., Bain, M., Krzywicki, A., Wobcke, W., Kim, Y., Compton, P., Mahidadia, A.: Learning collaborative filtering and its application to people to people recommendation in social networks. In: Proc. ICDM 2010, pp. 743–748 (2010)
6. Castells, P., Vargas, S., Wang, J.: Novelty and diversity metrics for recommender systems: Choice, discovery and relevance. In: DDR 2011 Workshop at ECIR 2011 (2011)
7. Delgado, J., Ishii, N.: Memory-based weighted-majority prediction for recommender systems. In: ACM SIGIR 1999 Workshop Recommender Systems (1999)
8. Deshpande, M., Karypis, G.: Item-based Top-N Recommendation Algorithms. ACM Transactions on Information Systems 22(1), 143–177 (2004)
9. Fleder, D.M., Hosanagar, K.: Recommender systems and their impact on sales diversity. In: Proc. ACM Electronic Commerce, pp. 192–199 (2007)
10. Hofmann, T.: Collaborative filtering via gaussian probabilistic latent semantic analysis. In: Proc. ACM SIGIR 2003, pp. 259–266 (2003)
11. Konstan, J., Miller, B., Maltz, D., Herlocker, J., Gordon, L., Riedl, J.: Grouplens: Applying collaborative filtering to usenet news. In: CACM, pp. 77–87 (1997)
12. Krzywicki, A., Wobcke, W., Cai, X., Mahidadia, A., Bain, M., Compton, P., Kim, Y.: Interaction-based collaborative filtering methods for recommendation in online dating. In: Chen, L., Triantafillou, P., Suel, T. (eds.) WISE 2010. LNCS, vol. 6488, pp. 342–356. Springer, Heidelberg (2010)
13. Linden, G., Smith, B., York, J.: Amazon.com recommendations: item-to-item collaborative filtering. IEEE Transactions on Internet Computing 7(1), 76–80 (2003)
14. Liu, T.Y.: Learning to Rank for Information Retrieval. Springer (2011)
15. Malinowski, J., Keim, T., Wendt, O., Weitzel, T.: Matching people and jobs: A bilateral recommendation approach. In: Proc. HICSS 2006, vol. 6, p. 137.3 (2006)
16. Pavlov, D., Pennock, D.: A maximum entropy approach to collaborative filtering in dynamic, sparse, high-dimensional domains. In: NIPS 2002, pp. 1441–1448 (2002)
17. Sarwar, B., Karypis, G., Konstan, J., Reidl, J.: Item-based collaborative filtering recommendation algorithms. In: Proc. WWW 2001, pp. 285–295 (2001)
18. Shardanand, U., Maes, P.: Social information filtering: Algorithms for automating "word of mouth". In: Proc. HCI 1995, pp. 210–217 (1995)
19. Terveen, L., McDonald, D.W.: Social matching: A framework and research agenda. ACM Transactions on Computer-Human Interaction 12(3), 401–434 (2005)
20. Yu, K., Schwaighofer, A., Tresp, V., Xu, X., Kriegel, H.: Probabilistic memory-based collaborative filtering. IEEE TKDE 16(1), 56–69 (2004)

Product and User Dependent Social Network Models for Recommender Systems

Min Li, Zhiwei Jiang, Bin Luo, Jiubin Tang, Qing Gu*, and Daoxu Chen

Department of Computer Science, Nanjing University,
Nanjing, Jiangsu 210000, China
minli@software.nju.edu.cn, jzwpm@163.com, tjb@telecomjs.com,
{luobin,guq,cdx}@nju.edu.cn

Abstract. Social network based applications such as Facebook, Myspace and LinkedIn have become very popular among Internet users, and a major research problem is how to use the social network information to better infer users' preferences and make better recommender systems. A common trend is combining the user-item rating matrix and users' social network for recommendations. However, existing solutions add the social network information for a particular user without considering the different characteristics of the products to be recommended and the neighbors involved. This paper proposes a new approach that can adaptively utilize social network information based on the context characterized by products and users. This approach complements several existing social network based recommendation algorithms and thus can be integrated with existing solutions. Experimental results on Epinions data set demonstrate the added value of the proposed solution in two recommendation tasks: rating prediction and top-K recommendations.

1 Introduction

Recommender Systems have achieved great success and are becoming increasingly popular in real world applications. For example, online stores, such as Amazon and Netflix, provide customized recommendations for products or services based on a user's history. Many techniques have been proposed to make recommendations for the users, among which collaborative filtering is one of the most popular approaches. The task of collaborative filtering is to predict the utility of items to a particular user based on the user's history and other users' ratings.

With the increasing popularity of social network based applications such as Facebook, Myspace and LinkedIn, how to make recommendations with additional information from a user's social network has become an important research topic. In real life, we often turn to our friends for some recommendations. Besides, people with close relationship are likely to have similar tastes. Therefore, a user's social network may have two effects in the real world: help us infer users' preferences and influence users' behaviors. Hence, social network info might be an important element that recommender algorithms can take advantage of. Recently, several researchers have started to tackle this problem [11][10][9].For

* Corresponding author.

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 13–24, 2013.

example, Jamali et al. proposed a a model-based approach utilizing matrix factorization techniques and incorporating trust propagation mechanisms [3]. Konstas et al. adopt a Random Walk framework and focus on investigating the role of additional relationships, such as friendships and social tags [6].

However, most of prior research only focused on a single-domain recommendation and thus the solutions are less likely to work well in open domain recommenders systems. There are three differences between the two kinds of recommender systems: 1) Data is sparser in the open-domain systems. Open-domain systems have much more items but less user feedback. That means the user-item rating matrix is sparser in open-domain systems. Thus traditional collaborative filtering cannot achieve as good performance as in the single-domain systems. 2) Data distribution varies according to the different domains. For example, in the Epinions data set, online stores get more reviews(average 10 ratings/item), yet books tend to get less reviews(average 2 ratings/item).[1] 3) The social network structure is more complicated than the single-domain system. Social network has been used to measure users' similarities and infer users' preferences in recommender systems. Most of prior research assumed that those people trusted by same user have the same influence for the user. However in the real world, people always selectively adopt others' opinions. Some persons are good at software, some persons are good at sports. People will consult different persons due to the products they want to purchase. Each person may involve in multiple social networks, we shouldn't consider them equally.

Recently, Hao et al.[8] introduced a framework combining social networks and collaborative filtering techniques for recommendation in an open domain data set *epinions.com*. However, similar to existing research on social network based recommender systems, their solution also combines the information using a static weight, without considering how to balance the weights between user-item ratings and social network information based on the context.

Motivated by early research on social network based recommender systems, this paper focuses on a problem that existing solutions have not addressed: how to differentiate the effect of social network info based on recommendation context. Without loss of generality, we focus on three variables that characterize the context: item category, the number of observed ratings for the user and characteristics of the neighbors. Our experiments are based on these three characteristics. We propose a solution to modify some existing social network based recommendation algorithms so that the context could be considered.

Based on experimental results, we found: 1) users' social networks influence users' behaviors and are useful for inferring users' preference; 2) how to balance the weights between user-item ratings and social network information is dependent on the recommendation context and neighbors involved. Our proposed approaches using adaptive weights can capture the recommendation context and thus outperform the approaches using a static weight; 3) utilizing social network information can help overcome the negative effect of rating variance, especially

[1] Based on the statistics of our crawled data.

in an open-domain recommender system; 4) weighted differentiation of each individual in a social network can better model the influence of the social network.

2 Social Recommendation Approaches

Our approach is to start with state-of-the-art social network recommendation algorithms, modify them so that product and neighborhood characteristics will be considered when we trade off the predicted user preferences (without considering social information) and user's neighbors' preferences.

Assume there are N items, M users in a recommender system. The rating of user i for item j is denoted by $r_{i,j}$. All the ratings from users to items are denoted by a user-rating matrix $R = \{r_{i,j}\}$. For some recommender systems, users are connected in a social network. For example, if user i selects user k as a trustable person or his/her friend, there is a directed connection from user i to user k. This network can be represented as a $M \times M$ matrix $S = \{s_{i,k}\}$, where $s_{i,k}$ denotes how well user i trust user k. In the simplest case, $s_{i,k} = 1$ means user i trusts user k, otherwise 0. The task is to recommend a list of items to a user, and good items are those that user is likely to purchase, rate high, or click.

2.1 Singular Value Decomposition

Singular Value Decomposition(SVD) is a widely used collaborative filtering algorithm. The central idea is factorizing the user-item rating matrix into low-rank approximation based on low-dimensional hidden representations of users and items, then utilizing them to predict the missing values in the rating matrix. Let $U \in \mathbb{R}^{D\times M}$ and $V \in \mathbb{R}^{D\times N}$ be latent user and item matrices, with column vectors $\mathbf{u}_i$ and $\mathbf{v}_j$ representing the latent/hidden vectors of user i and item j respectively. D is the dimension of latent vectors. There are various ways to find the latent representations of users and items. We can view it as a statistical modeling problem, where the observed ratings are generated as follows [12]

$$p(R|U, V, \sigma^2) = \prod_{r_{i,j}\in R} \mathcal{N}(r_{i,j}|\mathbf{u}_i^T\mathbf{v}_j, \sigma^2) \tag{1}$$

where $\mathcal{N}(x|\mu, \sigma^2)$ is a Gaussian distribution with mean μ and variance σ^2. The dot product of latent user and item vectors $\mathbf{u}_i^T\mathbf{v}_j$ is the expected mean of rating $r_{i,j}$. The latent vectors are assumed to be generated independently from Gaussian distributions of zero-mean: $p(U|\sigma_u^2) = \prod_{i=1}^{M} \mathcal{N}(\mathbf{u}_i|0, \sigma_u^2\mathbf{I})$ and $p(V|\sigma_v^2) = \prod_{j=1}^{N} \mathcal{N}(\mathbf{v}_j|0, \sigma_v^2\mathbf{I})$ where σ_u is the variance of the Gaussian distribution for users and σ_v is the variance of the Gaussian distribution for items. $\mathbf{I}$ is an identity matrix. Hence, the posterior distribution over the user and item latent vectors is given by

$$p(U, V|R, \sigma^2, \sigma_u^2, \sigma_v^2) \propto \prod_{r_{i,j}\in R} \mathcal{N}(r_{i,j}|\mathbf{u}_i^T\mathbf{v}_j, \sigma^2)\times\prod_{i=1}^{M}\mathcal{N}(\mathbf{u}_i|0, \sigma_u^2\mathbf{I})\times\prod_{j=1}^{N}\mathcal{N}(\mathbf{v}_j|0, \sigma_v^2\mathbf{I})$$

We can find $\mathbf{u}_i$ and $\mathbf{v}_j$ by maximizing the above posterior likelihood. The rating for user i and item j, if not available, can be predicted as $r_{i,j} = \mathbf{u}_i^T\mathbf{v}_j$.

2.2 Factorization with Social Network

In trust-aware recommender systems, users express trust for other users. When user u trusts user k, they may have similar preference to some extent, or user k may affect user u's decisions. Social Trust Ensemble is a probabilistic framework that naturally fused users' tastes and their trusted friends' favors [8]. In this framework, the conditional distribution over the observed ratings is modeled as:

$$\begin{aligned} p(R, U, V|S, \sigma^2,\ \sigma_U^2, \sigma_V^2) \propto & \prod_{i=1}^{M} \mathcal{N}(\mathbf{u}_i|0, \sigma_u^2\mathbf{I}) \times \prod_{j=1}^{N} \mathcal{N}(\mathbf{v}_j|0, \sigma_v^2\mathbf{I}) \\ & \times \prod_{r_{i,j}\in R} [\mathcal{N}(r_{i,j}|(\alpha_{i,j}\mathbf{u}_i^T\mathbf{v}_j + (1-\alpha_{i,j}) \sum_{k\in\tau(i)} s_{i,k}\mathbf{u}_k^T\mathbf{v}_j), \sigma^2)] \end{aligned} \tag{2}$$

The model assumes the ratings are generated from different Gaussian distributions. The mean of the Gaussian distribution that generates a rating $r_{i,j}$ is determined by the latent vectors of user $\mathbf{u}_i$ and item $\mathbf{v}_j$ as well as the users in user i's social network, which is denoted as $\tau(i)$. The contributions from the two parts are weighted by the parameter $\alpha_{i,j}$. In [8], $\alpha_{i,j}$ is fixed as the same value for different user i and item j. It ignores the recommendation context associated with ratings. We will discuss this issue and propose a new solution later.

2.3 Adaptive Weights Based on User and Product Characteristics

In formula (2), $\alpha_{i,j}$ and $s_{i,k}$ balance the information from users' own characteristics and their friends' favors. $\alpha_{i,j}$ controls how much the model should trust the user vs. the neighbors, and $s_{i,k}$ controls how much one should trust user k. A straightforward way is to define a fixed value for all the $\alpha_{i,j}$ [8]. For instance, $\alpha_{i,j} = 0.4$ for all $\langle i, j\rangle$ pairs means whatever the situation is, a user's own hidden representation contributes 40% and social network contributes 60%. However, in real life, how much to trust others depends on many factors. For example, if item k is a movie, user i may ask his/her friends or read reviews before watching it. If item k is a hard drive, the user i may have clear idea about his/her preferences (size, price range) and can judge the quality easily without consulting friends.

To make $\alpha_{i,j}$ context-sensitive, we propose to set the value of $\alpha_{i,j}$ based on the features of user i and item j using the following sigmoid function:

$$\alpha_{i,j} = sigmoid(\mathbf{w}^T\mathbf{f}_{i,j}) \tag{3}$$

where $\mathbf{w} \in \mathbb{R}^P$ and $\mathbf{f}_{i,j}$ is a P-dimensional feature vector about user i and item j. Each dimension of $\mathbf{f}_{i,j}$ corresponds to one feature, and each feature value could be binary or numeric. The features could include user characteristics (gender, location, etc.) and item characteristics(price, category, etc.). The features could also include interactions between users and items. For example, a binary value indicating whether user i is familiar with products in the same category/brand of item j, or the frequency of user i visiting the web pages mentioning product j. The sigmoid function is used to restrict the value of $\alpha_{i,j}$ between 0 and 1.

According to the formula (3), the recommendation algorithm decides how much to adopt the social networks' opinions based on the characteristics of users and items. The rating $r_{i,j}$ can be estimated as follows:

$$\hat{r}_{i,j} = sigmoid(\mathbf{w}^T \mathbf{f}_{i,j})\mathbf{u}_i^T \mathbf{v}_j + (1 - sigmoid(\mathbf{w}^T \mathbf{f}_{i,j})) \sum_{k \in \tau(i)} s_{i,k} \mathbf{u}_k^T \mathbf{v}_j \qquad (4)$$

We further assume $\mathbf{w}$ follows a Gaussian distribution $\mathcal{N}(0, \sigma_w^2 \mathbf{I})$. Thus the maximum likelihood estimation of the parameters can be learned by minimizing the following loss function (the negative log likelihood of the observation):

$$loss_{R,S,U,V,W} = \sum_{r_{i,j} \in R} \frac{1}{2}(\hat{r}_{i,j} - r_{i,j})^2 + \sum_{i=1}^{M} \frac{\lambda_u}{2} \parallel \mathbf{u}_i \parallel_2^2 + \sum_{j=1}^{N} \frac{\lambda_v}{2} \parallel \mathbf{v}_j \parallel_2^2 + \frac{\lambda_w}{2} \parallel \mathbf{w} \parallel_2^2$$

where $\lambda_u = \frac{\sigma^2}{\sigma_u^2}$, $\lambda_v = \frac{\sigma^2}{\sigma_v^2}$, $\lambda_w = \frac{\sigma^2}{\sigma_w^2}$.

The solution can be found using conjugate gradient algorithm. The gradient of $\mathbf{u}_i$, $\mathbf{v}_j$ and $\mathbf{w}$ can be calculated as below:

$$\frac{\partial loss}{\partial \mathbf{u}_i} = \sum_{r_{i,j} \in R} \alpha_{i,j}(\hat{r}_{i,j} - r_{i,j})\mathbf{v}_j + \sum_{t \in \varphi(i)} \sum_{r_{t,j} \in R} (1 - \alpha_{i,j})(\hat{r}_{t,j} - r_{t,j}) s_{t,i} \mathbf{v}_j + \lambda_u \mathbf{u}_i$$

$$\frac{\partial loss}{\partial \mathbf{v}_j} = \sum_{r_{i,j} \in R} (\hat{r}_{i,j} - r_{i,j})(\alpha_{i,j}\mathbf{u}_i + (1 - \alpha_{i,j}) \sum_{k \in \tau(i)} s_{i,k} \mathbf{u}_k) + \lambda_v \mathbf{v}_j$$

$$\frac{\partial loss}{\partial \mathbf{w}} = \sum_{r_{i,j} \in R} (\hat{r}_{i,j} - r_{i,j})(\mathbf{u}_i^T \mathbf{v}_j - \sum_{k \in \tau(i)} s_{i,k} \mathbf{u}_k^T \mathbf{v}_j)\alpha'_{i,j} \mathbf{f}_{i,j} + \lambda_w \mathbf{w}$$

where $\alpha'_{i,j} = exp(\mathbf{w}^T \mathbf{f}_{i,j})/(1 + exp(\mathbf{w}^T \mathbf{f}_{i,j}))^2$ is the derivative of the sigmoid function. $\varphi(i)$ is the set of all the users who trust user i.

2.4 Adaptive Weights Based on Individual Neighbors

$s_{i,k}$ captures how a particular neighbor k affects the prediction. According to the definition of $s_{i,k}$ and $\tau(i)$, we have the following three approaches:

Social Trust Model. A straightforward way is adopting a commonly used social network definition of recommender systems, which is so-called *social trust network*. In this scenario, $\tau(i)$ is the social network explicitly expressed by user i. For example, in *Epinions.com*, each user can express his/her Web of Trust by marking some other users as "trustable". Then the set $\tau(i)$ contains all the users who are selected by user i. There are several possible reasons that user i add user k into his/her trust list. First, they might know each other in the real life. Second, user i has read the reviews and ratings provided by user k, and found them valuable or consistent with his/her own tastes. In both cases, social trust network has much potential to be utilized for better inference of users' preferences. It is worth mentioning that the trust value is binary in most recommender systems. This means we do not know how much user i trusts each individual in the trust list. If we simply treat all trusted users on user i's list equally, the definition of $s_{i,k}$ is $s_{i,k} = \frac{1}{|\tau(i)|}$ where $|\tau(i)|$ is the number of trusted users by user i in the set $\tau(i)$.

Social Influence Model. The social trust model mentioned above utilizes a user's social network to infer the user's preference. Now we further discuss how we model the social network influencing users' behaviors. Consider a scenario in the real world, where user i knows nothing about the movie "Avatar" initially. He found more and more people around him have watched the movie, are talking about it and rate it highly. Then there is a high probability that user i will be influenced by people around and go to the theater for "Avatar", even if he usually does not watch Action Sci-Fi movies or movies in general.

To model the influence from one's social network, we restrict $s_{i,k}$ as follows: $s_{i,k} = 1$ if user k purchased or rated item j; otherwise $s_{i,k} = 0$. While predicting $r_{i,j}$, the social influence network being considered contains all the users who are trusted by user i and also purchased/rated the target item j.

Neighborhood Model with Implicit Social Network . The above models treat different individual's opinion in the social network equally. However, people adopt others' opinions differently. For close friends that people know well, they trust them highly. In this case, we probably want to use a high value for $s_{i,k}$. For people they are not familiar with, one may cautiously take the advice. In this case, we may want a low value for $s_{i,k}$. Even for the same person, people will trust him/her in varying degrees in different recommendation contexts. Besides, social network information is not always available for a recommender system. Based on above two considerations, we propose to utilize user's neighborhoods, which can be found using standard collaborative filtering algorithms, as implicit social network. In this model, $\tau(i)$ is the top-N nearest neighbors of user i. To calculate the similarity between users, several similarity measures have been proposed before. Without loss of generality, we use cosine similarity in the space of items. We use $\mathbf{U}_i$ and $\mathbf{U}_k$ to indicate the ith and kth row of the rating matrix. Then similarity between user i and user k is defined as $sim_{i,k} = \frac{\mathbf{U}_i \mathbf{U}_k}{\|\mathbf{U}_i\| \cdot \|\mathbf{U}_k\|}$. According to the similarities between users, we select top N nearest neighbors for each user i as the implicit social network $\tau(i)$. $s_{i,k}$ is defined based on the similarity $sim_{i,k}$ with a normalizing factor so that $\sum_{k \in \tau(i)} s_{i,k} = 1$:

$$s_{i,k} = \frac{sim_{i,k}}{\sum_{t \in \tau(i)} sim_{i,t}} \tag{5}$$

3 Experimental Methodology

We collect evaluation data set from *Epinions.com* which is a consumers review website. Users can review items and provide integer ratings from 1 to 5. Epinions also provides the user profiles and item descriptions, such as item category. As a trust-aware system, users can explicitly express the trust statements in Epinions. Each user maintains a "trust" list which includes some trustable users.

Researchers have used Epinions data set for various research on recommender systems, however, none of the existing data sets contain all the information we need. The data set used in this paper is a new collection we collected by crawling *Epinions.com* on Oct 2009. We first crawled the ratings and trust statements of the top reviewers and then move to the users who trust top reviewers or who are trusted by top reviewers. We crawled users' ratings and trust statements following users' social networks. As a result, we collected a data set that contains 56,859 users, 271,365 items, and 1,154,812 ratings. There are totally 603,686 trust statements. Most of the items are assigned into one category by *epinions.com.* 10,994(19.3%) users only rate one item. 26,712 users(47%) rated no more than 5 items. We use two sets of binary features to represent recommendation context. The first is item categories assigned by *Epinions.com.* The second is the group id that characterizes the number of items the user rated. We classify users into 7 groups (1:"1", 2:"2-5", 3:"6-10", 4:"11-20", 5:"21-40", 6:"41-80", 7:">80").

We carry out experiments on two recommendation tasks:

Rating Prediction Given a user i and an item j, the task is to predict the rating of user i on item j. For this task, we randomly select 80% rating data for training, 10% for testing, and 10% for cross validation (hold out data set). The prediction accuracy is measured by Root Mean Square Error(RMSE).

Top-K Recommendation In real life, a user wants the system to suggest a list of top K items that the user has not yet rated/purchased/seen before.

We design the experiments to answer the following questions: 1) How does the setting of the factor $\alpha_{i,j}$ affect the performance? 2) How does the selection of a user's social network $\tau(i)$ affect the performance? 3) Does weighting each neighbor' opinion differently improve the performance?

To answer question 1), we compare two different settings of $\alpha_{i,j}$. One is to define a fixed value for all the $\alpha_{i,j}$ [8]. The other is to assign adaptive weights based on characteristics of users and items (Section 2.3). To differentiate the two settings, we use "*A*" for the approaches with $\alpha_{i,j}$ that is adaptive for different users and items, and "*F*" for the approaches with a fixed value for all $\alpha_{i,k}$. To answer question 2), we compare the three models in Section 2.4 to utilize social network information. The models are denoted by "Trust", "Influence" and "Neighborhood" respectively. To answer question 3), we compare two settings of $s_{i,k}$ when using neighborhood as an implicit social network: using the similarities measure as formula (5) vs. assigning equal weights to all the neighbors.

The algorithms compared in our experiments are summarized as follows:

- *SVD*: Baseline approach as described in Section 2.1.
- *F-Trust*: Social trust network with a fixed α value (Section 2.4).
- *A-Trust*: Social trust network with adaptive α values (Section 2.3).
- *F-Influence*: Social influence network with a fixed α value (Section 2.4).
- *A-Influence*: Social influence network with adaptive α values.
- *F-Neighborhood*: This approach uses neighborhood as implicit social network (Section 2.4) and a fixed α value.

- *A-Neighborhood*: This approach uses neighborhood as implicit social network and adaptive α values.
- *F-Neighborhood-E*: A variation of *F-Neighborhood* that sets $s_{i,k} = 1/|\tau(i)|$.
- *A-Neighborhood-E*: A variation of *A-Neighborhood* that sets $s_{i,k} = 1/|\tau(i)|$.

All the approaches are based on the parameter setting $\lambda_u = \lambda_v = \lambda_w = 0.2$. For the *Neighborhood* based approaches, we use the top 10 nearest neighbors. Based on validation set, we found the fixed α values ($\alpha = 0.3$ for *F-Trust*, $\alpha = 1.0$ for *F-Influence*, and $\alpha = 0.2$ for *F-Neighborhood*.)

4 Experimental Results

4.1 Results on Rating Prediction

Table 1. Performance comparison

Model	Dimensionality		
	D=5	D=10	D=20
SVD	1.0747	1.0683	1.0812
F-Trust	1.0516	1.0434	1.0528
A-Trust	1.0481	1.0387	1.0416
F-Influence	1.0740	1.0673	1.0820
A-Influence	1.0682	1.0618	1.0664
F-Neighborhood	1.0262	1.0212	1.0270
A-Neighborhood	1.0238	1.0142	1.0022

Table 2. Performance on the subsets

Model	Influence Subset		Trust Subset	
	RMSE	α	RMSE	α
SVD	1.0467	-	1.0773	-
F-Trust	1.0009	0.3	1.0382	0.3
A-Trust	1.0002	-	1.0347	-
F-Influence	1.0447	0.9	1.0779	1.0
A-Influence	1.0373	-	1.0747	-
F-Neighborhood	1.0115	0.3	1.0323	0.2
A-Neighborhood	1.0023	-	1.0251	-

Table 1 summarizes the results on the whole test data. We conduct experiments on three latent vector dimensions: 5, 10, and 20. There are several things worth mentioning. First, it shows social network information is valuable. Social network based approaches outperformed baseline SVD. Second, it shows *A-Neighborhood* performs better than other methods. The improvement of using neighbors over SVD is not surprising. Because factorization captures global structure of the rating matrix, while neighborhood captures local regularization of the data space. Combining these complementary information has the same effect as the Netflix competition winner's solution, which combines nearest neighbors with factorization models [7]. However, it is interesting to see that neighborhood models perform better than social influence and social trust models, since neighborhood models do not use any user identified social network info. Third, it shows the performance of every approach improves when we vary α based on recommendation context (users and items). The improvements are different when using different social network information. One possible reason is the sparsity of social networks. In Epinions data set, almost every user has neighborhood, while only 59.5% of ratings in the test data have social trust information and only 18.2% have social influence information. Therefore, the overall performance may be dominated by the rating pairs without explicit social information. That also answers why *F-Influence* performs best with $\alpha = 1.0$.

Performance of Different Social Networks. To focus on the effect of different social networks, we created two test data subsets. One subset (*Trust Subset*) consists of ratings with social trust info. Both *Trust* and *Neighborhood* based approaches can be used to predict all the test cases, while *Influence* based approaches can not be used on part of this subset. Thus the second subset (*Influence Subset*) is much smaller and consists of the ratings with all three kinds of social info available. This data set contains 6618 users and 21,599 ratings. Table 2 shows the results on the two smaller test data sets.[2] We observe that *Trust* based approaches are comparable with *Neighborhood* based approaches, although *Neighborhood* based approaches are clearly better than the others on the whole test data (Table 1). On the influence subset, trust based approaches outperforms *Neighborhood* based approaches. The results suggest that a recommender system may want to use a hybrid neighborhood-trust network model.

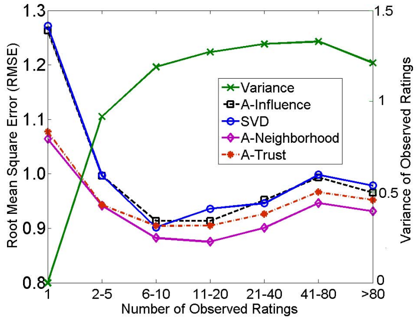


Fig. 1. Comparison on different number of observed ratings based on the whole test data

Table 3. Performance on two settings of $s_{i,k}$ when using neighborhood as implicit social network

identical $s_{i,k}$	
Model	RMSE
F-Neighborhood-E	1.0257
A-Neighborhood-E	1.0200

weighted $s_{i,k}$	
Model	RMSE
F-Neighborhood	1.0212
A-Neighborhood	1.0163

Performance on Different Users. We analyze how the size of training data per user affects the performance of different algorithms. We group all the users into 7 classes based on the number of observed ratings in the training data. Figure 1 shows the macro RMSE on different user groups. The horizontal axis describes how many training ratings are available for a user in that class. It shows that *A-Trust* and *A-Neighborhood* almost consistently outperform *SVD* and *A-Influence*, especially when users have less than 6 ratings. It's surprising that RMSE increases when the number of observed ratings is more than 10. To understand this, we look at the rating variance for each user group. We find that variance and RMSE have the similar trend, both of them tend to increase after observing more than 10 ratings (Figure 1). When user has fewer ratings, those ratings usually are about one or two aspects and thus the variance is small; when the user provide more ratings, those ratings consist of user's multiple interests. When we use all ratings to predict a rating in a specific aspect, products that are irrelevant to the target item may hurt the performance. Therefore, the initial decrease of RMSE is because the increase of observed ratings makes the model know more about users while the influence of rating variance confuses the model

[2] In the rest of this paper, all the experimental results are using 10-dimensional latent vector setting, where 10 is found by the validation data set.

and hurts the performance. It suggests that user ratings on one category may hurt the prediction of user ratings in another category.

Impact of Parameter $s_{i,k}$. In our approaches, the parameter $s_{i,k}$ indicates how much user i would trust user k. Table 3 shows the results of two settings of $s_{i,k}$ when using neighborhood as an implicit social network. It is clear that weighting others' opinions based on the similarity (*-Neighborhood) can achieve better performance than treating all opinions equally (*-Neighborhood-E).

4.2 Further Analysis about Social Influence

We did some further analysis by looking at the weights learned (the component values of $\mathbf{w}$)[3] for different contexts for social recommendation model, and the goal is to answer the following questions: 1) How does the number of observed ratings affect the weight of social network? 2) How does the size of a user's trust list affect the weight of social network? 3) Is a user more likely to be influenced when his uncertainty about the product is high? (We assume a user may be more uncertain about the product quality if the product quality tends to be subjective, such as for books/movies, instead of objective, such as for PC/memory.)

Figure 2(a) shows that, as the number of training ratings increases, the weights learned by *A-Neighborhood* become smaller, while the weights learned by *A-Trust* increase. The weight is a tradeoff between uncertainty about neighbors' ratings vs. uncertainty about the user's own ratings. In *A-Neighborhood*, the neighbors found are unreliable when the user has fewer ratings, therefore, *A-Neighborhood* does not weight neighbors' opinions high in these cases [1]. In *A-Trust*, the user's own prediction is more reliable when the number of ratings is high, thus *A-Trust* does not weight neighbors' opinions high.

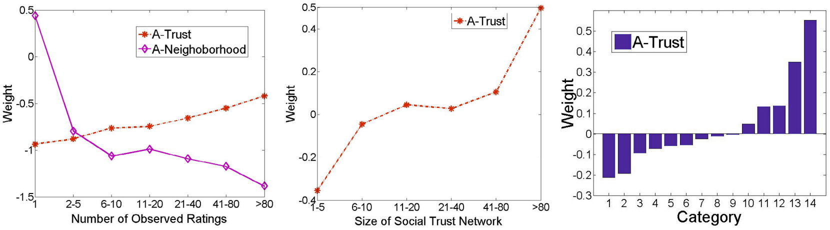
To answer question 2), we introduce a new feature, the size of social trust network, for *A-Trust*. Figure 2(b) shows the weights learned by *A-Trust* increases with the size of social trust network. That means the model considers larger social trust networks less reliable than smaller ones. One possible reason is that, a large social network is more likely be selected arbitrarily by a user, while a small social network tends to be selected more seriously and hence more reliable.

Figure 2(c) shows the learned weights for different categories. It seems that categories more related to personal experiences tend to have higher weights. Instead, the categories whose ratings are more subjective tend to have lower weights, probably because a user is more uncertain about these products and is likely to be influenced by people they trust.

4.3 Results on Top-K Recommendations

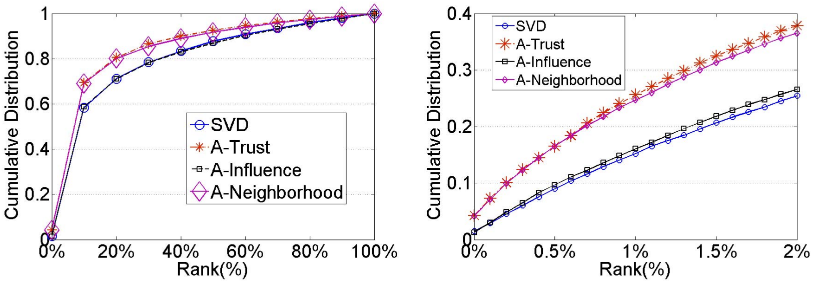
A more realistic task for a recommender system is to recommend K items that users may like. In this section, we simulate the real scenario and investigate the effect of our approaches on the task of top-K recommendations.

[3] According to formula (4), a larger weight value means that less emphasis is placed on social network information.



(a) (b) (c)

Fig. 2. Learned weights for different features. In Figure(c), item categories form 1 to 14 are: Books, Music, Kids & Family, Hotel & Travel, Software, Sports & Outdoors, Pets, Electronics, Games, Wellness & Beauty, Movies, Education, Online Stores & Services, Personal Finance.



(a) (b)

Fig. 3. Performance on Top-K Recommendation Task. The right plot concentrates on the top 2% ranking region.

Previous works on this task tend to adopt classic IR measures such as P@N and Recall [4][2]. However, without complete relevance judgements for each individual user, standard IR evaluation is almost infeasible. Thus we use a variation of the evaluation method in [7]. We randomly sample 10% from the rating data set $\langle i, j, r_{i,j} \rangle$. Then for each user in the sampled data set, we randomly choose one user-item pair with a 5-star rating. This gives 15,025 user-item testing pairs. To simulate the scenario that we only want to recommend the 5-star items to users, we treat 5-star pairs as relevant. The Epinions data is an open-domain data set with multiple categories. Intuitively, a book and a song are hard to compare. We assume that a user wants to purchase one specific kind of item, such as a book, and the system needs to rank items in this category. Therefore, for each testing pair $\langle i, j \rangle$, we randomly sample 1000 additional items which user i has not rated from the same category as item j. For example, if user i purchased a book, we randomly select 1000 additional books as candidates to be ranked.

Figure 3 compares four methods: *SVD*, *A-Trust*, *A-Influence*, *A-Neighborhood*. In real systems, only top K items might be recommended. Therefore, we focus on the top 2% ranking area (top 20 ranked items out of 1000) (Figure 3(b)). First, it shows all the social network based approaches outperform *SVD*. That means we can benefit from utilizing social network information in top-K recommendation task. Second, it shows *A-Influence* is not as good as *A-Trust* and *A-Neighborhood* due to the sparsity of social network.

5 Conclusions and Future Work

We investigated three ways to combine social network and matrix factorization for recommender systems. All three methods work better than the baseline method. This means social network is useful for recommendation. The three methods have different properties. When social trust information is applicable, Social Trust Model always works better than SVD, especially when a user has few ratings. Social Influence Model is not always applicable. When it is applicable, making recommendations using the influence from people a user trust can also improve the performance for two tasks. When the social network information is not available, we can find implicit N Nearest Neighbors and use the Neighborhood model to combine neighbors' predictions with the SVD prediction. This is a first step to adaptively weight the info from neighbors. Future work includes adapting the influence for other social network based recommendation methods.

Acknowledgement. This work is supported by the National High-Tech Research and Development Plan of China (863) under Grant 2006AA01Z177, the National Natural Science Foundation of China (NSFC) under Grant No. 60873027, 61021062, and the National 973 Program of China under Grant No. 2009CB320705.

References

1. Berkovsky, S., Kuflik, T., Ricci, F.: Distributed collaborative filtering with domain specialization. In: RecSys, pp. 33–40 (2007)
2. Gunawardana, A., Meek, C.: A unified approach to building hybrid recommender systems. In: RecSys, pp. 117–124 (2009)
3. Jamali, M., Ester, M.: A transitivity aware matrix factorization model for recommendation in social networks. In: IJCAI, pp. 2644–2649 (2011)
4. Karypis, G.: Evaluation of item-based top-n recommendation algorithms. In: CIKM, pp. 247–254 (2001)
5. Kautz, H.A., Selman, B., Shah, M.A.: Referral web: Combining social networks and collaborative filtering. Commun. ACM 40(3), 63–65 (1997)
6. Konstas, I., Stathopoulos, V., Jose, J.M.: On social networks and collaborative recommendation. In: SIGIR, pp. 195–202 (2009)
7. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: KDD, pp. 426–434 (2008)
8. Ma, H., King, I., Lyu, M.R.: Learning to recommend with social trust ensemble. In: SIGIR, pp. 203–210 (2009)
9. Ma, H., Zhou, D., Liu, C., Lyu, M.R., King, I.: Recommender systems with social regularization. In: WSDM, pp. 287–296 (2011)
10. Massa, P., Avesani, P.: Trust-aware recommender systems. In: RecSys, pp. 17–24 (2007)
11. McDonald, D.W.: Recommending collaboration with social networks: a comparative evaluation. In: CHI, pp. 593–600 (2003)
12. Salakhutdinov, R., Mnih, A.: Probabilistic matrix factorization. In: NIPS (2007)

Learning Representative Nodes in Social Networks

Ke Sun[1], Donn Morrison[2], Eric Bruno[3], and Stéphane Marchand-Maillet[1]

[1] Viper Group, Computer Vision & Multimedia Laboratory, University of Geneva, Switzerland
[2] Unit for Information Mining & Retrieval, Digital Enterprise Research Institute, NUIG, Galway, Ireland
[3] Knowledge Discovery & Data Mining, Firmenich S.A., Switzerland

Abstract. We study the problem of identifying representative users in social networks from an information spreading perspective. While traditional network measures such as node degree and PageRank have been shown to work well for selecting seed users, the resulting nodes often have high neighbour overlap and thus are not optimal in terms of maximising spreading coverage. In this paper we extend a recently proposed statistical learning approach called *skeleton learning* (SKE) to graph datasets. The idea is to associate each node with a random *representative* node through Bayesian inference. By doing so, a prior distribution defined over the graph nodes emerges where representatives with high probabilities lie in key positions and are mutually exclusive, reducing neighbour overlap. Evaluation with information diffusion experiments on real scientific collaboration networks shows that seeds selected using SKE are more effective spreaders compared with those selected with traditional ranking algorithms and a state-of-the-art degree discount heuristic.

1 Introduction

In a social network, a small subset of *representative* nodes can help establish a hierarchical messaging scheme: the correspondence with each individual node is through a nearby *representative*. Despite that the word "representative" can be interpreted in different ways in social analysis, here, the purpose of such a hierarchy is to broadcast information efficiently with constrained resources. Locally, these representatives should lie in hub positions so as to minimize the routing cost to their nearby nodes. Globally, there should be as few representatives governing different regions so as to save resources.

From a machine learning perspective, a closely related problem is *spectral clustering* [1–3], where the network is partitioned into a fixed number of densely-connected sub-networks with sparser connections between them. This technique has been applied to social networks, e.g., for community detection [4, 5] and spam nodes identification [6]. It is powerful in depicting complex clusters with simple implementations. It is computationally expensive for large datasets, especially when a proper number of clusters has to be searched over [4].

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 25–36, 2013.

In the data mining community, the graph-based ranking algorithms [7–9] have a profound impact on the present World Wide Web and citation analysis systems. They rank graph nodes based on the general idea that the value of one node is positively related to the value of its neighbours. These approaches are further investigated by machine learning researchers using spectral graph theory [10, 11] and random walks [12]. In our task of selecting representatives, the highly ranked nodes by these algorithms usually have high neighbour overlap because of the mutual reinforcement between connected high degree nodes.

Motivated by seeking effective marketing strategies, efforts have been made to select a set of influential individuals [13–15] and to maximize their influence through information diffusion [16, 17]. Although the optimization problem is generally NP hard [13], reasonable assumptions lead to polynomial-time solvable models [14] and efficient implementations with approximation bounds [15]. Targeting at similar objectives, methods from different perspectives are developed with improved speed and performance [18, 19].

This work provides a novel approach to measure the representativeness of graph nodes based on *skeleton learning* (SKE) [20]. It assigns each node a probability of being a representative and minimizes the communication cost from a random node to its corresponding representative. This method is different from other approaches in two aspects. First, the learned distribution has low entropy with the representative nodes having large probability and the non-representative nodes having probability close to zero. Second, the representative nodes are *mutually exclusive*: if a node already has a nearby representative, it will penalize the representativeness of other nearby candidates. Such exclusiveness is not implemented as heuristics in a greedy manner [18], but fits in a minimizing message length framework and allows global coordination in arranging the representatives.

The rest of this paper is outlined as follows. Section 2 introduces the skeleton learning. Section 3 presents the recent development of this approach on social network analysis. Section 4 and Section 5 show the experimental results on toy datasets and real social networks, respectively. Finally, Section 6 concludes.

2 Skeleton Learning

This section briefly reviews the recently proposed skeleton learning [20]. Given a set of samples $\mathcal{X} = \{\boldsymbol{x}_i\}_{i=1}^n \subset \Re^D$, this unsupervised method learns a probability α_i for each $\boldsymbol{x}_i$ ($\sum_{i=1}^n \alpha_i = 1$), so that the probability mass highlights the samples on the "skeleton" of the structures and diminishes on outliers.

The input samples are first encoded into a probability matrix $P_{n\times n} = (p_{i|j})$ as in Stochastic Neighbour Embedding [21], so that

$$p_{i|j} = \frac{\exp(-h_j||\boldsymbol{x}_i - \boldsymbol{x}_j||^2)}{\sum_{i:i\neq j}\exp(-h_j||\boldsymbol{x}_i - \boldsymbol{x}_j||^2)} \tag{1}$$

denotes the probability of node i receiving a message originated from node j with respect to space adjacency. In Eq.(1), $||\cdot||$ is 2-norm, and $h_j > 0$ is a kernel width parameter, which can be fixed so that the entropy of $p_{\cdot|j}$ equals to a prespecified constant [21]. The latent distribution $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)$ corresponds to a discrete random variable, or the index $j \in \{1, 2, \ldots, n\}$ of a random *skeleton point* $\boldsymbol{x}_j \in \mathcal{X}$. By assumption, this random point sends out a message with respect to $P_{n\times n}$. Any $\boldsymbol{x}_i \in \mathcal{X}$, upon receiving such a message, can infer the location of the skeleton point using Bayes' rule as

$$q_{j|i} = \frac{\alpha_j \cdot p_{i|j}}{\sum_{j:j\neq i} \alpha_j \cdot p_{i|j}}. \tag{2}$$

The objective is to optimally route from a random location in $\mathcal{X}$ to its skeleton point, which is implemented by minimizing

$$E(\boldsymbol{\alpha}) = -\sum_{i=1}^{n} \sum_{j:j\neq i} q_{j|i} \log p_{j|i} \tag{3}$$

with respect to $\boldsymbol{\alpha}$. Through such minimization, a compact set of skeleton positions with large α_i can be learned. As compared to clustering methods, the skeleton model is a prior distribution defined on the observations, and the effective number of skeleton points shrinks continuously during learning. Therefore no model selection is necessary to determine an appropriate number of clusters, and the learning process can be terminated at anytime to produce reasonable results. However, the effect of the kernel width parameter h_j must be carefully investigated depending on application. In image denoising [20], it shows better performance in preserving the manifold structure as compared to a state-of-the-art denoising approach [22]. The gradient-based algorithm has a complexity of $O(n^2)$ at each step, which limits its scalability.

3 Skeleton Learning on Graphs

The skeleton learning method introduced in Section 2 is performed on a set of coordinates for denoising and outlier detection. This section extends the idea to graph datasets and discusses related problems. Assume the input data is a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, 2, \ldots, n\}$ is the set of vertices and $\mathcal{E} = \{(i, j)\}$ is the set of edges. Throughout this paper, an undirected graph is treated as its directed version by replacing each edge $i \leftrightarrow j$ with two opposite arcs (i, j) and (j, i). We aim to discover a random *representative* characterized by a discrete distribution $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)$ defined on $\mathcal{V}$, so that any random node in $\mathcal{V}$ can communicate with it in the most efficient and economical way.

We first construct a channel between two random nodes so that the communication cost can be measured. The input graph $\mathcal{G}$ can be equivalently represented by its *normalized adjacency matrix* $A = (a_{ij})$ with

$$a_{ij} = \delta_{ij}/d_i, \quad \delta_{ij} = \begin{cases} 1 & \text{if } (i, j) \in \mathcal{E}; \\ 0 & \text{otherwise,} \end{cases} \tag{4}$$

where d_i is the outdegree of the node i. If a node i has at least one outgoing link, the i'th row of A defines a discrete distribution representing how likely i influences the other nodes according to the graph structure. To deal with nodes with no outgoing links or incoming links, we allow each node i to teleport to another random node with a small probability ν. In social networks, such teleportation models i's influence through external ways not restricted by the network [7, 23]. Consider i sending a message to one unique receiver other than i at time 0. The probability that node j receives this message in one time step can be defined as $p_{j|i} = (1-\nu)a_{ij} + \nu/(n-1)$ $(j \neq i)$. In matrix form it is equivalently

$$P = (1-\nu)A + \frac{\nu}{n-1}(\boldsymbol{e}\boldsymbol{e}^T - I), \tag{5}$$

where $\boldsymbol{e} = (1,\dots,1)^T$ and I is the identity matrix. If we allow this message to pass around in $\mathcal{G}$ for τ times $(\tau = 1, 2, \dots)$ after time 0, the probability for each node j holding the message is given by the i'th row of the matrix P^τ. It represents i's indirect influence over $\mathcal{G}$ through information spreading. In the extreme case when $\tau \to \infty$, all rows of P^τ will tend to be the same, or the equilibrium distribution corresponding to the PageRank (PR) measure [7]. To distinguish the "outgoing ability" of different nodes, τ should be a small value (e.g., 1 or 2) so that each node can only reach a local region around itself. Without loss of generality, we focus on the case $\tau = 1$ unless otherwise specified.

Assume a latent prior distribution $\boldsymbol{\alpha}$ of each node being the information source. The sender i, upon any node j receiving a message, can be identified with Bayesian inference in Eq.(2) (with i and j interchanged). The total communication cost for every node $j \in V$ to reply to its information source is given by Eq.(3). By minimizing such a cost, $\boldsymbol{\alpha}$ can be learned so that this communication loop is established in the optimal way.

More intuitively, consider without loss of generality the graph $\mathcal{G}$ as a social network of n persons. A directed link $(i,j) \in \mathcal{E}$ means that i could easily influence j because of personal relationship, etc. One real-life example could be j "follows" i on some microblogging website. In this context, the meaning of being a representative can be understood from Eq.(3). To minimize $E(\boldsymbol{\alpha})$, on average $-\log p_{j|i}$ should be small, which means the representative j can perceive news from its surrounding nodes easily. As another condition, $q_{\cdot|i}$ should have low entropy, which means each person i selects the candidate which influences i the most, and *deselects* other nearby candidates being its representative.

Implementation

The skeleton learning is implemented by gradient descent to minimize $E(\boldsymbol{\alpha})$. The gradient of $E(\boldsymbol{\alpha})$ has the form [20]

$$\frac{\partial E}{\partial \alpha_j} = -\frac{1}{\alpha_j} \sum_{i:i\neq j} q_{j|i,\alpha} \left(\log p_{j|i} - \sum_{l:l\neq i} q_{l|i,\alpha} \log p_{l|i} \right). \tag{6}$$

Algorithm 1. Skeleton Learning on Graph Datasets

```
Input: A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $n$ nodes; a small teleport probability $\nu$
Output: A discrete distribution $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)$ to measure the
        representativity of each node
begin
    $\boldsymbol{\alpha} \leftarrow (1/n, \ldots, 1/n)$; $\gamma \leftarrow \gamma_0$; $p_0 = \nu/(n-1)$        // $\gamma$: learning rate
    repeat
        $\bigtriangledown = (\bigtriangledown_1, \ldots, \bigtriangledown_n) \leftarrow \mathbf{0}$; $\bigtriangledown_c \leftarrow 0$
        foreach node $i$ in $\mathcal{S}$ (see comments in the end) do
            $p_i \leftarrow (1-\nu)\sum_{j:j\to i} \alpha_j/d_j + \nu(1-\alpha_i)/(n-1)$
                                                   // $d_j$ is the out-degree of node $j$
            $E_i \leftarrow -(1-\nu)\sum_{j:j\to i} \alpha_j \log p_{j|i}/d_j - \nu \sum_{j:j\neq i} \alpha_j \log p_{j|i}/(n-1)$
            $E_i \leftarrow E_i/p_i$
            foreach $j$ in Pre($i$) do     // Pre($i$) is the set of predecessors
                $\bigtriangledown_j \leftarrow \bigtriangledown_j - (1-\nu)(\log p_{j|i} + E_i)/(d_j p_i)$
            end
            foreach $j$ in Suc($i$) do      // Suc($i$) is the set of successors
                $\bigtriangledown_j \leftarrow \bigtriangledown_j - \nu(\log p_{j|i} - \log p_0)/\left((n-1)p_i\right)$
            end
            $\bigtriangledown_i \leftarrow \bigtriangledown_i + \nu(\log p_0 + E_i)/\left((n-1)p_i\right)$
            $\bigtriangledown_c \leftarrow \bigtriangledown_c - \nu(\log p_0 + E_i)/\left((n-1)p_i\right)$
        end
        $\bigtriangledown \leftarrow \bigtriangledown + \bigtriangledown_c$
        $\boldsymbol{\alpha} \leftarrow \boldsymbol{\alpha} \circ \exp(-\gamma \bigtriangledown \circ \boldsymbol{\alpha})$        // "$\circ$" is the element-wise product
        normalize $\boldsymbol{\alpha}$ so that $\sum_i \alpha_i = 1$
    until convergence or the number of iterations reaches a pre-specified value
end
/* If $\mathcal{S} = \mathcal{V}$, $\boldsymbol{\alpha}$ is updated on every full scan of the whole dataset;
   if $\mathcal{S}$ is a small random subset of $\mathcal{V}$, $\boldsymbol{\alpha}$ is updated with stochastic
   gradient descent (more efficient and scalable) */
```

Along $-\partial E/\partial\alpha_j$, the candidate weight α_j is adjusted at each step. Intuitively Eq.(6) says, for each node i within j's reachable range, j serves as a potential information source of i (the value $q_{j|i,\alpha}$ is significant enough based on Eq.(2)), and such i provides feedback to α_j based on how efficiently it can reach back to j. If the length $-\log p_{j|i}$ is shorter than the average length $-\sum_{l:l\neq i} q_{l|i,\alpha} \log p_{l|i}$, then $\partial E/\partial\alpha_j < 0$ and α_j increases, which means that i "votes" for j to become its representative. On the other hand, if the route $i \to j$ is too costly, i casts a negative vote for j. This type of gradient was discussed in a statistical machine learning framework [24] and further explored here in a non-parametric setting.

In general, each gradient descent step requires $O(|\mathcal{V}|^2)$ computation [20] because P is dense. However, the fact that most entries of the transition matrix P equal $\nu/(n-1)$ can lead to more efficient implementations. On graph datasets, the gradient in Eq.(6) is further written as

$$
\begin{aligned}
\frac{\partial E}{\partial \alpha_j} &= -\frac{1-\nu}{d_j}\sum_{i:j\to i}\Delta_{ij} - \frac{\nu}{n-1}\sum_{i:i\to j}(\Delta_{ij}-\Delta_{i0}) - \frac{\nu}{n-1}\sum_{i:i\neq j}\Delta_{i0},\\
\Delta_{ij} &= \frac{1}{p_i}\Big(\log p_{j|i} + E_i\Big), \quad \Delta_{i0} = \frac{1}{p_i}\left(\log\frac{\nu}{n-1} + E_i\right),\\
p_i &= (1-\nu)\sum_{j:j\to i}\frac{\alpha_j}{d_j} + \nu\frac{1-\alpha_i}{n-1},\\
E_i &= -\sum_{j:j\neq i} q_{j|i}\log p_{j|i} = -\frac{1-\nu}{p_i}\sum_{j:j\to i}\frac{\alpha_j}{d_j}\log p_{j|i}\\
&\quad - \frac{\nu}{p_i(n-1)}\sum_{j:i\to j}\alpha_j \log p_{j|i} - \frac{\nu}{p_i(n-1)}\log\frac{\nu}{n-1}\left(1-\alpha_i-\sum_{j:i\to j}\alpha_j\right).
\end{aligned}
$$

In Algorithm 1, the simple gradient descent has a computational complexity of $O(|\mathcal{E}|)$ in each iteration. The stochastic gradient descent (SGD) [25] version reduces this computation time to $O(\max(d_i)\cdot|\mathcal{S}|)$. Besides the learning rate, the algorithm has only one parameter ν. By default we set $\nu = 0.2$ in the following experiments. On real large social networks, the node degrees follow an exponential distribution, which may lead to trivial solutions if ν is too large. For example, one node with significant number of links could become the sole representative over the whole network and communicate with the unconnected nodes through teleport. In this case we have to lower the value of ν to penalize the teleport communication and to discover more representatives.

4 Toy Problems

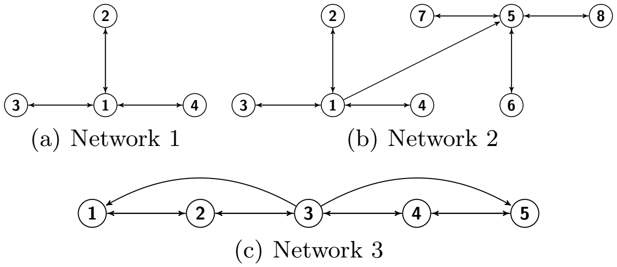
Figure 1 presents several toy social networks. Table 1 shows the α_i value and the PageRank value of each node. In Figure 1(a), only one person (node 1) is acquainted to all the others. SKE has successfully identified it as the sole representative. Figure 1(b) shows two groups of people, each with a central hub (node 1 and node 5), and a link from node 1 to node 5. The SKE values are very concentrated on these two centers, with node 5 having slightly larger weight due to the fact that no edge exists from node 5 to node 1. This type of penalty becomes clearer in the network shown in Figure 1(c). In this example, each node has exactly the same indegree. There is one node 3 which links to all the other nodes, while half of the linked nodes do not respond. Its α_i is close to zero, meaning that it has been identified as a spam node. In general, the PageRank values are less concentrated and do not reveal such information.

The proposed method is further tested on two "Primary School Cumulative Networks" [26] [1], where the nodes represent students or teachers and the edges represent their face-to-face interactions. We only consider *strong interactions*, which are defined as all such edges (A, B) if A and B has interacted for at least

[1] http://www.sociopatterns.org/datasets/primary-school-cumulative-networks

Table 1. SKE and PageRank results on the toy networks in Figure 1

	Node	1	2	3	4	5	6	7	8
Network 1	PageRank	0.46	0.18	0.18	0.18				
	SKE	**0.99**	0.00	0.00	0.00				
Network 2	PageRank	0.17	0.06	0.06	0.06	0.32	0.11	0.11	0.11
	SKE	**0.41**	0.01	0.01	0.01	**0.44**	0.04	0.04	0.04
Network 3	PageRank	0.17	0.22	0.22	0.22	0.17			
	SKE	0.20	**0.30**	0.00	**0.30**	0.20			



(a) Network 1 (b) Network 2

(c) Network 3

Fig. 1. Toy social networks

2 minutes and on at least 2 occasions. As a result, there are 236 nodes and 1954 edges in network 1, and there are 238 nodes and 2176 edges in network 2.

Figure 2 shows the visualization of the network on day 1, where the SKE value α_i is intuitively presented by the size of the corresponding circle and the node degree is presented by the color density. We see that the representatives (large circles) do not necessarily have high degrees (dense color), and vice versa. This is further confirmed by looking at the accurate measurements given in Table 2. Among the top ranked nodes, some have small degrees, such as node "1843" in day 1, node "1521" and "1880" in day 2. We further look at the average degree of their neighbours (D_n). All these three nodes have a relative small value of D_n, which means some of their neighbours are poorly-connected. The connections with these non-so-popular nodes are highly valued in the SKE measurement. On the other hand, among the bottom ranked nodes, some have large degrees, such as node "1628" and node "1428" in day 1, node "1766" and "1778" in day 2. Generally they have a relative large D_n value. Although they are well-connected, their relationships are mostly established with popular nodes, and thus have little value. We also see that the SKE measure has a "sharper" distribution as compared to PageRank, where the tail nodes have very small values. The effective number of skeleton points (given by $\exp\{-\sum_i \alpha_i \log \alpha_i\}$, or the number of uniformly distributed points with the same entropy as $\boldsymbol{\alpha}$) is 95.8 and 91.3 in network 1 and 2, respectively.

5 Information Diffusion on Collaboration Networks

We test the proposed approach as a seeding method for information diffusion in social networks [15], so that a small subset of *seeds* (corresponding to the representatives as discussed above) could influence as many nodes as possible.

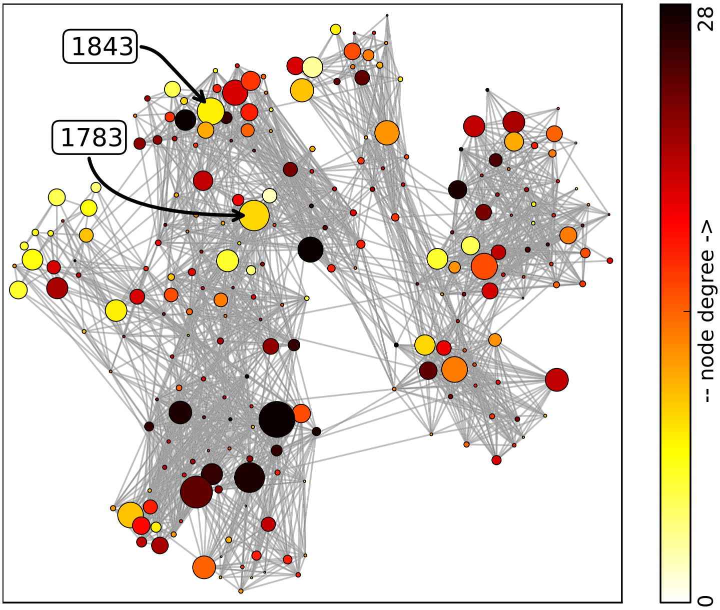


Fig. 2. SKE results on Primary School Cumulative Networks (day 1). The node size represents the SKE value. The node color represents its degree (the darker the higher).

Table 2. Different measures of nodes in the cumulative network dataset. For each day, the columns show (1) node ID (2) class ID if the node is a child, or "Teachers" (3) degree D (4) average degree D_n of its neighbours (5) PageRank (PR) in percentage (6) α_i (SKE) in percentage. The nodes are ordered by α_i.

Cumulative Network Day 1						Cumulative Network Day 2					
Node	Class	D	D_n	PR	SKE	Node	Class	D	D_n	PR	SKE
1890	2B	35	19.3	0.80	4.3	1745	Teachers	28	21.5	0.57	4.4
1650	Teachers	25	16.0	0.65	3.4	**1521**	Teachers	8	6.9	0.47	3.3
1783	1A	19	18.6	0.46	3.1	1668	Teachers	21	19.2	0.47	3.0
1743	2B	28	18.9	0.67	3.0	1443	5B	23	17.1	0.58	2.9
1843	3A	15	16.1	0.41	2.4	**1880**	4B	8	8.3	0.43	2.7
⋮						⋮					
1628	2A	23	21.3	0.57	0.0	**1766**	1A	11	18.7	0.31	0.0
1649	2A	7	16.3	0.26	0.0	1799	1A	9	19.6	0.26	0.0
1483	5A	4	16.0	0.18	0.0	1772	1A	8	15.9	0.26	0.0
1858	2B	7	21.0	0.23	0.0	1519	4A	2	21.0	0.13	0.0
1511	5A	9	20.3	0.27	0.0	1819	4B	5	12.6	0.27	0.0
1428	5B	12	21.1	0.33	0.0	**1778**	1A	11	22.9	0.29	0.0
1803	4B	6	18.2	0.22	0.0	1753	Teachers	3	23.0	0.14	0.0
1710	2B	5	24.0	0.19	0.0	1710	2B	8	29.6	0.21	0.0
1854	2B	5	23.4	0.18	0.0	1807	4B	2	11.5	0.16	0.0
1898	2B	5	23.8	0.19	0.0	1760	1A	3	23.7	0.14	0.0

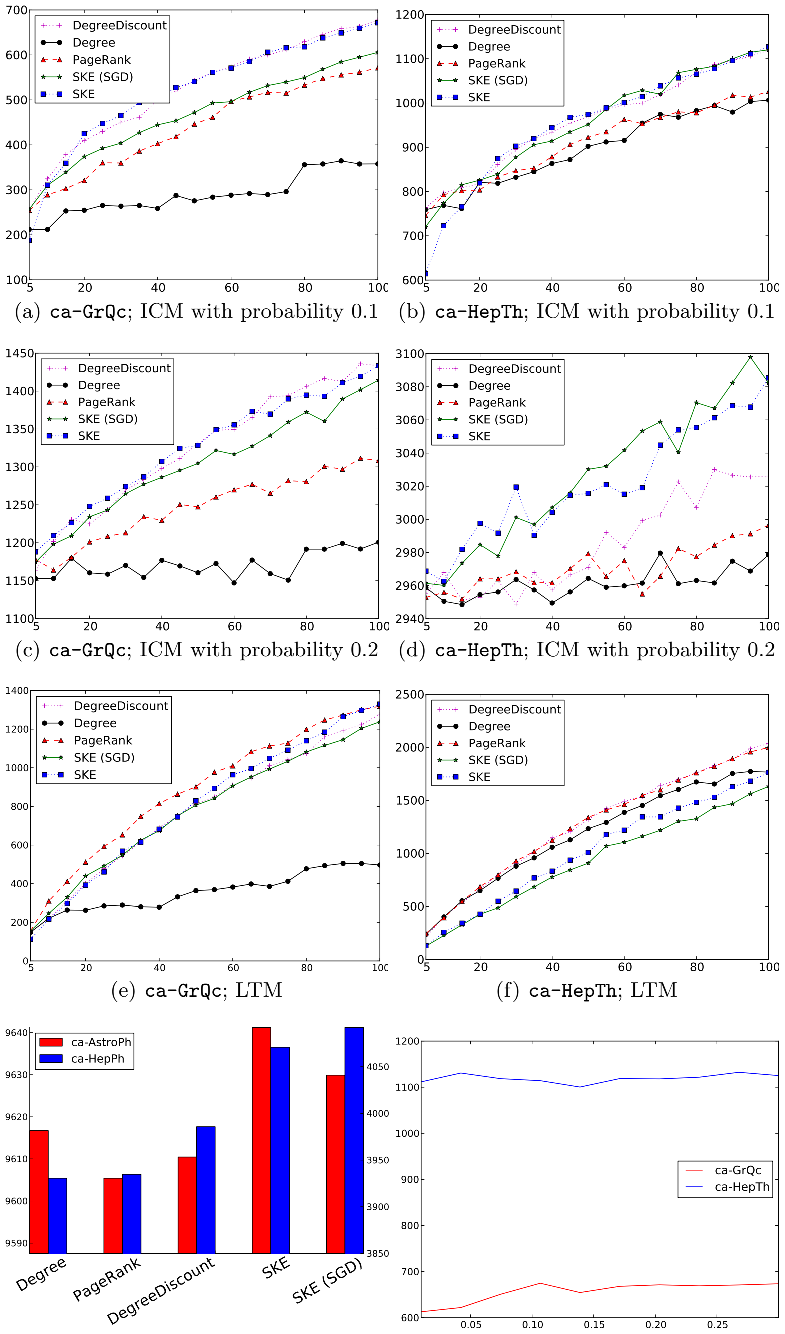
We use the collaboration networks in the Stanford Large Network Dataset Collection[2] [27]. `ca-GrQc` is a co-authorship network of physics publications, compiled from the General Relativity section of Arxiv. It has `5,242` nodes representing authors and `14,496` edges representing co-authorships. Similarly, `ca-HepTh`, `ca-HepPh` and `ca-AstroPh` are collaboration networks of different domains on Arxiv. Their sizes denoted by `#nodes/#edges` are `9,877/25,998`, `12,008/118,521`, `18,772/198,110`, respectively. All datasets are undirected and unweighted, which means the accurate number of times that two authors have collaborated is discarded. For each dataset, five different seeding methods are applied, which select seeds based on descending degree, descending PageRank value, descending SKE value (α_i) computed by simple gradient descent and stochastic gradient descent, and the degree discount heuristic [18], respectively. The damping factor in PageRank is set to be 0.85 by convention. The teleport probability ν in SKE is set to be 0.2.

To evaluate the seeding quality, we apply two different diffusion models, namely Independent Cascade Model (ICM) [17] and Linear Threshold Model (LTM) [16], once the selected seeds are marked as being *activated*. Both of these models expand the activated set of nodes in discrete steps with respect to the network structure. In ICM, an activated node v has a single chance to activate a neighbour w with the probability $p_{v,w}$. By discarding the number of times that two authors have cooperated, diffusion with ICM becomes harder because v has one link instead of several parallel links connecting with j. In LTM, a node v will be activated once the proportion of activated neighbours reaches a node-specific threshold θ_v. At convergence, the size of the activated set quantifies the influence of the initial seeds. To average the effect of random factors, the experiment for each seeding set is repeated for 100 times.

Figure 3(a-d) shows the size of the activated set varying with the number of seeds on `ca-GrQc` and `ca-HepTh` using ICM. Figure 3(g) condenses the results on `ca-AstroPh` and `ca-HepPh` using 100 seeds with ICM. Generally SKE and DegreeDiscount outperform PageRank, which in turn outperforms Degree. Basically SKE tries to place the minimum number of seeds while guaranteeing the influence coverage over different regions. The good performance of DegreeDiscount is due to a similar mechanism to penalize clustered seeds [18]. While the two approaches are comparable on `ca-GrQc`, seeding with SKE is obviously better on `ca-HepTh` (ICM probability=0.2), `ca-AstroPh` and `ca-HepPh` (ICM probability=0.1). Moreover, SKE can adapt to network changes through online learning, which is not straightforward for DegreeDiscount. Note, the SGD version of SKE has comparable performance with the simple implementation and is about ten times faster. Figure 3(h) shows the results with different values of ν in the range $[0.01, 0.3]$. We see that the influence coverage has a small variation and thus is not sensitive to this configuration.

As Figure 3(e-f) displays, the performance of SKE falls behind PageRank on the LTM experiments. Such results are expected. LTM, as well as the weighted independent cascade model [15], requires a certain proportion of v's neighbours

[2] `http://snap.stanford.edu/data/index.html`



(a) ca-GrQc; ICM with probability 0.1

(b) ca-HepTh; ICM with probability 0.1

(c) ca-GrQc; ICM with probability 0.2

(d) ca-HepTh; ICM with probability 0.2

(e) ca-GrQc; LTM

(f) ca-HepTh; LTM

(g) ca-AstroPh & ca-HepPh; 100 seeds; ICM with probability 0.1

(h) Effect of the parameter ν

Fig. 3. The number of influenced nodes on collaboration networks

to be activated in order for v to be activated. However, SKE places seeds so that they have less overlap and thus is not recommended in similar diffusion models, where more exposure to the spread increases the likelihood of activation.

6 Conclusion

We have extended a recently proposed skeleton learning approach [20] to social network analysis. From an information diffusion perspective, the method aims to identify representative individuals that have greater potential influence over the network. In a minimizing communication cost framework, the gradient-based optimization naturally allows nodes to cast negative votes to each other in order to derive a set of mutually exclusive candidates. Consequently, the resulting representatives lie in different regions which helps avoid overlap of neighbour sets. The computational complexity in each optimization step is improved from $O(|\mathcal{V}|^2)$ to $O(|\mathcal{E}|)$ ($\mathcal{V}$: node set; $\mathcal{E}$: edge set) and is further boosted with stochastic gradient descent. As presented in our experiments, this approach is able to discover important individuals who have fewer connections and are thus not considered by traditional methods such as PageRank. On real collaboration networks with the independent cascade model [17], the proposed method outperforms the traditional ranking algorithms and the degree discount heuristic [18]. As for future work, we are interested in varying this technique for linear threshold model [16] and exploring other application scenarios such as community finding [5, 28].

Acknowledgements. This work is supported by Swiss National Science Foundation (SNSF) via the NCCR (IM)2, the European COST Action on Multilingual and Multifaceted Interactive Information Access (MUMIA) via the Swiss State Secretariat for Education and Research (SER), and Irish CLIQUE Strategic Research Cluster (SFI grant no. 08/SRC/I1407).

References

1. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. 22(8), 888–905 (2000)
2. Ng, A.Y., Jordan, M.I., Weiss, Y.: On spectral clustering: Analysis and an algorithm. In: NIPS 14, pp. 849–856. MIT Press (2002)
3. Luxburg, U.: A tutorial on spectral clustering. Stat. Comput. 17(4), 395–416 (2007)
4. White, S., Smyth, P.: A spectral clustering approach to finding communities in graphs. In: Proc. SDM 2005, pp. 274–285 (2005)
5. Newman, M.E.J.: Finding community structure in networks using the eigenvectors of matrices. Phys. Rev. E 74(3), 036104 (2006)
6. Xu, K.S., Kliger, M., Chen, Y., Woolf, P.J., Hero III, A.O.: Revealing social networks of spammers through spectral clustering. In: Proc. ICC 2009, pp. 735–740 (2009)
7. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. Comput. Networks ISDN 30(1-7), 107–117 (1998)

8. Kleinberg, J.M.: Authoritative sources in a hyperlinked environment. J. ACM 46(5), 604–632 (1999)
9. Lempel, R., Moran, S.: The stochastic approach for link-structure analysis (SALSA) and the TKC effect. Comput. Netw. 33(1-6), 387–401 (2000)
10. Chung, F.R.K.: Spectral Graph Theory. Amer. Math. Soc. (1997)
11. Zhou, D., Huang, J., Schölkopf, B.: Learning from labeled and unlabeled data on a directed graph. In: Proc. ICML 2005, pp. 1036–1043 (2005)
12. Agarwal, A., Chakrabarti, S.: Learning random walks to rank nodes in graphs. In: Proc. ICML 2007, pp. 9–16 (2007)
13. Domingos, P., Richardson, M.: Mining the network value of customers. In: Proc. SIGKDD 2001, pp. 57–66 (2001)
14. Richardson, M., Domingos, P.: Mining knowledge-sharing sites for viral marketing. In: Proc. SIGKDD 2002, pp. 61–70 (2002)
15. Kempe, D., Kleinberg, J., Tardos, E.: Maximizing the spread of influence through a social network. In: Proc. SIGKDD 2003, pp. 137–146 (2003)
16. Granovetter, M.: Threshold models of collective behavior. Am. J. Sociol. 83(6), 1420–1443 (1978)
17. Goldenberg, J., Libai, B., Muller, E.: Talk of the network: A complex systems look at the underlying process of word-of-mouth. Market. Lett. 12(3), 211–223 (2001)
18. Chen, W., Wang, Y., Yang, S.: Efficient influence maximization in social networks. In: Proc. SIGKDD 2009, pp. 199–208 (2009)
19. Kitsak, M., Gallos, L., Havlin, S., Liljeros, F., Muchnik, L., Stanley, H., Makse, H.: Identification of influential spreaders in complex networks. Nature Phys. 6(11), 889–893 (2010)
20. Sun, K., Bruno, E., Marchand-Maillet, S.: Unsupervised skeleton learning for manifold denoising. In: Proc. ICPR 2012, pp. 2719–2722 (2012)
21. Hinton, G.E., Roweis, S.T.: Stochastic Neighbor Embedding. In: NIPS 15, pp. 833–840. MIT Press (2003)
22. Hein, M., Maier, M.: Manifold denoising. In: NIPS 19, pp. 561–568. MIT Press (2007)
23. Myers, S.A., Zhu, C., Leskovec, J.: Information diffusion and external influence in networks. In: Proc. SIGKDD 2012, pp. 33–41 (2012)
24. Xu, L.: Learning algorithms for RBF functions and subspace based functions. In: Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods and Techniques, pp. 60–94. IGI Global (2009)
25. Bottou, L.: Online algorithms and stochastic approximations. In: Online Learning and Neural Networks. Cambridge University Press (1998)
26. Stehlé, J., Voirin, N., Barrat, A., Cattuto, C., Isella, L., Pinton, J., Quaggiotto, M., Van den Broeck, W., Régis, C., Lina, B., Vanhems, P.: High-resolution measurements of face-to-face contact patterns in a primary school. PLoS ONE 6(8), e23176 (2011)
27. Backstrom, L., Leskovec, J.: Supervised random walks: predicting and recommending links in social networks. In: Proc. WSDM 2011, pp. 635–644 (2011)
28. Lancichinetti, A., Radicchi, F., Ramasco, J.J., Fortunato, S.: Finding statistically significant communities in networks. PLoS ONE 6(4), e18961 (2011)

Tracing Influential Nodes in a Social Network with Competing Information

Bolei Zhang, Zhuzhong Qian, Xiaoliang Wang, and Sanglu Lu

National Key Laboratory for Novel Software Technology
Nanjing University
Nanjing 210023, China
zhangbolei@dislab.nju.edu.cn, {qzz,waxili,sanglu}@nju.edu.cn

Abstract. We consider the problem of *competitive influence maximization* where multiple pieces of information are spreading simultaneously in a social network. In this problem, we need to identify a small number of influential nodes as first adopters of our information so that the information can be spread to as many nodes as possible with competition against adversary information. We first propose a generalized model of competitive information diffusion by explicitly characterizing the preferences of nodes. Under this generalized model, we show that the influence spreading process is no longer submodular, which implies that the widely used greedy algorithm does not have performance guarantee. So we propose a simple yet effective heuristic algorithm by tracing the information back according to a properly designed random walk on the network, based on the postulation that all initially inactive nodes can be influenced by our information. Extensive experiments are conducted to evaluate the performance of our algorithm. The results show that our algorithm outperforms many other algorithms in most cases, and is very scalable due to its low running time.

1 Introduction

Online social networks such as Facebook and Twitter are becoming an important medium for fast and widespread dissemination of ideas, innovations and products [6, 7]. Substantial attention has been gained to investigating the information spreading in these networks [1–4]. One interesting problem with practical importance, which is formally referred to as *influence maximization*, is to find a small set of influential nodes (seed nodes) properly, through which the information can be spread to as many nodes as possible under a cascade adoption in the network. Kempe *et al.* [1] first formulated the influence maximization problem by modeling the information diffusion as a discrete stochastic process. They further show that the influence spreading process has the properties of monotonicity and submodularity (i.e. having a diminishing marginal return property). Due to such properties, the greedy algorithm based on a hill-climbing strategy can achieve $(1 - 1/e)$ of the optimal solution.

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 37–48, 2013.

However, in many real world scenarios, there may be competing relationships between multiple pieces of information in the social network, such as the competition between iPhone *vs.* Android, Chrome *vs.* Firefox *vs.* IE, *etc.* For such competing pieces of information, one node usually accepts only one of them and discards all the others. In most cases, a node will accept the information which comes first. But when different pieces of information reach a node at almost the same time, the node needs to choose one of the competing information according to personal preference.

In deciding which information to adopt, several adoption models have been proposed to simulate such process when competitive information reaches a node simultaneously, with respect to different scenarios. For instance, Borodin *et al.* [14] consider that a node will choose uniformly at random one of the incoming information; Budak *et al.* [9] assume that the "good" information always beats the misinformation, while Xinran He *et al.* [13] address that people are more likely to believe the negative information. In contrast to all these works, in this paper, we present a generalized model for competitive information diffusion, where the preferences of nodes are characterized explicitly by a probability distribution and the information to be adopted is determined according to the distribution. As will be mentioned later, our model generalizes the adoption models proposed in [9, 13, 14].

Based on this generalized model, we present a comprehensive study of the *competitive influence maximization* problem [10]. In the presence of adversary information, the goal is to choose a set of seed nodes such that our information can be spread to as many nodes as possible. We show that, under this model, the influence spreading process is no longer submodular, which implies that the typical greedy algorithm cannot guarantee the worst-case performance anymore. Rather than applying the greedy approach, we propose a simple heuristic algorithm using a properly designed random walk on the social network. In this algorithm, by postulating that the specified information has been spread to every node in the network, we identify the most influential nodes by tracing the information back based on the random walk to find where it is most likely from. Extensive experiments are conducted to evaluate the performance and scalability of our algorithm on real social networks with high-clustering and scale-free properties. As shown by the results, our algorithm outperforms many other algorithms in most cases. Besides, compared with the greedy algorithm which is still effective, our algorithm achieves a comparable performance but is much more scalable due to its much less running time.

The rest of this paper is organized as follows: In Section 2, we show previous works on information diffusion processes in social networks. In Section 3, we introduce our generalized model and formalize the competitive influence maximization problem. The main algorithm is presented in Section 4. In Section 5, we compare the performance and scalability of our algorithm with some other heuristics. Section 6 concludes this paper.

2 Related Work

Extensive researches have studied the problem of *influence maximization*. It is the problem of identifying a small number of nodes as seed set so that the information spreading is maximized. Kempe *et al.* [1] first formulated influence maximization as a discrete optimization problem with *Independent Cascade Model* (ICM) and *Linear Threshold Model* (LTM). Both models have the properties of monotonicity and submodularity. With such properties, greedy algorithm using hill-climbing strategies can achieve $(1 - 1/e)$ of the optimal. However, this algorithm needs Monte Carlo method to simulate the network massive times so is computationally expensive. Many following works have been proposed to improve the efficiency of this algorithm [2–5], but scalability remains a key challenge. Moreover, they ignore the effects of competing information.

Recent researches show that there exists competing campaigns in real social networks. A lot of them have focused on deciding which information to choose when competing innovations or products reach at the same time. Barathi *et al.* [10] studied a similar problem where there are multiple players spreading their information to compete in one social network. How would each player choose the set of early adopters to begin the competing campaign? The authors augmented ICM by adding continuous time for each edge so information will not compete on the nodes. They also assumed that diffusion probability for different players is the same and show that the influence spreading process is submodular for the *last player*. Borodin *et al.* [14] considered the competitive information diffusion under threshold models. In their model, a node will choose randomly uniformly one of the incoming information to adopt. Budak *et al.* [9] investigated the problem of limiting misinformation in a social network. In the presence of misinformation, which k nodes should be chosen as “good” information adopters to limit the spreading of misinformation. In their diffusion model, the good information always beats the misinformation when they reach a person at the same time. The problem is submodular when the limiting campaign information is accepted by users with probability 1. Xinran He *et al.* [13] also studied the limiting problem but they thought misinformation always wins because people are more likely to believe negative opinions. This problem is submodular under LTM. Most of previous works try to give special cases of the diffusion model and prove the influence spreading process is submodular. However, in our generalized model, the competing influence maximization might not have such property and the general cases remain unexplored.

3 Competitive Information Diffusion in Social Networks

In this section, we first present our generalized model for competitive information diffusion. Then we will formulate the *competitive influence maximization* as a discrete optimization problem.

3.1 Diffusion Models

The social network is often represented as a directed graph $G = (V, E)$ where nodes represent individuals and edges represent social relationships between them. We use *Independent Cascade Model* (ICM) as the basic model for information diffusion. In a information cascade, we say that a node is *active* if it adopts the information, otherwise it is called *inactive*. Initially, only the seed nodes in S are active. The diffusion process starts from the set S and unfolds in discrete steps as follows: In each step t, the newly active nodes in S_t try to activate their neighbors with probability p_e independently, where p_e is an activation probability associated with each edge $e \in E$. The newly activated nodes are added into S_{t+1}. This process continues until no more nodes are activated at some step t, i.e. $S_t = \emptyset$.

We augment the ICM to present a generalized model called *Weighted Competitive Independent Cascade Model* (WCICM). In this model, competing information cascades start from disjoint seed sets $S_1, S_2, ..., S_r$ and spread in the social network to compete. We say a node is in color i if it adopts the ith information. Initially, the nodes in S_i are in color i and there is no color for the inactive nodes. The process unfolds in discrete steps as follows: at step t, a node u in $S_i^{(t)}$ tries to activate each of its inactive outgoing edge with probability p_{uv}^i. If edge (u, v) is activated, it is colored as $c_{uv} = i$ at *diffusion step* $T_{uv} = t$. If there is no other incoming edge to v at this step, then v will be colored as $c_v = i$ at *diffusion step* $T_v = t$. But if multiple processes reach v at the same step, v chooses one of them to adopt. In deciding which information to choose, users' decision is characterized explicitly by a probability distribution. By assigning a weight $\phi(i)$ for information with color i, v adopts color i with probability $Pr[v\ adopts\ color\ i] = \frac{\sum_{(u,v)\in E \wedge c_{uv}=i} \phi(i)}{\sum_{(s,v)\in E} \phi(c_{sv})}$, that is, the decision of v adopting color i is determined by the weight of color i versus the total weight of incoming information. If the weights are the same, v will choose uniformly at random one of the information. Or if the weight is positive for one information and 0 for others, this information will always win. So our model can generalize previous models introduced in [9, 13, 14].

3.2 Problem Formulation

Consider the problem when r players compete in a social network G. Each of them selects a disjoint set of seed nodes S_i to start the competing campaign sequentially. The information is spread in the network under the WCICM described above. Suppose the first $r-1$ players' strategies have been fixed, namely we have the knowledge of $S_1, S_2, ..., S_{r-1}$. We need to identify a seed set S_r of size k for the *last player*. The goal is to maximize the expected number of nodes in color r after information cascade. This *competitive influence maximization* problem can be formulated as:

$$\max_{S_r \subseteq V} \sigma(S_r)\ s.t.\ |S_r| = k \tag{1}$$

where $\sigma(S_i)$ is the expected number of nodes in color i when all information diffusion processes stop.

4 Competitive Influence Maximization

In this section, we first show the NP-hardness of the problem and prove it does not exhibit the submodular property. Then we will present a heuristic using properly designed random walk on the social network to find the influential nodes.

4.1 Hardness of Competitive Influence Maximization

Consider a special case of the competitive influence maximization problem when there are no competing adversaries. Then it is exactly the problem of influence maximization problem under the ICM, which can be reduced from the NP-complete *Set Cover* problem in [1].

Theorem 1. *The competitive influence maximization problem is NP-hard under the WCICM.*

Influence spreading of single seed set is monotone and submodular under the ICM. Typically, a function $\sigma(\cdot)$ is said to be submodular if it satisfies: $\sigma(S \cup \{v\}) - \sigma(S) \geq \sigma(T \cup \{v\}) - \sigma(T)$ for all elements v when $S \subseteq T$. With this property, the greedy algorithm using hill-climbing method can achieve near optimal solution with performance guarantee. However, in a competing campaign, the influence spreading process is not submodular.

Claim. There exists counter example which is not submodular in the competitive influence maximization under the WCICM.

Proof. Figure 1 shows a counter example: Adversary information starts from node S_1 with color 1. Define $S = \emptyset$ and $T = \{w\}$ with color 2. We use identical weight for each information, so a node will choose uniformly if two precesses reach it at the same time. The edge in color i means the probability of information diffusion of S_i on this edge is 1. Initially, we have $\sigma(S) = 0$ and $\sigma(T) = 2.5$ before adding v. When node v is set as color 2, we have $\sigma(S \cup \{v\}) = 1$ and $\sigma(T \cup \{v\}) = 4$. So there is $\sigma(S \cup \{v\}) - \sigma(S) < \sigma(T \cup \{v\}) - \sigma(T)$, which implies that the process does not exhibit the property of submodularity.

4.2 Random Walk to Find Influential Nodes

Since the influence spreading process under WCICM does not exhibit submodular property, the greedy algorithm does not have performance guarantee. We propose a heuristic using a properly designed random walk to find the influentials. In this algorithm, we assume an ideal situation where all initially inactive

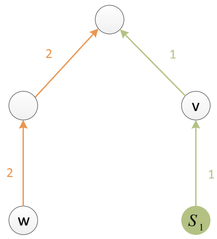


Fig. 1. Counter example where the competitive influence spreading process does not exhibit submodular property

nodes have been influenced by our information with color r. Under this postulation, the information on each node traces back to find where it most likely to be from. The nodes aggregating most information can be identified as our seed nodes. To trace back, information random walks in the *reverse direction* of where it was propagated from. The transfer probability of the information walking from node u to node v is:

$$P_{uv} = \frac{w_{vu}}{\sum_{s:(s,u)\in E} w_{su}} \tag{2}$$

where w_{vu} is the weight of directed edge (v, u). So P_{uv} is the weighted proportion of edge (v, u) among all incoming edges of u. It is easy to see that $\sum_v P_{uv} = 1$, and the transition probability of node u is only dependent on its current state. So this is a Markov process with transition matrix P.

Specifically, if we define w_{uv} as 1 for any u and v, then this process is Pagerank [15] with reverse directions. However, Pagerank neglects the impacts of competing information. In this work, the weight of each edge is determined by how much v can influence u. Generally, if moving the information on u to v can increase the probability of v influencing u, then the edge (v, u) has higher weight. So we define the weight of (v, u) as:

$$w_{vu} = Pr[v \ activates \ u | v \in S_r] \tag{3}$$

This means that the weight of (v, u) can be represented as the probability that v can influence u given that v is a seed node. In a competing campaign, if v is a seed node: *a)*, the information from v can reach u first of all, $Pr[v \ activates \ u] = 1$; *b)*, the information from v will reach u with adversary information at the same time; *c)*, the information from v won't reach u or the information will reach u after some adversary information, $Pr[v \ activates \ u] = 0$. Above all three cases, the probability depends on when will adversary information reach u. However, it is hard to estimate the *diffusion step* of adversary information in a stochastic diffusion process.

One possible solution is to simulate the network massive times to get deterministic graphs each time. We can view the information diffusion on edges and nodes as the results of coin flip with bias. In a deterministic graph, the diffusion step of adversary information can be obtained using a BFS linear scan of the graph. The result is the average information value after convergence of each outcomes of deterministic graphs. However, simulating the graph still takes too much time. To get the influential nodes more efficiently, we follow the idea of *Shortest Path Model* (SPM) introduced in [16]. In this model, a node can only be activated through the shortest path between the initially active seed set. So the *diffusion step* of each node is the shortest path between the node and the seed nodes.

Claim. Assume the inactive node can only be influenced by the nearest seed node, a node w will be colored as i if $\exists v \in S_i$ such that $\forall u \in S_j$ $(j \neq i \wedge j \in \{0, 1, ..., r\})$, $|SP(v, w)| < |SP(u, w)|$ where $SP(v, w)$ denotes the shortest path from node v to node w.

Figure 2 presents one example of the diffusion process. The nodes 0 and 9 are adversary seed nodes. The diffusion step is labeled in the brackets above each node. The value is the length of shortest path from the seed nodes. The nodes that are unreachable from seed nodes have step ∞, they are nodes 4 and 5.

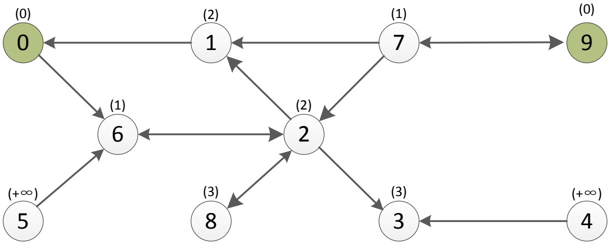


Fig. 2. One sample example of information diffusion process of adversary information

When multiple pieces of information reach u at the same time, the information from v would compete with adversary information to influence u. Since the information diffusion process is probabilistic, we need to get the *propagation probability* of the edge. In most cases, the activation probability p is often not uniformly distributed, and there might be multiple edges between a pair of nodes. Suppose the edges between node v and u are $(v, u)^1, (v, u)^2, ...,$ each with probability p_{vu}^i to be activated independently. Then the *propagation probability* of edge (u, v) is

$$pp_{vu} = 1 - \prod_{i:(v,u)^i \in E} (1 - p_{vu}^i) \tag{4}$$

Suppose v is a seed node with color r, and the adversary information from node w will reach u at diffusion step T_{wu}, then the probability of v can influence u is

$$Pr[v\ activates\ u|v \in S_r] = \frac{\phi(c_{vu}) * pp_{vu}}{\sum_{w:(w,u)\in E \wedge T_{wu}=1} \phi(c_{wu}) * pp_{wu} + \phi(c_{vu}) * pp_{vu}} \tag{5}$$

which is the weight of directed edge (v, u): w_{vu}. For a large scale-free social network, it is almost impossible that the transition matrix P is periodic. The random walk will converge to a stationary distribution. In the experiment, we show that the distribution converges very fast. The detail of this algorithm is presented in Algorithm 1. We call this algorithm *CompeteRank*.

Algorithm 1. $CompeteRank(G, S_{[r-1]}, k, p, max\_iterations, min\_delta)$

```
$S_{adversary} \leftarrow \bigcup_{i\in\{1,2,\ldots,r-1\}} S_i$;
foreach $v \in V/S_{adversary}$ do $T_v \leftarrow \infty$, $I_v \leftarrow \frac{1}{|V-S_{adversary}|}$;
foreach $v \in S_{adversary}$ do $T_v \leftarrow 0$;
for $v \in V$ do                                  // Get the diffusion step of each node
    for $s \in S_{adversary}$ do
        Get shortest path $SP(s, v)$, $u$ is the second last node on $SP(s, v)$;
        if $|SP(s, v)| < T_v$ then
            $T_v \leftarrow |SP(s, v)|$;
            $v.weights \leftarrow pp_{uv} * \phi(c_{uv})$;
        else if $|SP(v, s)| = T_v$ then
            $v.weights \leftarrow v.weights + pp_{uv} * \phi(c_{uv})$;
for $u \in V/S_{adversary}$ do                      // Compute the transition matrix $P$
    for $v : (v, u) \in E$ do
        $w_{uv} \leftarrow \frac{pp_{vu}*\phi(c_{vu})}{u.weights+pp_{vu}*\phi(c_{vu})}$;
        $P_{uv} \leftarrow \frac{w_{vu}}{\sum_{s:(s,u)\in E} w_{su}}$;
for $i \leftarrow 1$ to $max\_iterations$ do          // Random walk to find influentials
    foreach $v \in V$ do $I'_v \leftarrow I_v$;
    for $u \in V/S_{adversary}$ do
        for $v : (v, u) \in E$ do
            $I_u \leftarrow I_u + I'_v * P_{vu}$;
    $diff \leftarrow \sum_{v\in V} |I'_v - I_v|$;
    if $diff < min\_delta$ then
        break;
pick the first $k$ nodes as $S_r$;
```

5 Evaluation

5.1 Experiment Setup

We use 3 social network data sets from [17] to evaluate our algorithms. The first one is ca-GrQc: an academic collaboration network from scientific collaborations between authors' papers submitted to General Relativity and Quantum

Cosmology category with 5242 nodes and 28980 edges. The nodes in the network represent authors and the edges indicate coauthor relationships. Each coauthor paper is represented as a single chance for one author to influence another. The second one is ca-HepTh: a collaboration network from papers submitted to High Energy Physics with 9877 nodes and 51971 edges. The third one is ca-CondMat: also a collaboration network from papers submitted to Condense Matter category with 23133 nodes and 186936 edges. All of them are scale-free networks with high clustering. Without loss of generality, we use identical activation probability for the edges as 0.1 or 0.01. In *CompeteRank*, $max\_iterations = 1000$ and $min\_delta = 0.00001$ are considered to be reasonable threshold values. Since the influence spreading is a stochastic process, we use Monte-Carlo method to simulate the graph for $R = 1000$ times to get the average influence value of the competing seed sets under the WCICM. The algorithms compared to *CompeteRank* are:

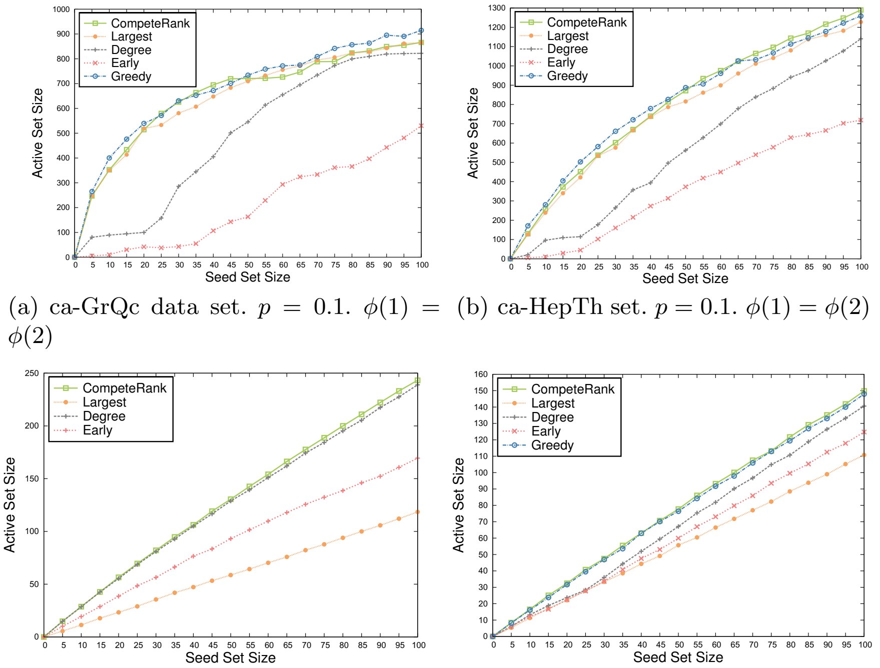
- **Greedy**: This algorithm uses a hill-climbing strategy to greedily find the node that has maximal influence at each step.
- **Degree Centrality**: The heuristic identifies the nodes with highest degree.
- **Early Infectees**: It chooses seeds that are expected to be infected first. The graph is simulated R times, and the nodes are ordered by the number of simulations they are firstly infected.
- **Largest Infectees**: This heuristic chooses seeds that are expected to the most nodes if they were to be infected themselves. A more detailed description can be referred to [9]. The graph should also be simulated R times.

5.2 Competitive Influence Spreading

We first evaluate the influence spreading of different algorithms on the data sets. The adversary seed nodes are chosen using *Degree Centrality* with fixed size as 100. Since we do not care the influence spreading of adversary nodes, they can be assumed from single player. Figure 3 presents the results of our experiments.

Figure 3(a) and Figure 3(b) are the results of ca-GrQc and ca-HepTh data set with $p = 0.1$. We use identical weight for different information, so when multiple pieces of information reach a node at the same time, it will choose randomly uniformly one of them. In this case, *Greedy*, *Largest* and *CompeteRank* all perform very well. Even the influence spreading function is not submodular, *Greedy* is still effective. This might be the reason that counter examples rarely exist. *Early* performs very poor in both experiments. So blocking the influence spreading of adversary nodes does not help much in the spreading of our information. *CompeteRank* outperforms the *Degree* over about 50% sometimes. The gap is even larger than influence maximization of single set. This is because *Degree* neglects the effect of adversary information.

Figure 3(c) is the result of ca-CondMat data set with $p = 0.01$. We do not include the *Greedy* algorithm in this experiment since it takes too much time. When p is small, the effect of competing nodes also becomes smaller. So *Largest*



(a) ca-GrQc data set. $p = 0.1$. $\phi(1) = \phi(2)$

(b) ca-HepTh set. $p = 0.1$. $\phi(1) = \phi(2)$

(c) ca-CondMat data set. $p = 0.01$. $\phi(1) = \phi(2)$

(d) ca-HepTh data set. $p = 0.01$. $\phi(1) = 1, \phi(2) = 0$

Fig. 3. Influence Spreading of Seed Sets

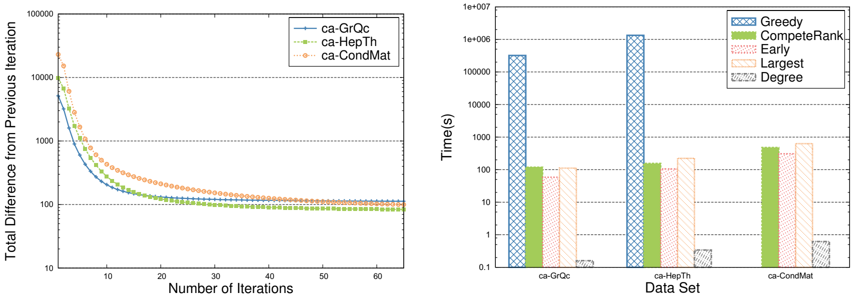
is not effective as before while *Degree* achieves much better performance. Clearly, our *CompeteRank* performs the best of all.

In Figure 3(d), we show the results of ca-HepTh with $p = 0.01$. We have $\phi(1) = 1$ and $\phi(2) = 0$, so adversary information always beats our information when they reach a node at the same time. In this case, *CompeteRank* performs even better than the *Greedy* algorithm. And both of them significantly outperform others heuristics.

5.3 Convergence and Scalability

In Figure 4(a) we show the iterations for our algorithm to converge. We use $I_v = 1$ for the each of the node initially. The algorithm can converge to a reasonable tolerance in about 50 iterations in all three cases. Generally, the social networks are expander-like graphs. The random walk on an expander which has an eigenvalue separation is rapidly-mixing.

Figure 4(b) presents the running time of our algorithms. In this experiment, we use identical $p = 0.01$ for all the data sets to select 100 seed nodes. It is evident that *Greedy* takes too much time. What is worse, some improvements like the "CELF" proposed by Leskovec *et al.* [2] does not work here, because the influence spreading process is not submodular. *Degree* is the most efficient



(a) Convergence of *CompeteRank* on different data sets

(b) Running time of different algorithms. The number of adversary node is fixed as 100 with $p = 0.01$

Fig. 4. Convergence and Scalability

of all. The running time of *CompeteRank*, *Largest* and *Early* are similar with an acceptable running time. Moreover, the time of *CompeteRank* does not grow much with the increase size of data sets.

6 Conclusion

In this work, we studied the problem of *competitive influence maximization* in a social network. We introduced a generalized model called WCICM for competitive information diffusion which could characterize users' preference for each information explicitly. In this model, the influence spreading process is no longer submodular, so greedy approach does not have performance guarantee. We proposed a simple yet effective heuristic algorithm called *CompeteRank*. In this algorithm, the influential nodes can be identified by tracing the information back according to a properly designed random walk on the network, based on the postulation that all the nodes have been influenced. Extensive experiments on different data sets were conducted. The results revealed that even without submodular property, the greedy algorithm can still be effective. However, the computation cost is too expensive. Our algorithm is very comparable to the greedy approach and outperforms other well-known heuristics in most cases. Some of them, like *Largest* and *Degree*, only perform well in certain circumstances. We also analyzed the convergence and scalability of our algorithm. The results showed that *CompeteRank* can converge very fast with low running time.

Acknowledgment. The authors would like to thank Bin Tang for his insightful remarks. This work is partially supported by the National Natural Science Foundation of China under Grant No. 61073028, 61021062, 61202113, 61003213; the National Basic Research Program of China (973) under Grant No. 2009CB320705; Jiangsu Natural Science Foundation under Grant No. BK2011510.

References

1. Kempe, D., Kleinberg, J.M., Tardos, É.: Maximizing the spread of influence through a social network. In: KDD, pp. 137–146 (2003)
2. Leskovec, J., Krause, A., Guestrin, C., Faloutsos, C., VanBriesen, J.M., Glance, N.S.: Cost-effective outbreak detection in networks. In: KDD, pp. 420–429 (2007)
3. Chen, W., Wang, Y., Yang, S.: Efficient influence maximization in social networks. In: KDD, pp. 199–208 (2009)
4. Chen, W., Wang, C., Wang, Y.: Scalable influence maximization for prevalent viral marketing in large-scale social networks. In: KDD, pp.1029–1038 (2010)
5. Zhang, Y., Gu, Q., Zheng, J., Chen, D.: Estimate on Expectation for Influence Maximization in Social Networks. In: Zaki, M.J., Yu, J.X., Ravindran, B., Pudi, V. (eds.) PAKDD 2010, Part I. LNCS, vol. 6118, pp. 99–106. Springer, Heidelberg (2010)
6. Tang, J., Sun, J., Wang, C., Yang, Z.: Social influence analysis in large-scale networks. In: KDD, pp. 807–816 (2009)
7. Jo, Y., Hopcroft, J.E., Lagoze, C.: The web of topics: discovering the topology of topic evolution in a corpus. In: WWW, pp. 257–266 (2011)
8. Gomez-Rodriguez, M., Leskovec, J., Krause, A.: Inferring networks of diffusion and influence. In: KDD, pp. 1019–1028 (2010)
9. Budak, C., Agrawal, D., El Abbadi, A.: Limiting the spread of misinformation in social networks. In: WWW, pp. 665–674 (2011)
10. Bharathi, S., Kempe, D., Salek, M.: Competitive Influence Maximization in Social Networks. In: Deng, X., Graham, F.C. (eds.) WINE 2007. LNCS, vol. 4858, pp. 306–311. Springer, Heidelberg (2007)
11. Kostka, J., Oswald, Y.A., Wattenhofer, R.: Word of Mouth: Rumor Dissemination in Social Networks. In: Shvartsman, A.A., Felber, P. (eds.) SIROCCO 2008. LNCS, vol. 5058, pp. 185–196. Springer, Heidelberg (2008)
12. Pathak, N., Banerjee, A., Srivastava, J.: A Generalized Linear Threshold Model for Multiple Cascades. In: ICDM, pp. 965–970 (2010)
13. He, X., Song, G., Chen, W., Jiang, Q.: Influence Blocking Maximization in Social Networks under the Competitive Linear Threshold Model. In: SDM, pp. 463-474 (2012)
14. Borodin, A., Filmus, Y., Oren, J.: Threshold Models for Competitive Influence in Social Networks. In: Saberi, A. (ed.) WINE 2010. LNCS, vol. 6484, pp. 539–550. Springer, Heidelberg (2010)
15. Page, L., Brin, S., Motwani, R., Winograd, T.: The PageRank Citation Ranking: Bringing Order to the Web. Technical Report. Stanford InfoLab (1999)
16. 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)
17. Collaboration networks, http://snap.stanford.edu/data/

ViStruclizer: A Structural Visualizer for Multi-dimensional Social Networks

Bing Tian Dai, Agus Trisnajaya Kwee, and Ee-Peng Lim

Living Analytics Research Centre
Singapore Management University
80 Stamford Road, Singapore 178902
{btdai,aguskwee,eplim}@smu.edu.sg

Abstract. With the popularity of Web 2.0 sites, social networks today increasingly involve different kinds of relationships among different types of users in a single network. Such social networks are said to be multi-dimensional. Analyzing multi-dimensional networks is a challenging research task that requires intelligent visualization techniques. In this paper, we therefore propose a visual analytics tool called *ViStruclizer* to analyze structures embedded in a multi-dimensional social network. *ViStruclizer* incorporates structure analyzers that summarize social networks into both node clusters each representing a set of users, and edge clusters representing relationships between users in the node clusters. *ViStruclizer* supports user interactions to examine specific clusters of users and inter-cluster relationships, as well as to refine the learnt structural summary.

1 Introduction

Web 2.0 sites widely adopt online social networks as the means to connect their users with one another to encourage users to participate in information sharing (e.g., friendship network in Facebook[1]) as well as to collaborate with one another (e.g., collaboration networks in Wikipedia[2]). Unlike traditional social networks which involve a single type of nodes or actors and a single relation type, these online social networks are *heterogeneous* and of *large scale*, where multiple types of nodes and relations may exist in the same network.

In network analysis, the term “mode” refers to a distinct set of entities [14], a network with different types of objects is therefore called a ***Multi-Mode Network***.

Example 1. An online social network provides a platform for friends to share photos and videos. There are three types of objects in this network, namely people, photos and videos. This network is therefore a multi-mode network.

We would like to make a remark about the modes here. The modes of nodes give an explicit grouping of the nodes. This grouping is however, not always

[1] www.facebook.com

[2] www.wikipedia.org

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 49–60, 2013.

informative as there are other implicit groupings of nodes that are more interesting. The implicit structure to be discovered should be orthogonal to the modes of the nodes, and it reveals the implicit groups from the observed interactions among the nodes. Hence, for multi-mode networks, it is important to extract their implicit structures even though the modes are known.

There are also networks with different types of relations, each representing one type of interaction. It is often that multiple relations co-exist between a pair of individuals. For example, a user in the network can send a message to another user, or comment on another user's status. Since each type of interaction suggests a different association between two users [5], if we had only one relation to represent different types of interactions, there would have been too much loss in the network information. Therefore, we define a ***Multi-Relational Network*** to be one that describes the relationship from one object to another as a composition of different relations.

Example 2. Individuals in the social network, mentioned in Example 1, send messages or comments to one another. Each user may also upload a photo or a video, or comment on others' photos or videos. There may exist some interaction links between a photo and a video, if they are taken at the same location, or the photo is captured from the video. This network is illustrated by Figure 1, where multiple types of relations and objects can be found.

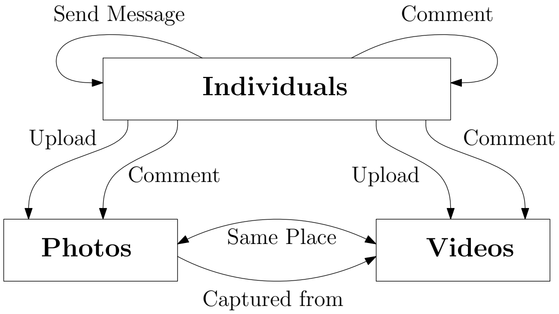


Fig. 1. A Multi-Dimensional Social Network

The network in Figure 1 is multi-mode and multi-relational at the same time. A network with heterogeneous types of nodes and relations is therefore regarded as a ***Multi-Dimensional Network***. Multi-mode networks and multi-relational networks are just special cases of multi-dimensional networks. The analysis of multi-dimensional networks is known to be harder than simple networks and is currently an active research topic [4].

As social networks grow in size, they become more difficult to analyze as there are many nodes and relation edges. A typical user can only handle less than a hundred of nodes and edges. Beyond that, visualization techniques have to be used [2,8,11]. These techniques usually summarize large social networks into smaller and simpler networks that are human-interpretable. This summarization process essentially groups nodes and edges in an original network into a simple network of node clusters and relationships between node clusters. The grouping however is not arbitrary. It ought to preserve information contained in the original network. Otherwise in the extreme case, the summarized network may consist of only one node cluster containing all nodes and such a summarized network does not help user-interpretation.

Many existing network visualization and summarization techniques [10,7,13,16] are designed mainly for simple social networks, but not for multi-dimensional networks. Recent works, OntoVis[12] and FacetAtlas[3], focus on visualizations of heterogeneous networks. However, OntoVis is mainly designed for multi-mode networks, as edge types are determined by node types and multiple types of relations between two nodes are not taken into consideration. FacetAtlas visualizes multi-dimensional networks in multiple facets. But again, their nodes are connected by at most one type of relation, i.e., an edge within a facet.

In this paper, we focus on visualizing multi-dimensional networks by designing and implementing a network visual analytics system, called ***ViStruclizer***, which presents a summarized network structure from a multi-dimensional network as an overlay over the original set of nodes and relations for human users to interpret. Given that multi-mode and multi-relational networks are special classes of multi-dimensional networks, *ViStruclizer* also works well on these networks.

In the absence of research in visualizing multi-dimensional networks, we define two key research objectives for the *ViStruclizer* project. The first objective is to introduce a design framework that can be adopted by *ViStruclizer* and other future visual analytics systems for multi-dimensional networks. The framework identifies the required system components, their corresponding functionalities and how these components interact with one another. The second research objective is to develop the working visual analytics system and to demonstrate features that support the important visualization requirements. To meet the above objectives, we have three major contributions in this paper, namely:

- We propose a network visualizer system design framework that covers the essential system modules and the desired functions for analyzing a multi-dimensional network. The design framework also defines the network structure representation that summarizes the original network.
- We have developed a visualizer called *ViStruclizer* to realize our proposed system design framework. *ViStruclizer* is capable of deriving a network structure from a large multi-dimensional social network, and allowing users to examine and navigate the derived network structure. The structure captures the positions of the individuals and their roles in the network.

- We propose a means in *ViStruclizer* to allow users to exercise their own judgments on the structures of social networks and to refine the automatically learnt network structures according to the users' preferences. For example, one may want to correct an individual's position in the learnt network structure. This is achieved by making *ViStruclizer* a *semi-supervised* system.

2 The *ViStruclizer* System Design Framework

In this section, we discuss the system design framework of *ViStruclizer* for multi-dimensional networks in three parts. We first describe the representation of multi-dimensional networks in some high level summary networks in Section 2.1. This is followed by the architecture framework of *ViStruclizer* and its components (see Section 2.2). We finally outline the interactive visualization requirements to be supported by *ViStruclizer* in Section 2.3.

2.1 Summary of Multi-Dimensional Networks

Let $G = (V, E, M, R, m, r)$ denote a multi-dimensional network, where V is the set of vertices, E is the set of edges, $M = \{M_1, \ldots, M_t, \ldots\}$ is the set of vertex modes and $R = \{R_1, \ldots, R_s, \ldots\}$ is the set of relations. The second last element of G, $m : V \to M$, is the mapping from vertices to their modes; each vertex V_i has a definite mode $m(V_i)$. Compared to the vertex modes, it is more complicated to represent the edges since an edge is a combination of different type of relations. Thus the last element of G, $r : E \to R_1 \times \cdots \times R_s \times \cdots$, is defined as a mapping from edges to the Cartesian product on R, i.e., $r(E_{i,j}) = (r_1, \ldots, r_s, \ldots)$, each r_s is the number of times the relation R_s is observed from vertex V_i to V_j. For example, $r(E_{i,j}) = (1, 1, 2)$ indicates that $E_{i,j}$ consists of one occurrence of relation R_1, one occurrence of R_2, and two occurrences of R_3. We also denote the value of $r(E_{i,j})$ on relation R_s by $r_s(E_{i,j})$.

When the network has large number of vertices and edges, summarizing them by a representative high level network structure becomes necessary. For example, one way to summarize a social network is to group individuals with similar connections into one community, and many social network visualizers focus on density-based community structures, e.g., Vizter [7] and HiMap [13].

As edges are heterogeneous in multi-dimensional networks, not only is it necessary to group similar vertices, but similar edges are also expected to be grouped. *Positional and Role Analysis* groups individuals into one position if they behave similarly, and similar relationships into roles. The structure derived by this analysis treats relationships across different communities as important as relationships within a community, which is more informative than community structures. White, Boorman and Breiger [15] observed that the most informative positional and role analysis requires multiple ties, i.e., a multi-relational network, and they introduced the concept of *blockmodel* to study roles and positions. Wasserman and Faust [14] defined a blockmodel as a partition of vertices into k positions (vertex clusters). Roles from one position to another are modeled by blocks (edge clusters). Airoldi [1] extended blockmodels to mixed membership blockmodels (MMB), where the position of each vertex is modeled by a probabilistic distribution. Dai, Chua and Lim [5] generalized MMB [1] further on multi-relational networks. The blockmodel developed in [5] is called *Generalized Stochastic Blockmodels* (GSBM), which models (i) the position of each vertex by a probabilistic distribution; and (ii) each block by a multivariate distribution over the set of relations.

Without loss of generality, we assume a structural summary for a multi-dimensional network provides two kinds of clusters, vertex clusters and edges clusters. $C = \{C_1, C_2, \dots, C_k\}$ is the set of vertex clusters; for vertex $V_i \in V$, $\pi_{i,u}$ is the membership probability of V_i assigned to cluster C_u, subject to $\sum_{u=1}^{k} \pi_{i,u} = 1$. The *weight of the cluster* C_u is the accumulated probability of all vertices assigned to C_u, i.e.,

$$w(C_u) = \sum_{i=1}^{|V|} \pi_{i,u} \tag{1}$$

The *weight of mode* M_t *of the cluster* C_u is proportional to the accumulated probability of all vertices of mode M_t assigned to C_u, i.e.,

$$w(M_t|C_u) \propto \sum_{m(V_i)=M_t} \pi_{i,u} \tag{2}$$

We also define $B = \{B_{u,v}|C_u, C_v \in C\}$ as the edge clusters from C_u to C_v in the structural. Similarly, $\rho_{i,j,u,v}$ is the membership probability of edge $E_{i,j}$ assigned to $B_{u,v}$, subject to $\sum_{u,v=1}^{k} \rho_{i,j,u,v} = 1$. The *weight of an edge cluster* $B_{u,v}$ is the accumulated probability of all its edges, that is:

$$w(B_{u,v}) = \sum_{E_{i,j}} \rho_{i,j,u,v} \|r(E_{i,j})\| \tag{3}$$

where $\|r(E_{i,j})\| = \sum_{R_s \in R} r_s(E_{i,j})$ is the *1-norm* of $r(E_{i,j})$ since all $r_s(E_{i,j})$ are positive. $\|r(E_{i,j})\|$ can also be other form of norm. The *width* of an edge cluster $B_{u,v}$ is logarithmically proportional to the weight of the edge cluster. The weight of relation R_s of the edge cluster $B_{u,v}$ is proportional to the accumulated probability of relation R_s, i.e.,

$$w(R_s|B_{u,v}) \propto \sum_{E_{i,j}} \rho_{i,j,u,v} r_s(E_{i,j})$$

This is valid with $\|r(E_{i,j})\|$ being the 1-norm, and it ought to be altered for other kinds of norm.

2.2 Architecture Framework

ViStruclizer takes a multi-dimensional social network as input, and its primary objective is to visualize the network through its structural summary. With vertices and edges grouped into clusters with some latent semantics, it helps users to understand the original network.

The architectural framework of *ViStruclizer* is shown in Figure 2. There are three components in *ViStruclizer*, namely the structure analyzer, the back-end database and the visualizer which serves as the frontend interface of *ViStruclizer*.

Our objective is to make *ViStruclizer* general enough to accommodate different structure analyzers that can return different structural summaries. The structure analyzer in *ViStruclizer* either takes input directly from a multi-dimensional network or reads the network from the database, and constructs a structural summary to be presented by the visualizer. The structural summary consists of a mandatory element and another optional element, which will be discussed shortly.

The visualizer component in *ViStruclizer* takes both the original multi-dimensional network from the database and the structural summary as input and presents the network to the users, as shown by the solid arrows in Figure 2. The dash-dotted arrows show that the users can provide their feedback to *ViStruclizer*. The user feedback allows the structure analyzer to further refine the structural summary, and presents the network using refined structural summary to the users.

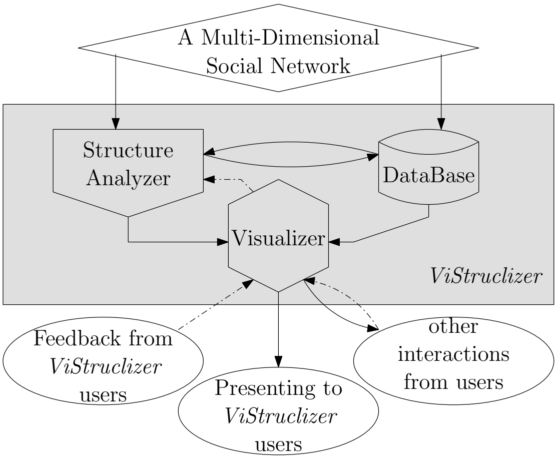

A Multi-Dimensional Social Network
Structure Analyzer
DataBase
Visualizer
ViStruclizer
Feedback from *ViStruclizer* users
Presenting to *ViStruclizer* users
other interactions from users

Fig. 2. The System Design of *ViStruclizer*

Within the structural summary returned by the structure analyzer, the mandatory element refers to the cluster membership for all vertices in the original network. If a structure analyzer returns a hard clustering, the cluster membership for a vertex is then its cluster indicators, i.e., cluster id. In *ViStruclizer*, we consider mixed cluster memberships derived by soft clustering or fuzzy clustering techniques, e.g. EM algorithm [9], which are widely adopted in cluster analysis. The mixed cluster memberships are typically represented in the form of probabilistic distributions, i.e., a vertex is assigned to multiple clusters with different probabilities. In Section 2.1, such probabilities are denoted by $\pi_{i,u}$, which is the probability of assigning vertex V_i and cluster C_u, subject to $\sum_{u=1}^{k} \pi_{i,u} = 1$. Hence the cluster memberships in the structural summary are probabilistic distributions for vertices assigned to vertex clusters.

The optional element for the structure analyzer is to return the cluster memberships for the edges. Similar to the vertex cluster membership, the cluster membership of an edge indicates between which pair of clusters the edge belongs to. If k vertex clusters are extracted by structure analyzer, there exist k^2 edge clusters, representing every possible pairs of k vertex clusters.

If a hard clustering for the vertices is adopted, for an given edge, its edge cluster is defined by the cluster of the source vertex to the cluster of the target vertex. However, when vertices are associated with vertex clusters with mixed memberships, the membership distributions of the edges belonging to an edge cluster become more complicated. This is denoted by $\rho_{i,j,u,v}$ in Section 2.1, which is the cluster membership of edge $E_{i,j}$ assigned to cluster pair (C_u, C_v), subject to $\sum_{u,v=1}^{k} \rho_{i,j,u,v} = 1$. This issue will be discussed in greater detail in Section 3. For now, we would like to design *ViStruclizer* to be able to cope with both kinds of edge memberships.

2.3 Interactive Visualization Requirements of *ViStruclizer*

For *ViStruclizer* to be an effective tool for analyzing multi-dimensional networks interactively, it has to satisfy a few essential visualization requirements mentioned below:

Network Search. Finding a user vertex in a large multi-dimensional network is like searching a needle in the haystack. Vertex search by label and other attributes is therefore essential. Once some target vertices are found, they can serve as the lead to explore the rest of network. When structural summary is available, vertex search can also return the vertex clusters of vertices meeting the search criteria and help users to determine the relevance of these vertices.

Summary Network Visualization. A summary network consists of vertex clusters and edge clusters, and each cluster has its own salient attributes, i.e., cluster weights, cluster modal weights. These clusters and their attributes should be visualized clearly using graphical properties such as shape, color and size. The composition of vertex types and edge types in clusters is also another piece of information to be visually presented clearly. In the visualizer component in *ViStruclizer*, the size of a vertex cluster is determined by its weight. The width of an edge cluster is also logarithmically proportional to the weight of the edge cluster. We use multiple colors to distinguish different kinds of vertices and relations in a multi-dimensional network. An edge cluster is thus represented by a directed multi-color edge from the source vertex cluster to the target vertex cluster such that each color represents a different relation, and the proportion of each color represents the weight of the corresponding relation.

Summary Network Exploration. For a user to find out how vertices are grouped into vertex clusters, *ViStruclizer* has to support user interactively expanding or collapsing vertex clusters. Such a visual operation can be non-trivial for very large vertex clusters each with too many vertices for the user to examine. The main challenge of summary network exploration is therefore to expand the cluster without being overwhelmed by the large number of vertices. This can be achieved by selectively expanding a vertex cluster as opposed to complete vertex cluster expansion. When vertex clusters are derived by soft clustering, one can use a probabilistic threshold to control the extent of vertex cluster expansion by expanding only the vertices with membership probabilities above this threshold. This threshold can be set globally for all vertex clusters, or locally for only one vertex cluster.

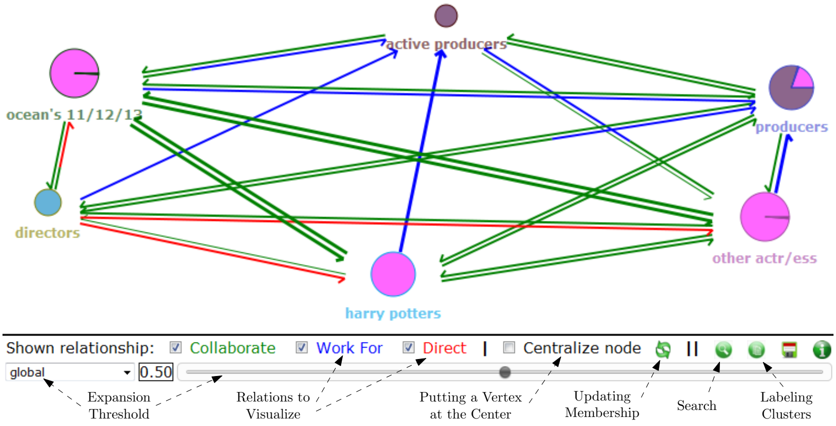
Structure Refinement. Structural summary automatically learnt from multi-dimensional networks may not always match user expectation. When a user disagrees with the way a structural summary summarizes the underlying network, she may want to refine the cluster membership distribution of a vertex. Such a refinement will require the structural summary to be revised based on user input. We therefore require the structure analyzer to be *semi-supervised.* Depending on how the structure analyzer clusters the vertices, the change in one vertex's membership distribution may cause change in others' membership distributions. As the structure analyzer refines the summary structure, the visualizer has to update the summary network accordingly with three kinds of interactions: i) change the membership distribution of one or more vertices; ii) create a new cluster, and indicate several members of the cluster; and iii) merge two existing clusters.

3 Case Study for Structure Analysis and Visualization

Based on our proposed framework, a working *ViStruclizer* has been developed with its visualizer and structure analyzer components implemented in Javascript and C++ respectively. *ViStruclizer* uses MySQL database system for backend storage. In this section, we demonstrate the capabilities of *ViStruclizer* using a multi-dimensional network extracted from IMDb.

3.1 The IMDb Network and Its Structure

We focus on the people involved in the movie industry and their network. There are more than 4 million of them and their network is very sparse. In our case study, we selected a denser subset of the network as follows. We started with a set of eleven directors, James Cameron, Chris Columbus, Jon Favreau, Ron Howard, Doug Liman, Christopher Nolan, Guy Ritchie, Martin Scorsese, Steven Soderbergh, Steven Spielberg and David Yates. These directors directed 73 movies from year 2000 to year 2010. We then further expanded to 486 people, including directors, producers and actors/actresses, who are involved in at least



Shown relationship: ☑ Collaborate ☑ Work For ☑ Direct | □ Centralize node

global 0.50

Fig. 3. The Structure of IMDb Network by *ViStruclizer*

two of the 73 movies. There are 3 modes for people in this network, namely, `actors/actresses`, `directors` and `producers`.

We also established three relations among the 486 people, i.e., `collaborate`, `direct` and `work_for`. For each movie, relation `collaborate` is observed among actors/actress, among directors, or among producers; relation `direct` is from directors to their directed actors/actresses; and relation `work_for` is observed from both actors/actresses and directors to producers. Therefore, this IMDb network is a multi-dimensional network with three modes and three relations.

We then incorporate the structure analyzer GSBM into *ViStruclizer* to visualize the extracted IMDb network. Empirically, we set the number of vertex clusters in the resultant structural summary to be $k = 6$. Besides grouping vertices into six vertex clusters, GSBM also returns edge clusters membership distributions, which is the optional element mentioned in Section 2. In GSBM, the probability of observing $r(E_{i,j})$ in edge cluster $B_{u,v}$ is given by

$$p(r(E_{i,j})|B_{u,v}) = \prod_{s=1}^{h} \text{Pois}(r_s(E_{i,j})|B_{u,v,s})$$

where $B_{u,v,s}$ is the Poisson parameter of edge cluster $B_{u,v}$ on relation R_s. Therefore, the probability of $r(E_{i,j})$ being modeled by $B_{u,v}$ is

$$\rho_{i,j,u,v} = p(u,v|r(E_{i,j}),B) \propto p(u,v,r(E_{i,j})|B) = \pi_{i,u}\pi_{j,v}p(r(E_{i,j})|B_{u,v})$$

Hence, the edge cluster membership distribution for edge $E_{i,j}$ is determined.

Again, this component is optional, and not every structure analyzer provides this functionality. For those structure analyzers which do not have this component, *ViStruclizer* simply takes $\rho_{i,j,u,v} = \pi_{i,u}\pi_{j,v}$ with the assumption that the probabilities of observing any particular edge in all edge clusters are the same.

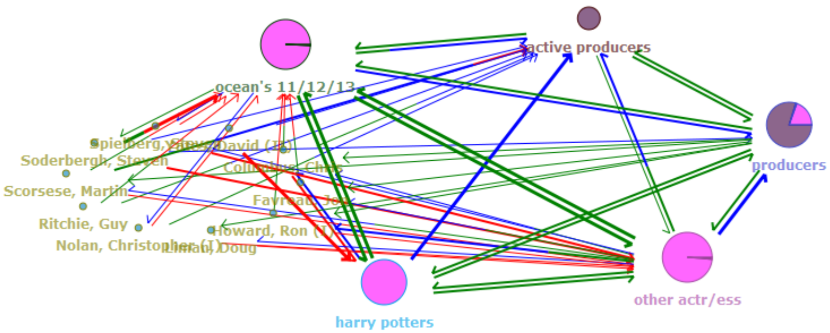


Fig. 4. The Structure of IMDb Network with `director` Cluster Expanded

3.2 Visual Representations

The summarized network structure of our IMDb network is shown in Figure 3. Each vertex cluster is represented by a pie chart with a user-assigned cluster label, size and pie segments determined by the weight and modal weights of the vertex cluster respectively (as computed using Equation 1 and 2). Different colors are assigned to different vertex modes, i.e., magenta for `actors/actresses`, royal blue for mode `directors` and purple for `producers`.

There are two producer clusters, the active producers who produce relatively more movies, and the other consisting of producers who may also act in the movies, as shown by the magenta sector in the pie chart of the cluster `producers` in Figure 3. Actors and actresses are divided into three clusters, i..e, "harry potters", "ocean's 11/12/13", and "other actr/ess". The first and second groups act in Harry Potter series and Ocean's series respectively, while the third group includes the others. The royal blue pie chart represents the `directors` cluster.

Between two vertex clusters are edges representing edge clusters. The width of the edge represents the weight of the corresponding edge cluster (the self-loops on each vertex cluster are omitted), as computed by Equation 3. Three colors are also used for three relations, green for `collaborate`, blue for `work_for` and red for `direct`. The proportions of the three colors tell how the two vertex clusters are related.

3.3 User Interactions in the IMDb Network

The visual representation of the original network and summary network in *ViStruclizer* is meant to be interactive. Using the control panel at the bottom of screen (see Figure 3), one can control the expansion threshold values to be used, select the relations to be included in the visualization, search vertices, and perform other operations on the network.

Figure 4 shows how *ViStruclizer* looks like after the "directors" cluster gets expanded. An edge between the expanded vertex cluster and another vertex

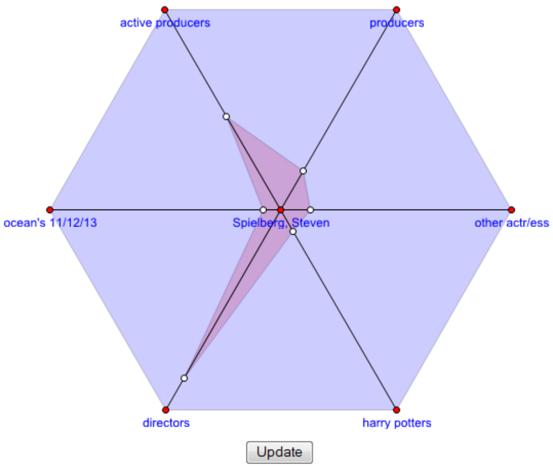


Fig. 5. Updating of the Membership of a Vertex

cluster is broken down into edges between vertices and the vertex cluster. Similarly, the edge between a vertex and a vertex cluster is aggregated from edges linking this particular vertex and all vertices in the vertex cluster. By adjusting the expansion threshold, one can choose to only expand the vertices assigned to clusters with high membership probabilities. *ViStruclizer* also allows any selected vertex to be placed at the center of the screen to use it as the focus.

One can change the membership probabilities $\pi_{i,u}$ of a selected vertex V_i by invoking a membership refinement function. As shown in Figure 5, the membership probabilities of the selected vertex in different vertex clusters can be displayed in a radar chart. Each radial axis of [0,1] value range corresponds to a vertex cluster and one can choose any point along the axis. The membership probabilities $\pi_{i,u}$ will then be recomputed so that the sum remains 1, and the visualization of the vertex clusters and the edge clusters will altered accordingly.

4 Conclusion and Future Work

This paper presents *ViStruclizer*, a network visual analytics system designed and implemented based on a framework for visualizing multi-dimensional networks using their summary network structures. Multi-dimensional network models are new in social network analysis and there have not been many visualization techniques specially designed for them. *ViStruclizer* represents one of these pioneering efforts. With the incorporation of a structure analyzer, which performs positional and role analysis, *ViStruclizer* effectively allows users to explore a multi-dimensional network along with its summary network. Its visualization capabilities on an IMDb network have also been demonstrated. To carry this work further, we plan to improve the structure analyzer and visualizer components. In particular, other network models for multi-dimensional networks and efficient learning of these models will be studied. For example, if we regard a specific topic as a type of relations, the Twitter[3] network can be visualized by different topics between clusters of users [6]. The visualizer component can also be improved by introducing new visual constructs that help user to identify interesting communities and anomalies in the networks.

References

1. Airoldi, E.M., Blei, D.M., Fienberg, S.E., Xing, E.P.: Mixed membership stochastic blockmodels. Journal of Machine Learning Research 9, 1981–2014 (2008)
2. Batagelj, V., Mrvar, A.: Pajek - Analysis and Visualization of Large Networks. In: Mutzel, P., Jünger, M., Leipert, S. (eds.) GD 2001. LNCS, vol. 2265, pp. 477–478. Springer, Heidelberg (2002)
3. Cao, N., Sun, J., Lin, Y.R., Gotz, D., Liu, S., Qu, H.: Facetatlas: Multifaceted visualization for rich text corpora. IEEE Transactions on Visualization and Computer Graphics 16(6), 1172–1181 (2010)
4. Contractor, N.S.: The emergence of multidimensional networks. Journal of Computer-Mediated Communication 14(3), 743–747 (2009)
5. Dai, B.T., Chua, F.C.T., Lim, E.P.: Structural analysis in multi-relational social networks. In: SDM, pp. 451–462 (2012)
6. Dai, B.T., Lim, E.P., Prasetyo, P.K.: Topic discovery from tweet replies. In: MLG: The Workshop on Mining and Learning with Graphs (2012)
7. Heer, J., Boyd, D.: Vizster: Visualizing online social networks. In: INFOVIS, p. 5 (2005)
8. Henry, N., Fekete, J.D.: Matrixexplorer: a dual-representation system to explore social networks. IEEE Transactions on Visualization and Computer Graphics 12(5), 677–684 (2006)
9. McLachlan, G.J., Krishnan, T.: The EM Algorithm and Extensions. Wiley Series in probability and Statistics. Wiley (2008)
10. Newman, M.E.J.: Fast algorithm for detecting community structure in networks. Physical Review E 69(6), 066133 (2004)
11. Shannon, R., Quigley, A.J., Nixon, P.: Graphemes: self-organizing shape-based clustered structures for network visualisations. In: CHI Extended Abstracts, pp. 4195–4200 (2010)
12. Shen, Z., Ma, K.L., Eliassi-Rad, T.: Visual analysis of large heterogeneous social networks by semantic and structural abstraction. IEEE Transactions on Visualization and Computer Graphics 12(6), 1427–1439 (2006)
13. Shi, L., Cao, N., Liu, S., Qian, W., Tan, L., Wang, G., Sun, J., Lin, C.Y.: Himap: Adaptive visualization of large-scale online social networks. In: PacificVis, pp. 41–48 (2009)
14. Wasserman, S., Faust, K.: Social Network Analysis: Methods and Applications. Cambridge University Press (1994)
15. White, H.C., Boorman, S.A., Breiger, R.L.: Social structure from multiple networks. i. blockmodels of roles and positions. The American Journal of Sociology 81(4), 730–780 (1976)
16. Zinsmaier, M., Brandes, U., Deussen, O., Strobelt, H.: Interactive level-of-detail rendering of large graphs. IEEE Transactions on Visualization and Computer Graphics 18(12), 2486–2495 (2012)

[3] `twitter.com`

Influential Nodes in a One-Wave Diffusion Model for Location-Based Social Networks*

Hao-Hsiang Wu[1,2] and Mi-Yen Yeh[2]

[1] Graduate Institute of Networking and Multimedia,
National Taiwan University, Taipei, Taiwan
[2] Institute of Information Science, Academia Sinica, Taipei, Taiwan
{haohsiangwu,miyen}@iis.sinica.edu.tw

Abstract. Taking the Foursquare data as an example, this paper investigates the problem of finding influential nodes in a location-based social network (LBSN). In Foursquare, people can share the location they visited and their opinions to others via the actions of checking in and writing tips. These check-ins and tips are likely to influence others on visiting the same places. To study the influence behavior in LBSNs, we first propose the attractiveness model to compute the influence probability among users. Then, we design a one-wave diffusion model, where we focus on the direct impact of the initially selected individuals on their first degree neighbors. Base on these two models, we propose algorithms to select the k influential nodes that maximize the influence spread in the complete-graph network and the network where only the links with friendship are preserved. We empirically show that the k influential nodes selected by our proposed methods have higher influence spread when compared to other methods.

1 Introduction

Due to the advances in wireless communication and positioning technologies, people can surf the Internet and share their locations through mobile devices almost anytime, anywhere. This fosters the emergence of location-based social networks (LBSNs), where people can interact with each other while sharing their location information. Example applications include Foursquare[1] and Gowalla[2]. The main difference between an LBSN and a general social network is that the former introduces a new dimension of physical location that brings social networks to reality and bridges the gap between the physical world and online social networking services [1]. In an LBSN, the act of users sharing their current locations is called *check-in*. By a check-in action at certain locations, people can

* The research in this paper was supported in part by the National Science Council of Taiwan, R.O.C., under Contracts NSC100-2221-E-001-023 and NSC101-2221-E-001-013. All opinions, findings, conclusions and recommendations in this paper are those of the authors and do not necessarily reflect the views of the funding agency.

[1] https://foursquare.com

[2] http://blog.gowalla.com

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 61–72, 2013.

also associate it with other additional information such as their comments about the place, the visiting time and their companions.

Prior studies on an LBSN usually focus on the human movement behavior analysis by mining their trajectories of visited locations, such as user movement prediction [2,3] and travel recommendations [4,5]. As an LBSN is also a kind of social networks that is currently a popular medium for people to share location information, we are interested in how people influence each other on the check-in behavior, how people will be attracted by other's comments on the shared locations and who are potentially influential in an LBSN to spread the location information. The answers of these problems are important to the location-based advertising applications because they can help enlarge the visibility and adoption of the products they promote. To the best of our knowledge, none of the existing works discuss finding influential nodes in an LBSN.

In this paper, taking the Foursquare data as an example, we propose to find the influential nodes in an LBSN. Foursquare is an LBSN application that provides a platform for users to share with friends or the public about their locations, which is called *venues*, by doing the check-in action through any GPS-equipped handhold devices. In addition, users can write comments, which is called *tips*, for each venue. By viewing the tips of others, a user is possibly attracted by some of them. Moreover, each user can add their interested tips to his/her *todo* list, and mark them as *done* if they did visit the corresponding venues. This information is useful for us to inference the potential influential users of the entire network especially when we do not have other explicit information of the influence behavior due to the privacy issues of Foursquare. Now, our challenges of finding influential nodes in an LBSN become the following two: How to leverage the available information to inference the influence probability between users? How to find the influential nodes under a suitable information diffusion model?

We begin with the modeling of the influence probability among users. To be more specific, we compute the likelihood that user u_i is attracted by user u_j's tips according to the proposed *attractiveness model*, which is based on the popularity of the mutual venues they have been visited and the popularity of tips written by u_j. In addition, we design the *one-wave diffusion model*, where we focus on the direct impact of the initially selected individuals on their first degree neighbors. With the attractiveness model and the one-wave diffusion model, we further design algorithms to select k influential nodes that maximize the influence spread in the complete-graph network, where a link weighted by an influential probability is built between every pair of nodes. In addition, to scale down the search space, we also consider to find influential nodes in the friendship network, where only the link between two nodes having friendship is left, and compare the results to the nodes found in the complete-graph network. By collecting the historical tip data of Foursquare, we evaluate the effectiveness of proposed models and algorithms. We report our findings on the influential nodes found in the tip data of New York and Los Angeles. The results show that the influence spread of the k nodes found in the friendship network is very close to the spread of those found in the complete-graph network under our attractiveness model and one-wave diffusion model.

The remainder of the paper is organized as follows. We discuss the related work in Section 2. In Section 3, we introduce the attractiveness model to compute the influence probability between users. Section 4 presents the one-wave diffusion model and our proposed algorithms to find the k influential nodes in LBSNs. We report the experiment results in Section 5. Section 6 concludes the paper.

2 Related Work

The influence maximization is to find a set of k nodes that maximize the information spread in a social network under some information diffusion model. Domingos and Richardson [6] were the first to study the influence maximization problem to analyze the value of customers in business. Kempe et al. [7,8] formulated the problem as an optimization problem and proved it is NP-hard under the linear threshold model and the independent cascade model. Prior studies about finding influential nodes focused on general social networks without considering the features of location-based social networks. To the best of our knowledge, we are the first to find influential nodes in LBSNs.

The linear threshold (LT) model and the independent cascade (IC) model are two generally studied information diffusion models. Kempe et al. [8] gave a comprehensive concept of these two models. Essentially, a node can be active or inactive. An inactive node may be influenced by any of its active neighbor according to a weight between them. The diffusion process starts with a set of active nodes while all other nodes are inactive. At each step, an active node remains active and the inactive one can become active only when the the total weight of its active neighbor exceeds a pre-selected threshold between 0 and 1. On the other hand, the independent cascade model works as follows. At each step of the diffusion process, only the newly active node has a chance to influence each of its inactive neighbors with a diffusion probability. When the inactive node is influenced successfully, it becomes active in the next step. Once an active node has tried to influence its neighbors in some step, it can never influence others in the following steps.

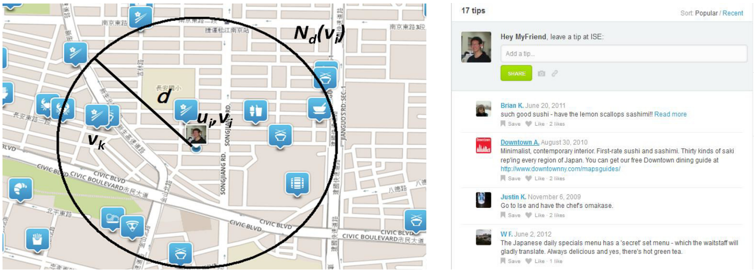
Note that both the LT and IC models consider multiple waves of influence propagation from a node to the entire network. In contrast, our proposed diffusion model only considers one wave of the influence between any initially selected node to its neighbors. In other words, the diffusion process of our proposed diffusion model involves only one step. It is because we only care the direct influence of a node to others but not the second-hand influence.

3 Modeling Attractiveness between Users

3.1 User Scenario of Foursquare

Foursquare is an location-based social networking application that provides a platform for users to share their locations, by doing the *check-in* action, with friends through any GPS-equipped handheld devices. A location is called *venue* in Foursquare. By locating the current position of a user as the center, the Foursquare application will provide the venues that fall in the neighborhood within some radius d, as shown in Fig. 1(a). All these venues are contributed by the Foursquare users and verified by the Foursquare administrators. Each venue on the map is marked by an icon showing its category. There are nine main types of categories: "Arts & Entertainment", "College & University", "Food", "Professional & Other Places", "Nightlife Locations", "Residences", "Great Outdoors", "Shops & Services" and "Travel & Transport".

In addition to check in at some venue, users can also write *tips*, namely the review comments, about the venue. Due to the interface design of Foursquare, when a user writes a tip of a certain venue, all other tips of the same venue will be listed out at the same page as shown in Fig. 1(b). Therefore, we can assume that the user will definitely be attracted by some of them. Note that these tips may come either from the friends or non-friends of the user. As the Foursquare application provides the venues within the neighborhood of radius d, the tips of these venues will also have chances to be seen by the user. In other words, any user may be attracted by the tips of venues in the neighborhood of radius d centered at his/her current location. Finally, when viewing the tips left by others, a user can add any interesting tips into his/her *todo* list. The user can further mark each todo tip to a *done* status if he/she completes the visit to the corresponding venue.



17 tips

Sort: Popular / Recent

Hey MyFriend, leave a tip at ISE:

Add a tip...

SHARE

Brian K. June 20, 2011
such good sushi - have the lemon scallops sashimi!! Read more
Save Like · 2 likes

Downtown A. August 30, 2010
Minimalist, contemporary interior. First-rate sushi and sashimi. Thirty kinds of saki rep'ing every region of Japan. You can get our free Downtown dining guide at http://www.downtownny.com/mapsguides/
Save Like · 2 likes

Justin K. November 6, 2009
Go to Ise and have the chef's omakase.
Save Like · 2 likes

W F. June 2, 2012
The Japanese daily specials menu has a 'secret' set menu - which the waitstaff will gladly translate. Always delicious and yes, there's hot green tea.
Save Like · 1 like

(a) Venues of the same category in the d-neighborhood of v_i

(b) The tip list of some venue v_k

Fig. 1. User scenario of Foursquare

3.2 The Attractiveness Model

By collecting the Foursquare data, we would like to study the influence relationship among the users and build a model to study the influential power among users in the location-based social network. In our attractiveness model, an LBSN is modeled as a graph $G = (V, E)$, where V denotes the set of nodes representing users, E is the link set representing the weighted connections between any two users. To be more specific, given two users u_i and u_j, the weight $w_{i,j}$ of the link between them is the likelihood that u_i's behavior on visiting some venues will be attracted by u_j's activities in Foursquare. In other words, $w_{i,j}$ is the influence probability of user u_j to user u_i. In the following, we show how to compute the $w_{i,j}$ value.

First, we introduce a tip and its attributes we can collect from Foursquare.

Definition 1 (A tip and its attributes). *A tip, denoted as s, has the following attributes: the user who writes this tip s.u, the category s.c, the recorded time s.t, the corresponding venue s.v, and the sum of the number of todos and the number of dones s.tdsum.* ■

As the act of adding a tip written by others into the todo list and marking it as done can be regarded as a positive feedback of a user to that tip [10], we can use this information to model how likely the later tips are attracted by the earlier ones. Due to the privacy settings of Foursquare, we cannot access the todo/done list of each user. Therefore, we cannot inference the attractiveness between users directly. However, for each tip, we can know its total number of todos and dones added and marked by different users. If a tip has a large number of todos and dones, it means that it is focused by a lot of people. Implicitly, it also shows that the user who writes this tip has some influential power.

According to the user scenario introduced in Section 3.1, a user u_i, who visits venue v_i and writes a tip for it at some time t_i, may have a chance to see other tips existing before t_i and be attracted by some of them for venues of the same category within the neighborhood of radius d centered at v_i. We denote all the venues in the d-neighborhood of v_i and having the same category of v_i as $N_d(v_i)$. Suppose g_k represents the probability that user u_i reads the tip list of venue $v_k \in N_d(v_i)$, as shown in Fig. 1(b). Among all the tips for v_k, some of them existing before t may be written by user u_j. As a result, user u_i may have a probability p_{jk} to be attracted by these tips. Then, we can compute the attractiveness of u_j to u_i, i.e., u_i is attracted by at least one tip written by u_j, for u_i to write a tip at v_i as follows.

$$P(u_i \rightsquigarrow u_j, N_d(v_i)) = 1 - \prod (g_k * (1 - p_{jk})). \tag{1}$$

Note that the above equation is under the convenient assumption that p_{jk} and g_k are independent for different v_k and different user u_j.

To compute g_k in Eq.(1), our intuition is that if a venue is hot and has a high chance to be viewed by a user, it may have a lot of tips, many of which certainly have a high number of todos and dones. As a result, we compute g_k as follows.

$$g_k|_{v_k \in N_d(v_i)} = \frac{\sum_{s \in S(v_k)} s.tdsum}{\sum_{s \in S(N_d(v_i))} s.tdsum}, \tag{2}$$

where $S(v_k)$ refers to all the tips for v_k that are written before u_i writes a tip s_i for v_i at time t_i and $S(N_d(v_i))$ refers to all the tips written before t_i for all venues in $N_d(v_i)$. Note that in case there are tips with zero sum of todos and dones, we can add one to *s.tdsum* of every tip in advance.

Next, we compute p_{jk} in Eq.(1) as follows.

$$p_{jk} = \frac{\sum_{s \in S_j(v_k)} s.tdsum}{\sum_{s \in S(v_k)} s.tdsum}, \tag{3}$$

where $S_j(v_k)$ refers to tips in $S(v_k)$ that are written by u_j. The intuition of this equation is that if a user u_j has high attractiveness, not only the number of tips that u_j writes is high but also the sum of todo and done numbers of these tips are high.

Finally, the value $w_{i,j}$, which is the influence probability of u_j to u_i, is computed as the expected value of $P(u_i \rightsquigarrow u_j, N_d(v_i))$ defined in Eq.(1).

$$w_{i,j} = \sum_{s \in S_i} P(u_i \rightsquigarrow u_j, N_d(v_i)), \tag{4}$$

where S_i are tips written by user u_i in our collected data set.

It is noted that if $w_{i,j} > w_{i,k}$, then user u_j is more attractive to user u_i compared to user u_k. In addition, when collecting the tip data of Foursquare, we can also know if two users are friends. Foursquare regularly recommends users for the tips of their friends. Also, users can actively search the tips written by their friends for venues adjacent to their interested places. Therefore, if $P(u_i \rightsquigarrow u_j, N_d(v_i)) > 0$ and u_i and u_j are friends in Foursquare, we set $P(u_i \rightsquigarrow u_j, N_d(v_i)) = 1$. This is to model that tips written by user u_i must be attracted by those written earlier by u_i's friend u_j.

4 Finding Influential Nodes

After introducing the attractiveness model, in this section, we propose *one-wave diffusion model*, which is used to model the tip information diffusion, followed by the algorithm of finding the k influential nodes that maximize the influence spread in an LBSN.

4.1 One-Wave Diffusion Model

In our attractiveness model, we show how to compute $w_{i,j}$, which is the attractiveness, or the influence power of u_j to u_i and defined in Eq.(4). Before running the diffusion process, we do normalization for these values and obtain the diffusion probability from u_j to u_i as follows.

$$q_{i,j} = \frac{w_{i,j}}{w_{max} + w_{min}}, \tag{5}$$

where w_{max} and w_{min} are the maximum and the minimum $w_{i,j}$ of the whole network.

Algorithm 1 BaseLine (G, k)

```
Input: $G = (V, E)$, number of seeds $k$
Output: A set $R$ of $k$ influential nodes
1: $R = \emptyset$
2: for $i = 1$ to $k$ do
3:     for each node $v \in V \backslash R$ do
4:         $IS_v = InfluenceSpread(v)$ /* A node will not be considered to be influenced
           by $v$ if that node is already influenced by nodes in $R$.*/
5:     $R = R \cup argmax_v\{|IS_v|, v \in V \backslash R\}$
6: return $R$
```

The diffusion process of the one-wave diffusion model works as follows. Given a start node, say v_j, it influences each of its neighbors v_i if $q_{i,j} > 0$. The total number of the influenced nodes is regarded as the influence spread of v_j. Please note that, for k seed nodes, we run the above process sequentially. Also, once a node is influenced by some seed, it cannot be further influenced by other seeds. Finally, instead of using the well-know IC or LT models in this study, our intuition of adopting the one-wave diffusion model is that we care only the influence spread of the initially selected seeds, but not that of the active nodes influenced by the seeds.

4.2 Algorithms for Influence Maximization

Given the one-wave diffusion model, in this section we show how to select k influential nodes that maximize the influential spread. Kempe et al. [8] has described a greedy algorithm to solve the k-seed selection problem, which we modify based on our one-wave diffusion model as shown in Algorithm 1.

In Algorithm 1, the $InfluenceSpread$ function computes the influence spread of v according to the one-wave diffusion process and put the v with the largest influence spread into R as the solution. If there is a tie at line 5, the node with the smaller ID number wins. Because this is an exhausted search on a complete graph G (i.e., $O(|V|^2)$ links), the BaseLine approach is regarded as the benchmark but lacks efficiency. Instead, we seek if we can search influential nodes in a smaller space.

One method is called GreedyAlgorithmOnFriends(GAOF) as shown in Algorithm 2. In GAOF, we search the influential nodes on a graph where only the friendship links are retained. To be more specific, we collect the friendship between each pair of users from the Foursquare data, and remove the edge between two nodes who do not have friendship from G to generate $G_f = \{V_f, E_f\}$. First we compute the influence spread of each node, denoted as IS_v, in G_f(Lines 2-3). Finally, we extract the k nodes as the result of GAOF(Lines 4-5). If there are tie at line 5, the node with the smaller ID number wins.

Algorithm 2 GAOF (G_f, k)

```
Input: $G_f = (V_f, E_f)$, number of seeds $k$
Output: A set $R_{GAOF}$ of $k$ influential nodes
 1: $R_{GAOF} = \emptyset$
 2: for each node $v \in V_f$ do
 3:     $IS_v = InfluenceSpread(v)$ /* each node $v$ has its own $IS_v$ */
 4: for $i = 1$ to $k$ do
 5:     $R_{GAOF} = R_{GAOF} \cup argmax_v\{|IS_v|, v \in V_f \backslash R_{GAOF}\}$
 6: return $R_{GAOF}$
```

Algorithm 3 CGAOF (G_f, k)

```
Input: $G_f = (V_f, E_f)$, number of seeds $k$
Output: A set $R_{CGAOF}$ of $k$ influential nodes
 1: $R_{CGAOF} = \emptyset$
 2: $R = \emptyset$
 3: for each node $v \in V_f$ do
 4:     $IS_v = InfluenceSpread(v)$ /* each node $v$ has its own $IS_v$ */
 5:     compute $a_v$ of node $v$
 6: for $i = 1$ to $2k$ do
 7:     $R = R \cup argmax_v\{|IS_v|, v \in V_f \backslash R\}$
 8: for $i = 1$ to $k$ do
 9:     $R_{CGAOF} = R_{CGAOF} \cup argmax_v\{a_v, v \in argmax_v\{|S_v|, v \in R \backslash R_{CGAOF}\}\}$
10: return $R_{CGAOF}$
```

An further improvement of GAOF is called ClassifyingGreedyAlgorithmOnFriends(CGAOF) as shown in Algorithm 3. Similar to GAOF, we compute the influence spread of each node in G_f(Lines 3-4). However, in contrast to GAOF extracting k nodes by using the influence spread only, CGAOF extracts k nodes by considering three extra criteria: $|IS_v|$, $|S_v|$ and a_v, which denotes the number of influence spread of v, the number of tips written by v, and

$$a_v = \frac{\sum_{i \in V_f}\{w_{i,v} \mid \text{node } i \text{ is influenced by node } v, i \neq v\}}{|IS_v|}, \tag{6}$$

respectively. We set $a_v = 0$ when $|IS_v| = 0$. If a node v has a high a_v value, it means the attractiveness of this user to his/her friends is higher. In CGAOF, we extract the nodes by comparing $|IS_v|$ and $|S_v|$ (Lines 8-9). If there is a tie at line 7 and 9, the node with the smaller ID number wins. The intuition of this algorithm is that we think high $|S_v|$ and high a_v are also important features to compare the influence power of two nodes having only a small difference in influence spread. The node with high $|S_v|$ will have more chances to influence other nodes while the node of a_v implies this user usually writes more attractive tips.

4.3 Complexity Analysis

In this section, we analyze the complexity of each algorithm. The complexity of the one-wave diffusion process of a node v, that is $InfluenceSpread(v)$, is $O(D_v)$, where D_v denotes the degree of v in G. Thus the complexity of BaseLine is $O(k|V|D_v)$, where it extracts k influential nodes and runs $|V|$ times of the diffusion process for k nodes and selects the node with the largest influence spread. The complexity of getting R_{GAOF} from Algorithm 2 is $O(|V|D_{vf}+k|V|)$, where D_{vf} denotes the degree of some node v in G_f, and the algorithm extracts k nodes with the highest influence spread in $k|V|$ time (Algorithm 2: Line 4, 5). The complexity of getting R_{CGAOF} from Algorithm 3 is $O(|V|D_{vf} + k|V| + k^3)$, where it selects $2k$ nodes with the highest influence spread in $O(k|V|)$ and selects the highest $|S_v|$ from $2k$ nodes, then selects the highest a_v nodes from these the highest $|S_v|$ nodes, and continuously runs the selecting process k times (Algorithm 3: Line 8, 9). Because G_f only reserves the edges between two nodes that have friendship from G, $D_{vf} \ll D_v$. As a result, the time cost of GAOF and CGAOF is much smaller compared to BaseLine when $|V| \gg k$.

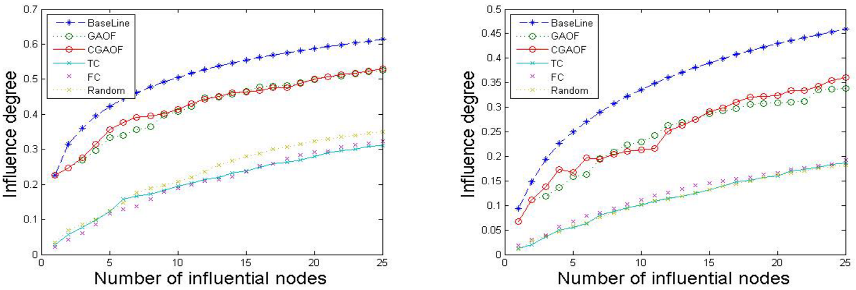
5 Performance Study

We conducted experiments on two real data sets collected from Foursquare to evaluate the effectiveness of our proposed models and algorithm. All experiments were run on a workstation with an Intel Xeon E5530 2.40 GHz CPU and 70GB RAM using C.

5.1 Settings

By crawling the Fousquare data, we obtained 47218 users who wrote tips for the venues in New York City (NYC) and 30196 users who wrote tips for the venues in Los Angeles City (LAC). There were about 410,000 tips from 2008/5 to 2011/7 in the NYC dataset and about 260,000 tips from 2009/2 to 2011/7 in the LAC dataset. The d used for $N_d(v_i)$ was set to 1 KM.

We compared the proposed three methods, BaseLine, GAOF, CGAOF, with three other methods: FriendCentrality (FC), TipsCentrality (TC), and Random on selecting the k seeds for influence maximization. We compute the influence degree, the number of influenced nodes divided by the total number of nodes in a network, of the k seeds selected by different methods. The FC and TC methods always selected k nodes having the largest number of friends and the largest number of tips, respectively, as the seeds. For the Random method, it just randomly selected k nodes as seeds. We run FC, TC, and Random several times to get their average influence degree. Please note that BaseLine, TC, FC and Random methods selected seeds from the complete network G. Only GAOF and CGAOF select k seeds from the network with only friendship links existed, i.e., G_f. In addition, for all methods, only users writing at least 50 tips were considered as the candidate of seeds. There were 754 candidates nodes in LAC and 1137 in NYC. This was to reduce the search space and speed up the initial seed selection in the two large networks.



(a) NYC (b) LAC

Fig. 2. The influence degree of each algorithm

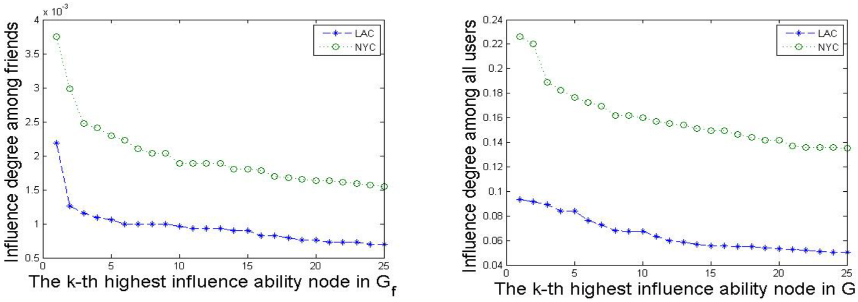
5.2 Degree of Influence Maximization

Fig. 2 shows the influence degree of the 25 seeds chosen by different methods. Note that although the five approaches selected different sets of k seeds from either G or G_f, the influence degrees here were examined for each set of k on G.

In general, the influence degree increased as the number of seeds increased on both data sets. The influence degrees of Baseline were the highest because the influential nodes were selected on the whole network G. The influence degrees of GAOF and CGAOF were lower compared to Baseline, but still of high enough values. On the NYC data set as shown in Fig. 2(a), the influence degree of the 25 nodes selected by CGAOF was high to 86.3% of that of Baseline, and that of GAOF was high to 85.5%. On the LAC data set as shown in Fig. 2(b), the influence degree of the 25 nodes selected by CGAOF was high to 78.5% of that of the BaseLine method and that of GAOF was high to 73.5%. TC, FC, and Random had poor performance on the influence degrees for both data sets showing that users who wrote the most tips or had the most friends were not necessary the most influential nodes. Instead, finding the influential nodes on G_f was a good alternative when the time cost was an primary issue. Finally, when k increased from 1 to 25, we found that CGAOF was better than GAOF in general because it considered both the number of tips written and the influential spread of a user simultaneously. Sometimes CGAOF was worse probably because G_f had some users writing many tips to attract their friends while these tips might not attract non-friends in G, where the influence degree was examined.

5.3 Independent Influence Spread among Friends and All Nodes

Here, we run the $InfluenceSpread(v)$ of each v independently on G using the Baseline approach and on G_f using GAOF. We selected the top 25 nodes of the highest influence degree values. Fig. 3(a) and 3(b) show the influence degrees of



(a) Influence among friends

(b) Influence among all users

Fig. 3. The influence ability of each node in LAC and NYC

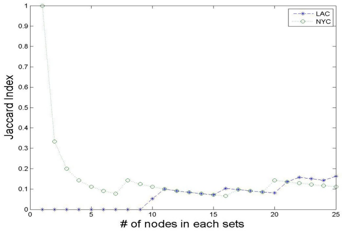


Fig. 4. Similarity between the set of the top k highest influence ability nodes in G and the set of the top k highest influence ability nodes in G_f

	LAC	NYC
# of candidate nodes	754	1137
Avg. # of tips of candidate nodes	102.2	97.9
Avg. degree per node in G	30195	47217
Avg. degree per node in G_f	28.7	48.9

Fig. 5. The statistics of G_f and G

the top 25 nodes selected from G_f and G, respectively, on both the NYC and LAC data sets. We denoted the set of 25 nodes on G_f as A and that on G as B and computed the Jaccard index $J(A, B) = \frac{|A \cap B|}{|A \cup B|}$. The higher the Jaccard index represented that the two sets were more similar to each other.

Fig. 4 shows the Jaccard indexes of A and B on both the NYC and LAC data sets. We observed there were overlaps between the two sets. Moreover, the top 1 influential node were even the same for the NYC data. We show the statistics of G_f and G in Fig. 5. It shows that although the average number of degree per node in G_f was much smaller than that of G, G_f was still useful to find the subset(about 20%-30% with k=25 in our experiments) of the top k highest influence ability nodes in G. As we mention in Section 4.3, $D_v \gg D_{vf}$ and $|V| \gg k$ in our experiments, so it was very useful to improve the efficiency. Due to our attractiveness model, if a node has friends rarely or never wrote tips, then the node could hardly or not influence its friends. Fig. 5 shows that the candidate nodes in NYC and LAC wrote the same number of tips, but NYC had a higher average number of D_{vf} compared to LAC. This means that the average number of friends of users in the LAC data was smaller compared to the NYC data such that the influence degree among friends in LAC was smaller. We also think that the influence degree may become a reason that if users in G have more friends writing tips in the same area, then it should be more efficient to use the G_f to find the influential nodes in G.

6 Conclusions

In this paper, taking the Foursquare data as an example, we studied the problem of finding influential nodes in location-based social networks. Based on the popularity of the mutual venues visited and the popularity of the tips written, we built the attractiveness model to compute the influence probability between two users. Furthermore, a one-wave diffusion model was designed to focus the direct impact of the initial seeds on their first degree neighbors. With these two models, we proposed algorithms to find the k influential nodes in LBSNs, on both a complete-graph network and a friendship network. Under our attractiveness and one-wave diffusion models, we empirically showed that the k influential nodes selected by our proposed methods in the the complete-graph and friendship networks had higher influence spread when compared to other methods.

References

1. Zheng, Y., Zhou, X.: Computing with Spatial Trajectories (2011)
2. Cho, E., Myers, S.A., Leskovec, J.: Friendship and Mobility: User Movement in Location-Based Social Network. In: Int. Conf. on KDD, pp. 1082–1090 (2011)
3. Noulas, A., Scellato, S., Mascolo, C., Pontil, M.: An Empirical Study of Geographic User Activity Patterns in Foursquare. In: Int. Conf. on ICWSM (2011)
4. Ye, M., Yin, P., Lee, W.C., Lee, D.L.: Exploiting Geographical Influence for Collaborative Point-of-Interest Recommendation. In: Int. Conf. on SIGIR, pp. 325–334 (2011)
5. Zheng, Y., Zhang, L., Xie, X., Ma, W.Y.: Mining interesting locations and travel sequences from GPS trajectories. In: Int. Conf. on WWW, pp. 791–800 (2009)
6. Domingos, P., Richardson, M.: Mining the Network Value of Customers. In: Int. Conf. on KDD, pp. 57–66 (2001)
7. Kempe, D., Kleinberg, J.M., Tardos, É.: Influential Nodes in a Diffusion Model for Social Networks. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 1127–1138. Springer, Heidelberg (2005)
8. Kempe, D., Kleinberg, J., Tardos, E.: Maximizing the Spread of Influence through a Social Network. In: Int. Conf. on KDD, pp. 137–146 (2003)
9. Granovetter, M.: Threshold Models of Collective Behavior. American Journal of Sociology 83, 1420–1443 (1978)
10. Vasconcelos, M.A., Ricci, S.M.R., Almeida, J.M., Benevenuto, F., Almeida, V.A.F.: Tips, Dones and Todos: Uncovering User Profiles in Foursquare. In: Int. Conf. on WSDM, pp. 653–662 (2012)

Modeling Social Information Learning among Taxi Drivers

Siyuan Liu[1], Ramayya Krishnan[1], Emma Brunskill[1], and Lionel M. Ni[2]

[1] Carnegie Mellon University
[2] Hong Kong University of Science and Technology

Abstract. When a taxi driver of an unoccupied taxi is seeking passengers on a road unknown to him or her in a large city, what should the driver do? Alternatives include cruising around the road or waiting for a time period at the roadside in the hopes of finding a passenger or just leaving for another road enroute to a destination he knows (e.g., hotel taxi rank)? This is an interesting problem that arises everyday in many cities worldwide. There could be different answers to the question poised above, but one fundamental problem is how the driver learns about the likelihood of finding passengers on a road that is new to him (as in he has not picked up or dropped off passengers there before). Our observation from large scale taxi drivers behavior data is that a driver not only learns from his own experience but through interactions with other drivers. In this paper, we first formally define this problem as **Socialized Information Learning (SIL)**, second we propose a framework including a series of models to study how a taxi driver gathers and learns information in an uncertain environment through the use of his social network. Finally, the large scale real life data and empirical experiments confirm that our models are much more effective, efficient and scalable that prior work on this problem.

1 Introduction

The study of how a person gathers information and makes decisions has a long and varied literature. In the previous research, collective intelligence [13, 19, 23, 27], intelligent agent [7, 10, 12, 21, 22], transfer learning [6, 25, 26] and evidence-based reasoning [4, 8, 14] and other methods are proposed to investigate an agent's learning theory. But due to the new challenges raised in dynamic uncertain environment [11, 17, 18], prior work on this topic is either inefficient or inaccurate. Now consider the following problem.

In our application context, there are 3,187 taxi drivers, and among them there are 25.2% new drivers (less than one year driving experience), 47.1% normal drivers (one to two years driving experience) and 27.7% experienced drivers (greater than two years driving experience). When a driver comes to road to pick up passengers, there are two actions (action set) to choose: *waiting* at the current location or *cruising* to other locations. The knowledge of the taxi driver can be described as the histogram of the waiting time before picking up a passenger, the number of picking-ups, and the revenue (income) at the given time and location. Given a situation that the drivers come to an unknown road, in a survey of 1,000 taxi drivers, we study how the drivers follow their own experienced knowledge and the socialized knowledge (the knowledge

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 73–84, 2013.

obtained from other drivers) to accordingly make actions. We find that different drivers have very different learning preferences. The new drivers prefer to follow the socialized knowledge, but the experienced drivers prefer to follow their experienced knowledge. The standard deviation is larger in new drivers than the experienced drivers. Hence, different drivers would take various knowledge sources.

In our dataset, we also have the communication records to indicate the socialization of taxi drivers. When a new driver was assigned to different group compositions, we studied the income changes (one week). We take 70% as the threshold to define a group composition, e.g., *New* means more than 70% of the drivers in the group are new drivers. We also tested other thresholds larger than 50%, and the results are very similar. It is very interesting that even the new driver has less social closeness to the *Experienced* group than the *New* group, but not only individual income but also total income in the *Experienced* group have the greater increases than the *New* group. Hence the more socializations may not give the more accurate knowledge and better income. For example, the experienced drivers could give the new driver more accurate knowledge than other drivers.

In a word, the problem is that how the drivers socialize with each other to construct the accurate knowledge in an uncertain environment, which we define as **Socialized Information Learning**. To tackle the problem, we have two steps to be accomplished, 1) how to retrieve an accurate knowledge set for an individual driver in an uncertain environment; 2) how to utilize a social network to learn the information. In this paper, we first propose an **Individual Information Model** to describe taxi drivers' information collection, considering the features of the required places and the similarity between the required places and the experienced places. Second, we introduce the social network structure with a probability weighting function into the model to describe the non-linear socialized information learning, called **Socialized Information Model**.

To summarize the contributions of our work, first, we are the first to discover the **Socialized Information Learning** problem in taxi drivers, and we define it as a new uncertain information learning problem; second, we propose a framework including a series of novel models to solve the socialized information learning problem (not only model taxi drivers' behaviors by themselves, but also their social behaviors via the group information) and investigate it in the dynamic field; third, we employ large scale real life datasets to test our models, and the empirical results show that our models outperform the state-of-the-art in terms of effectiveness, efficiency and scalability.

The paper is structured as follows.In Section 2, we formally define the socialized information learning problem. We propose a series of models to solve the new socialized information learning problem in Section 3, and the empirical experiment results are illustrated and analyzed in Section 4. The related work to our study is surveyed in Section 5. Finally, we conclude our work and give directions to the future work.

2 Problem Definition

2.1 Definitions

Definition 1. **(Agent)** *Agent is defined as an entity that is capable of perceiving knowledge and accordingly do action.*

In our work, an agent is a taxi driver.

Definition 2. (Agent group) *Agent group is defined as a set of agents that are capable of perceiving knowledge from each other and accordingly do action.*

In our work, an agent group is a predefined group of taxi drivers by a taxi company. We define the knowledge of a taxi driver as experienced knowledge and socialized knowledge as below.

Definition 3. (Experienced knowledge) *The experienced knowledge (EK) is defined as a set of information collected from the agent's own experience, that is, historical records.*

In our work, the experienced knowledge is a set of the CDFs of waiting-time, picking-up and revenue distributions at given locations and times, from a given taxis historical GPS logs and business records. For the road without a given taxi driver's experienced knowledge is called an unknown road.

Definition 4. (Socialized knowledge) *The socialized knowledge (SK) is defined as a set of information collected from other agents' information in the same group, that is, other agents' historical records.*

In our work, the socialized knowledge is a set of CDFs of waiting-time, picking-up and income distributions at given locations and times, from a given taxis group members GPS logs and business records at the same given locations and times.

Definition 5. (Action) *Action of an agent is defined as a selection of a mutual exclusion set.*

Action of a taxi driver is defined as cruising or waiting for a passenger. At a given location and time, a taxi driver can select an action (make a decision) of cruising to other locations until picking up a passenger or waiting for a time period at the given location until picking up a passenger.

Definition 6. (Socialization) *Socialization of an agent is defined as a communication between two agents.*

Socialization of a taxi driver is a call between two taxi drivers in the same group. Each socialization is recorded a vector: $(i, j, t_s, t_e, \iota_s, \iota_e)$, where i is the caller taxi ID, j is the callee taxi ID, t_s is the call start time, t_e is the call end time, ι_s is the calling start location (longitude and latitude) and ι_s is the calling end location.

Definition 7. (Socialization closeness) *Socialization closeness between two agent is defined as a function of communications between the two agents.*

The socialization closeness between two taxi drivers is defined as:

Definition 8. (Socialization closeness of taxi drivers) *Given two taxi driver i and j in a group, a time interval t, i has the communication attribute set $S_i^t = \{s_i^{t,1}, ..., s_i^{t,f}, ..., s_i^{t,m}\}$, where $1 \leq f \leq m$ and m is the number of attributes. The socialization closeness within the time interval t, $\gamma_{i,j}^t$ is*

$$\gamma_{i,j}^t = \frac{1}{m} \sum_{f=1}^{m} w_f s_{i,j}^{t,f} \tag{1}$$

where w_f^t is the weight of an attribute f in the time interval t. The socialization closeness set is $\Gamma^t = \{..., \gamma_{i,j}^t, ...\}$, where $i, j \in \mathbb{N}$, $i, j = 1, 2, ..., n$, and n is the number of taxi drivers in a given group.

In our study, we take the mean of three attributes (m=2), the number of calls and the call duration as socialization closeness values. Our solution can be extended to other cases where $m > 2$ and other functions. The default time interval is set as a minute, which is set by the communication service company. In this work, we equally take the weights in 8, which is predefined by users.

For the input communication data, we can construct a social network, $G = (V, E)$, where V is a set of nodes (taxi drivers), and E is a set of edges (socialization with socialization closeness as the weight on the edge).

Definition 9. (Decision knowledge) *The decision knowledge (DK) is defined as the information taken by an agent to make an action.*

For a taxi driver, the decision knowledge is based on a set of CDFs of waiting-time, picking-up and income distributions at given locations and times to make a certain action.

2.2 Socialized Information Learning

The formal definition of **Socialized Information Learning** (SIL) is as below.

Given: a set of agents Q, a set of experienced knowledge E, a set of socialized knowledge S, and a social network G with socialization closeness Γ.

Goal: a set of decision knowledge D.

Specifically, given a taxi driver with experienced knowledge and socialized knowledge, under a social network, to make a good action to pick up a passenger in an unknown road, what is decision knowledge to support the given taxi driver's action?

In our work, the decision knowledge utilized by a taxi driver to decide the next move is calculated by a decision function as below.

Definition 10. (Decision function of a Taxi Driver) *$P[v^t(\iota) \,|n^t(\iota) \geq \varpi] \geq \omega$, where ι is a location index, t is a time index, $v^t(\iota)$ is the revenue, $n^t(\iota)$ is the number of passengers, ϖ and ω are thresholds.*

The above function means if given a probability of a certain number of passenger is greater than a given threshold (ϖ), the conditional probability of revenue is greater than ω, the driver is going to wait for passengers at the given location, otherwise, cruise to other locations.

3 Socialized Information Learning Framework

3.1 Individual Information Model

The basic idea of **Individual Information Model** is as follows. First, based on the utilities of the road and buildings, we label each grid and cluster the grids into different clusters. Second, given a location, we evaluate the similarity between the given location and the taxi's experienced locations. Third, we weight the similar experienced knowledge and the socialized knowledge at the given location and time, and finally get the decision knowledge. To make the following expression clear, we take the revenue as a knowledge example to illustrate the model.

Definition 11. *Given a physical location ι =(x,y) and the report set Φ, the revenue spectrum $V_{\Phi}^{(t)}(\iota)$ is the set of all the reported revenues (of the given taxi) at time t in location ι in Φ, i.e.,*

$$V_{\Phi}^{(t)}(\iota) = \{v|\exists\phi_m^{(t)} \in \Phi, (x_m^{(t)}, y_m^{(t)}, v_m^{(t)}) = (x, y, v)\}$$

The revenue spectrum at all time instances is also written as $V_{\Phi}(\iota) = \cup_{t\in[0,+\infty]} V_{\Phi}^{(t)}(\iota)$. Since the time is discrete, Φ contains a finite number of reports and thus the revenue spectrum is finite as well.

Definition 12. *The location revenue $v_{\Phi}^{(t)}(\iota)$ is defined as the average of the revenue spectrum in location ι at t,*

$$v_{\Phi}^{(t)}(\iota) = \frac{1}{|V_{\Phi}^{(t)}(\iota)|} \sum\nolimits_{v_m \in V_{\Phi}^{(t)}(\iota)} v_m \tag{2}$$

In the real data, the revue spectrum is very sparse and lossy, hence we employ a moving average technique to reconstruct a sufficient spectrum as below.

Definition 13. *The experienced revenue knowledge $\mathrm{H}^{(t)}(\iota)$ of a given location $\iota = (x, y)$ is defined as an exponential moving average of the complementary cumulative distribution function (CCDF) of the instant revenue over the revenue spectrum, i.e.,*

$$\mathrm{H}^{(t)}(\iota) = \alpha_{\iota} \cdot \mathrm{H}^{(t-\tau_{\iota})}(\iota) + (1 - \alpha_{\iota}) \cdot (1 - \mathrm{P}(v \leq v_{\Phi}^{(t)}(\iota))) \tag{3}$$

where $\mathrm{P}(v \leq v^{(t)}(\iota))$, $v \in V_{\Phi}(\iota)$ represents the probability that the revenue at ι is less than or equal to the location instant revenue $v_{\Phi}^{(t)}(\iota)$. α_{ι} and τ_{ι} are two parameters to capture the dynamism of ι.

The parameter α_{ι} is a smoothing factor of exponential moving average in H. It is used to capture the degree of dynamism of the location dynamics. In general, a smaller α_{ι} indicates a higher dynamism and vice versa. The parameter τ_{ι} is the interval between two moving averages. It reflects the periodic property of the location dynamics.

Different locations will present distinctive dynamic behaviors, resulting in various settings. In order to systematically study the speed distribution, we apply Fourier Transformation (FT). FT can transform the function from time domain to frequency domain, revealing inherent periodic property of original function as well as the amplitude of the corresponding frequency. Specifically, given the revenue distribution function over time $v^{(t)}(\iota)$ at a location ι, its FT can be calculated by,

$$\hat{f}(\xi) = \int_{-\infty}^{+\infty} v^{(t)}(l) e^{-2\pi it\xi} dt, \tag{4}$$

where t is the variable.

To calculate the $\mathrm{H}^{(t)}(\iota)$, we maintain the location history information for six months. As the computation is carried out at a data center in a centralized manner, the computational and storage cost is acceptable.

Definition 14. *The similarity between the given location and the taxi's experienced locations is defined by the linear correlation coefficient.*

Given the similar experienced revenue knowledge $\mathrm{H}^{(t)}(\iota)$ and the socialized revenue knowledge $\mathrm{H}^{'(t)}(\iota)$ from other drivers, we have the decision revenue knowledge as below.

$$\hat{\mathrm{H}}^{(t)}(\iota) = \beta \mathrm{H}^{(t)}(\iota) + (1 - \beta)\mathrm{H}^{'(t)}(\iota), \tag{5}$$

where β is a parameter capturing the weight of a taxi driver following the own experienced revenue knowledge, which is determined by users (taxi drivers). In our work, we learned this parameter from the real life data, which is elaborated in Section 4. The method to retrieve the socialized revenue knowledge $\mathrm{H}^{'(t)}(\iota)$ is proposed in the next section.

3.2 Socialized Information Model

Given a taxi driver, there exists knowledge limitation (hard to possess the knowledge of the whole city), hence the taxi driver may consult other taxi drivers via a social network G. In this subsection, we propose a **Socialized Information Model** to learn knowledge from other drivers in a social network.

Definition 15. *(**Socialization probability**) The probability $p_{i,j}$ of a given taxi driver i socializing with a taxi driver j is the percentage of socialization closeness between the two drivers over the total socialization closeness among other drivers being socialized by the driver in a given time period.*

Hence the socialized knowledge for a given taxi driver can be described by a probability-based weighting function over a set of knowledge. Under the expected utility theory, a taxi driver weights probabilities of learning information linearly. However, the evidence suggests that the taxi drivers weight probabilities in a non-linear manner. An example is given as following.

$$\mathrm{H}_i^{'(t)}(\iota) = (\frac{1}{n} \cdot \sum_{k=1}^{n} [\mathrm{p}_{i,k}\mathrm{H}_k^{(t)}(\iota)]^m)^{\frac{1}{m}}, \tag{6}$$

where m is a parameter in generalized mean, determining the appropriate mean (in our work, we select as 2). $\mathrm{p}_{i,k}\mathrm{H}_k^{(t)}(\iota)$ in Eq. 6 means the socialized knowledge of driver i from driver j, under the probability of driver i being able to access the experienced knowledge of driver k.

Unfortunately, this method does not work well under the following two cases: 1) overweight small probabilities and underweight large ones (S1); 2) do not choose stochastically dominated options when such dominance is obvious (S2). Hence we utilize a probability weighting function to conduct a non-linear weighting of socialized knowledge.

The particular probability weighting function is

$$w(p) = \frac{p^{\lambda}}{\left[p^{\lambda} + (1-p)^{\lambda}\right]^{\frac{1}{\lambda}}}, \tag{7}$$

where $0.5 < \lambda \leq 1$, $w(p)$ is a weighted probability and p is $p_{i,j}$ in Eq. 6.

After weighting $p_{i,j}$, we utilize Eq. 6 to compute the socialized knowledge.

4 Empirical Experiments

4.1 Experiment Setup

Datasets Description: We collected one year taxi operation data records, including taxi tracking records, taximeter records and communication records, in a large city in China. The scale of the whole dataset is almost 1 Terabytes. Taxi tracking records provide taxis' group information and traces, including location and time information; taximeter records provide taxis' revenue, waiting time and picking-up with location-time logs; communication records provide taxis' social information, e.g., social closeness. We employ six months data as training data and the other six months data as test data. We also collected one month traffic surveillance video data in the city. The traffic data can provide us the traffic flow, traffic lights, taxis' picking-up and dropping-off information, which we employ as the ground truth of taxis' behaviors in the city.

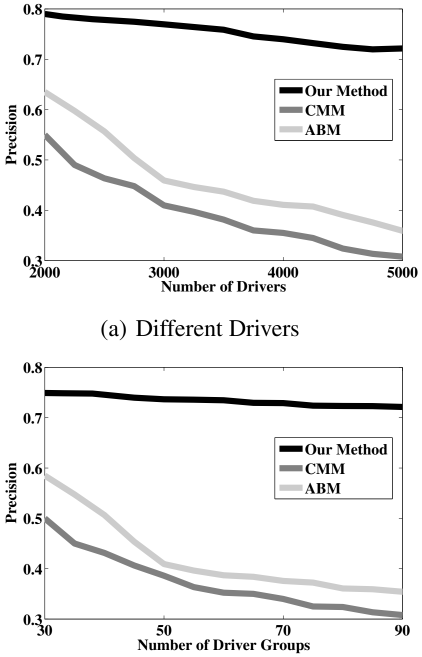
Experiment Environment: A server with four Intel Core Quad CPUs (Q9550 2.83 GHz) and 32 GB main memory.

Baseline Methods: We compare our method with two baseline methods: one is the classic method in collective intelligence, called CMM [13], which is popular and generate many latest approaches; the other is the representative method in agent modeling in intelligent agent and social system, called ABM [21]. The parameter settings of the above two methods in our work follow their parameters in their papers [13, 21].

Evaluation Metrics: In our experiments, we employ time cost, scalability to evaluate efficiency, and precision, recall, and F1 to evaluate effectiveness.

4.2 Parameter Learning

Given a taxi driver, we learn the parameters β in Eq. 5 from the driver's historical behaviors as follows. When the driver comes to an unknown road, if the driver makes no call to other drivers to consult the given road's information, we assume the driver follow the own experienced knowledge; else if the driver makes calls to other drivers who have the experienced knowledge of the given road, and the given driver accordingly makes an action, we assume the driver follow the socialized knowledge. Based on the records in the historical data, we can have the percentage of a given driver's follow behavior. In our study, we take the percentage of following the own experienced knowledge as β. For different drivers, β is different and updating along the new records coming to the dataset. The parameter updating is intuitive which is not elaborated in the paper. In the following experiments, we utilize the percentage number as the parameter.



(a) Different Drivers

(b) Different Groups

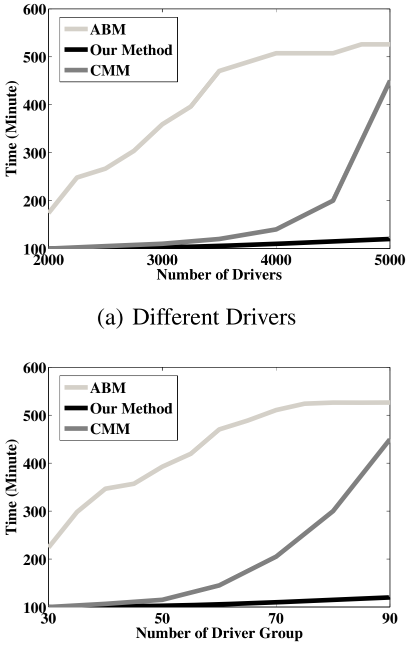
Fig. 1. Precision in Different Drivers and Groups

4.3 Effectiveness Evaluation

We evaluate effectiveness by *precision*, *recall*, and *F1*. *Precision* is the fraction of retrieved results that are relevant to the search, that is, the number of waiting/ cruising actions resulting from our model over the number of all waiting/ cruising actions made by taxis resulting from our model. *Recall* is the fraction of retrieved results that are relevant to the query that are successfully retrieved, that is, the number of waiting/ cruising actions resulting from our model over the number of all waiting cruising actions made by taxis.

Table 1. Our Method's Effectiveness in Different Driver Categories

DC	N	P	R	F
New	1260	67.8%	57.8%	62.3%
Normal	2355	71.7%	67.3%	69.4%
Experienced	1385	80.2%	78.7%	79.4%



(a) Different Drivers

(b) Different Groups

Fig. 2. Efficiency in Different Drivers and Groups

Table 2. Baseline Method's Effectiveness in Different Driver Categories

DC	N	P	R	F
New	1260	31.2%	17.8%	22.7%
Normal	2355	39.7%	27.3%	32.3%
Experienced	1385	45.9%	36.5%	40.8%

To evaluate the effectiveness of our method, we design two categories of experiments.

Category 1: effectiveness in different driver categories. The results of our method are listed in Table 1. DC is the driver category, N is the number of drivers in the category, P is *Precision*, R is *recall*, and F is *F1*. The results of ABM are listed in Table 2. In our experiment, CMM returns much worse results than ABM.

Category 2: effectiveness in different drivers, groups and time series. We conducted *precision*, *recall*, and *F1* tests in different drivers, groups and time series. Due to page limitation, we only show the *precision* results. In Figure 1 (a) and (b), we test the accuracy in one month data. The results show that our method returns much more accurate results than the baseline methods, not only in different drivers scenario, but also in different groups. In a conclusion, our method's accuracy is much better than baseline methods, and the accuracy also shows great scalability.

4.4 Efficiency Evaluation

We conducted efficiency tests in different drivers, groups and time series. The efficiency is measured by the time cost in a knowledge learning process. In Figure 2 (a) and (b), we test the efficiency in one month data. The results show that our method costs much less time than the baseline methods, not only in different drivers scenario, but also in different groups. In a conclusion, our method's efficiency is much more better than baseline methods, and the efficiency also shows great scalability.

5 Related Work

In [9], they proposed an approach to the problem of driving an autonomous vehicle in normal traffic. In [1], they discussed the spatial dispersion problem. But the work in this category either does not consider the social structure to retrieve the accurate information, or does not consider the dynamics in the learning process. In organized learning theory, this category work assumes that the sum of individual knowledge does not equate to organizational knowledge [2, 3, 5, 15, 16, 24]. In [4], they studied the distinction between individual knowledge and organizational knowledge, and prove the assumption. In [21], Ronald *et al.* demonstrated the design and implementation of an agent-based model of social activity generation. Szuba *et al.* [23] attempted to formally analyze the problem of individual existence of a being versus its existence in a social structure through the evaluation of collective intelligence efficiency. Heylighen *et al.*[13] argued that the obstacles created by individual cognitive limits and the difficulty of coordination could be overcome by using a collective mental map (CMM). Deng *et al.* [7] explored the use of active learning techniques to design more efficient trials. Rettinger *et al.*[20] studied the learning of trust and distrust in social interaction among autonomous, mentally-opaque agents. Wang *et al.*[25] presented an algorithm for finding the structural similarity between two domains, to enable transfer learning at a structured knowledge level. Cao *et al.*[6] proposed an adaptive transfer learning algorithm to adapt the transfer learning schemes by automatically estimating the similarity between a source and a target task. Zhu *et al.*[27] turned the co-training algorithm into a human collaboration policy. Unfortunately current work can not work well in our socialized information learning because of the challenges from dynamic updating along the time and large scale socializations.

6 Conclusion and Future Work

In this paper, we model the social information learning among taxi drivers and employ large scale real life data and empirical experiments to confirm our models in terms of much better effectiveness, efficiency and scalability than the state-of-the-art. Our models could be relevant to other domains, e.g., studying animal behavior, or where people go to sell things. We leave taxi driver decision model as the future work. How to model taxi driver's decision based on their collected information is a very interesting but challenging topic. Our current work can make such future work on the accurate and updated information.

Acknowledgment. This research is supported by the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office, and Huawei Corp. grant YBCB2009041-27, and University Transportation Center grant (DTRT12-G-UTC11) from the US Department of Transportation.

References

1. Alpern, S., Reyniers, D.: Spatial dispersion as a dynamic coordination problem. Theory and Decision 53(1) (2002)
2. Argote, L., Miron-Spektor, E.: Organizational learning: From experience to knowledge. Organization Science 22 (2011)
3. Argyris, C., Schn, D.: Organizational learning: A theory of action perspective. Addison-Wesley (1978)
4. Bhatt, G.: Information dynamics, learning and knowledge creation in organizations. The Learning Organization 7(2) (2000)
5. Bontis, N., Coss, V.M.: Managing an organizational learning system by aligning stocks and flows. Journal of Management Studies 39 (2002)
6. Cao, B., Pan, J., Zhang, Y., Yeung, D., Yang, Q.: Adaptive transfer learning. In: Proc. of AAAI (2010)
7. Deng, K., Pineau, J., Murphy, S.: Active learning for developing personalized treatment. In: Proc. of UAI (2011)
8. Devlin, K.: A framework for modeling evidence-based, context-influenced reasoning. In: Proc. of CONTEXT (2003)
9. Forbes, J., Huang, T., Kanazawa, K., Russell, S.: The batmobile: Towards a bayesian automated taxi. In: Proc. of IJCAI (1995)
10. Fu, W., Song, L., Xing, E.P.: Dynamic mixed membership blockmodel for evolving networks. In: Proc. of ICML (2009)
11. Ge, Y., Xiong, H., Liu, C., Zhou, Z.-H.: A taxi driving fraud detection system. In: Proc. of ICDM (2011)
12. Glaubius, R., Tidwell, T., Gill, C., Smart, W.: Real-time scheduling via reinforcement learning. In: Proc. of UAI (2010)
13. Heylighen, F.: Collective intelligence and its implementation on the web: Algorithms to develop a collective mental map. Comput. Math. Organ. Theory 5(3), 253–280 (1999)
14. Kay, J., Niu, W.T., Carmichael, D.J.: Oncor: ontology- and evidence-based context reasoner. In: Proc. of IUI (2007)
15. Lee, S.J., Popović, Z.: Learning behavior styles with inverse reinforcement learning. ACM Trans. Graph. 29 (July 2010)
16. Li, P., Yu, J.X., Liu, H., He, J., Du, X.: Ranking individuals and groups by influence propagation. In: Huang, J.Z., Cao, L., Srivastava, J. (eds.) PAKDD 2011, Part II. LNCS, vol. 6635, pp. 407–419. Springer, Heidelberg (2011)
17. Liu, S., Liu, C., Luo, Q., Ni, L., Krishnan, R.: Calibrating large scale vehicle trajectory data. In: Proc. of IEEE MDM (2012)

18. Liu, S., Liu, Y., Ni, L.M., Fan, J., Li, M.: Towards mobility-based clustering. In: Proc. of ACM SIGKDD (2010)
19. Nickel, M., Tresp, V., Kriegel, H.-P.: A three-way model for collective learning on multi-relational data. In: Proc. of ICML (2011)
20. Rettinger, A., Nickles, M., Tresp, V.: Statistical relational learning of trust. Mach. Learn. 82(2) (2011)
21. Ronald, N., Dignum, V., Jonker, C., Arentze, T., Timmermans, H.: On the engineering of agent-based simulations of social activities with social networks. Inf. Softw. Technol. 54(6) (2012)
22. Russell, S.: Learning agents for uncertain environments (extended abstract). In: Proc. of COLT (1998)
23. Szuba, T., Polański, P., Schab, P., Wielicki, P.: On Efficiency of Collective Intelligence Phenomena. In: Nguyen, N.T., Kowalczyk, R. (eds.) Transactions on CCI III. LNCS, vol. 6560, pp. 50–73. Springer, Heidelberg (2011)
24. Templeton, G.F., Lewis, B.R., Snyder, C.A.: Development of a measure for the organizational learning construct. J. Manage. Inf. Syst. 19(2) (2002)
25. Wang, H., Yang, Q.: Transfer learning by structural analogy. In: Proc. of AAAI (2011)
26. Xiao, L., Zhou, D., Wu, M.: Hierarchical classification via orthogonal transfer. In: Proc. of ICML (2011)
27. Zhu, X., Gibson, B., Rogers, T.: Co-training as a human collaboration policy. In: Proc. of AAAI (2011)

Mining User Interests from Information Sharing Behaviors in Social Media

Tingting Wang[1], Hongyan Liu[2,*], Jun He[1,*], and Xiaoyong Du[1]

[1] Key Labs of Data Engineering and Knowledge Engineering, Ministry of Education, China
School of Information, Renmin University of China, 100872, China
{wtt526,hejun,duyong}@ruc.edu.cn

[2] Department of Management Science and Engineering, Tsinghua University, 100084, China
hyliu@tsinghua.edu.cn

Abstract. Mining user interests and preference plays an important role for many applications such as information retrieval and recommender systems. This paper intends to study how to infer interests for new users and inactive users from social media. Although some recently proposed methods can mine user interests efficiently, these works cannot make full use of relationship between users in their social network. In this paper, we propose a novel approach to infer interests of new users or inactive users based on social connections between users. A random-walk based mutual reinforcement model combining both text and link information is developed in the approach. More importantly, we compare the contribution of different social connections such as “follow”, “retweet”, “mention”, and “comment” to interest sharing. Experiments conducted on real dataset show that our method is effective and outperforms existing algorithms, and different social connections have different impacts on mining user interests.

Keywords: Interest inferring, Social networks, Information sharing behaviors.

1 Introduction

With the advance of web technology, many micro-blogging websites are emerging rapidly. This kind of websites allows users not only to publish their views but also to share interests online. Twitter is one of the most famous micro-blogging services [14, 15], while in China Tencent Weibo is one of the largest micro-blogging websites, and it provides the similar social-networking functionality with Twitter.

A user registered in Tencent provides his profile information such as gender and date of birth, and publishes messages from time to time. Different people have different publishing frequency. Some are very active and some are very inactive. Besides, unlike other social network services that require users to send a request to make friends, another important part is that every user is allowed to “follow” the others without seeking any permission. In this case, the user who initiated this behavior is called “follower”, while the one who is followed is called “friend”. After a “follow”

[*] Corresponding authors.

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 85–98, 2013.

relationship is built, follower can read the messages published by his friends. Furthermore, follower can retweet a message or leave a comment to it if he or she is interested in the content. Unlike users registered in Twitter, users in Tencent can append some words when he retweets the message from others. Besides, a user is allowed to mention someone else when he publishes a message. To infer user interests, we use all of the above-mentioned information. For convenience, we call user's profile information and messages posted ***text*** information and social connection relationships regarding *follow*, *retweet*, *comment* and *mention* ***link*** information.

As we know, the number of micro-blogging users increased rapidly. A statistic from a newspaper shows that, the number of Tencent Weibo users has reached 310 million and the number of active users has reached 50 million up to November 2011. Furthermore, each day there are thousands of enthusiastic new users adding into the existing billions of actively engaged users. Although the large number of Tencent Weibo users benefit each other by information sharing, too much information results in information overload problem, which needs systems such as searching and recommender system to solve. Thus, it's really important to capture user interests and then provide personalized results and meet user's needs individually according to one's interests.

There are many good ways [1, 2, 3] to infer user's interests, however, most of them are not proposed for the social network. In these approaches user interest models are built according to the behavior history during web search, such as click-through web pages. For users in social network, the mainly behaviors are communication with the other users. Based on this difference, recently there are some methods proposed for social network. These algorithms mine user's interests through user contents such as micro-blog messages and user-generated tags. However, in social network there are some users preferring to read messages from other active users rather than publish some information about themselves. In this case, user-generated tags are not always representative of all of his interests, and only a small part of users provide tags. Especially for a user who just registered or the user who is not very active, both user-generated tags and micro-blog messages are difficult to get. Another kind of existing works inferred user interests from social neighbors to solve the problem of inactive users. However, this solution focuses on the three-degree ego network of a user and uses the information in a deterministic way. After inferring interests for one target user, its interests will not affect the target user's neighbors, which is unreasonable. In this paper, we emphasize mutual reinforcement between users through a process similar to random walk [16, 18].

Our proposed method is based on the homophily in social network. The phenomenon of homophily means that individuals with similar characteristics tend to associate with each other. Although homophily has been validated in different social networks such as twitter and student homepages [13, 17, 19], it isn't studied deeply in Tencent Weibo. As Tencent Weibo is mainly for Chinese users, and the culture and some information sharing mechanisms are different, it is necessary to study which kind of social connections reflect homophily and which one contributes more to interest sharing. To do this, before developing the interest inferring method, we conducted some statistical tests to study the relationship between social connections and similarity of user interests (Detail is explained in Section 3.1). Existing work has studied semantics of the follow and retweet relationship in twitter [20]. In this paper, we studied more relationships such as comment, besides follow, retweet and mention.

According to this study, a conclusion that users with communication behaviors share more interests than those without can be made. Based on this finding, we propose a novel approach to infer user interests and we develop an algorithm to implement the approach. In this algorithm, first a directed graph to indicate potential interest propagation among users is constructed. And then text information is used to generate initial interests and link information is utilized to show how users affect each other in interests. Then, a mutual reinforcement process based on a random walk model is conducted to infer interests for new users and inactive users.

Our work offers two contributions.

First, we studied the relationship between social connections such as follow, retweet, comment and mention and common interests between users. Statistical study shows that different social behaviors have different influence on the interest similarity between users. We find that follow and retweet mean more strong connections for users in common interests than mention and comment.

Second, we propose a novel approach combining users' text information and link information (information about social connection) to infer interests for inactive users. In this approach, the mutual reinforcement between users is emphasized by a random walk model. Experiments show that this approach can improve the accuracy by up to 21.4%. Especially for inactive users, this approach can address the shortcomings of too little information.

The rest of the paper is organized as follows. In section 2, problem definition is given. The proposed algorithm is introduced in section 3. Experiment setup and results are described in section 4. In section 5, related work is discussed. Finally, conclusions are drawn in section 6.

2 Problem Definition

Let U be the set of users registered in a social media website such as Tencent WeiBo. Each user has a unique ID assigned by the system. In this paper, interest is defined as a pair of keyword and its weight about this keyword. For active users, keywords can be extracted from the text information of the user. Weight shows the favorite degree of the user to this keyword. The larger the weight is, the more the user likes the interest. One user can have one or more interests. Thus, a vector of pairs of keyword and weight is used to express the interests of users as shown in the definition below:

Definition 1. The interests of a user are expressed by a set of pairs of keyword and weight.

$$\{keyword_1{:}weight_1;\ keyword_2{:}weight_2;\ \ldots\ \ldots;\ keyword_l{:}\ weight_l\}$$

Example of interest information is shown in Table 1, where each integer represents a keyword, followed by weight. Further, the interests of a user a can be expressed by a vector I_a=<$weight_1$, $weight_2$, …, $weight_{N>}$, where N is the size of the union of all of user's interest keywords.

Table 1. Interest information

UserID	Interest Vector
10001	<101:0.4; 102:0.3; ······
……	……
10005	<101:0.4; 103:0.3; ······

Table 2. Follow relationship

UserID	Interest Vector
10001	<101:0.4; 102:0.3; ······
……	……
10005	<101:0.4; 103:0.3; ······

Table 3. Link information between users

User *a*	User *b*	RTnum	MEnum	CMnum
10001	10002	10	2	5
……	……	……	……	……
10004	10007	3	0	0

The other kind of information is the link information between every pair of users. For users registered in Tencent Weibo, the basic behavior information between two users is the "follow" relationship. Besides, there are several other behaviors between two users, including "retweet (publishing other user's message)", "mention (mentioning other users when publish a message)", "comment (having a comment on other user's message)" and so on. These behaviors create links between users, which will be introduced in details in the next section. The *follow* relationship information is showed in Table 2. And the other behavior information, such as "retweet", "mention" and "comment" is given in Table 3, which shows the numbers of times of these different behaviors happened from user *a* to user *b*.

According to link information, a directed behavior graph $G(V, E)$ can be constructed to show the relationships among users. V is the node set which contains all the registered users. E is the edge set. Suppose a and b are two registered users. If user a has any action of follow, retweet, mention and comment to user b, an edge (a, b) is formed from user a to user b.

After constructing behavior graph $G(V, E)$, a directed graph $G'(V, E')$ called ***propagation*** graph is constructed to model how user interest propagates, in which a node is also on behalf of a user registered in the website. V is the same node set as that in graph G, and each edge (b, a) in E' corresponds to edge (a, b) in E. That is, if user a initiated an action such as "follow", "retweet", "comment" or "mention" to user b, there is an edge from the node b to node a. The direction of the edge is exactly opposite to the one in E. Because when an action is initiated from user a to user b, it reveals that user b's interests attract user a, and user b has some influence on user a about his interests. Thus, the interests should propagate from user b to user a. Besides, there is a weight assigned to this edge to indicate the influence on interests user b has on user a. And also an interest vector is assigned to each node according to the text information of users. However, first, not every user has this value, because some new registered users have little information published. And second, it's not easy to collect the interest information for every user especially for those inactive users who have little information. Thus, our mining task is to infer interests for these users in the network.

3 Interest Inferring Method

3.1 Hypothesis Tests

To infer user interests from his link relationship, several questions need to be answered to prove whether this approach is valid.

Questions 1, 2, 3 and **4:** Do users with "follower-friend", "retweet", "comment" or "mention" relationships in micro-blogging system in China have more similar interests than those without?

Besides these four questions directly related to the four kinds of link information we mentioned before, another factor indirectly related to the "follow" relationship is analyzed, that is the ratio of common friends of two users. In the next sections, this information is also discussed with the link information. Thus, another similar question is raised.

Question 5: Do users who have more common friends in Chinese micro-blogging system have more similar interests than those without?

To answer these questions, we give the definition of interest similarity of two users as follows:

Definition 2. Interest similarity of two users *a* and *b* can be measured by Equation 1.

$$ISim_{ab} = cos(v_a, v_b) \tag{1}$$

v_a and v_b are interest vectors of user *a* and *b* respectively, extracted from their text information.

Question 1 can be formalized as a two-sample *t*-test. Let u_{follow} be the mean interest similarity of the pairs of users with "follower-friend" relationship, while $u_{nofollow}$ the mean interest similarity of the pairs of users without. Let H_0 be the null hypothesis: $u_{follow}=u_{nofollow}$, and H_1 be the alternative hypothesis: $u_{follow}>u_{nofollow}$. Results show the null hypothesis is rejected at significant level α =0.01 with a *p*-value of 3.14×10^{-5}. Question 2, 3, 4 and 5 are formalized as a two-sample *t*-test separately, too. Results show that the answers of Question 2, 3, and 4 are positive, and the null hypothesis is rejected at significant level $\alpha = 0.05$. To conduct a hypothesis test on Question 5, Equation 2 is used to measure the ratio of the common friends between two users.

$$cf_{ab} = \frac{|F_a \cap F_b|}{|F_a \cup F_b|} \tag{2}$$

F_a and F_b are the friend sets of user *a* and *b* separately. When selecting users who are used in hypothesis test, the users whose "common friends" measurement are larger than 0.8 are selected. Result shows that the null hypothesis is rejected at significant level α= 0.05 with a *p*-value of 2×10^{-3}.

From these tests, we know that all the answers to these questions are positive, which shows that users who have these behaviors (follow, retweet, comment, mention and common friends) are more similar than users who don't. Based on this outcome, a novel approach to infer user interests is proposed in the next part.

3.2 Random Walk Based Inference Model

In this section, we focus on the problem of how to infer user interests after we construct the propagation graph. We will explain how to construct the graph later.

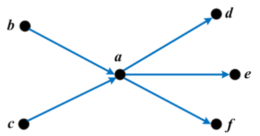


Fig. 1. User follow relationship

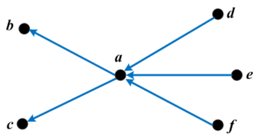


Fig. 2. User influence relationship on interests

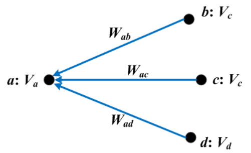


Fig. 3. One user and his in-degree neighbors

For one user in the social network, its local directed graph is shown in Fig. 1 to show its follow relationship.

And the directed graph to show its influence relationship in interests is shown in Fig. 2, right opposite to the direction in Fig. 1. In Fig. 1, user *a* follows user *d*, *e*, and *f*, and also is followed by user *b* and *c*. In Fig. 2, the direction of interest propagation is right opposite. Interests are propagated from user *d*, *e*, *f* to user *a*, while user *a* propagates interests to user *b* and *c*.

From Fig. 2, the interests of user *a* can be collected from two aspects. One is the text information of user *a*. For a user who has published some messages, an interest vector can extracted from the messages he published, retweeted or commented. On the other hand, based on the finding in the previous section, some information sharing behaviors indicate the common interests between users. Thus, interest information can also be inferred from those users who have link relationship with them. The interests are propagated to the user in a certain probability, which is expressed by weight of the corresponding in-degree edge. We denote this probability by w_{ij} on edge (u_j, u_i). For example, the probability of the interests propagating from user *d* to user *a* is denoted as w_{ad}, as shown in Fig. 3. Combining the two resources of information, for user *a*, his interests according to this method is inferred by Equation 3.

$$I_a = \alpha \cdot \sum_{i \in U} w_{ai} \cdot I_i + v_a \tag{3}$$

In Equation 3, *U* is the set of all users in the network, v_a is the interest vector of user *a* extracted from the text information, and I_a is user *a*'s overall interest vector considering both text and link information, and α is the decaying factor of influence from user's in-degree neighbors. The lower α is, the less influence a user will be got from his friends, and vice versa.

According to this formula, the interests can be computed recursively, because users influence each other during the information sharing behaviors. Thus, a random walk process is utilized to implement the mutual reinforcement between users.

Suppose the whole propagation matrix is denoted as *P*. *P* is a |*U*|×|*U*| matrix, where each entry is equal to w_{ij}, as we described above. All users' interest vectors are collected into a |*U*|×*N* matrix, *v*, where *N* is the total number of keywords. Each row v_i of the matrix *v* is the interest vector extracted from text information of user *i*. Then the interests of all users in network can be computed.

The interest matrix of the users, denoted as *I*, where row *j* represents user *j*'s interest vector. Matrix *I* can be calculated iteratively by Equation 4. I_t is a |*U*|×*N* matrix and represents the interests after *t* times of iterations, $t > 0$. Initially, $I_0 = v$.

$$I_t = \alpha \cdot P \cdot I_{t-1} + v \tag{4}$$

According to the property of Markov chain, convergence is guaranteed if *P* is stochastic. In the next sections, we introduce how to compute the weight of the propagation graph and make sure that *P* satisfies this requirement.

3.3 Generating Interest Vector from Text Information

There are several methods to produce initial interest vectors for users. Usually, user-generated tags can be considered as a way to express user interests. However, most people add few tags in the system, thus other information, such as messages one posted, can be utilized. All tweets posted by one user can be collected as a document for the user. These tweets include not only the tweets and comments published by the user himself and also those retweeted from others. And for all users in the website, a set of documents can be collected and used. In this paper, a typical method, *Latent Dirichlet Allocation* (*LDA*) [5, 12] model is applied to these documents, which is an unsupervised machine learning technique to identify latent topic from large document collection.

3.4 Assigning Weights to Edges

In the social media websites like Tencent Weibo micro-blogging system, users can communicate with each other by retweet, comment and mention behaviors. According to these different communications, five different factors are defined to compute the weight of edge (*b, a*) in propagation graph.

Based on Retweet

We measure the influence of user *b* to user *a* based on the amount of user *b*'s tweets retweeted by user *a*. The more tweets retweeted by user *a*, the more influence user *b* has to user *a*, and the more common interests occurs between user *a* and user *b*. Let RT_{ab} be the number of tweet retweeted by user *a* from user *b*. The weight is measured by Equation 5.

$$w_{rt} = \frac{RT_{ab}}{\sum_{i \in U} RT_{ai}} \tag{5}$$

Based on Comment

The number of comments which user *a* gives to user *b* measures the degree user *a* shows interests on the tweets of user *b*. Let CM_{ab} be the number of comments user *a* gives to user *b*. Then the weight of the edge from user *b* to user *a* is calculated like by Equation 6.

$$w_{cm} = \frac{CM_{ab}}{\sum_{i \in U} CM_{ai}} \tag{6}$$

Based on Mention

Mention action is another communications between two users. To some extent, the frequency of this action can show the influence user b has to user a. let ME_{ab} be the number of "mention" actions user *a* gives to user *b*, then we measure the weight according to Equation 7.

$$w_{me} = \frac{ME_{ab}}{\sum_{i \in U} ME_{ai}} \tag{7}$$

Based on Follow Relationship

The “follow” relationship is the basis and most usual action in the social network. Mostly, user *a* will follow user *b* if he is interested in the tweets posted by user *b* or user *b* himself. Thus, this kind of relationship can reflect the relationship between two users and their interests. f_{ab} is used to show whether user *a* follow user *b*. According to this, the weight of the edge from user *b* to user *a* is measured by Equation 8, where f_{ab}=1 if user *a* follows user *b*, otherwise, f_{ab}=0.

$$w_f = \frac{f_{ab}}{\sum_{i \in U} f_{ai}} \tag{8}$$

Based on Intersection of Friends

According to the “follow” relationships of one user, a list of his friends can be got. For user *a* and user *b*, the larger the set of intersection of their friends, the more interests they share. Let F_a be the set of the friends of user *a*, and F_b be the set of the friends of user *b*. Then the influence user *b* has on user *a* can be calculated according to this Equation 2. And then the weight on the edge (*b*, *a*) in the propagation graph is measured by Equation 9.

$$w_{cf} = \frac{cf_{ab}}{\sum_{i \in U} cf_{ai}} \tag{9}$$

Considering this will generate a matrix with so many non-zero numbers, we neglect those values which are smaller than 0.1. Through this process, the non-zero values are reduced to 422380, which makes the matrix sparse and improves the efficiency of iteration.

If the denominators in Equations 5, 6, 7, 8, 10 are zeros, 1/|*U*| is used to replace the formula. Combining these five factors, a comprehensive computation formula is proposed in Equation 10.

$$w_{ab} = \frac{w_{rt} \cdot RT_{ab} + w_{cm} \cdot CM_{ab} + w_{me} \cdot ME_{ab} + w_f \cdot f_{ab} + w_{cf} \cdot cf_{ab}}{\sum_{i \in U} (w_{rt} \cdot RT_{ai} + w_{cm} \cdot CM_{ai} + w_{me} \cdot ME_{ai} + w_f \cdot f_{ai} + w_{cf} \cdot cf_{ai})} \tag{10}$$

After these factors are defined, the propagation matrix on interests can be identified. From these computation formulas, each row in the propagation matrix is sum up to 1, which makes the propagation matrix stochastic. This makes sure the iteration process will converge.

4 Experiment

4.1 Dataset

A large dataset collected from Tencent weibo is provided by the Tencent Company. To test the proposed method, a relatively small network is extracted by a *BFS* algorithm. After this extraction process, the total number of users in *U* is 5238. For every pair of users in *U*, the corresponding information is also extracted, including the follow relationship, the number of “retweet”, “comment”, and “mention” actions. Table 4 shows some information of this dataset *U*. The distribution of the followers for each user is shown in Fig. 4.

Basically, this distribution of followers per user follows a power-law distribution approximately. That is, most people have small number of followers, while only a small of users have a large number of followers, which proves that the experiment data extracted is reasonable and representative.

Table 4. Basic information of dataset *U*

items	value
# of users	5,238
# of users in training	4,190
# of users in test set	1,048
# of follow relation-	133,825

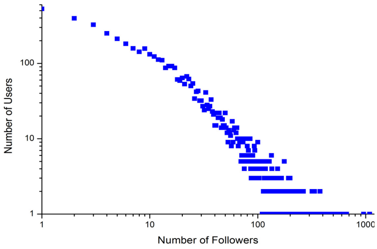


Fig. 4. Distribution of followers per user

For each user in the network, keywords information is extracted from messages posted. The number of distinct keywords for all 5238 users is 22376, and the average number of keywords which one user has is 29. A five-fold cross validation is conducted in this paper. We split these 5238 interest vectors into five parts. For every time, four parts of the vectors are used as training set, and the rest one is test set, whose interest vectors are regarded as truth. In the experiment, the text information of users in training set and the link information of users both in training set and test set are utilized to infer interests for users in test set.

4.2 Performance Comparison

We conduct experiments based on five different behavior factors, “retweet”, “mention”, “comment”, “follow” and “common friends”. Correspondingly, we denote these methods as “*RT*”, “*ME*”, “*CM*”, “*Follow*” and “*Common Friends* (or *CF*)”. In the next tables, these abbreviations will be used to show the comparisons. For each factor, a separate experiment is conducted to compare which factor performs better. To combine all of these factors, five weights need to be determined. We extract a part of data from training set and compute five *NDCG* values according to different factors separately. Based on the *NDCG* values, we set the five weights. The larger the *NDCG* value is, the larger the weight of the factor. In this paper, we set these weights, w_{rt}, w_{cm}, w_{me}, w_f, and w_{cf} as 0.31, 0.25, 0.25, 0.32, and 0.34 respectively. The method combining the five factors is called “*combination*”. When determining whether to stop the iterations, the sum of absolute errors of each entry of the result matrix is used. In our experiment, this value is set to 0.1. When the sum of the absolute errors is smaller than 0.1, the iterations will stop and the method get final results for each user in the test set.

Comparison against related algorithm is also conducted. The work in [4] is one of the classic related works in inferring interests. In this paper, user interests are inferred from his social connections, that is, his friends, friends’ friends and 3-degree friends. This method is called “3*D-Friends*” here. For each user, a 3-degree ego network is constructed to infer the interests for inactive users. These results are compared in Table 5.

Table 5. Comparison of different methods

Methods	NDCG
RT	0.3120
CM	0.2533
ME	0.2576
Follow	0.3215
Common Friends (*CF*)	0.3360
Combination	0.3493
3*D-Friends*	0.2878

Table 6. Num. of edges in propagation graphs

Factor	# of edges
RT	112039
CM	25526
ME	27018
Follow	133825
Common Friends	422380
Combination	511676

From Table 5 we observe that the factor of common friends has more significant impact than the other four factors. The method based on follow relationship also works better, with *NDCG* only less than the one based on common friends. However, the methods based on "comment" and "mention" don't work very well. The reason why the performance of these two methods is not very well will be studied later. Besides, the method based on five comprehensive factors outperforms all the other methods. Our best method increases the quality of interest inferring than the existing method, 3*D-Friends*, by 21.4%.

The number of edges of the propagation graph based on each factor is shown in Table 6. For a graph that has 5238 nodes, the total number of edges of complete graph is 5238×5238=27436644. When the factor of "*Common Friends*" is considered, the graph is complete with a large number of edges. We reduce the number of edge constructed based on "*Common Friends*" to improve the efficiency. If the weights computed based on "*Common Friends*" factor is smaller than 0.1, the corresponding edge is removed. Through this process, the non-zero values in the propagation matrix are reduced to 422380, which make the matrix sparse and improve the efficiency of computation. Accordingly, the number of edges of the "*Combination*" method is not very large, too. After this edge removal step, the number of edges is reduced to 511676, which makes the propagation matrix sparse, too. For the other four factors, the number of edges is small, especially for the factors of "comment" and "mention". The propagation matrix based on "retweet" or "follow" has more non-zero values than those based on "comment" and "mention". This tells us that users prefer to follow the others and retweet the tweets more than comment or mention others. Basically, based on the comparison we can conclude that different user behaviors have different impact on user interests, which is same with the conclusion with Adamic and Adar [19], that is, some factors are better indicators of social connections than others.

We also compare the efficiency of our method with 3*D-Friends*, which is illustrated in Fig. 5. From this Figure, we know that the time spent is proportional to the number of the non-zero values in the propagation matrix. The efficiency of our method based on mention, comment, retweet or follow is better than the method 3*D-Friends*. Time spent on common friend graph and the combination graph is more than 3*D-Friends*, because these two graphs are much denser than the graph 3*D-Friends* uses.

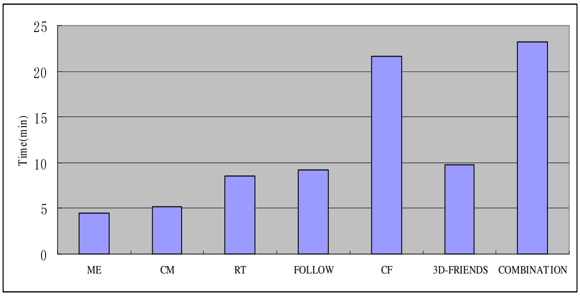
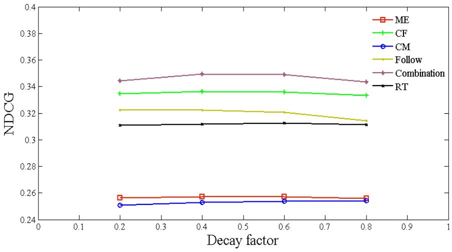


Fig. 5. Efficiency of different methods

4.3 Effect of Decay Factor

In this experiments described in the last section, we set the decaying factor α to 0.5. However, this decaying factor determines the important degree of the influence from a user's friends. The result of the algorithm will differ according to different decaying factor. Fig. 6 shows the changes of the results *NDCG* of all our six algorithms based on different decaying factor values.



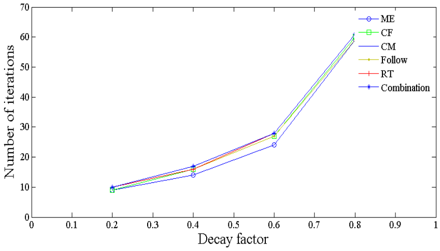


Fig. 6. Effect of α to *NDCG*

Fig. 7. Effect of α to number of iterations

From Fig. 6 we can observe that in most scenarios *NDCG* gets the best when α is between 0.4 to 0.6. However, the changes caused by different decaying factors are not very significant. No matter what value the decaying factor is, the basic trend among the results of the five methods remains similar.

The value of decaying factor has influence on not only the accuracy of the results, but also the efficiency of the algorithms. The number of iterations for different decaying factors is shown in Fig. 7. The larger the number of iterations is, the more time the corresponding method spends. From Fig. 7 we can see that the growth trends follow an exponential distribution. In our experiment, the decaying factor α is set to 0.5, and the number of iterations is about 21. When α is larger than 0.6, the run time increases a lot, and neighbor's influence becomes too heavy in the meantime.

5 Related Work

In this section, we briefly introduce related work. We category related work about user interest study into three groups: 1) based on user contents 2) based on user behavior 3) based on social cues.

Based on User Contents. Simply, explicit interests can be specified directly from users' profiles. In addition, other sources can also indicate users' interests. Some users prefer to use descriptive tags to express what they are interested in. Therefore, some researchers proposed approaches to find social interest based on user-generated tags [1, 9, 10]. However, these tags are not always representative, and some users don't like to add these tags to themselves. Although the tags extracted from user micro-blogs can replace the user-generated tags, it still cannot work well for new users and inactive users who have few micro-blogs.

Based on User Behaviors. Several algorithms have been proposed based on user behaviors during web search and browsing. Interests are captured from click through data or visited web pages [2, 6]. Qiu and Cho [7] focus on disambiguating the true intention of a query based on past click history. Kim and Chan [8] proposed a divisive hierarchical clustering algorithm to learn a user interest hierarchy from a set of web pages. These methods based on user behaviors, especially based on click-through history and web pages are mainly used to personalize user interests in the web search community. However, for users in social network, the click history is too sparse to be utilized to infer user interests. Most of users' behaviors are to share the messages from their friends.

Based on Social Cues. All of these above-mentioned methods use user individual information to infer interests. For the inactive users who have not many profiles and behaviors, these approaches cannot work well. Therefore, some researchers propose to infer user interests by leveraging social cues from other users. Similar with collaborative filtering systems, Glodberg et. al [3] proposed a method to mine the interests from the users who have similar opinions on a set of items. Their basic idea is that users who have similar behaviors will share similar interests. Similarly, White et.al [11] proposed a method to find a user's interests from other users that visit the same page as the user. In addition, another new approach is proposed by Wen and Lin [4]. It focuses on social cues from user's neighbors. In this work, for one target user, the neighbors in his three-degree ego network are considered. That is, for each user, a 3-degree ego network is constructed to infer interests. Relationships between users are built based on electronic communication data such email and instant messaging and Web2.0 social content such as social bookmarks and file sharing. The interests of active users in the network are extracted by LDA model from text information. Then inactive users' interests are computed based on their neighbors in a deterministic way, without considering user's mutual influence. Besides, Welch et al [20] demonstrate that in Twitter platform retweeting is a better indicator of topical interest than following behavior through the PageRank algorithm.

In our study, we focus on leveraging the social network to infer user interests. The contents from active users are considered as initial interests. These interests are propagated through the social network, which is built according to the interest similarity between users. This approach has important differences from the above-mentioned existing work in two aspects. First, unlike the existing methods based on user content and behavior, our proposed approach works well especially for users who have no text information. Second, different from the methods based on social cues, we consider the social connections and emphasize the mutual reinforcement among users, instead of directly inferring from a couple of friends or similar users.

6 Conclusions and Future Work

In this paper, we propose a novel approach for user interests inferring especially for new users and inactive users who have few messages published. In this approach, a random walk on a propagation graph model is used to emphasize the mutual reinforcement between users. When constructing the propagation graph, both text information and link information of users are taken into account. Besides, we prove by statistical tests that information sharing behaviors such as follow, retweet, comment and mention are related to the common interests between users. And the experimental results conducted on real social network data set show that different kind of social connections have different influence to common interests. Experimental results demonstrate that our methods get a better performance not only in the quality but also in efficiency. In the future, we will utilize the approach to provide better results for recommender system in social network.

Acknowledgement. This work was supported by the 973 program of China under Grant No. 2012CB316205 and the NSF of China under Grant No. 71272029, 70890083, 71110107027 and 61033010.

References

1. Li, X., Guo, L., Zhao, Y.E.: Tag-based social interest discovery. In: Proc. of the 17th International Conference on World Wide Web, pp. 675–684. ACM, New York (2008)
2. Agichtein, E., Brill, E., Dumais, S.T.: Improving web search ranking by incorporating user behavior information. In: Proc. of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 19–26. ACM, New York (2006)
3. Glodberg, D., Nichols, D., Oki, B.M., Terry, D.: Using collaborative filtering to weave an information tapestry. Communications of the ACM-Special Issue on Information Filtering 35(12), 61–70 (1992)
4. Wen, Z., Lin, C.-Y.: On the quality of inferring interests from social neighbors. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2010, pp. 373–382. ACM, New York (2010)
5. Blei, D.M., Nq, A.Y., Jordan, M.I.: Latent dirichlet allocation. The Journal of Machine Learning Research 3, 993–1022 (2003)
6. Claypool, M., Brown, D., Le, P., Waseda, M.: Inferring user Interest. IEEE Internet Computing, 32–39 (2001)
7. Qiu, F., Cho, J.: Automatic identification of user interest for personalized search. In: Proceedings of the 15th International Conference on World Wide Web, WWW 2006, pp. 727–736. ACM, New York (2006)
8. Kim, H.R., Chan, P.K.: Learning implicit user interest hierarchy for context in personalization. In: Proceedings of the 8th International Conference on Intelligent User Interfaces, IUI 2003, pp. 101–108. ACM, New York (2003)
9. Shepitsen, A., Gemmell, J., Mobasher, B., Burke, R.D.: Personalized recommendation in social tagging systems using hierarchical clustering. In: Proceedings of the 2008 ACM Conference on Recommender Systems, RecSys 2008, pp. 259–266. ACM, New York (2008)

10. Stoyanovich, J., Amer-Yahia, S., Markow, C., Yu, C.: Leveraging tagging to model user interests in del.icio.us. In: AAAI Spring Symposium on Social Information Processing (2008)
11. White, R.W., Bailey, P., Chen, L.: Predicting user interests from contextual information. In: Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2009, pp. 363–370. ACM, New York (2009)
12. Steyvers, M., Griffiths, T.: Probabilistic topic models. In: Landauer, T., McNamara, D., Dennis, S., Kintsch, W. (eds.) Latent Semantic Analysis: A Road to Meaning. Lawrence Erlbaum (2007)
13. Weng, J., Lim, E.-P., Jiang, J., He, Q.: TwitterRank: Finding topic-sensitive influential twitterers. In: Proceedings of the Third International ACM Conference on Web Search and Data Mining, pp. 261–270 (2010)
14. Micro-blogging, http://en.wikipedia.org/wiki/Micro-blogging
15. Milstein, S., Chowdhury, A., Hochmuth, G., Lorica, B., Magoulas, R.: Twitter and the micro-messaging revolution: Communication, connections, and immediacy-140 characters at a time. O'Reilly Report (November 2008)
16. Random Walk, http://en.wikipedia.org/wiki/Random_walk
17. McPherson, M., Smith-Lovin, L., Cook, J.M.: Birds of a feather: Homophily in social networks. Annual Review of Sociology 27(1), 415–444 (2001)
18. Tong, H., Faloutsos, C., Pan, J.-Y.: Fast random walk with restart and its applications. In: Proceeding of the 6th international Conference on Data Mining, pp. 613–622.
19. Adamic, L.A., Adar, E.: Friends and Neighbors on the Web. Social Networks 25(3), 211–230 (2003)
20. Welch, M.J., Schonfeld, U., He, D., Cho, J.: Topic semantics of twitter links. In: Proceedings of the Fourth International ACM Conference on Web Search and Data Mining (WSDM 2011), Hong Kong, China (2011)

Anonymization for Multiple Released Social Network Graphs

Chih-Jui Lin Wang[1], En Tzu Wang[2], and Arbee L.P. Chen[3,*]

[1] Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan, R.O.C.
s9862588@m98.nthu.edu.tw

[2] Cloud Computing Center for Mobile Applications, Industrial Technology Research Institute, Hsinchu, Taiwan
m9221009@em92.ndhu.edu.tw

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

Abstract. Recently, people share their information via social platforms such as Facebook and Twitter in their daily life. Social networks on the Internet can be regarded as a microcosm of the real world and worth being analyzed. Since the data in social networks can be private and sensitive, privacy preservation in social networks has been a focused study. Previous works develop anonymization methods for a single social network represented by a single graph, which are not enough for the analysis on the evolution of the social network. In this paper, we study the privacy preserving problem considering the evolution of a social network. A time-series of social network graphs representing the evolution of the corresponding social network are anonymized to a sequence of sanitized graphs to be released for further analysis. We point out that naively applying the existing approaches to each time-series graph will break the privacy purposes, and propose an effective anonymization method extended from an existing approach, which takes into account the effect of time for releasing multiple anonymized graphs at one time. We use two real datasets to test our method and the experiment results demonstrate that our method is very effective in terms of data utility for query answering.

Keywords: social network, privacy, anonymization, time-serial data.

1 Introduction

Recently, people share their information and participate in various activities on the Internet via social platforms such as Google+, Facebook, and Twitter. The data in social networks are worth being analyzed in social science research since they can reflect the real social activities. Since the data in social networks include personal information and interactions, which can be private and sensitive, there is a need to anonymize the data to protect users' privacy before their release for some analysis.

A social network can be represented by a graph consisting of nodes and edges between the nodes. The nodes are used to represent users while the edges represent

[*] Corresponding author.

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 99–110, 2013.

the interactions between the users. A trivial method for protecting users' privacy in the released graph is to replace the identities of users (e.g. ID or user name) by random values. This is not enough since the privacy may still be revealed by *malicious attackers* [2]. Achieving complete privacy protection and at the same time offering high utility of the social network data are challenging [4][8][12]. The existing approaches on anonymizing a social network graph are categorizing into two types including nodes with attributes [1][3] and without attributes [6][10][11].

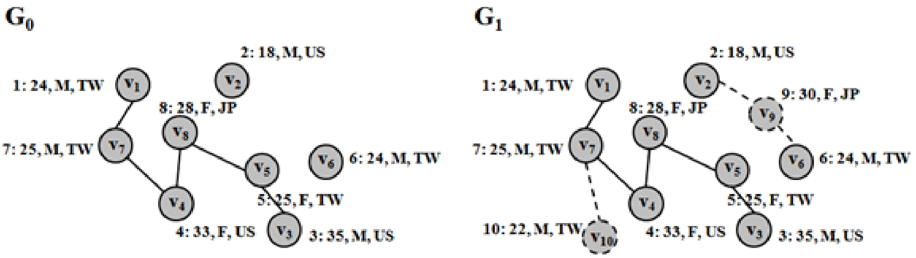
Consider a scenario as follows. A service provider, e.g. Facebook, has social network data since 2004 and wants to anonymize and release the corresponding social network graph from 2004 to 2006 to a data mining company to find interesting patterns from the evolution of the social network. Since the previous studies focus on anonymizing a single social network graph, we may consider applying the existing approaches to anonymize the social network graphs at different timestamps by considering the graph at each timestamp as a specific social network graph. However, to be detailed in Subsection 2.2, this may make the privacy of the social network data leaked.

In this paper, we consider a privacy preserving problem on a time-series of social network graphs representing an evolution of a specific social network. A time-series of social network graphs are anonymized to a sequence of sanitized graphs to be released (called *multiple releases* in this paper). Based on [1], which groups nodes into classes for achieving the privacy purposes, we design a constraint in the grouping procedure, taking into account the effect of time to avoid revealing privacy with multiple releases. Rooted in this constraint, we propose an overall method to anonymize all these time-series social network graphs at distinct timestamps at one time. We use two real datasets to test our method and the experiment results demonstrate that our method is very effective in terms of data utility for query answering.

The remainder of this paper is organized as follows. The preliminaries of this paper are introduced in Section 2. We describe the details on anonymizing a single social network graph proposed in [1] and point out that if we apply it to deal with the time-series social network graphs, the privacy guarantees may fail. Our approach is detailed in Section 3 and after that, the experiment results are presented in Section 4. Finally, Section 5 concludes this work

2 Preliminaries

We consider a privacy preserving problem on time-series social network graphs in this paper. A time-series of social network graphs denoted $g = <G_0, G_1, \ldots, G_T>$ represent the evolution of a specific social network. The social network graph at time t is denoted $G_t = (V_t, E_t, L_t)$, where V_t is a set of vertices representing users at time t, E_t is a set of edges representing the interaction among users at time t, and L_t is a set of labels, each of which is used to descript a specific user. We assume that g is *incremental*, i.e. the vertices and edges are only added to but not deleted from the graph in the next timestamp. Accordingly, $V_{t-1} \subseteq V_t$, $E_{t-1} \subseteq E_t$, and $L_{t-1} \subseteq L_t$, where t = 1 to T. Obviously, G_T is equal to the union of $G_0, G_1, \ldots, G_T$. Figure 1 shows two snapshots of a time-serial social network graph at $t = 0$ and $t = 1$. The corresponding vertex of a user is associated with a label and each label has a number of attributes, e.g. age, gender, and location, to descript the user.



(*a*) $t = 0$ (*b*) $t = 1$

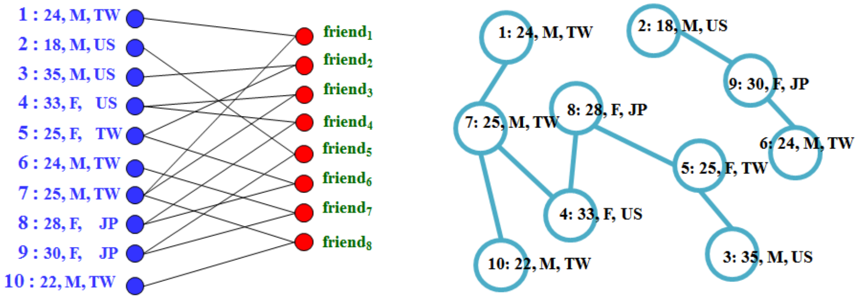
Fig. 1. Snapshots of a time-series of social network graphs at $t = 0$ and $t = 1$

Problem Statement. Given a time-series of social network graphs $g = <G_0, G_1, \ldots, G_T>$ and a constant k, we want to release the anonymized graphs $<G_0', G_1', \ldots, G_T'>$, each of which should follow the guarantees below:

1. For any edge e in an anonymized graph, an attacker who has no background knowledge about the original graph can correctly guess that a specific user u participates in e with a probability at most equal to $1/k$.
2. For any two users u_x and u_y, an attacker who has no background knowledge about the original graph can correctly guess that these users have interaction with a probability at most equal to $1/k$.

2.1 Single Graph Anonymization

Distinct from [1], only one type of interaction is considered in this paper. Then, the bipartite graph to represent the interaction among users in [1] as shown in Figure 2 (a) is easily transformed into a general social network graph as shown in Figure 2 (b).



(*a*) The interaction graph (one type) (*b*) The general graph

Fig. 2. The transformation of an interaction graph

In order to achieve the above privacy objectives, a *label list* denoted $l(v)$ is used in [1] to replace the true label of a node v and moreover, the true label of v must be contained in $l(v)$. Label lists are generated by dividing all nodes into classes with a size equal to k. Nodes in the same class have the same label list. Accordingly, an attacker has a probability of $1/k$ to guess the correct label of a node. For example, suppose $k = 2$. The nodes shown in Figure 2(b) are divided into classes with a size equal to 2. We then get five classes including $A = \{1, 2\}$, $B = \{3, 4\}$, $C = \{5, 6\}$, $D = \{7, 8\}$, and $E = \{9, 10\}$. Then, each node is assigned a label list according to its corresponding class. Figure 3 shows the anonymized graph.

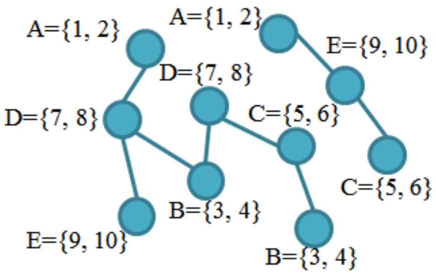


Fig. 3. The full list anonymized graph at k = 2

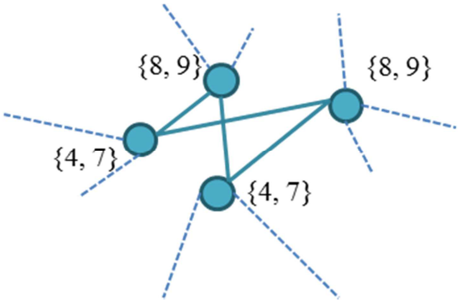


Fig. 4. An example of the dense links between two classes at k = 2

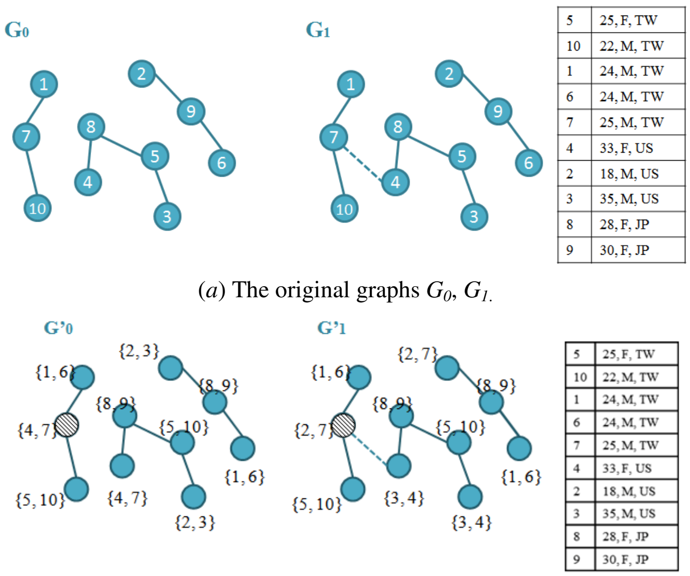
Merely partitioning the nodes into classes cannot achieve the privacy objectives. If the links among nodes in a same class are dense, an attacker can imply with a high probability that a certain link must exist. For example, suppose user 1 and user 7 in Figure 2(b) are grouped in the same class at $k = 2$. Then, an attacker can be sure that user 1 and user 7 have interaction, without recognizing the true labels related to the nodes. Moreover, the links between two classes should not be dense. For example, the interaction between two classes is dense as shown in Figure 4, and an attacker can exactly know user 4 and user 7 have the same friends including user 8 and user 9. Bhagat et al. propose the *class safety condition* in [1], defined as follows, to avoid the above attacks.

Definition 1. *Class Safety Condition [1]: Division of nodes V into classes satisfies the Class Safety Condition if any node* $v \in V$ *and any class* $C \subset V$ *follow 1)* $\forall(v, w)$ *and* $(v, z) \in E$*: if* $w \in C \wedge z \in C \Rightarrow w = z$ *and 2)* $\forall(v, w) \in E$*: if* $v \in C \wedge w \in C \Rightarrow v = w$.

A simple greedy approach for partitioning nodes into classes is proposed in [1]. To improve the utility of the anonymized graph, they consider sorting the attributes of nodes according to their importance to queries and then follow the sorted *attribute priority list* in the division procedure to make sure that nodes with similar attributes are divided into the same or nearby classes under the condition of observing the class safety condition. Following [1], we assume that the attribute priority list is given.

2.2 Privacy Revealed across Multiple Releases

To solve the problem addressed in this paper, a naïve solution based on [1] is descripted as follows.



5	25, F, TW
10	22, M, TW
1	24, M, TW
6	24, M, TW
7	25, M, TW
4	33, F, US
2	18, M, US
3	35, M, US
8	28, F, JP
9	30, F, JP

(*a*) The original graphs G_0, G_1.

5	25, F, TW
10	22, M, TW
1	24, M, TW
6	24, M, TW
7	25, M, TW
4	33, F, US
2	18, M, US
3	35, M, US
8	28, F, JP
9	30, F, JP

(*b*) The released graphs G'_0, G'_1.

Fig. 5. An example of privacy revealed across multiple releases at $k = 2$

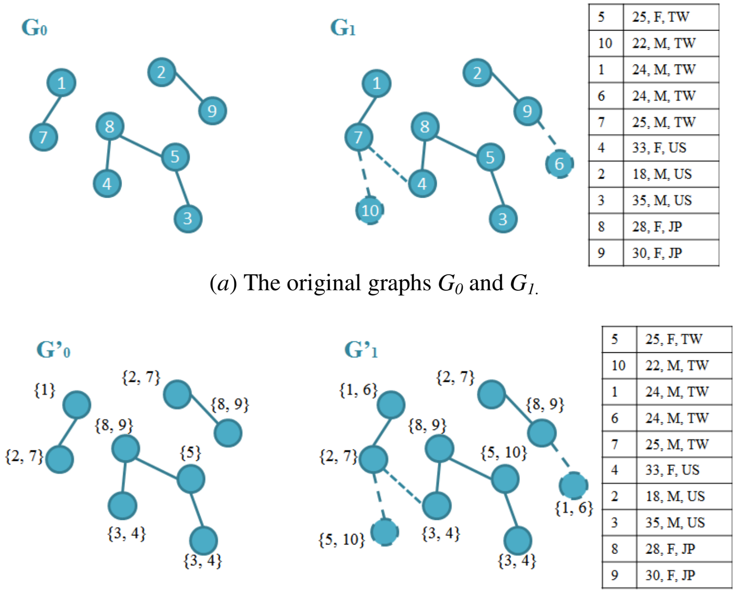
Naïve Solution. G_0, G_1, …, G_T are individually anonymized using the approach proposed in [1]. Since the classes may be different at distinct timestamp, the label list of v may change at distinct timestamp.

For example, given two snapshots of a time-serial social network graph, G_0 and G_1, at $t = 0$ and 1 respectively, a constant $k = 2$, and the sorted nodes according to the attribute priority list <location, gender, age> as input of the approach proposed in [1], G_0' and G_1' are generated as shown in Figure 5(b). Let us focus on the grey node v in G_0' and G_1', the label lists of v at two timestamps are $l_0(v) = \{4, 7\}$ and $l_1(v) = \{2, 7\}$, respectively. Since we assume the evolution of a specific social network graph is incremental and only the grey node links both to {1, 6} and {5, 10} in G_0' and G_1', the true identity of v must be 7. Obviously, it violates the privacy purposes.

Observation 1. Given a time-series of social network graphs $g = <G_0, G_1, …, G_T>$, a constant k, and an attribute priority list denoted $list_{ap}$, we first anonymize G_T to generate G_T' and in the next step, we generate G_{t-1}' by removing the edges and nodes arriving at t from G_t' for $t = 1$ to T at each iteration. In this case, the privacy may be revealed when the nodes are removed since the size of the label list containing the removed entities may be less than k.

For example, given two snapshots of a time-serial social network graph, G_0 and G_1, at $t = 0$ and 1 respectively, a constant $k = 2$, and the sorted nodes according to the attribute priority list <location, gender, age> as input shown in Figure 6(a), we first anonymize G_1 to generate G_1' using the approach in [1] as shown in Figure 6(*b*). Next,

G_0' is generated by deleting the edges and nodes arriving at $t = 1$ from G_1'; furthermore, the corresponding labels of the removed nodes should be deleted from the label lists as shown in Figure 6(*b*). Obviously, the identity of node e.g. user 1, is revealed. To solve the problem of revealing the identities of nodes, we may put the nodes arriving at the same timestamp into the same classes. Therefore, the nodes arriving at the same time and their corresponding classes will be deleted together. The main idea of our solution extended from Observation 1 is detailed in the following section.



5	25, F, TW
10	22, M, TW
1	24, M, TW
6	24, M, TW
7	25, M, TW
4	33, F, US
2	18, M, US
3	35, M, US
8	28, F, JP
9	30, F, JP

(*a*) The original graphs G_0 and G_1.

5	25, F, TW
10	22, M, TW
1	24, M, TW
6	24, M, TW
7	25, M, TW
4	33, F, US
2	18, M, US
3	35, M, US
8	28, F, JP
9	30, F, JP

(*b*) The anonymized graphs G'_0 and G'_1.

Fig. 6. An example of privacy revealed in Observation 1 at $k = 2$

3 The Anonymizing Method

Our solution to anonymizing a time-series of social network graphs is detailed in this section. The definition of the *Time-Series Class Safety Condition* (TSCSC) used in our solution is described in Subsection 3.1 and then, we present the main algorithm in Subsection 3.2 and discuss the security of our solution in Subsection 3.3.

3.1 Time-Series Class Safety Condition

We modify the class safety condition in Definition 1 for the time-series social network graphs to obtain the *Time-Series Class Safety Condition* (TSCSC). Dividing nodes into classes that satisfy TSCSC can achieve the privacy objectives mentioned. How TSCSC to guarantee the privacy objectives will be discussed later.

Definition 2. *Time-Series Class Safety Condition: Division of nodes V_t into classes satisfies the Time-Series Class Safety Condition if any node $v \in V_t$ and any class $C \subset V_t$ follow 1) $\forall v \in V_t$ and $w \in V_{t'}$: if $v \in C \wedge w \in C \Rightarrow t = t'$, 2) $\forall (v, w) \in E_t$: if $v \in C \wedge w \in C \Rightarrow v = w$, and 3) $\forall C_a$ and $C_b \subset V_t$, n_e is the number of edges between C_a and $C_b \Rightarrow n_e \leq k$.*

The first condition indicates that the nodes in a same class arrive at the same time. The second condition similar to Definition1 constraints that at each timestamp, no edges exist in a class. The third condition constraints the number of interaction between any pairs of classes at each timestamp.

3.2 The Anonymizing Method for Time-Series Social Network Graphs

Our greedy algorithm name *DMRA* (***D****ecreasing* ***M****ultiple* ***R****eleases* ***A****nonymization*) for anonymizing the time-series social network graphs is descripted in this subsection. Given a sequence of time-series social network graphs $g = <G_0, G_1, \ldots, G_T>$, a constant k, and an attribute priority list denoted $list_{ap}$:

Step 1. We sort all vertices in G_T according to $list_{ap}$ to generate an order list of vertices, V_{list}, in which the vertices with similar attributes are nearby. Then, we start to anonymize G_T.

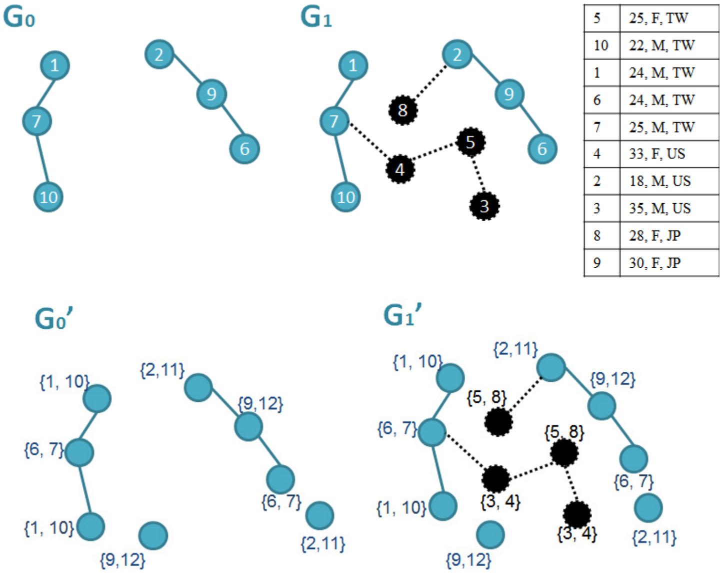
Step 2. Initially, no class exists; we thus create a new class only containing the first vertex in V_{list}. Then, for each $v \in V_{list}$, we sequentially insert the node v into the first fit class which contains the nodes with a number smaller than k and moreover, the insertion must satisfy the Time-Series Class Safety Condition. If we cannot find a class with a size smaller than k or observing TSCSC after considering v, a new class is created.

Step 3. After Step 2, some classes may not have k nodes. To reduce the number of classes with a size smaller than k, the classes need refinement. Since this is a heuristic method, it is possible that partitioning nodes into classes fails while interaction is dense. However, the social network usually follows *Power Law Distribution*, making many nodes with low degrees. This can be effectively used to reduce the number of classes with a size smaller than k, also mentioned [1]. For each class with a size smaller than k, we check whether it can be merged with another class with a size smaller than k (observing TSCSC), if yes, we merge the two classes to reduce the number of classes with a size smaller than k.

Step 4. Now, we add *dummy nodes* to those classes still with a size smaller than k. The attributes of a dummy node are decided by randomly picking from the attributes of the nodes of the class which it belongs to. For example, suppose Class A = {2, 7} and k = 3, user 2 is with attributes = {18, M, US} and user 7 is with attributes = {25, M, TW}. We add a dummy node corresponding to user 11 to Class A and its attributes will be either {18, M, US} or {25, M, TW}. Finally, each class has k nodes and we assign the corresponding label list to each node according to its class to get G_T'.

Step 5. After Step 4, G_T' is generated. Then, we can generate G_{t-1}' by removing all vertices $\in V_t \setminus V_{t-1}$ and all edges $\in E_t \setminus E_{t-1}$ from G_t' for $t = 1$ to T.

As shown in Figure 7, we sort all vertices in G_1 according to $list_{ap}$ = <location, gender, age> to generate V_{list}. Following Steps 2 and 3, the nodes are divided into A = {5, 8}, B = {1, 10}, C = {6, 7}, D = {3, 4}, E = {2}, and F = {9}. Since Classes E and F are both with a size smaller than k, dummy nodes are added to them. E becomes {2, 11} and F becomes {9, 12}. Next, we assign the corresponding label list to each node according to its class to get G_1'. Finally, we generate G_0' by removing the vertices and edges belong to $V_1 \setminus V_0$ and $E_1 \setminus E_0$ from G_1', respectively.



5	25, F, TW
10	22, M, TW
1	24, M, TW
6	24, M, TW
7	25, M, TW
4	33, F, US
2	18, M, US
3	35, M, US
8	28, F, JP
9	30, F, JP

Fig. 7. An illustration of the running DMRA at $k = 2$

3.3 The Security of Time-Series Class Safety Condition

Three conditions for ensuring the privacy objectives of our method on multiple releases for time-series social network graphs are descripted. The first condition ($\forall$ $v \in V_t$ *and* $w \in V_{t'}$: *if* $v \in C \wedge w \in C \Rightarrow t = t'$) indicates that nodes in the same class arrive at the same time in each anonymized graph. As a result, DMRA ensures that the deleted nodes are in the same classes, thus making each class to have k members.

The second condition ($\forall(v, w) \in E_t$: *if* $v \in C \wedge w \in C \Rightarrow v = w$) constraints no edges exist in a class at a timestamp. Accordingly, if there is an edge between two nodes in the anonymized graph at timestamp t, the true labels of the two nodes must belong to different classes. Then, to an edge, there will be k candidate labels for both endpoints.

The third condition ($\forall$ C_a and $C_b \subset V_t$, n_e is the number of edges between C_a and $C_b \Rightarrow n_e \leq k$) ensures the number of interaction between any pairs of classes at each timestamp to less than or equal to k. Given two classes C_a and C_b at the same timestamp, suppose that an entity u_x is in C_a and an entity u_y is in C_b, the probability of guessing these two entities u_x and u_y having interaction can be formulated as following:

$$p\left(e\left(u_x, u_y\right)\right) = \frac{(|C_a|-1)! \times (|C_b|-1)! \times n_e}{|C_a|! \times |C_b|!} \tag{1}$$

Since $|C_a|$ and $|C_b|$ are both equal to k and $p(e(u_x, u_y))$ should be less than or equal to $1/k$, we can imply $n_e \leq k$.

4 Experiments

How to evaluate the utility of the anonymized social network graphs is an important issue. Some researchers [1] [9] [11] conduct aggregation queries on the anonymized social network graphs. In this paper, we use two kinds of queries including the *single hop queries* and *two hops queries* (also used in [1] [9] [11]) to evaluate the utility of the anonymized time-series graphs The formal description of the single hop queries is as follows: How much interaction between the user with one specific attribute and another user with another specific attribute at a time period. For example, how much friendship between the users located in United States and the users located in Japan at the measurement period? The two hops queries involve three user attributes. For example, how much friendship satisfying that Americans have friendship with Japanese, who also have friendship with Chinese at the measurement period? We measure the utility using *average relative error* [1] [9] [11]). The relative error is equal to $|d - d'| / |d|$, where d and d' are the results of querying on the original graphs and on the anonymized graphs, respectively. Since we do not know the true label of each node in the anonymized graphs, how to perform the queries on the anonymized graphs is an issue. We use the Sampling Consistent Graphs method [1] to randomly sample a graph that is consistent with the anonymized graph. The query is performed on the sampled graph.

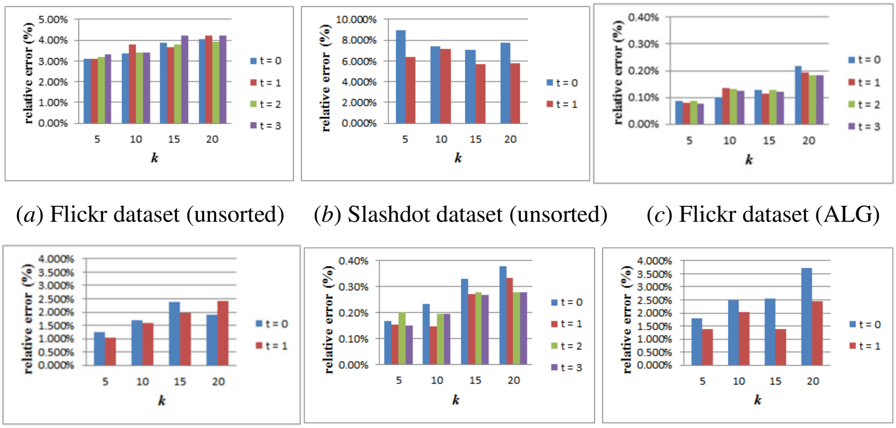
Table 1. The Flickr dataset

t	Timestamp	Nodes	Edges	Nodes added	Edges added
0	Dec 3 ‘06	1,277,145	6,042,807	1,277,145	6,042,807
1	Mar 3 ’07	1,572,674,	8,374,733	295,529	2,331,926
2	Apr 3 ’07	1,712,227	9,166,282	139,553	791,549
3	May 18 ‘07	1,856,342	10,301,741	144,115	1,135,459
Total		1,856,342	10,301,741	1,856,342	10,301,741

Table 2. The Slashdot dataset

t	Timestamp	Nodes	Edges	Nodes added	Edges added
0	Nov 6 ‘08	70,668	358,981	70,668	358,981
1	Feb 1 ’09	79,940	723,428	9,272	364,447
Total		79,940	723,428	79,940	723,428

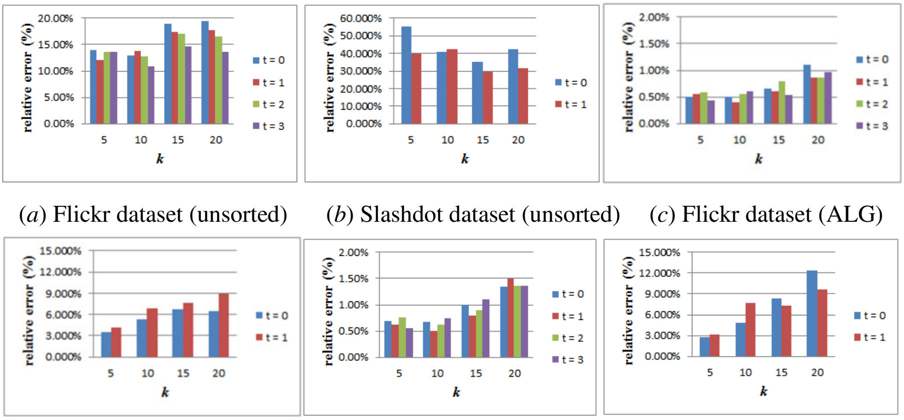
Two real datasets Flickr [7] and Slashdot [5] with synthetic labels are used to test our solution. Flickr was daily crawled from the Flickr network between November 2nd, 2006 and December 3rd, 2006, and again between February 3rd, 2007 and May 18th, 2007. This dataset has a total number of nodes and edges about 1.8M and 10M, respectively. We separate the dataset into four partitions to simulate different timestamps. Slashdot is a technology-related news website. The dataset was collected in November 6th, 2008 and February 1st, 2009, which consists of 79K nodes and 723K edges. We separate this dataset into two partitions to simulate two timestamps. Because the original datasets have no labels, we generate labels containing three attributes, age (10~60), gender (male/female) and location (50 countries) for each entity and all values are within the uniform distribution. Our algorithm is implemented in C++ and performed on a PC with the Intel Core 2 Quad 2.66GHz CPU, 8GB memory, and under the Ubuntu v11.04 64bits operating system.



(*a*) Flickr dataset (unsorted) (*b*) Slashdot dataset (unsorted) (*c*) Flickr dataset (ALG)

(*d*) Slashdot dataset (ALG) (*e*) Flickr dataset (GLA) (*f*) Slashdot dataset (GLA)

Fig. 8. The average relative errors on single hop queries

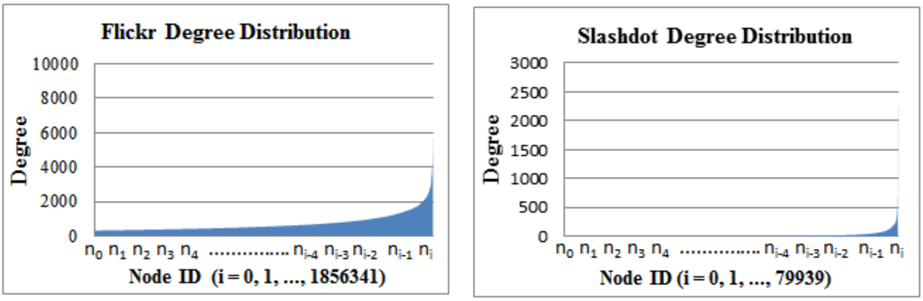


(*a*) Flickr dataset (unsorted) (*b*) Slashdot dataset (unsorted) (*c*) Flickr dataset (ALG)

(*d*) Slashdot dataset (ALG) (*e*) Flickr dataset (GLA) (*f*) Slashdot dataset (GLA)

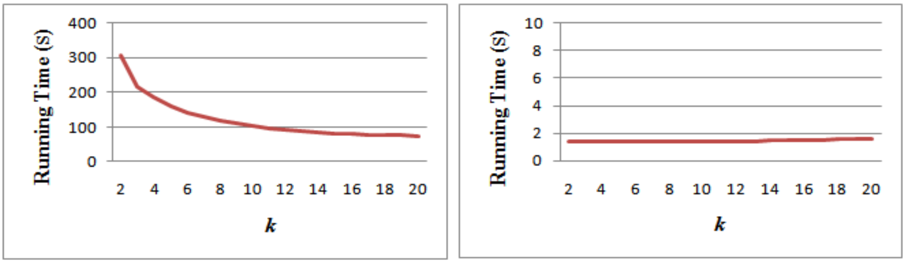
Fig. 9. The average relative errors on two hops queries

We consider 50 queries and three sorting methods including unsorted, $list_{ap}$ = <**A**ge, **L**ocation, **G**ender>, and $list_{ap}$ = <**G**ender, **L**ocation, **A**ge> in the experiments. The experiment results regarding single hop queries and two hops queries are shown in Figures 8 and 9, respectively. Obviously, since the larger k leads to the more possible labels for each entity, the average relative error rate increases as the increasing of k. The benefits of sorting vertices before partitioning them into classes are shown in Figures 8 and 9. As can be seen, the unsorted cases have the higher average relative error rates in either Figures 8 or 9. Usually, the results on single hop queries are better than those on two hops queries since the two hops queries involve much interaction. The degree distributions of the two datasets are shown in Figure 10, which indicate that the degree distributions of the two datasets follow the power law distribution. Accordingly, most of the nodes are quickly grouped together since there are many nodes with low degrees in the datasets and therefore, we only need to add few dummy nodes in most of the cases.



(*a*) The distribution of Flickr (*b*) The distribution of Slashdot

Fig. 10. The degree distributions of the two datasets



(*a*) Flickr dataset (AGL) (*b*) Slashdot dataset (AGL)

Fig. 11. The running time of DMRA

The running time of DMRA is shown in Figure 11, which decreases with the increasing of k, since the number of classes will decrease with the increasing of k, making the comparisons on checking *Time-Series Class Safety Condition* reduced. Either single hop queries or two hops queries only consider the interaction among entities with different user attributes. Since dummy nodes do not have any edges in the released graphs, adding dummy nodes does not change the total number of interactions and does not seriously affect the results of queries.

5 Conclusions

In this paper, we address a new problem on privacy preserving for releasing multiple time-series social network graphs. Naively applying the existing approach to each time-series graph will break the privacy purposes. For achieving the privacy purposes, we propose Time-Series Class Safety Condition and DMRA for releasing multiple anonymized graphs at one time. The experiments demonstrate that DMRA is very effective in terms of data utility. Moreover, if we know which attributes are more important and often used in the queries in advance and follow the sorted vertex ordering in our anonymizing algorithm, the relative error rate will be reduced.

References

1. Bhagat, S., Cormode, G., Krishnamurthy, B., Srivastava, D.: Class-Based Graph Anonymization for Social Network Data. In: Proceedings of the 35th International Conference on Very Large Data Base, pp. 766–777 (2009)
2. Backstrom, L., Dwork, C., Kleinberg, J.: Wherefore Art Thou R3579X*?* Anonymized Social Networks, Hidden Patterns, and Structural Steganography. In: Proceedings of the 16th International Conference on World Wide Web, pp. 181–190 (2007)
3. Cormode, G., Srivastava, D., Yu, T., Zhang, Q.: Anonymizing Bipartite Graph Data Using Safe Groupings. In: Proceedings of the 34th International Conference on Very Large Data Base, pp. 833–844 (2008)
4. Liu, K., Das, K., Grandison, T., Kargupta, H.: Privacy-Preserving Data Analysis on Graphs and Social Networks. In: Kargupta, H., Han, J., Yu, P., Motwani, R., Kumar, V. (eds.) Next Generation Data Mining, ch. 21, pp. 419–437. CRC Press (December 2008)
5. Leskovec, J., Lang, K., Dasgupta, A., Mahoney, M.: Community Structure in Large Networks: Natural Cluster Sizes and the Absence of Large Well-Defined Clusters. University of Massachusetts Technical Report, Internet Mathematics 6(1), 29–123 (2009)
6. Liu, K., Terzi, E.: Towards Identity Anonymization on Graphs. In: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, pp. 93–106 (2008)
7. Mislove, A., Koppula, H.S., Gummadi, K.P., Druschel, P., Bhattacharjee, B.: Growth of the Flickr Social Network. In: Proceedings of the 1th ACM SIGCOMM Workshop on Online Social Networks, pp. 25–30 (2008)
8. Wu, X., Ying, X., Liu, K., Chen, L.: A Survey of Privacy-Preservation of Graphs and Social Networks. In: Aggarwal, C.C., Wang, H. (eds.) Managing and Mining Graph Data, vol. 40, pp. 421–453. Springer US (2010)
9. Yuan, M., Chen, L., Yu, P.: Personalized Privacy Protection in Social Networks. In: Proceedings of the 37th International Conference on Very Large Data Base, pp. 141–150 (2010)
10. Zou, L., Chen, L., Ozsu, M.T.: K-automorphism: A General Framework for Privacy Preserving Network Publication. In: Proceedings of the 35th International Conference on Very Large Data Base, pp. 946–957 (2009)
11. Zhou, B., Pei, J.: Preserving Privacy in Social Networks against Neighborhood Attacks. In: Proceedings of the 24th IEEE International Conference on Data Engineering, pp. 506–515 (2008)
12. Zhou, B., Pei, J., Luk, W.-S.: A Brief Survey on Anonymization Techniques for Privacy Preserving Publishing of Social Network Data. In: Proceedings of the SIGKDD Explorations, pp. 12–22 (2008)

A New, Fast and Accurate Algorithm for Hierarchical Clustering on Euclidean Distances

Elio Masciari[1], Giuseppe Massimiliano Mazzeo[1], and Carlo Zaniolo[2]

[1] ICAR-CNR
[2] UCLA
{masciari,mazzeo}@icar.cnr.it, zaniolo@cs.ucla.edu

Abstract. A simple hierarchical clustering algorithm called CLUBS (for CLustering Using Binary Splitting) is proposed. CLUBS is faster and more accurate than existing algorithms, including k-means and its recently proposed refinements. The algorithm consists of a divisive phase and an agglomerative phase; during these two phases, the samples are repartitioned using a least quadratic distance criterion possessing unique analytical properties that we exploit to achieve a very fast computation. CLUBS derives good clusters without requiring input from users, and it is robust and impervious to noise, while providing better speed and accuracy than methods, such as BIRCH, that are endowed with the same critical properties.

1 Introduction

The clustering challenge. Cluster analysis represents a fundamental and widely used method of knowledge discovery, for which many approaches and algorithms have been proposed over the years [9]. Among the most popular methods, we find partition-based clustering (e.g. *k-means*[10]), density based clustering (e.g. *DBScan*[6]), hierarchical methods (e.g. *BIRCH*[14]) and grid-based methods (e.g. *STING* [13]). The continuous stream of clustering algorithms proposed over the years underscores the fact that the logical and algorithmic complexities of this many-facet problem have yet to be tamed completely, and that along with the great progress achieved in the past, significant progress should be expected in the future. In particular, it is well known that no clustering algorithm completely satisfies both accuracy and efficiency requirements, thus a good clustering algorithm has to be evaluated w.r.t. some external criteria that are independent from the metric being used to compute clusters. Indeed, in this paper we propose an algorithm that significantly improves the state of the art in clustering analysis, with respect to *speed, repeatability,* and *accuracy* whose performances have been evaluated using widely accepted clustering validity metric.

Current Solutions: Who is the best in terms of speed and accuracy? In order to compare our performances we preliminary tested existing clustering solutions. We chose algorithms that are widely used by data miners due to their general purpose nature. More in detail, we evaluated algorithms that satisfy user needs in a wide variety of application scenarios. In terms of *speed* of computation, the standard of paragon is set by the k-means [10] algorithm that has as objective minimizing the average distance of the samples from their cluster centroids. Owning to its efficiency, simplicity, and the

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 111–122, 2013.

naturalness of its objective, k-means has become the most widely used clustering algorithm in practical applications. But k-means also suffers from serious shortcomings: in particular, the algorithm does not assure *repeatability* of results, which instead depends on the choice of the initial k points. In practice, therefore, the data mining analyst will have to select the value of k, and the initial k points, via some exploratory dry runs. k-means' shortcomings have motivated much research work [11,2,4], seeking to reduce the variability of the results it produces and bring it closer to the 'unsupervised learning' archetype. A second line of research has instead produced different mining algorithms to overcome k-means' problem while keeping with its general objective of clustering the points around centroids. Approaches such as grid-based clustering work by partitioning the data space into cells arranged in a grid and then merging them to build clusters, while density based approaches search for fully-connected dense regions. Finally, hierarchical clustering methods work by performing a hierarchical decomposition of data in either bottom-up (agglomerative) or top-down (divisive) way. In this respect hierarchical approaches offer good performances w.r.t. the accuracy of clustering. All the techniques mentioned here present advantages and weakness that will be discussed in detail in the related work section. For the goal of assessing the quality of our approach, it is worth noticing that the hierarchical clustering algorithm BIRCH [14] is *stable* (i.e. its results do not vary depending on some initial parameter setting), *accurate*, *impervious to noise* and *scalable to very large data sets*. However, BIRCH is typically not as fast as k-means.

Our Solution. The above thought-provoking discussion guided our search for a new clustering algorithm. Indeed, in this paper we propose a new hierarchical algorithm called CLUBS (for CLustering Using Binary Splitting) whose *speed* performances are better than k-means *and* whose *accuracy* overcomes previous hierarchical algorithms while operating in a *completely unsupervised* fashion. The first phase of the algorithm is divisive, as the original data set is split recursively into miniclusters through successive binary splits: the algorithm's second phase is agglomerative since these miniclusters are recombined into the final result. Due to its features our algorithm can be used also for refining other approaches performances. As an example it can be used to overcome k-means initial assignment problem since its low complexity will not affect the overall complexity while the accuracy of our results will guarantee an excellent initial assignment of cluster centroids. Further, our approach induces during execution a *dynamic hierarchical* grid that will better fit the dataset w.r.t. classical grid approaches that exploit a fixed grid instead. Finally, the algorithm exploits the analytical properties of the Quadratic Sums of Squares (SSQ in the following) function to minimize the cost of merge and split operations, and indeed the approach results really fast. One may argue that many different measures could be used for cluster computation but the accuracy of SSQ is as good as other cluster distance measures (e.g. Single Link, Complete Link, Average) for real case scenarios *and* its computation can be made faster than other measures. These properties are discussed in Section 2.

Main Difference of CLUBS w.r.t. other approaches. CLUBS works in a completely unsupervised way and overcomes the main limitations that beset other algorithms. In particular, we have that (1) CLUBS is not tied to a fixed grid, (2) it can backtrack on previously wrong calculation, and (3) it performs also well on non-globular clusters where clusters are not spherical in shape, this feature will be intuitively understood after the partitioning and recombination strategy will be detailed in Section 3 (BIRCH does not perform as well, because it uses the notion of radius or diameter to control the boundary of a cluster, and the same drawback also affects k-means like algorithms). Moreover we have that (4) CLUBS can detect the natural clusters present in data, while in Birch each node in the auxiliary tree exploited (called CF tree) can hold only a limited number of entries due to its size thus a CF tree node does not always correspond to what a user may consider a natural cluster. Finally, (5) density based algorithms like DBSCAN are very sensitive to clustering parameters like Minimum Neighborhood Points and they fail to identify clusters if density varies and if the data set is too sparse and different sampling affects density measures, however we compared CLUBS against OPTICS that allows to detect clusters with different densities instead. As will be clear by experimental evaluation, CLUBS does not suffer these limitations due to unique features of SSQ and the two-phase algorithm.

2 Background

After recalling some basic notions used in our algorithm, we discuss binary partitioning and the cluster quality measures there used. Throughout the paper, for each dataset a d-dimensional data distribution D is assumed. D will be treated as a multi-dimensional array of integers with volume n^d (without loss of generality, we assume that all dimensions of D have the same size). The number of non-zero elements of D will be denoted as N. A *range* ρ_i on the i-th dimension of D is an interval $[l..u]$, such that $1 \leq l \leq u \leq n$. Boundaries l and u of ρ_i are denoted by $lb(\rho_i)$ (*lower bound*) and $ub(\rho_i)$ (*upper bound*), respectively. The size of ρ_i will be denoted as $size(\rho_i) = ub(\rho_i) - lb(\rho_i) + 1$. A *block* b (of D) is a d-tuple $\langle \rho_1, \ldots, \rho_d \rangle$ where ρ_i is a range on the dimension i, for each $1 \leq i \leq d$. Informally, a block represents a "hyper-rectangular" region of D. A block b of D with all zero elements is said to be a *null block*. The volume of a block $b = \langle \rho_1, \ldots, \rho_d \rangle$ is given by $size(\rho_1) \times \ldots \times size(\rho_d)$ and will be denoted as $vol(b)$. Given a point in the multidimensional space $\mathbf{x} = \langle x_1, \ldots, x_d \rangle$, we say that $\mathbf{x}$ belongs to the block b (written $\mathbf{x} \in b$) if $lb(\rho_i) \leq x_i \leq ub(\rho_i)$ for each $i \in [1..d]$.

Given a block $b = \langle \rho_1, \ldots, \rho_d \rangle$, let x be a coordinate on the i-th dimension of b such that $lb(\rho_i) \leq x < ub(\rho_i)$. Coordinate x divides the range ρ_i of b into $\rho_i^{low} = [lb(\rho_i)..x]$ and $\rho_i^{high} = [(x+1)..ub(\rho_i)]$, thus partitioning b into $b^{low} = \langle \rho_1, \ldots, \rho_i^{low}, \ldots, \rho_d \rangle$ and $b^{high} = \langle \rho_1, \ldots, \rho_i^{high}, \ldots, \rho_d \rangle$. The pair $\langle b^{low}, b^{high} \rangle$ is said to be the *binary split* of b along the dimension i at the position x; dimension i and coordinate x are said to be the *splitting dimension* and the *splitting position*, respectively.

Informally, a binary partition can be obtained by performing a binary split on D (thus generating the two sub-blocks D^{low} and D^{high}), and then recursively partitioning these two sub-blocks with the same binary hierarchical scheme.

Definition 1. *Given a d-dimensional data distribution D with volume n^d, a* binary partition *BP of D is a binary tree such that the root of BP is the block $\langle [1..n], \ldots, [1..n] \rangle$ and for each internal node p of BP the pair of children of p is a binary-split of p.* □

Clustering Computation Preliminaries. Given a dataset $\mathcal{DS}$ cluster analysis aims at producing a clustering $C = \{C_1, \cdots, C_n\}$ that is a subset of the set of all subsets of $\mathcal{DS}$ such that C contains disjoint (non-overlapping) subsets, covering the whole object set (we refer in this paper exclusively to hard clustering problem, where every data point belongs to one and only one cluster). Consequently, every point $x \in \mathcal{DS}$ is contained in exactly one and only one set C_i. These sets C_i are called clusters.

Definition 2. *Let C_s be a cluster (set) of N d-dimensional points. Let $\mathbf{S} = (S_1, \ldots, S_d) = \sum_{\mathbf{p} \in C_s} \mathbf{p}$ be the vector representing the sum of points in C_s. The center of C_s is $\mathbf{C^0_s} = \frac{\mathbf{S}}{N}$. Let $\mathbf{Q} = (Q_1, \ldots, Q_d)$, where $Q_i = \sum_{\mathbf{p} \in C} p_i^2$, be the vector whose i-th coordinate is the sum of the squared i-th coordinates of the points in S. The SSQ (Sum of Squares) of C_s is defined as:*

$$SSQ(C_s) = \sum_{\mathbf{p} \in C_s} dist^2(\mathbf{p}, \mathbf{C^0_s}) = \sum_{\mathbf{p} \in C} \sum_{i=1}^{d} (p_i - C_s^0)^2 =$$
$$\sum_{i=1}^{d} \sum_{\mathbf{p} \in C} (p_i - C_s^0)^2 = \sum_{i=1}^{d} \sum_{\mathbf{p} \in C} (p_i^2 - 2 \cdot p_i \cdot C_s^0 + (C_s^0)^2) =$$
$$\sum_{i=1}^{d} \sum_{\mathbf{p} \in C} p_i^2 - 2 \cdot C_s^0 \cdot \sum_{\mathbf{p} \in C} p_i + N \cdot (C_s^0)^2)$$

we recall that N is the number of points in C and

$$\sum_{\mathbf{p} \in C} p_i = C_s^0 \cdot N$$

thus we obtain by substituting:

$$\sum_{i=1}^{d} \sum_{\mathbf{p} \in C} p_i^2 - \sum_{i=1}^{d} \frac{(\sum_{\mathbf{p} \in C} p_i)^2}{N}$$

finally by definition of Q_i and S_i we obtain:

$$SSQ(C_s) = \sum_{i=1}^{d} (Q_i - \frac{S_i^2}{N}) \quad (1)$$

From the latter, it is clear that, in order to quickly compute the SSQ of a cluster, we need only to store $\mathbf{Q}$, $\mathbf{S}$, and N. In the next section we will show how these information can be used effectively and efficiently to optimize the divisive and agglomerative steps of the CLUBS algorithm.

3 CLUBS: A New Clustering Algorithm

In order to obtain a good tradeoff between accuracy and efficiency we exploit in this paper a new really fast hierarchical approach. Among hierarchical algorithms, bottom-up approaches tend to be more accurate but have a higher computational cost than the top-down approaches [9]. The higher cost is due to the higher number of candidate clusters to be taken into account. To overcome this limitation, in our approach, the agglomerative step is only used on mini-clusters generated by a first divisive process, this results in a remarkable efficiency increase. Top-down partitioning exploiting greedy algorithms has been widely used in the multidimensional data compression due to its efficiency. Here we use a similar divisive approach to minimize the SSQ among the data belonging to clusters, we recall again that in literature many measure have been proposed (e.g. EES) that works in a similar way as SSQ but we chose SSQ since it offers a really fast computation while maintaining an high accuracy in cluster model evaluation. Thus, our clustering algorithm consists of two steps, where in the first step we use binary hierarchical partitioning to produce a set of mini-clusters and in the second step, we pairwise merge the mini-clusters so obtained in a bottom-up fashion. In both steps the clusters are defined by a hierarchical partition of the multi-dimensional space. The partition can be compactly represented by a binary tree (BT in the following), where:1) each node is associated with a range of the multi-dimensional domain; 2) the root is associated with the whole data domain; 3)for each inner node n, its children are associated with a pair of ranges representing a (rectangular) partition of n.

Each node also maintains summary information about points inside its range, to expedite the clustering computation. The top-down splitting works as follows. As auxiliary structure, we maintain a priority queue of clusters whose elements are ordered on the basis of the SSQ of each cluster. At each iteration, the algorithm performs the following two steps: A) select the cluster C_s that exhibits the highest SSQ (i.e. the one on top of the priority queue), and then B) partition this C_s in such a way that the overall SSQ reduction, denoted ΔSSQ, is maximized. For step B, we compute $\Delta SSQ(i, j)$ for each dimension i and for each cutting position j; then we choose the position j that guarantees the maximum ΔSSQ. This computation can be done very efficiently since we pre-compute Q and S, and therefore we need a single scan of the data. We repeat these two steps, A and B above, while ΔSSQ is greater than the average SSQ. We recall that the partition (i.e., the cluster tree) is built by exploiting a greedy strategy. To this end, the tree is constructed top-down, by means of leaf-node splitting. At each step, the leaf with the largest SSQ is chosen, and it is split as to maximize ΔSSQ . Being SSQ a measure of a range skewness, we perform splits as long as ΔSSQ remains "significant". After the early splits that yield large SSQ reductions, the values of ΔSSQ become smaller and smaller, until after n splits both SSQ and ΔSSQ become 0 (since each point has become its own cluster). Thus, the average SSQ reduction per split is SSQ_0/n, and we will compare this value against the current ΔSSQ to decide when we should stop splitting, The rationale for this criterion is clearly illustrate by Fig. 1, where the typical ΔSSQ slope is displayed against the average SSQ: there is no gain in splitting beyond the turning point (marked with a solid circle) since the SSQ reduction is less than the average ΔSSQ and thus imputable to random distributions rather than cluster-like ones.

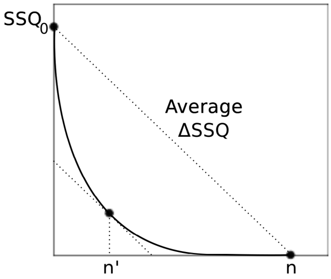


Fig. 1. Average SSQ and ΔSSQ example plots

The splitting process just described is tied to the grid partitioning and thus may cause a non-optimal splitting of some clusters. The successive phase overcomes this limitation since the merging is performed considering all the possible pairs of adjacent mini-clusters, and recombining those that offer best SSQ reduction. This agglomerative process offers significant advantages. One is that it merges clusters in different grid partitions, thus overcoming non optimal splits obtained in the first phase (see Fig. 3(b) and Fig. 3(c)). The second critical advantage is that the computational complexity of this bottom-up step is very low since the number of merging steps is related to the number clusters that is very low compared to usual dataset sizes. The final advantage is that this phase also halts automatically, producing an algorithm that does not require any seeding or other parameters from the user a really nice feature that is not shared by all clustering algorithms.

The Clustering Algorithm. Fig. 2 provides a more formal description of the CLUBS algorithm. We use the *initializeTree* to load the dataset into the root of the auxiliary tree structure BT exploited for partitioning. Once the tree structure has been initialized the *topdownsplitting* step starts. In particular, the root of BT is added to a priority queue whose ordering criterion is based on the SSQ values of clusters stored in the queue. The initial cluster assignment performed by *initializeClusters* is composed by the root r of BT and the initial SSQ is the one computed on r. The function *computeAverageDeltaSSQ* averages the actual SSQ for all the points in the cluster. The function *computeWeightedDeltaSSQ* is applied to the cluster C_s that is currently on top of the priority queue. The $weighted\Delta_{SSQ}$ is computed as the average gain of SSQ obtained by splitting C_s as explained above for Δ_{SSQ}, i.e. we pre-compute the marginal sums (S and Q) for a given splitting point (w.r.t the coordinates ordering) and reassigning the splitting point based on these partial sums. In order to improve the effectiveness of splits the value of Δ_{SSQ} is raised to a power of p, $p < 1$, thus obtaining $weighted\Delta_{SSQ}$ value. If $weighted\Delta_{SSQ}$ is greater than $avgDeltaSSQ$ computed by *computeAverageDeltaSSQ* then we proceed with the split, otherwise we do not. We use values of p that are less than 1, since for $p \geq 1$ we would end up splitting clusters where the gain does not exceed the Average Δ_{SSQ} associated with a random distribution. This would result in a large number of small clusters, where both intra-cluster and inter-cluster distances small. We instead seek values of p that reduce the former while magnifying the latter. We determined experimentally that the best value is $p = 0.8$ regardless the dataset feature thus the user is not required to set any parameter, due to space limitations we cannot report here the detailed discussion of the experiments being conducted.

When no more top-down splits are possible, the *topDownSplitting* ends and we begin the *bottomUpMerging*. In order to obtain more compact clusters, we select (by running *selectBestPair*) the pair of clusters that, if merged, yields the least SSQ increase (that is assigned to $minInc$ by function *computeSSQIncrease*). This merging step is repeated until $minInc$ becomes larger than $avgDeltaSSQ$. Fig. 3 shows the algorithm in action. After three steps, the initial samples in 3(a) are partitioned according to the grid shown in Fig. 3(b). The algorithm takes seven more splitting steps producing the partition of Fig. 3(c). The merging phase produces the final five clusters that a human will instinctively recognize at a glance Fig. 3(d).

```
**Input:**
A dataset $DS$ of $n$ points
**Output:**
A set of clusters $C$.
**Vars:**
An auxiliary binary tree $BT$;
An initial cluster assignment $C'$.
**Method: CLUBS**
$BT :=$ initializeTree($DS$);
$C' :=$ topDownSplitting($BT$);
$C :=$ bottomUpMerging($C'$);
**return** $C$;

**Function** topDownSplitting($BT$) : $C'$;
**Vars:**
A priority queue $PQ$;
A boolean $finished$;
A double $\Delta_{SSQ}$;
A double $avgDeltaSSQ$;
**Method:**
$PQ :=$ add($BT.root()$);
$C' =$ initializeClusters;
$finished = false$;
$avg\Delta_{SSQ} =$ computeAverageDeltaSSQ();
**while** $!finished$ **do begin**
    $C_s = PQ.get()$;
    $weighted\Delta_{SSQ} =$ computeWeightedDeltaSSQ($C_s$);
    **if** ($weighted\Delta_{SSQ} > avg\Delta_{SSQ}$) **then**
        $C' :=$ update($C'$);
    **else** $finished = true$;
**end while**
**return** $C'$;

**Function** bottomUpMerging($C'$) : $C$;
**Vars:**
A pair of cluster $Pair$;
A double $avgDeltaSSQ$;
A double $minInc$;
**Method:**
$C := C'$;
$Pair :=$ selectBestPair($C'$);
$minInc :=$ computeSSQIncrease($Pair$);
$avgDeltaSSQ =$ computeAverageDeltaSSQ();
**while** $minInc < avgDeltaSSQ$ **do begin**
    $C :=$ merge($Pair$);
    $Pair :=$ selectBestPair($C$);
    $minInc :=$ computeSSQIncrease($Pair$);
**end while**;
**return** $C$;
```

Fig. 2. The CLUBS clustering algorithm

We point out that producing axis parallel cuts is not a limitation, we can still obtain, in our approach, non parallel cuts however this will not improve the performances of the algorithm. Furthermore, also grid based approaches are tied to parallel cuts since they allow more efficient computation without paying any accuracy loss.

In terms of computational complexity, we see that, in order to split, we have to compute the SSQ for each dimension and for each splitting point. Thus, each split has a complexity $O(n \cdot d \cdot l)$ and we perform s splits. The bottom-up step contributes to the overall complexity with a term $O(k^2)$ where k is the number of clusters, since for each cluster we have to consider all the possibly adjacent clusters for merging; but since $k << n$ we can disregard this term. Thus the complexity is as follows:

Proposition 1. *Algorithm CLUBS works in $O(n \cdot d \cdot l \cdot s)$ where n is the number of points, d is the number of dimensions, l is the number of splitting positions for each dimension and s is the number of splits.*

4 Experimental Evaluation

An extensive set of experiments was executed to evaluate the performance of CLUBS. In particular, we compared our method with BIRCH [14], K-means++ [2](we refer to it as KM++) and k*-means [4] (we refer to it as SMART) and OPTICS [1].

Our test suite encompasses a large number of widely used benchmarks over a wide spectrum of different characteristics. Due to space limitations we can present here a small subset of the results, so we choose the really interesting results we obtained on a severe test bench, i.e. microarray data. We used two publicly available dataset on Gene Expression Omnibus Database: a dataset provided by [8], *Dataset 1* hereafter, and a dataset provided by [5], *Dataset 2* hereafter.

As regards *Dataset 1*, authors examined 42 patients by using Affymetrix HU133A (Affymetrix, Santa Clara CA) microarrays. Patients were subdivided in three groups.

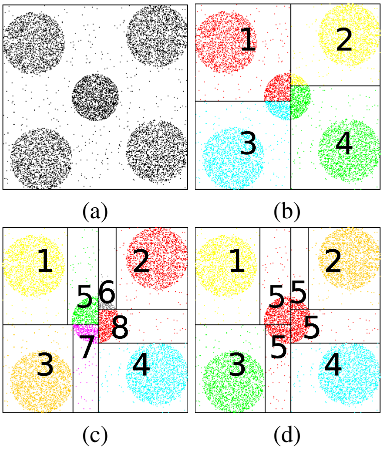


Fig. 3. Execution steps of CLUBS

Women at usual breast cancer risk undergoing mammoplasty reduction (RM) , women with breast cancer undergoing surgery for either an ER+ or ER- breast tumor (HN), and high-risk patients, consisting of women undergoing prophylactic mastectomy (PM). Dataset providers selected 98 differentially expressed genes in HN w.r.t. RM and they built a matrix of these genes for all three groups. The resulting dataset was analyzed by clustering in order to catch the difference among three groups.

Dataset 2 comprises samples extracted from human breast cancer cells analyzed using the Affymetrix U133A 2.0 gene chips (Affymetrix, Santa Clara, CA). Dataset provider considered 4 group of cells treated with 20 lh/ml of actein at 6 and 24 hours, and cells treated with 40 lg/ml of actein at 6 and 24 hours in order to elucidate the effect of actein. The initial preprocessing was performed using the GCRMA method. The statistical significance of differential expression with respect to the same reference value was calculated using the empirical Bayesian LIMMA (LI Model for MicroArrays).

We started our analysis considering these preprocessed datasets on which we used CLUBS and the other clustering algorithms for the sake of comparison. The obtained results are reported in Table 1 where values represent SSQ per dataset and milliseconds.

The results obtained are quite convincing both for the accuracy and the execution times where CLUBS offer best performances. In particular our clustering method correctly detected the number of clusters in the data (3 clusters for Dataset 1 and 4 clusters for Dataset 2). Indeed, CLUBS showed a nice feature when clustering Dataset 1: the HN group contains two subgroups ER+ and ER-, CLUBS during the splitting step identified these two subgroups that have been collapsed in a single cluster after the merging step. To asses, the validity of the approach we exploited several method-independent quality measure that are reported in the following.

Quality of Clustering Results. Here we will evaluate the quality of the results CLUBS produces and its reliability. The issue of finding method-independent measures for clustering results has been the source of much topical discussions, but over time sound measures have emerged that can be used reliably to compare the quality of the results produced by a wide range of clustering algorithms [3]. In particular the following three measures have sound theoretical and practical bases. The *Variance Ratio* measures the ratio between the average distance between points belonging to different clusters and the average distance between points within the same cluster [3]. The range of variance ratio is $[0, \infty)$ and larger values of variance ratio indicate better clustering quality. The *Relative Margin* reports the average of the *Relative Point Margin* defined as the ratio between the distance of a given point x to the center of the cluster it belongs to

Table 1. Accuracy and Time Performances for our test datasets

Algorithm	$Dataset1$		$Dataset2$	
	SSQ	time	SSQ	time
CLUBS	2.01E+8	2.513	1.77E+2	0.0784
OPTICS	3.55E+8	5.271	1.79E+2	0.2456
BIRCH	2.67E+8	9.124	1.78E+2	0.3522
KM++	4.31E+8	2.913	1.76E+2	0.1154
SMART	4.65E+8	3.025	1.81E+2	0.1243

and the distance between x and the closest cluster center different from the cluster x belongs to [3]. The range of relative margin is $[0, 1)$, and lower relative margin indicates a better clustering. The *Weakest Link* measure is defined as the maximal value of weakest link over all pairs of points belonging to the same cluster, divided by the shortest between-cluster distance [3]. The range of values of weakest link is $[0, \infty)$. Lower values of weakest link represent better clusterings. The results obtained for the above mentioned quality measures are given in Table 2(a): they show that CLUBS outperforms other methods significantly, producing values for Relative Margin & Weakest Link (resp. Variance Ratio) that are significantly lower (larger) than those other methods, i.e. clusters of much better quality. These results also confirm that CLUBS finds the exact number of clusters and the quality of the found cluster is overwhelming w.r.t the other methods.

Additional Quality Measures. SSQ is a natural and widely used norm of similarity, but a devil's advocate can point out that other clustering algorithms might not measure their effectiveness in terms of SSQ or even the compactness of each cluster around its centroid. Thus, in this section we will measure the quality of the clusters produced by CLUBS using very different criteria inspired by the nearest subclass classifiers that were previously used in a similar role in [12] and [7].

A first relevant evaluation measure in this approach is the error rate of a k-Nearest Neighbor classifier defined by the clustering results. This value provide relevant information about the ability of the clustering method under evaluation to minimize the errors due to incorrect assignment of points to the proper cluster. Indeed, this information is crucial for biological data analysis. Thus, for each point, we can check whether the dominant class of the k closer elements allows to correctly predict the actual class of membership (there is no relationship between the value of k used here and that of k-means). Thus, the total number of points correctly classified measures the effectiveness

Table 2. Clustering Quality Measures Evaluation

(a)

Dataset 1	#Clusters	Variance Ratio	Relative Margin	Weakest Link
M-CLUBS	3	75.41	0.098	0.817
OPTICS	5	56.18	0.135	2.045
BIRCH	6	63.42	0.176	1.934
KM++	3	65.44	0.157	4.152
SMART	3	64.77	0.198	4.789
Dataset 2	#Clusters	Variance Ratio	Relative Margin	Weakest Link
M-CLUBS	4	81.33	0.066	0.713
OPTICS	4	67.18	0.153	1.876
BIRCH	4	70.41	0.182	1.943
KM++	4	68.67	0.201	3.412
SMART	4	69.97	0.225	3.725

(b)

Dataset 1			
method/index	ε	$e_{k=10}$	$q_{k=10}$
CLUBS	0.0661	0.0984	0.9998
OPTICS	0.1253	0.1976	0.8934
BIRCH	0.1154	0.2010	0.9756
KM++	0.1002	0.1974	0.9803
SMART	0.1086	0.2101	0.9057
Dataset 2			
method/index	ε	$e_{k=10}$	$q_{k=10}$
CLUBS	0.0054	0.0352	0.9999
OPTICS	0.0432	0.1312	0.9875
BIRCH	0.0165	0.0953	0.9923
KM++	0.0487	0.1657	0.9764
SMART	0.0568	0.1789	0.9734

of the clustering at hand. Formally, the error $e_k(D)$ of a k-NN classifier exploiting a the distance matrix among every pair of points. D can be defined as

$$e_k(D) = \frac{1}{N} \sum_{i=1}^{N} \gamma_k(i)$$

where N is the total number of points, and $\gamma_k(i)$ is 0 if the predicted class of the i-th point (x_i) coincides with its actual class, and 1 otherwise. Low values of the $e_k(D)$ index denote high-quality clusters.

Following [7], we can go deeper in our evaluation by measuring the average number of elements, in a range of k elements (we recall again that we use the expected cluster size value), having the same class as the point under consideration. Practically, we define q_k as the average percentage of points in the k-neighborhood of a generic point belonging to the same class of that point. Formally:

$$q_k(D) = \frac{1}{N} \sum_{i=1}^{N} \frac{|\mathcal{N}_k(i) \cap Cl(i)|}{min(k, n_i)}$$

where $Cl(i)$ represents the actual class associated with the i-th point in the dataset, $n_i = |Cl(i)|$, and $\mathcal{N}_k(i)$ is the set of k points having the lowest distances from x_i, according to the distance used at hand. This value will provide a really interesting information, in fact it will measure the *purity* of the clusters since it take into account the number of points wrongly assigned to a cluster. In principle, a Nearest Neighbor classifier exhibits a good performance when q_k is high. Furthermore, q_k provides a measure of the stability of a Nearest-Neighbor: high values of q_k make a k-NN classifier less sensitive to increasing values k of neighbors considered. The sensitivity of the clustering can also be measured by considering, for a given group of points x, y, z, the probability that x and y belong to the same class and z belongs to a different class, but z is more similar to x than y is. We denote this probability by $\varepsilon(D)$, estimated as:

$$\varepsilon(D) = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{1}{(n_i - 1)(N - n_i)} \sum_{Cl(j)=Cl(i), j \neq i} \; \sum_{Cl(k) \neq Cl(i)} \delta_D(i, j, k) \right)$$

where δ_D is 1 if $D(i, j) < D(i, k)$, and 0 otherwise. This value gives information about the ambiguity in cluster assignments. Here too, low values of $\epsilon(D)$ denote a good performance of the clustering under consideration.

The results in Table 2(b) show that CLUBS produces better results than the other algorithms. Table 2(b) shows that CLUBS offers the best performance on all indices and in particular the really high values of q_k (it is practically 1 since it detects exactly the number of clusters for each dataset and the point assignment to cluster is correct) allow to asses that the clusters are well defined, and CLUBS outperforms both BIRCH and OPTICS. In measuring e_k and q_k, we used neighborhoods of size 10 (this value is the actual cluster size available by datasets provider). The overall structure of the clusters and the points distribution for Dataset 1 (results in Table 2(b)) produced superior performance for CLUBS on every index, with particularly low values of ε. This result is confirmed also for Dataset 2 and suggests that CLUBS exhibits the highest effectiveness compared to the other approaches even when SSQ is not the chosen metric.

5 Conclusion

The naturalness of the hierarchical approach for clustering objects is widely recognized, and also supported by psychological studies of children's cognitive behaviors[1]. CLUBS is providing the analytical and algorithmic advances that have turned this intuitive approach into a data mining method of superior accuracy, robustness and speed. The speed achieved by our approach is largely due to CLUBS' ability of exploiting the analytical properties of its quadratic distance functions to simplify the computation. We conjecture that similar benefits might be at hand for situations where the samples are in data streams or in secondary store. These situations were not studied in this paper, but represent a promising topic for future research.

References

1. Ankerst, M., Breunig, M.M., Kriegel, H.P., Sander, J.: Optics: Ordering points to identify the clustering structure. SIGMOD Record 28(2), 49–60 (1999)
2. Arthur, D., Vassilvitskii, S.: k-means++: the advantages of careful seeding. In: SODA, pp. 1027–1035 (2007)
3. Ben-David, S., Ackerman, M.: Measures of clustering quality: A working set of axioms for clustering. In: NIPS, pp. 121–128 (2008)
4. Cheung, Y.M.: k*-means: A new generalized k-means clustering algorithm. Pattern Recognition Letters 24(15), 2883–2893 (2003)
5. Einbond, L.S., Su, T., Wu, H., Friedman, R., Wang, X., Ramirez, A., Kronenberg, F., Weinstein, I.B.: The growth inhibitory effect of actein on human breast cancer cells is associated with activation of stress response pathways. I. J. of Cancer 121(9), 2073–2083 (2007)
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 (1996)
7. Flesca, S., Manco, G., Masciari, E., Pontieri, L., Pugliese, A.: Fast detection of xml structural similarity. TKDE 17(2), 160–175 (2005)
8. Graham, K., De Las Morenas, A., Tripathi, A., King, C., Kavanah, M., Mendez, J., Stone, M., Slama, J., Miller, M., Antoine, G., Willers, H., Sebastiani, P., Rosenberg, C.L.: Gene expression in histologically normal epithelium from breast cancer patients and from cancer-free prophylactic mastectomy patients shares a similar profile. Br. J. Cancer 102(8), 1284–1293 (2010)
9. Han, J., Kamber, M.: Data Mining: Concepts and Techniques. Morgan Kaufmann (2000)
10. MacQueen, J.B.: Some methods for classification and analysis of multivariate observations. In: 5-th Berkeley Symposium on Mathematical Statistics and Probability, pp. 281–297 (1967)
11. Ostrovsky, R., Rabani, Y., Schulman, L.J., Swamy, C.: The effectiveness of lloyd-type methods for the k-means problem. In: FOCS (2006)
12. Veenman, C.J., Reinders, M.J.T.: The nearest subclass classifier: A compromise between the nearest mean and nearest neighbor classifier. IEEE Trans. Pattern Anal. Mach. Intell. 27(9), 1417–1429 (2005)
13. Wang, W., Yang, J., Muntz, R.R.: Sting: A statistical information grid approach to spatial data mining. In: VLDB, pp. 186–195 (1997)
14. Zhang, T., Ramakrishnan, R., Livny, M.: Birch: An efficient data clustering method for very large databases. In: SIGMOD, pp. 103–114 (1996)

[1] Jodie M. Plumert, Flexibility in Children's Use of Spatial and Categorical Organizational Strategies in Recall Developmental Psychology 1994. Vol. 30. No. 5. 738-747.

Clustering Patient Medical Records via Sparse Subspace Representation

Budhaditya Saha[1], Duc-Son Pham[2], Dinh Phung[1], and Svetha Venkatesh[1,2]

[1] Center for Pattern Recognition and Data Analytics
School of Information Technology, Deakin University, Geelong, Australia
[2] Institute for Multi-sensor Processing and Content Analysis
Department of Computing, Curtin University, Western Australia
budhaditya.saha@deakin.edu.au

Abstract. The health industry is facing increasing challenge with "big data" as traditional methods fail to manage the scale and complexity. This paper examines clustering of patient records for chronic diseases to facilitate a better construction of care plans. We solve this problem under the framework of subspace clustering. Our novel contribution lies in the exploitation of sparse representation to discover subspaces automatically and a domain-specific construction of weighting matrices for patient records. We show the new formulation is readily solved by extending existing ℓ_1-regularized optimization algorithms. Using a cohort of both diabetes and stroke data we show that we outperform existing benchmark clustering techniques in the literature.

Keywords: subspace clustering, medical data, sparse representation.

1 Introduction

Traditional methods fail to manage the scale and complexity of "big data". The health sector is at the epicenter of this "big data" - data on admissions, diagnosis, outcomes, spanning a bewildering and disconnected web of images, computerized records and registries. There are no systems to manage this big data. The result is "write only data", mostly unused. Critically it has potential to identify critical safety issues, as well as service and clinical efficiency. This paper explores the pressing need, to construct data analytic to inform such clinical decisions. The outcomes are critically important from economic, patient safety and systems perspectives.

Historically, classical statistical methods have been used to verify stated hypotheses. This requires a priori assumption, for example, on data distributions. As the scale, distribution and diversity of data increase, this approach leads to sub-optimal use of this information. This paper examines new ways to analyze cohorts of patients with chronic diseases, such as Diabetes mellitus (diabetes) and stroke. Chronic care is expensive to administer. One crucial problem in the management of chronic patients is to deliver care plans, such that in majority of cases patients can be manged in the community without hospitalization.

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 123–134, 2013.

This requires us to find sub-groups of patients with same disease characteristics, without any prior assumptions on grouping.

Considering the complexity and nature of the datasets, we propose to model the data by a union of subspaces [1] where each subspace corresponds to patients with similar diagnostic conditions. This model has been used in many applications, such as lossy compression of images [2][3], motion segmentation in video sequences [4,5,6,7,8] etc. Early subspace clustering methods include mixture of Gaussian, factorization, algebraic, compressed sensing/low-rank [9] methods, and examples range from K subspaces [10], mixture of probabilistic PCA [11], multi-stage learning [12] etc. These algorithms typically require prior knowledge about the subspaces - the number of subspaces or their dimensions [13]. The computation is also exponential with the number and/or dimensions. Recently, Elhamifar and Vidal [13] propose sparse subspace clustering (SSC), in which the clustering is solved by seeking a sparse representation of data points. By computing an affinity graph on the sparse representations for all data points, SSC automatically discovers the subspaces and their dimensions. However, the previous results by SSC show that there are many instances in which the sparse coefficients corresponding to points outside a cluster of interest are significantly non-zero. This suggests that enforcing constraints that discourage points further apart will prevent them from entering the same cluster [14]. This was also exploited in [5], who propose a weighted version (WL-SSC).

Inspired by the related success of sparse subspace clustering in computer vision, this paper proposes a novel application of this powerful approach in the context of health care data. Here, it requires a careful modeling and interpretation of subspaces in health care data as well as novel construction of weighting matrices. The weighting matrix acts as the prior knowledge on the similarity between patient records and is computed directly from the data. We explore the decomposition into union of linear subspaces (WL-SSC) and extend the model to consider decomposition of a union of affine subspaces (WA-SSC). To decide on the weighting constraints, we consider three different ways of specifying proximity of points in a k-neighborhood - RBF, cosine and 0-1 matrix. We apply the models across a cohort of 1580 diabetes patients with 551 disease codes, and 1159 stroke patients with 805 codes. The data is collected over a period of 5 years, and each time the patient comes to hospital, a diagnosis code is assigned. Evaluation of such algorithms, with real-world data is notoriously hard. We propose the use of the recently introduced ρ-measure- this method allows ground-truth to be allocated based on degree of similarity between two points. Using this measure, we can compute the Rand-Index and F-measure for a given ρ. We show that our methods outperform the unweighted version and many competitive clustering methods such as affinity propagation (AP) [15], locality-preserving projection (LPP)[16] and k-means [17]. We show that further improvement can be achieved with a weighted union of affine subspace model. We also show tag clouds for clusters in the diabetes cohort and demonstrate how the sub-groups discovered are qualitatively meaningful.

The novelty in our paper is threefold: (a) it applies weighted sparse subspace clustering to a unique medical dataset problem to improve service efficiency, (b) it proposes a new affine, weighted subspace clustering method, and (c) uses a novel principled way to evaluate real world clustering results for which no ground-truth can be obtained.

The significance of the problem lies in the ability to save costs with efficient sub-group identification, leading to targeted care plans. Both chronic diseases chosen have reached epidemic status. For example, Diabetes mellitus (diabetes) is spreading so rapidly that recent studies show that the total number of diabetic people across the world was 171 million in 2001 and it is estimated to 230 million by the end of 2030 [18,19].

2 Related Background

2.1 Sparse Subspace Clustering (SSC)

Consider a set of N data points collected in a $D \times N$ matrix $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]$ where $\mathbf{x}_i \in \mathbb{R}^D$ and D is the number of features. SSC [4] clusters the datapoints via the subspace principle. Intuitively, a linear representation of a datapoint with respect to the whole set gives more preferences to those points that belong to the same subspace. Denote as $\mathcal{S}_i$ the subspace (cluster) that $\mathbf{x}_i$ belongs to. Then, the linear representation of a datapoint can be written as follows:

$$\mathbf{x}_i = \sum_{j \neq i} c_{ij}\mathbf{x}_j = \sum_{i \in \mathcal{S}_i, j \neq i} c_{ij}\mathbf{x}_j + \sum_{j \notin \mathcal{S}_i} c_{ij}\mathbf{x}_j = \mathbf{X}\mathbf{c}_i \tag{1}$$

Here, $\mathbf{c}_i \doteq [c_{i1}, c_{i2}, \ldots, c_{iN}]^T$ are the coefficients of the representation. In the ideal case, the coefficients in the second summation of the right term are zero, giving rise to sparse coefficient vector $\mathbf{c}_i$. However, the solution of (1) is generally not unique when the number of features D is usually much less than the number of observations N. Recent advances in sparse learning [20,21] show that it is possible to regularize the solution and at the same time achieve sparse solution, which is consistent to the ideal case, by enforcing the ℓ_1-norm of the coefficient vector, $\|\mathbf{c}_i\|_1 = \sum |c_{ik}|$, to be small. Using this principle, SSC [4] advocates to find the solution with two variations as follows.

Linear Sparse Subspace formulation (L-SSC). Under this formulation, we assume that data points in $\mathbf{X}$ are sampled from a union of linear subspaces. Then the sparse coefficients are obtained by solving following optimization problem without employing any others constraints on coefficient vector $\mathbf{c}_i$.

$$\arg\min_{\mathbf{c}_i} ||\mathbf{c}_i||_1 \quad \text{s.t. } \mathbf{x}_i = \mathbf{X}\mathbf{c}_i, \;\; c_{ii} = 0 \tag{2}$$

Affine Sparse Subspace formulation (A-SSC). L-SSC can be extended to union of affine subspaces by enforcing an additional equality constraint over the sparse coefficient vector $\mathbf{c}_i$ as follows:

$$\arg\min_{\mathbf{c}_i} ||\mathbf{c}_i||_1 \quad \text{s.t. } \mathbf{x}_i = \mathbf{X}\mathbf{c}_i, \;\; \mathbf{c}_i^T\mathbf{1} = \mathbf{1} \quad c_{ii} = 0 \tag{3}$$

The coefficients are then used to compute a balanced affinity matrix for final spectral clustering: $\bar{\mathbf{C}} = (\mathbf{C} + \mathbf{C}^T)/2$. Then, the Laplacian matrix $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1/2}\bar{\mathbf{C}}\mathbf{D}^{-1/2}$ is computed, with $\mathbf{I}$ being the identity matrix and $\mathbf{D}$ being a diagonal matrix where $\mathbf{D}_{ii} = \sum_{j=1}^{N} \bar{c}_{ij}$. The smallest eigenvalues of $\mathbf{L}$ is used to estimate number of subspaces and the corresponding data points are obtained using the k-means algorithm.

3 Proposed Method

3.1 Weighted Sparse Subspace Clustering (W-SSC)

In the ideal case, the coefficients c_{ij} are zero if data points $\mathbf{x}_i$ and $\mathbf{x}_j$ are sampled from two different subspaces. However, there are cases where they significantly deviate from zero due to numerical properties of the data matrix $\mathbf{X}$ [5]. To avoid undesirable sparse solutions, it has been suggested to introduce a weighting scheme in the sparse formulation [5]. Under this scheme, a weight matrix $\mathbf{W} \in \mathbb{R}^{N\times N}$ is used to enforce sparse coefficients to better fall into the same subspace they deem to belong to. Such a desired solution is encouraged by minimizing the weighted ℓ_1-norm $\|\mathbf{w}_i \odot \mathbf{c}_i\|_1$ instead of $\|\mathbf{c}_i\|_1$. Here, $\odot$ denotes element-wise product of two vectors. Inspired by this principle, we also propose to employ the weighting scheme in our method. The remaining challenge is to construct a suitable weighting matrix for the data, which we detail next.

3.2 Construction of Weighting Matrix W

An optimal weighting matrix can be constructed if we have ground-truth knowledge of the clusters to suppress cross-cluster coefficients (by setting w_{ij} large or small for inter- or intra-cluster coefficients respectively). However, as this knowledge is not available, we propose to use the information within the data to approximate the optimal weighting matrix. We rely on the principle that the weights for inter-cluster coefficients are large whilst those for intra-cluster coefficients are small. Denote as $\mathbf{x}_i \in \mathcal{N}(\mathbf{x}_j)$ as $\mathbf{x}_i$ is k-nearest neighbor of $\mathbf{x}_j$, and $\mathbb{I}$ the indicator (0/1) function. We propose the following choices:

- **Inverse RBF** : $w_{ij} = \mathbb{I}_{\mathbf{x}_i \notin \mathcal{N}(\mathbf{x}_j)} \times \exp^{\frac{||\mathbf{x}_i - \mathbf{x}_j||_2^2}{2\sigma^2}}$
- **0-1** : $w_{ij} = \mathbb{I}_{\mathbf{x}_i \notin \mathcal{N}(\mathbf{x}_j)}$
- **Cosine**: $w_{ij} = \mathbb{I}_{\mathbf{x}_i \notin \mathcal{N}(\mathbf{x}_j)} \times \frac{\langle \mathbf{x}_i, \mathbf{x}_j \rangle}{||\mathbf{x}_i||_2 ||\mathbf{x}_j||_2}$

3.3 Weighted Formulation

Extending the basic SSC algorithms, we propose to adapt to the idea in [5] and solve the following *basic* weighted formulation with linear subspace assumption

$$\arg\min_{\mathbf{c}_i} ||\mathbf{w}_i \odot \mathbf{c}_i||_1, \quad \text{s.t.} \mathbf{x}_i = \mathbf{X}\mathbf{c}_i, c_{ii} = 0 \tag{4}$$

The above basic formulation assumes noiseless data generation. Considering noise while modeling the data points sampled from the union of subspaces, we assume that each data points $\mathbf{x}_i$ is contaminated with noise $\mathbf{e}_i$. i.e. $\mathbf{x}_i = \mathbf{x}_i^{true} + \mathbf{e}_i$ where $\mathbf{x}_i^{true}$ is the true value of the i-th variable and $\mathbf{e}_i$ is bounded: $||\mathbf{e}_i||_2 \leq \epsilon$. Thus, it is more realistic to extend the basic model to account for noise by considering the noise-aware version of the formulation

$$\arg\min_{c_i} ||\mathbf{w}_i \odot \mathbf{c}_i||_1 \quad \text{s.t. } ||\mathbf{x}_i - \mathbf{X}\mathbf{c}_i||_2^2 \leq \epsilon, \quad c_{ii} = 0 \tag{5}$$

This can be more conveniently written in a Lagrangian form

$$\arg\min_{\mathbf{c}_i} \lambda||\mathbf{w}_i \odot \mathbf{c}_i||_1 + \frac{1}{2}||\mathbf{x}_i - \mathbf{X}^{-i}\mathbf{c}_i||_2^2 \tag{6}$$

Here, λ is regularization parameter, $\mathbf{X}^{-i}$ is $\mathbf{X}$ with the ith column removed, and we implicitly ignore the i^{th} entry of $\mathbf{c}_i$. When considering the affine subspace modeling, the above Lagrangian formulation can be extended to account for the additional affine constraints as follows

$$\arg\min_{\mathbf{c}_i} \lambda||\mathbf{w}_i \odot \mathbf{c}_i||_1 + \frac{1}{2}||\mathbf{x}_i - \mathbf{X}^{-i}\mathbf{c}_i||_2^2, \quad \mathbf{c}_i^T\mathbf{1} = 1, \tag{7}$$

Next, we discuss optimization algorithms to solve (7) (note that (6) can be readily solved by a slight modification of many efficient compressed sensing solver, such as reweighting the column of $\mathbf{X}^{-i}$ by the inverse of the corresponding weights and working on the reweighted variables [5]). As they are convex problems, off-the-shelf solvers, such as CVX, can be used, but we do not seek to use them because they are rather inefficient. We show that it is possible to solve (7) more efficiently with the alternative direction method of multipliers (ADMM) [22]. For notational simplicity, we drop the subscript/superscript of $\mathbf{c}_i, \mathbf{x}_i$ and $\mathbf{X}^{-i}$. Under the ADMM framework, we decouple the ℓ_1 regularization term from the quadratic terms by introducing a new variable $\mathbf{z}$ such that $\mathbf{z} - \mathbf{c} = 0$ and consider the augmented Lagrangian

$$\begin{aligned}\mathcal{L}(\mathbf{c}, \mathbf{z}, \mathbf{y}, v) = \frac{1}{2}\|\mathbf{x} - \mathbf{X}\mathbf{c}\|_2^2 + \lambda\|\mathbf{z}\|_1 + \mathbf{y}^T(\mathbf{c} - \mathbf{z}) + \frac{\rho_1}{2}\|\mathbf{c} - \mathbf{z}\|_2^2 \\ + v(\mathbf{1}^T\mathbf{c} - 1) + \frac{\rho_2}{2}(\mathbf{1}^T c - 1)^2.\end{aligned} \tag{8}$$

Here, $\mathbf{y}$ and v are the dual parameters corresponding to the inequality constraints $\mathbf{c} - \mathbf{z} = 0$ and $\mathbf{1}^T\mathbf{c} - 1 = 0$ respectively; ρ_1 and ρ_2 are small parameters to improve numerical stability (see [22] for ADMM background). By using the normalized dual variables $\mathbf{u}_1 = (\mathbf{y}/\rho_1)$ and $u_2 = (v/\rho_2)$ we derive the following ADMM updates that solve (7)

$$\mathbf{c}^{k+1} = (\mathbf{X}^T\mathbf{X} + \rho_1\mathbf{I} + \rho_2\mathbf{1}\mathbf{1}^T)^{-1}(\mathbf{X}^T\mathbf{x} + \rho_1(\mathbf{z}^k - \mathbf{u}_1^k) + \rho_2\mathbf{1}(1 - u_2)) \tag{9}$$

$$\mathbf{z}^{k+1} = \mathsf{S}_{\lambda/\rho_1}(\mathbf{c}^{k+1} + \mathbf{u}_1) \tag{10}$$

$$\mathbf{u}_1^{k+1} = \mathbf{u}_1^k + (\mathbf{c}^{k+1} - \mathbf{z}^{k+1}) \tag{11}$$

$$u_2^{k+1} = u_2^k + (\mathbf{1}^T\mathbf{c}^{k+1} - 1). \tag{12}$$

Here $\mathsf{S}_{\boldsymbol{\tau}}(\mathbf{c})$ is the soft-thresholding shrinkage operator, defined as a vector r such that $r_i = \mathsf{sign}(c_i)\max(|c_i| - \tau_i, 0)$ (see [22]).

Once the coefficient vectors $\mathbf{c}_i$'s are found, the spectral clustering part proceeds in the same way as the original SSC algorithm [4].

4 Experiments

4.1 Datasets

We validate our approach on two real-world datasets collected from patients having diabetes and heart (stroke) diseases collected over a period of five years from 2007 to 2011 and has diagnosis records from 9878 patients. Each patient has been diagonised several times over a period of five years and assigned unique diagnosis code(s). An example of a record for a patient over time might be (`E1172, I10, E1172, Z9222`). Table 1 and 2 shows the description of some codes. Patients may be assigned similar code more than once over time.

We remove records without codes, patients diagonised less than twice and also duplicated codes. This results in 1580 diabetes patients with 551 unique codes. We construct a code-patient matrix, where codes are used as features and each patient is an observation, analogous to term-document matrix for text data analysis. In our second data set (stroke patients), there are 1159 patients with 805 diagnostic codes.

4.2 Evaluation Method

As no ground-truth is available for latent groups, it is impossible to measure the clustering performance by standard evaluation metrics. Thus, we evaluate the performance using a novel ρ-measure method as follows:

1. Each data point $\mathbf{x}_i \in \mathbb{R}^N$ is mapped to a binary vector $\bar{\mathbf{x}}_i$ where $\bar{x}_{ij} = \mathbb{I}_{x_{ij} \neq 0}$.
2. Compute relative similarity metric $s_\rho(\bar{\mathbf{x}}_i, \bar{\mathbf{x}}_j)$

$$s_\rho(\bar{\mathbf{x}}_i, \bar{\mathbf{x}}_j) = \frac{\sum(\bar{\mathbf{x}}_i \odot \bar{\mathbf{x}}_j)}{\sum_{k=1}^N \bar{\mathbf{x}}_{ik} + \sum_{k=1}^N \bar{\mathbf{x}}_{jk} - \sum(\bar{\mathbf{x}}_i \odot \bar{\mathbf{x}}_j)} \tag{13}$$

3. Construct a ground-truth matrix $\mathbf{G}_\rho \in \mathbb{R}^{N \times N}$ with element $g_{ij} = \mathbb{I}_{s_\rho(\bar{\mathbf{x}}_i, \bar{\mathbf{x}}_j) \geq \rho}$
4. Construct a cluster membership matrix $\mathbf{V}$ with element $v_{ij} = \mathbb{I}_{ID_K(i) = ID_K(j)}$

Table 1. Examples of code description

Codes	Description of Codes
E1172	Type 2 diabetes mellitus with features of insulin resistance
I10	Essential (primary) hypertension
Z9222	Personal history of long-term (current) use of other medicament, insulin
R63Z	Chemotherapy

Table 2. Diabetes dataset

Patient Id	Diagnosis Codes
P1	E1172,I10,E1172,Z9222
P2	M81403,Z511,R63Z,R63Z
P3	E1023,E1023,E1012

Next, we compute the standard*Precision* (P), *Recall* (R) and *F-measure* (F):

$$P = \frac{TP}{TP + FP}, \ R = \frac{TP}{TP + FN}, \ F = \frac{2 \times P \times R}{P + R} \tag{14}$$

Here, true positive (TP) is scored when two similar data points in the ground-truth are grouped together in the obtained results, a true negative (TN) is scored when two dissimilar data points are grouped separately, a false positive (FP) is scored when two dissimilar data points are grouped together and a false negative (FN) is scored when two similar data points are grouped separately. Similarly, the rand index (RI) is defined as

$$RI = \frac{TP + TN}{TP + FP + FN + TN}$$

where high RI and F indicates the better accuracy.

Algorithm 1 show the overall method of computing F-measure. Note that, we compute F measure over a matrix of $N \times N$ variables, instead of N number of data points.

4.3 Results and Comparisons

Performance against Other Methods. We compare our proposed clustering method against competitive sparse subspace clustering and baseline alternatives, including affinity propagation (AP) [15], locality preserving projection (LPP) [16], and k-means [17]. In all experiments, we set ρ to 0.9, regularization parameter λ to 0.001.

Table 3 presents the clustering results obtained from SSC methods for diabetes and stroke data. Clearly, our proposed method outperforms both L-SSC and A-SSC variants by obtaining larger RI and F scores. The F measure scores of WL-SSC and WA-SSC have improved over L-SSC and A-SSC by large margins of **47**% and **45**% for the diabetes data and **236**% and **257**% for the stroke data respectively.

Algorithm 1. Computing F measure

```
Input: Groundtruthed Matrix $\mathbf{G}_\rho$ and Cluster Index matrix $\mathbf{V}$.
Output: $F$ − measure
Intialize: Set TP=TN=FP=FN=0.

– for $i = 1$ to $N$
    • for $k =$ 1to $N$
        ∗ if ($\mathbf{g}_{ik} = 1$) and ($\mathbf{v}_{ik} = 1$) $TP = TP + 1$; // Two similar data points grouped together.
        ∗ else if ($\mathbf{g}_{ik} = 0$) and ($\mathbf{v}_{ik} = 0$) $TN = TN + 1$; // Two dissimilar data points grouped separately.
        ∗ else if ($\mathbf{g}_{ik} = 0$) and ($\mathbf{v}_{ik} = 1$) $FP = FP + 1$;//Two dissimilar data points grouped similar.
        ∗ else ($\mathbf{g}_{ik} = 1$) and ($\mathbf{v}_{ik} = 0$) $FN = FN + 1$;//Two similar data points grouped separately.
    • end
– end
– Calculate $F$-measure following the equation 14.
```

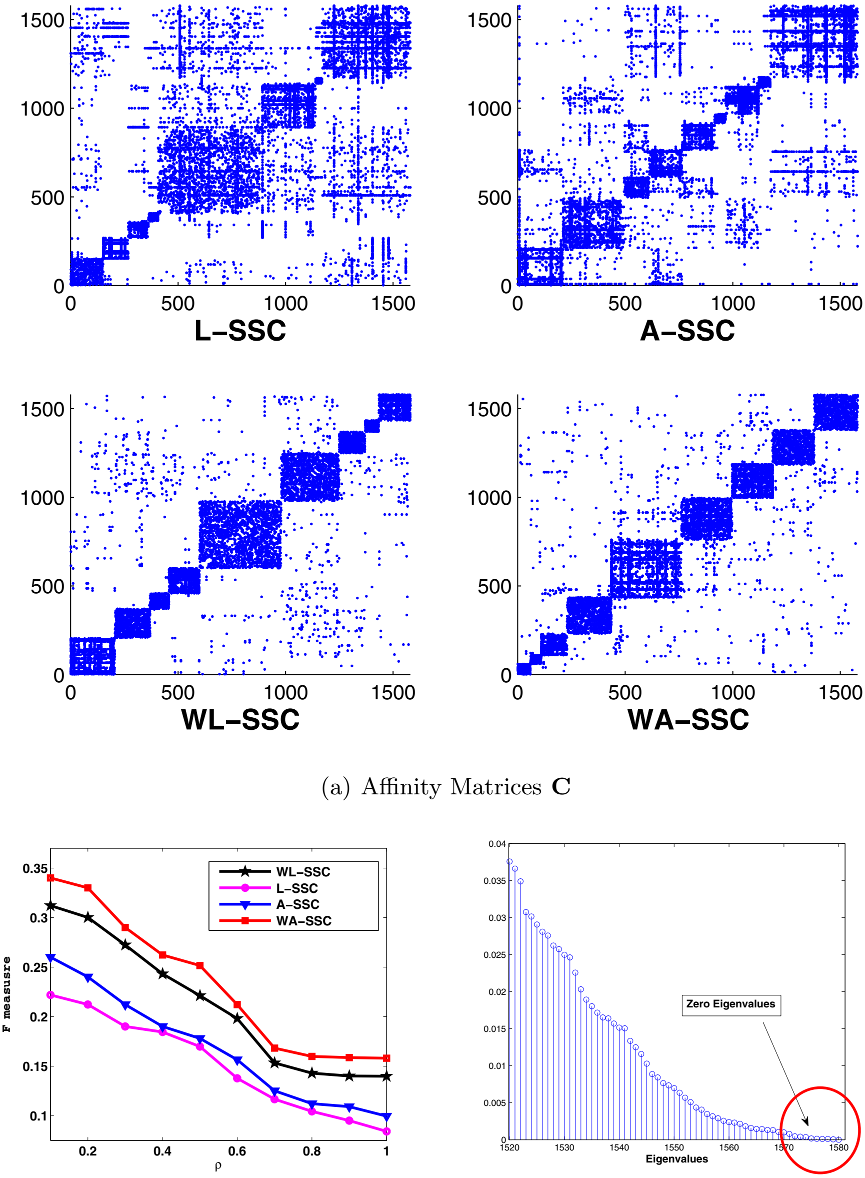
Likewise, the F measure is improved by **275**% (AP), **85**% (LPP), **388**% (k-means) for diabetics datasets, whereas the betterment in RI is **87**% (AP), **14**% (LPP), **10**% (k-means) respectively. For the strokes data, F measure is improved by **173**% (AP), **54**% (LPP), **465**% (k-means) and *Rand Index* is **71**% (AP), **13**% (LPP), **139**% (k-means) respectively.

Table 3. Performance comparison

Datasets	Diabetics Data		Strokes Data	
Methods	F measure	Rand Index	F measure	Rand Index
AP	0.0423	0.4639	0.062	0.522
LPP	0.0854	0.7654	0.11	0.8045
k-means	0.0325	0.4312	0.0294	0.3845
L-SSC	0.0951	0.7817	0.0475	0.5210
A-SSC	0.1092	0.7862	0.0619	0.7324
WL-SSC	**0.1401**	**0.8652**	**0.1597**	**0.90**
WA-SSC	**0.1587**	**0.8982**	**0.1697**	**0.91**

Table 4. Performance analysis using different weighting schemes

Datasets	Diabetes Data		Strokes Data	
Weighting Schemes	WL-SSC	WA-SSC	WL-SSC	WA-SSC
RBF	**0.1401**	**0.1587**	**0.1597**	**0.1697**
0-1	0.1199	0.1221	0.1191	0.1201
cosine	0.1352	0.1444	0.1390	0.1382

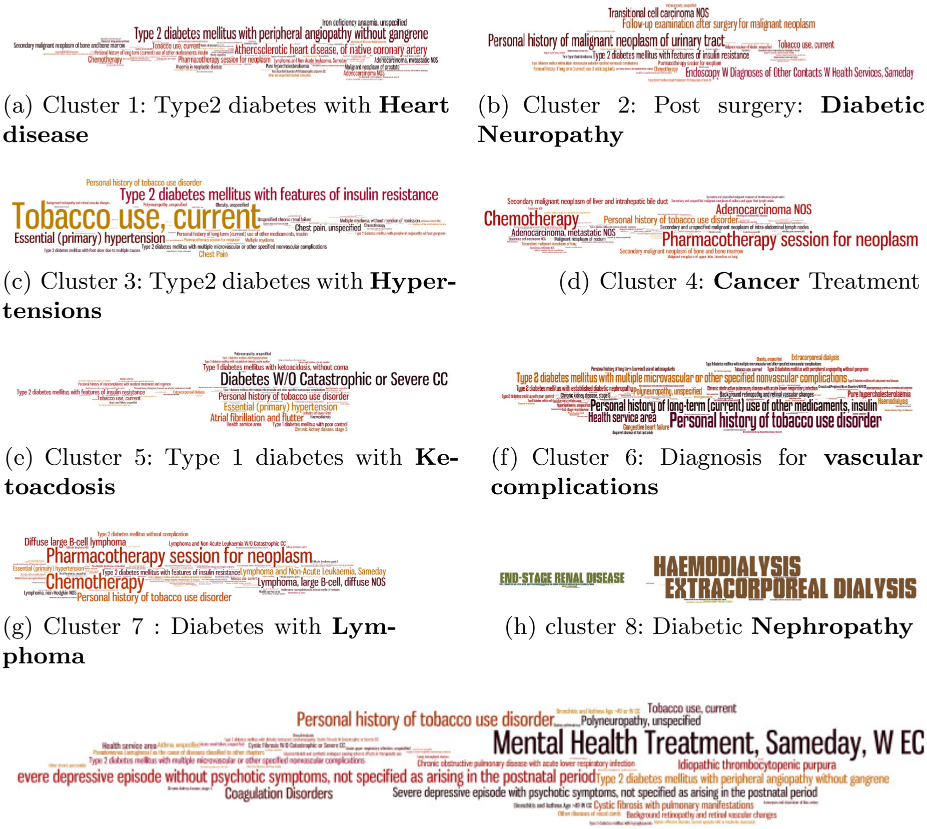


(a) Affinity Matrices **C**

(b) F-measure

(c) Eigenvalues of **L**

Fig. 1. Plots for Qualitative Evaluations



(a) Cluster 1: Type2 diabetes with **Heart disease**

(b) Cluster 2: Post surgery: **Diabetic Neuropathy**

(c) Cluster 3: Type2 diabetes with **Hypertensions**

(d) Cluster 4: **Cancer** Treatment

(e) Cluster 5: Type 1 diabetes with **Ketoacdosis**

(f) Cluster 6: Diagnosis for **vascular complications**

(g) Cluster 7 : Diabetes with **Lymphoma**

(h) cluster 8: Diabetic **Nephropathy**

(i) Cluster 9: Diabetes with **Psychiatric Disorders**

Fig. 2. Diagnostic Clouds

Influence of Weighting Schemes. Table 4 include the performance for different weighting schemes and it is found that the RBF choice provides better performance than the other choices.

Discovered Clusters. The number of clusters K equals to the number of zero eigenvalues of of Laplacian matrix $\mathbf{L}$. Fig. 1(c) shows the eigenvalue plot of $\mathbf{L}$ for the diabetes data where the number of zero eigenvalue equals to 9. Similarly, we found 12 sub-groups for stroke data.

Since ρ is the relative similarity between the two data points, which means high value of ρ denotes two observations are highly similar, we vary ρ varies from 0.1 to 1 in a separate experiment on diabetes data and plots are shown in . Figure 1(b). As expected, F-measure is high for small values of ρ and F-measure is low when ρ is increasing.

Figures 1 and 2 show the qualitative evaluation of clusters for the diabetes data. Figure 1(a) shows the affinity matrices, whilst Figure 2 shows the tag clouds of the diagnosis codes in each cluster. As anticipated the clusters are qualitatively different in terms of disease differentiation within diabetes: diabetes with heart disease, with cancer, with dialysis. Type 1 and 2 are clearly differentiated.

5 Conclusion

We have demonstrated a novel application of the sparse subspace clustering theory in solving the clustering problem of health care data. Our novel contributions includes special construction of the weighting matrices to obtain better sparse solution and the efficient algorithm to solve the formulation with affine constraints. To evaluate realistic health care data where no ground-truth is available, we have also suggested a novel evaluation method of clustering results. Compared with competitive alternatives in the literature, our proposed method achieve much better F and RI scores, and discovers meaningful patients subgroups.

References

1. Eldar, Y., Mishali, M.: Robust recovery of signals from a structured union of subspaces. IEEE Transactions on Information Theory 55(11), 5302–5316 (2009)
2. Hong, W., Wright, J., Huang, K., Ma, Y.: A multiscale hybrid linear model for lossy image representation. In: Proc. ICCV, pp. 764–771 (2005)
3. Yang, A., Wright, J., Ma, Y., Sastry, S.: Unsupervised segmentation of natural images via lossy data compression. Computer Vision and Image Understanding 110(2), 212–225 (2008)
4. Elhamifar, E., Vidal, R.: Sparse subspace clustering. In: Proc. CVPR, pp. 2790–2797. IEEE (2009)
5. Pham, D.-S., Saha, B., Phung, D., Venkatesh, S.: Improved subspace clustering via exploitation of spatial constraints. In: Proc. CVPR. IEEE (2012)
6. Wang, S., Yuan, X., Yao, T., Yan, S., Shen, J.: Efficient subspace segmentation via quadratic programming. In: Proc. AAAI (2011)
7. Yu, Y., Schuurmans, D.: Rank/norm regularization with closed-form solutions: Application to subspace clustering. Arxiv preprint arXiv:1202.3772 (2012)
8. Vidal, R., Tron, R., Hartley, R.: Multiframe motion segmentation with missing data using power factorization and GPCA. IJCV 79(1), 85–105 (2008)
9. Liu, G., Lin, Z., Yu, Y.: Robust subspace segmentation by low-rank representation. In: Proc. ICML (2010)
10. Ho, J., Yang, M., Lim, J., Lee, K., Kriegman, D.: Clustering appearances of objects under varying illumination conditions. In: Proc. CVPR, vol. 1, pp. I–11. IEEE (2003)
11. Tipping, M., Bishop, C.: Mixtures of probabilistic principal component analyzers. Neural Computation 11(2), 443–482 (1999)
12. Gruber, A., Weiss, Y.: Multibody factorization with uncertainty and missing data using the EM algorithm. In: Proc. CVPR (2004)
13. Elhamifar, E., Vidal, R.: Sparse subspace clustering. In: Proc. CVPR, pp. 2790–2797 (2009)

14. Candes, E., Wakin, M., Boyd, S.: Enhancing sparsity by reweighted l1 minimization. Journal of Fourier Analysis and Applications 14(5), 877–905 (2008)
15. Frey, B., Dueck, D.: Clustering by passing messages between data points. Science 315(5814), 972–976 (2007)
16. He, X., Cai, D., Liu, H., Ma, W.: Locality preserving indexing for document representation. In: Proc. ACM SIGIR, pp. 96–103 (2004)
17. Kanungo, T., Mount, D., Netanyahu, N., Piatko, C., Silverman, R., Wu, A.: An efficient k-means clustering algorithm: Analysis and implementation. IEEE Transactions on Pattern Analysis and Machine Intelligence 24(7), 881–892 (2002)
18. Fabris, P., Floreani, A., Tositti, G., Vergani, D., De Lalla, F., Betterle, C.: Type 1 diabetes mellitus in patients with chronic hepatitis c before and after interferon therapy. Alimentary Pharmacology & Therapeutics 18(6), 549–558 (2003)
19. Young, J., McAdam-Marx, C.: Treatment of type 1 and type 2 diabetes mellitus with insulin detemir, a long-acting insulin analog. Clinical Medicine Insights. Endocrinology and Diabetes 3, 65 (2010)
20. Candès, E., Romberg, J., Tao, T.: Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. IEEE Transactions on Information Theory 52(2), 489–509 (2006)
21. Donoho, D.: Compressed sensing. IEEE Transactions on Information Theory 52(4), 1289–1306 (2006)
22. Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J.: Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. In: Jordan, M. (ed.) Foundations and Trends in Machine Learning, vol. 3(1), pp. 1–122. Now Publisher (2011)

A Unified Metric for Categorical and Numerical Attributes in Data Clustering

Yiu-ming Cheung[1,2] and Hong Jia[1]

[1] Department of Computer Science and Institute of Computational and Theoretical Studies, Hong Kong Baptist University, Hong Kong SAR, China
[2] United International College,
Beijing Normal University - Hong Kong Baptist University, Zhuhai, China
{ymc,hjia}@comp.hkbu.edu.hk

Abstract. Most of the existing clustering approaches are applicable to purely numerical or categorical data only, but not both. In general, it is a nontrivial task to perform clustering on mixed data composed of numerical and categorical attributes because there exists an awkward gap between the similarity metrics for categorical and numerical data. This paper therefore presents a general clustering framework based on the concept of object-cluster similarity and gives a unified similarity metric which can be simply applied to the data with categorical, numerical, or mixed attributes. Accordingly, an iterative clustering algorithm is developed, whose efficacy is experimentally demonstrated on different benchmark data sets.

1 Introduction

To discover the natural group structure of objects represented in numerical or categorical attributes [1], clustering analysis has been widely applied to a variety of scientific areas. Traditionally, clustering analysis mostly concentrates on purely numerical data only. The typical clustering algorithms include the k-means [2] and EM algorithm [3]. Since the objective functions of these two algorithms are both numerically defined, they are not essentially applicable to the data sets with categorical attributes. Under the circumstances, a straightforward way to overcome this problem is to transform the categorical values into numerical ones, e.g. the binary strings, and then apply the aforementioned numerical-value based clustering methods. Nevertheless, such a method has ignored the similarity information embedded in the categorical values and cannot faithfully reveal the similarity structure of the data sets [4]. Hence, it is desirable to solve this problem by finding a unified similarity metric for categorical and numerical attributes such that the metric gap between numerical and categorical data can be eliminated. Subsequently, a general clustering algorithm which is applicable to various data types can be presented based on this unified metric.

In this paper, we will propose a unified clustering approach for both categorical and numeric data sets. Firstly, we present a general clustering framework based

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 135–146, 2013.

on the concept of object-cluster similarity. Then, a new metric for both of numerical and categorical attributes is proposed. Under this metric, the object-cluster similarity for either categorical or numerical attributes has a uniform criterion. Hence, transformation and parameter adjustment between categorical and numerical values in data clustering are circumvented. Subsequently, analogous to the framework of k-means, an iterative algorithm is introduced to implement the data clustering. This algorithm conducts an efficient clustering analysis without manually adjusting parameters and is applicable to the three types of data: numerical, categorical, or mixed data, i.e. the data with both of numerical and categorical attributes. Empirical studies have shown the promising results.

2 Related Works

Roughly, the existing clustering approaches dealing with data sets which contain categorical attributes can be summarized into the four categories [5]. The first category of the methods is based on the perspective of similarity. For example, based on Goodall similarity metric [6] that assigns a greater weight to uncommon feature value matching in similarity computations without assuming the underlying distributions of the feature values, paper [7] presents the Similarity Based Agglomerative Clustering (SBAC) algorithm. This method has a good capability of dealing with the mixed numeric and categorical attributes, but its computation is quite laborious. Beside the similarity concepts, the second category is based on graph partitioning. A typical example is the CLICKS algorithm [8], which mines subspace clusters for categorical data sets. This novel method encodes a data set into a weighted graph structure, where each weighted vertex stands for an attribute value and two nodes are connected if there is a sample in which the corresponding attribute values co-occur. It is experimentally demonstrated that CLICKS outperforms ROCK algorithm [9] and scales better for high-dimensional data sets. However, this algorithm is not applicable to data mixed with categorical and numerical attributes and its performance also depends upon a set of parameters whose tuning is quite difficult from the practical viewpoint. The third category is entropy-based methods. For example, the COOLCAT algorithm [10] utilizes the information entropy to measure the closeness between objects and presents a scheme to find a clustering structure via minimizing the expected entropy of clusters. The performance of this algorithm is stable for different data sizes and parameter settings. Nevertheless, this method can only be applied to purely categorical data and cannot handle numerical attributes. The last category of approaches attempts to give a distance metric between categorical values so that the distance-based clustering algorithms (e.g. the k-means) can be directly adopted. Along this line, the most cost-effective one may be the k-prototype algorithm proposed by Huang [11]. In this method, the distance between two categorical values is defined as 0 if they are the same, and 1 otherwise while the distance between numerical values is quantified with Euclidean distance. Subsequently, the k-means paradigm is utilized for clustering. However, since different metrics are adopted for numerical and categorical attributes, a user-defined parameter is utilized to control the proportions of numerical distance and categorical distance. Nevertheless, various settings of this parameter will lead to a totally different clustering result. A simplified version of k-prototype algorithm namely k-modes [12, 13], which is applicable for purely categorical data clustering, has also been widely utilized due to its satisfactory efficiency, and different improvement strategies have been explored on this method [14–16].

3 Object-Cluster Similarity Metric

The general task of clustering is to classify the given objects into several clusters such that the similarities between objects in the same group are high while the similarities between objects in different groups are low [17]. Therefore, clustering a set of N objects, $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$, into k different clusters, denoted as C_1, C_2, …, C_k, can be formulated to find the optimal $\mathbf{Q}^*$ via the following objective function:

$$\mathbf{Q}^* = \arg\max_{\mathbf{Q}} F(\mathbf{Q}) = \arg\max_{\mathbf{Q}} [\sum_{j=1}^{k} \sum_{i=1}^{N} q_{ij} s(\mathbf{x}_i, C_j)] \tag{1}$$

where $s(\mathbf{x}_i, C_j)$ is the similarity between object $\mathbf{x}_i$ and Cluster C_j, and $\mathbf{Q} = (q_{ij})$ is an $N \times k$ partition matrix satisfying

$$\sum_{j=1}^{k} q_{ij} = 1, \text{ and } 0 < \sum_{i=1}^{N} q_{ij} < N, \tag{2}$$

with

$$q_{ij} \in [0, 1], \; i = 1, 2, \ldots, N, j = 1, 2, \ldots, k. \tag{3}$$

Evidently, the desired clusters can be obtained by (1) as long as the metric of object-cluster similarity is determined. In the following sub-sections, we shall therefore study the similarity metric.

3.1 Similarity Metric for Mixed Data

This sub-section will study the object-cluster similarity metric for mixed data. Suppose the mixed data $\mathbf{x}_i$ with d different attributes consists of d_c categorical attributes and d_u numerical attributes, i.e. $d_c + d_u = d$. $\mathbf{x}_i$ can be therefore denoted as $[\mathbf{x}_i^{cT}, \mathbf{x}_i^{uT}]^T$ with $\mathbf{x}_i^c = (x_{i1}^c, x_{i2}^c, \ldots, x_{id_c}^c)^T$ and $\mathbf{x}_i^u = (x_{i1}^u, x_{i2}^u, \ldots, x_{id_u}^u)^T$. Then, we have x_{ir}^u $(r = 1, 2, \ldots, d_u)$ belonging to $\mathbf{R}$ and x_{ir}^c $(r = 1, 2, \ldots, d_c)$ belonging to $dom(A_r)$, where $\{A_1, A_2, \ldots, A_{d_c}\}$ are the d_c categorical attributes and $dom(A_r)$ contains all possible values that can be chosen by attribute A_r. For categorical attributes, the value domains are finite and unordered, $dom(A_r)$ with m_r elements can be therefore represented with $dom(A_r) = \{a_{r1}, a_{r2}, \ldots, a_{rm_r}\}$.

Firstly, we focus on the difference between categorical attributes and numerical attributes. For categorical attributes, each attribute can usually represent an important feature of the given object. Therefore, when we conduct classification or clustering analysis, we often investigate the categorical attributes one by one such as Decision Tree method. By contrast, the numerical attributes are often treated as a vector and handled together in clustering analysis. Based on these observations, for the mixed data $\mathbf{x}_i$, the d_u numerical attributes can be treated as a whole but the d_c categorical attributes should be investigated individually. Let the object-cluster similarity between $\mathbf{x}_i$ and cluster C_j, denoted as $s(\mathbf{x}_i, C_j)$, be the average of the similarity calculated based on each attribute, we will have

$$\begin{aligned} s(\mathbf{x}_i, C_j) =& \frac{1}{d} s(x_{i1}^c, C_j) + \frac{1}{d} s(x_{i2}^c, C_j) + ... + \frac{1}{d} s(x_{id_c}^c, C_j) + \frac{d_u}{d} s(\mathbf{x}_i^u, C_j) \\ =& \frac{1}{d} \sum_{r=1}^{d_c} s(x_{ir}^c, C_j) + \frac{d_u}{d} s(\mathbf{x}_i^u, C_j). \end{aligned} \tag{4}$$

That is, the similarity between each numerical attribute and the cluster C_j is replaced with the similarity between the cluster and the whole numerical vector $\mathbf{x}_i^u$. Moreover, if we denote the similarity between $\mathbf{x}_i^c$ and C_j as $s(\mathbf{x}_i^c, C_j)$, we can get

$$s(\mathbf{x}_i^c, C_j) = \frac{1}{d_c} \sum_{r=1}^{d_c} s(x_{ir}^c, C_j) = \sum_{r=1}^{d_c} \frac{1}{d_c} s(x_{ir}^c, C_j). \tag{5}$$

Then, (4) can be further rewritten as

$$s(\mathbf{x}_i, C_j) = \frac{d_c}{d} \sum_{r=1}^{d_c} \frac{1}{d_c} s(x_{ir}^c, C_j) + \frac{d_u}{d} s(\mathbf{x}_i^u, C_j) = \frac{d_c}{d} s(\mathbf{x}_i^c, C_j) + \frac{d_u}{d} s(\mathbf{x}_i^u, C_j), \tag{6}$$

where $s(\mathbf{x}_i^c, C_j)$ is actually the similarity on categorical attributes and $s(\mathbf{x}_i^u, C_j)$ is the similarity on numerical attributes. Subsequently, the object-cluster similarity metric can be obtained based on the definitions of $s(\mathbf{x}_i^c, C_j)$ and $s(\mathbf{x}_i^u, C_j)$.

Similarity Metric for Categorical Attributes. In (5), we have assumed that each categorical attribute has the same contribution to the calculation of similarity on categorical part. However, from the practical viewpoint, due to the different distributions of attribute values, categorical attributes each often have unequal importance for clustering analysis. In light of this characteristic, (5) should be further modified with

$$s(\mathbf{x}_i^c, C_j) = \sum_{r=1}^{d_c} w_r s(x_{ir}^c, C_j), \tag{7}$$

where w_r is the weight of categorical attribute A_r satisfying $0 \leq w_r \leq 1$ and $\sum_{r=1}^{d_c} w_r = 1$. That is, the object-cluster similarity for categorical part is the weighted summation of the similarity between the cluster and each attribute value. Weight factor w_r describes the importance of each categorical attribute and is utilized to control the contribution of attribute-cluster similarity to object-cluster similarity.

Definition 1. *The similarity between a categorical attribute value* x_{ir}^c *and cluster* C_j*,* $i \in \{1, 2, \ldots, N\}$*,* $r \in \{1, 2, \ldots, d_c\}$*,* $j \in \{1, 2, \ldots, k\}$*, is defined as:*

$$s(x_{ir}^c, C_j) = \frac{\sigma_{A_r = x_{ir}^c}(C_j)}{\sigma_{A_r \neq NULL}(C_j)}, \tag{8}$$

where NULL refers to empty, and $\sigma_{A_r = x_{ir}^c}(C_j)$ *counts the number of objects (also called* instances *hereinafter) that have the value* x_{ir}^c *for attribute* A_r *in cluster* C_j*.*

From *Definition 1*, we can find that this metric of attribute-cluster similarity has the following properties:

(1) $0 \leq s(x_{ir}^c, C_j) \leq 1$;
(2) $s(x_{ir}^c, C_j) = 1$ only if all the instances belonging to cluster C_j have the value x_{ir}^c for attribute A_r, and $s(x_{ir}^c, C_j) = 0$ only if no instance belonging to cluster C_j has the value x_{ir}^c for attribute A_r.

According to (7) and (8), the object-cluster similarity for categorical part can be therefore calculated by

$$s(\mathbf{x}_i^c, C_j) = \sum_{r=1}^{d_c} w_r s(x_{ir}^c, C_j) = \sum_{r=1}^{d_c} w_r \frac{\sigma_{A_r = x_{ir}^c}(C_j)}{\sigma_{A_r \neq NULL}(C_j)}, \tag{9}$$

where $i \in \{1, 2, \ldots, N\}$, and $j \in \{1, 2, \ldots, k\}$.

Remark 1. Since $0 \leq s(x_{ir}^c, C_j) \leq 1$ and $\sum_{r=1}^{d_c} w_r = 1$, we have:

$$s(\mathbf{x}_i^c, C_j) = \sum_{r=1}^{d_c} w_r s(x_{ir}^c, C_j) \geq \sum_{r=1}^{d_c} (w_r \cdot 0) = 0,$$

and

$$s(\mathbf{x}_i^c, C_j) = \sum_{r=1}^{d_c} w_r s(x_{ir}^c, C_j) \leq \sum_{r=1}^{d_c} (w_r \cdot 1) = \sum_{r=1}^{d_c} w_r = 1.$$

That is, for any $i \in \{1, 2, \ldots, N\}$ and $j \in \{1, 2, \ldots, k\}$, the value of $s(\mathbf{x}_i^c, C_j)$ will fall into the interval $[0, 1]$.

Next, we discuss how to estimate the importance of each categorical attribute. From the view point of information theory, the significance of an attribute can be regarded as the inhomogeneity degree of the data set with respect to this attribute. Furthermore, it is described in [18] that if the information content of an attribute is high, the inhomogeneity of the data set is also high for this attribute. Hence, the importance of any categorical attribute A_r ($r \in \{1, 2, \ldots, d_c\}$) can be calculated by

$$H_{A_r} = -\sum_{t=1}^{m_r} p(a_{rt}) \log p(a_{rt}) \tag{10}$$

with

$$p(a_{rt}) = \frac{\sigma_{A_r=a_{rt}}(X)}{\sigma_{A_r \neq NULL}(X)}, \tag{11}$$

where $a_{rt} \in dom(A_r)$, $p(a_{rt})$ is the probability of attribute value a_{rt}, m_r is the total number of values that can be chosen by A_r and X is the whole data set. Furthermore, according to (10), the more different values an attribute has, the higher its significance is. However, in practice, an attribute with too many different values may have little contribution to clustering. For example, the ID number of instances is unique for each instance, but this information is useless for clustering analysis. Hence, (10) can be further modified with

$$H_{A_r} = -\frac{1}{m_r} \sum_{t=1}^{m_r} p(a_{rt}) \log p(a_{rt}). \tag{12}$$

That is, the importance of an attribute is quantified by its average entropy over each attribute value. The weight of each attribute is then computed as

$$w_r = \frac{H_{A_r}}{\sum\limits_{t=1}^{d_c} H_{A_t}}, r = 1, 2, \ldots, d_c. \tag{13}$$

Subsequently, the object-cluster similarity on categorical part can be given by

$$s(\mathbf{x}_i^c, C_j) = \sum_{r=1}^{d_c} \left(\frac{H_{A_r}}{\sum\limits_{t=1}^{d_c} H_{A_t}} \cdot \frac{\sigma_{A_r=x_{ir}^c}(C_j)}{\sigma_{A_r \neq NULL}(C_j)} \right). \tag{14}$$

In practice, for an attribute A_r, if all the instances to be classified have the same value a, it can be obtained from (12) and (11) that the importance of this attribute will be 0 as $p(a) = 1$ and $log(1) = 0$. Then, the corresponding attribute weight will also be zero and this attribute will have no contribution to the whole clustering learning.

Similarity Metric for Numerical Attributes. Since the distance between each vector $\mathbf{x}_i^u$ can be numerically calculated, the similarity metric for numerical attributes can be defined based on the measure of distance. According to [19] and [20], it is a universal law that the distance and perceived similarity between numerical vectors are related via an exponential function as follows:

$$s(\mathbf{x}_A, \mathbf{x}_B) = \exp(-Dis(\mathbf{x}_A, \mathbf{x}_B)), \tag{15}$$

where Dis stands for a distance measure. Moreover, it can be observed that the magnitudes of distances between instances from variant data sets may have a significant difference in practice. To avoid the potential influence of this scenario, we can further use proportional distance instead of absolute distance to estimate the similarity between numerical vectors.

Definition 2. *The object-cluster similarity between numerical vector* $\mathbf{x}_i^u$ *and cluster* C_j*,* $i \in \{1, 2, \ldots, N\}$*,* $j \in \{1, 2, \ldots, k\}$*, is given by*

$$s(\mathbf{x}_i^u, C_j) = \exp\left(-\frac{Dis(\mathbf{x}_i^u, \mathbf{c}_j)}{\sum\limits_{t=1}^{k} Dis(\mathbf{x}_i^u, \mathbf{c}_t)}\right), \tag{16}$$

where $\mathbf{c}_j$ *is the center of all numerical vectors in cluster* C_j*.*

It can be seen from *Definition 2* that the values of this similarity metric also fall into the interval [0, 1]. In practice, different distance metrics can be utilized to calculate $Dis(\mathbf{x}_i^u, \mathbf{c}_j)$. For example, if the Minkowski distance is adopted, we shall have:

$$Dis(\mathbf{x}_i^u, \mathbf{c}_j) = \left(\sum_{r=1}^{d_u} |x_{ir}^u - c_{jr}|^p\right)^{1/p}, \tag{17}$$

where $p > 0$ is a constant which characterizes the distance function. A typically special case of (17) is the Euclidean distance with $p = 2$.

Finally, according to (6), (14), and (16), the object-cluster similarity metric for mixed data is defined as

$$s(\mathbf{x}_i, C_j) = \frac{d_c}{d}\sum_{r=1}^{d_c}\left(\frac{H_{A_r}}{\sum\limits_{t=1}^{d_c} H_{A_t}} \cdot \frac{\sigma_{A_r = x_{ir}^c}(C_j)}{\sigma_{A_r \neq NULL}(C_j)}\right) + \frac{d_u}{d}\exp\left(-\frac{Dis(\mathbf{x}_i^u, \mathbf{c}_j)}{\sum\limits_{t=1}^{k} Dis(\mathbf{x}_i^u, \mathbf{c}_t)}\right), \tag{18}$$

where $i = 1, 2, \ldots, N$, $j = 1, 2, \ldots, k$. It can be seen that the defined similarities for categorical and numerical attributes in (18) are in the same scale. Hence, unlike k-prototype method, there is no need any more to manually adjust the parameter to control the proportions of numerical and categorical distances for different data sets.

4 Iterative Clustering Algorithm

This paper concentrates on hard partition only, i.e., $q_{ij} \in \{0, 1\}$, although it can be easily extended to the soft partition in terms of posterior probability. Under the circumstances, given a set of N objects, the optimal $\mathbf{Q}^* = \{q_{ij}^*\}$ in (1) can be given by

$$q_{ij}^* = \begin{cases} 1, & \text{if } s(\mathbf{x}_i, C_j) \geq s(\mathbf{x}_i, C_r), 1 \leq r \leq k, \\ 0, & \text{otherwise.} \end{cases} \tag{19}$$

Therefore, similar to the learning procedure of k-means, an iterative algorithm, denoted as OCIL, can be conducted to implement the clustering analysis as shown in Algorithm 1.

The first step in OCIL algorithm, i.e. Step 1, is a procedure for the calculation of object-cluster similarity. Thus, we can find that the iterative steps of

Algorithm 1 Iterative clustering learning based on object-cluster similarity metric (OCIL)

```
Require: data set $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$, number of clusters $k$
Ensure: cluster label $Y = \{y_1, y_2, \ldots, y_N\}$
Calculate the importance of each categorical attribute according to (12), if applicable
Set $Y = \{0, 0, \ldots, 0\}$ and randomly select $k$ initial objects, one for each cluster
repeat
    Initialize $noChange = true$
    for $i = 1$ to $N$ do
        $y_i^{(new)} = \arg \max_{j \in \{1,\ldots,k\}} [s(\mathbf{x}_i, C_j)]$
        if $y_i^{(new)} \neq y_i^{(old)}$ then
            $noChange = false$
            Update the information of clusters $C_{y_i^{(new)}}$ and $C_{y_i^{(old)}}$, including the frequency of each categorical value and the centroid of numerical vectors
        end if
    end for
until $noChange$ is $true$
return $Y$
```

OCIL algorithm is the same as the k-means algorithm and the only difference is the measurement of similarity between object and clusters. Therefore, the effectiveness of the proposed similarity metric can be easily evaluated by comparing OCIL with other similar algorithms, such as k-means and k-prototype. Next, we further give the time complexity analysis of OCIL algorithm. It can be observed that the computation cost of Step 1 is $O(mNd_c)$, where m is the average number of different values that can be chosen by each categorical attribute. For each iteration, the cost of the "**for**" statement is $O(mNkd_c + Nkd_u)$. Hence, the total time cost of this algorithm is $O(t(mNkd_c + Nkd_u))$, where t stands for the number of iterations. From the practical viewpoint, k, m and t can be regarded as a constant in most cases. Therefore, the time complexity of this algorithm approaches to $O(dN)$. Hence, the proposed algorithm is efficient for data clustering, particularly for a large data set.

5 Experiments

This section is to investigate the effectiveness of the proposed approach to data clustering. We applied it to various categorical and mixed data sets obtained from UCI Machine Learning Data Repository[1] and compared its performance with the existing counterparts. Since the proposed method on numerical data degenerates to the k-means algorithm, the effectiveness of OCIL algorithm on numerical data set is transparent. Hence, there is no need to investigate it any more. Each algorithm was coded with MATLAB and all experiments were implemented by

[1] See http://archive.ics.uci.edu/ml/

a desktop PC computer with Intel(R) Core(TM)2 Quad CPU, 2.40 GHz main frequency, and 4GB DDR2 667 RAM.

Moreover, in our experiments, the clustering accuracy [21] for measuring the clustering performance was estimated by

$$ACC = \frac{\sum_{i=1}^{N} \delta(c_i, map(r_i))}{N},$$

where N is the number of instances in the data set, c_i stands for the provided label, $map(r_i)$ is a mapping function which maps the obtained cluster label r_i to the equivalent label from the data corpus by using the Kuhn-Munkres algorithm, and the delta function $\delta(c_i, map(r_i)) = 1$ only if $c_i = map(r_i)$, otherwise 0. Correspondingly, the clustering error rate is computed as $e = 1 - ACC$.

5.1 Performance on Mixed Data Sets

In the following experiments, we will investigate the performance of the proposed algorithm on real data sets in comparison with the existing counterparts. Firstly, experiments were conducted on mixed data and the information of selected data sets is shown in Table 1. The performance of the proposed method has been compared with the k-prototype algorithm [11] and k-means algorithm, whose time complexity are also $O(Nd)$. In k-prototype method, the distance regulation parameter γ was set at 0.5σ [11], where σ is the average standard deviation of numerical attributes. When utilizing k-means, the categorical values were transformed into integers in our experiments. Moreover, the Euclidean distance has been adopted as the distance metric of numerical vectors for consistency. Each algorithm has been run 100 times on each data set and the clustering results are summarized in Table 2.

Table 1. Statistics of mixed data sets

Data set	Instance	Attribute ($d_c + d_u$)	Class
Statlog Heart	270	$7 + 6$	2
Heart Disease	303	$7 + 6$	2
Credit Approval	653	$9 + 6$	2
German Credit	1000	$13 + 7$	2
Dermatology	366	$33 + 1$	6
Adult	30162	$8 + 6$	2

It can be seen that, both with random initializations, the proposed algorithm OCIL has an obvious superiority in terms of clustering accuracy over the k-prototype and k-means methods. This result shows that, in comparison with numerically representing the distance between categorical values, the proposed similarity metric in this paper is a more reasonable measurement for clustering

Table 2. Clustering errors of OCIL on mixed data sets in comparison with k-prototype and k-means

Data set	K-means	K-prototype	OCIL
Statlog	0.4047±0.0071	0.2306±0.0821	**0.1716±0.0065**
Heart	0.4224±0.0131	0.2280±0.0903	**0.1644±0.0030**
Credit	0.4487±**0.0016**	0.2619±0.0976	**0.2519**±0.0966
German	0.3290±0.0014	0.3289±**0.0006**	**0.3057**±0.0007
Dermatology	0.7006±**0.0216**	0.6903±0.0255	**0.3051**±0.0896
Adult	0.3869±**0.0067**	0.3855±0.0143	**0.3079**±0.0305

Table 3. Comparison of average convergent time and iterations between k-prototype and OCIL

Data set	Time		Iterations	
	K-prototype	OCIL	K-prototype	OCIL
Statlog	0.0519s	**0.0516**s	3.09	**3.07**
Heart	0.0639s	**0.0576**s	3.54	**3.02**
Credit	**0.1323**s	0.1625s	**3.18**	4.26
German	0.2999s	**0.2023**s	5.29	**3.15**
Dermatol	0.3674s	**0.1888**s	7.27	**4.32**
Adult	15.2795s	**9.6774**s	10.93	**6.78**

Table 4. Statistics of categorical data sets

Data set	Instance	Attribute	Class
Soybean	47	35	4
Breast	699	9	2
Vote	435	16	2
Zoo	101	16	7

Table 5. Comparison of clustering errors on categorical data sets

Data set	H's k-modes	N's k-modes	OCIL
Soybean	0.1691±0.1521	**0.0964**±0.1404	0.1017±**0.1380**
Breast	0.1655±0.1528	0.1356±0.0016	**0.0934±0.0009**
Vote	0.1387±0.0066	0.1345±0.0031	**0.1213±0.0010**
Zoo	0.2873±0.1083	0.2730±**0.0818**	**0.2681**±0.0906

analysis on mixed data. Moreover, comparing the average running time of OCIL and k-prototype algorithms listed in Table 3, we can find that, although OCIL needs additional time to calculate the weight of each categorical attribute, its total running time is no more than k-prototype's one. A plausible reason can be found from Table 3 is that the convergence speed of OCIL is usually faster than k-prototype in most cases we have tried so far. Therefore, the proposed similarity metric is efficient for mixed data clustering.

5.2 Performance on Categorical Data Sets

We further investigated the performance of the proposed algorithm on purely categorical data. The information of four different benchmark data sets we utilized has been summarized in Table 4. To conduct comparative studies, we have also implemented the other two existing categorical data clustering algorithms: original k-modes (H's k-modes) [12] and k-modes with Ng's dissimilarity metric (N's k-modes) [16]. In this experiment, each algorithm was conducted with random initializations. Table 5 lists the average value and standard deviation in error obtained by OCIL and the other two algorithms, respectively. It can be seen that the proposed clustering method has competitive advantage in terms of clustering accuracy and robustness compared with the other two methods.

6 Conclusion

In this paper, we have proposed a general clustering framework based on object-cluster similarity, through which a unified similarity metric for both categorical and numerical attributes has been presented. Under this new metric, the object-cluster similarity for categorical and numerical attributes are with the same scale, which is beneficial to clustering analysis on various data types. Subsequently, analogous to k-means method, an iterative algorithm has been introduced to implement the data clustering. The advantages of the proposed method have been experimentally demonstrated in comparison with the existing counterparts.

Acknowledgment. The work described in this paper was supported by the Faculty Research Grant of Hong Kong Baptist University with the Project Code: FRG2/11-12/067, and the NSFC under grant 61272366.

References

1. Michalski, R.S., Bratko, I., Kubat, M.: Machine learning and data mining: methods and applications. Wiley, New York (1998)
2. MacQueen, J.B.: Some methods for classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, pp. 281–297 (1967)
3. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society, Series B (Methodological) 39(1), 1–38 (1977)
4. Hsu, C.C.: Generalizing self-organizing map for categorical data. IEEE Transactions on Neural Networks 17(2), 294–304 (2006)

5. Cesario, E., Manco, G., Ortale, R.: Top-down parameter-free clustering of high-dimensional categorical data. IEEE Transactions on Knowledge and Data Engineering 19(12), 1607–1624 (2007)
6. Goodall, D.W.: A new similarity index based on probability. Biometric 22(4), 882–907 (1966)
7. Li, C., Biswas, G.: Unsupervised learning with mixed numeric and nominal data. IEEE Transactions on Knowledge and Data Engineering 14(4), 673–690 (2002)
8. Zaki, M.J., Peters, M.: Click: Mining subspace clusters in categorical data via k-partite maximal cliques. In: Proceedings of the 21st International Conference on Data Engineering, pp. 355–356 (2005)
9. Guha, S., Rastogi, R., Shim, K.: Rock: A robust clustering algorithm for categorical attributes. Information Systems 25(5), 345–366 (2001)
10. Barbara, D., Couto, J., Li, Y.: Coolcat: An entropy-based algorithm for categorical clustering. In: Proceedings of the 11th ACM Conference on Information and Knowledge Management, pp. 582–589 (2002)
11. Huang, Z.: Clustering large data sets with mixed numeric and categorical values. In: Proceedings of the First Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 21–24 (1997)
12. Huang, Z.: A fast clustering algorithm to cluster very large categorical data sets in data mining. In: Proceedings of the SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery, pp. 1–8 (1997)
13. Huang, Z., Ng, M.K.: A note on k-modes clustering. Journal of Classification 20(2), 257–261 (2003)
14. Khan, S.S., Kant, S.: Computation of initial modes for k-modes clustering algorithm using evidence accumulation. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007), pp. 2784–2789 (2007)
15. He, Z., Deng, S., Xu, X.: Improving k-modes algorithm considering frequencies of attribute values in mode. In: Hao, Y., Liu, J., Wang, Y.-P., Cheung, Y.-m., Yin, H., Jiao, L., Ma, J., Jiao, Y.-C. (eds.) CIS 2005. LNCS (LNAI), vol. 3801, pp. 157–162. Springer, Heidelberg (2005)
16. Ng, M.K., Li, M.J., Huang, J.Z., He, Z.: On the impact of dissimilarity measure in k-modes clustering algorithm. IEEE Transactions on Pattern Analysis and Machine Intelligence 29(3), 503–507 (2007)
17. Jain, A.K.: Data clustering: 50 years beyound k-means. Pattern Recognition Letters 31(8), 651–666 (2010)
18. Basak, J., Krishnapuram, R.: Interpretable hierarchical clustering by constructing an unsupervised decision tree. IEEE Transactions on Knowledge and Data Engineering 17(1), 121–132 (2005)
19. Shepard, R.N.: Toward a universal law of generalization for physical science. Science 237, 1317–1323 (1987)
20. Santini, S., Jain, R.: Similarity measures. IEEE Transactions on Pattern Analysis and Machine Intelligence 21(9), 871–883 (1999)
21. He, X., Cai, D., Niyogi, P.: Laplacian score for feature selection. In: Advances in Neural Information Processing Systems (2005)

An Extension of the Infinite Relational Model Incorporating Interaction between Objects

Iku Ohama[1], Hiromi Iida[1], Takuya Kida[2], and Hiroki Arimura[2]

[1] Corporate R&D Division, Panasonic Corporation, Osaka 571-8501, Japan
{ohama.iku,iida.hiromi}@jp.panasonic.com
[2] Division of Computer Science, Hokkaido University, Hokkaido 060-0814, Japan
{kida,arim}@ist.hokudai.ac.jp

Abstract. The *Infinite Relational Model* (IRM) introduced by Kemp *et al.* (Proc. AAAI2006) is one of the well-known probabilistic generative models for the co-clustering of relational data. The IRM describes the relationship among objects based on a stochastic block structure with infinitely many clusters. Although the IRM is flexible enough to learn a hidden structure with an unknown number of clusters, it sometimes fails to detect the structure if there is a large amount of noise or outliers. To overcome this problem, in this paper we propose an extension of the IRM by introducing a *subset mechanism* that selects a part of the data according to the interaction among objects. We also present posterior probabilities for running collapsed Gibbs sampling to learn the model from the given data. Finally, we ran experiments on synthetic and real-world datasets, and we showed that the proposed model is superior to the IRM in an environment with noise.

Keywords: relational data, co-clustering, generative model, subset selection.

1 Introduction

A *relational data* among m objects and n objects is a bipartite network on a set of m vertices and another set of n vertices, which describes the relationships among objects in social, physical, and other phenomena. Equivalently, a relational data is represented by a matrix with m rows and n columns. For example, POS data is a relational data between customers and items, and a friend list of a social network service (SNS) such as the Facebook is a relational data among users.

With the emergence of such large amounts of relational data, there has been an increase in the interest in methods that can efficiently discover hidden interaction patterns among objects from given relational data. For example, enterprises involved in e-commerce and SNS might want to know about the following relationships:

- Which type of items does a customer purchase using e-commerce?
- Which other users are in a relationship with a SNS user?
- To which user does another user re-tweet when communicating on Twitter?

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 147–159, 2013.

Clustering methods are among the most effective approaches to obtain answers to such questions, and several methods have been proposed so far [5,3,4,16]. The *Infinite Relational Model* (IRM) [11] is a well-known and important generative model that represents processes for generating relational data. Co-clustering based on the model can produce a proper set of clusters that summarizes the relationships among objects. Moreover, the number of clusters is automatically estimated from the input data, even when the cluster structure and its size are unknown.

However, the IRM might fail to detect unknown structures when the data has a large amount of noise or the model can describe only a part of the data. Owing to the use of infinite clustering based on the Dirichlet Process (DP) [6], the IRM works to some extent, but it finds many small clusters to adapt itself to contradicting data. In fact, the problem of the co-clustering of real-world datasets is often difficult, because the data are noisy or sparse. For example, a spam blog that leaves comments randomly on other blogs has too many links. Such a noisy blog makes it difficult to analyze the relationship among blogs. Moreover, an inactive blog, which the author is not eager to write, has very few links. Such an insignificant blog also becomes an obstacle in finding important clusters. As we show later in Section 5, co-clustering with the IRM on such ill-formed data finds ineffective clusters.

To handle these ill-formed data, we incorporated a *subset selection mechanism* into the IRM and proposed a new relational model. In our model, the *relevance* of each object is parameterized by an individual Bernoulli parameter. The relevance indicates the degree of confidence with which an object forms informative relations coming from the latent cluster structure. For example, for POS data, an active customer tends to generate relevant relations with many items, as done by a well-known item as well. Their relevance becomes comparatively high in our model. Then, either a *relevant relation* or an *irrelevant relation* is generated stochastically for pair-wise objects according to the interaction of their relevance parameters.

Our contributions in this paper are summarized as follows:

- We proposed a new generative model, which is an extension of the IRM and incorporates a subset selection mechanism, whereby a subset of the relational data is determined by the interaction of the objects' relevances. By estimating the relevance of each object from the data, we diminished the effect of the irrelevant relations and performed co-clustering accurately.
- We derived posterior probabilities for running the Collapsed Gibbs Sampling [12] in order to infer the parameters of the model.
- We performed experiments on synthetic and real-world datasets. The experimental results for the synthetic datasets showed that our model significantly improved the performance of co-clustering compared with the IRM. For the real-world datasets, our model could successfully find major categories as clusters from the datasets. An estimated relevance of object can be viewed as the popularity or representativeness of the object within a cluster.

Therefore, the proposed method is effective in analyzing noisy relational data.

1.1 Related Works

Hoff *et al.* [8] discussed an ill-formed problem with clustering vector data. They introduced a background distribution that describes irrelevant elements within the vector data, so that their model can find cluster robustly against noise based on a relevant subset of the data.

Ishiguro *et al.* [10] extended the IRM with a similar idea. They introduced switch variables to indicate whether an object is relevant for cluster analysis, or is an irrelevant troublesome one. In their model, only relationships among relevant objects are analyzed. That is, their model is an object-wise subset model. However, in some cases, it would not be preferable to select subset of objects for clustering target. For example, when we utilize co-clustering results for recommendation, we want to suggest the nearest cluster for any object. In our new model, the clustering target is selected in a relation-wise manner.

2 Relational Data and the Infinite Relational Model

In this section, we first define the relational data discussed in this paper. Then, we discuss the IRM, a generative model for co-clustering relational data.

Let $T^1 = \{O_i^1\}_{i=1}^{N^1}$ and $T^2 = \{O_j^2\}_{j=1}^{N^2}$ be the sets of objects. We define the relational data between T^1 and T^2 as $R : T^1 \times T^2 \to \{0, 1\}$. If $R(i, j) = 1(0)$, we say that there is a *link* (*non-link*) between O_i^1 and O_j^2 [1]. For a purchase dataset, T^1 and T^2 are the sets of customers and items, respectively. We can represent customer i's purchase of item j by $R(i, j) = 1$, while $R(i, j) = 0$ indicates that customer i have not bought item j. The co-clustering problem on relational data is to estimate cluster assignments $\boldsymbol{z}^1 = \{z_i^1\}_{i=1}^{N^1} \in C^1$ and $\boldsymbol{z}^2 = \{z_j^2\}_{j=1}^{N^2} \in C^2$ based on given data $\boldsymbol{R}$, where $C^1 = \{1, 2, \cdots, K\}$ and $C^2 = \{1, 2, \cdots, L\}$ are the sets of cluster indices for T^1 and T^2, respectively.

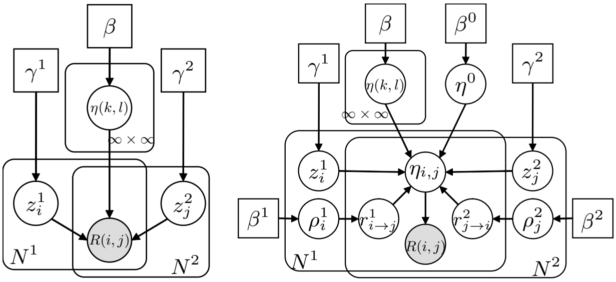
The IRM proposed by Kemp *et al.* [11] is a generative model for relational data that can co-cluster objects based on the similarities of the relationships among the objects. In the IRM, the Dirichlet Process (DP) [6] is used as a prior distribution for the number of clusters. The DP is a nonparametric stochastic process that can be viewed as an infinite-dimensional Dirichlet distribution, and can generate any-dimensional multinomial distributions. Therefore, the IRM can adaptively estimate the number of clusters for the observed data. The generative model of the IRM is described as follows:

$$z_i^1 \,|\, \gamma^1 \sim \mathrm{CRP}(\gamma^1), \qquad z_j^2 \,|\, \gamma^2 \sim \mathrm{CRP}(\gamma^2), \tag{1}$$

$$\eta(k, l) \,|\, \beta \sim \mathrm{Beta}(\beta, \beta), \tag{2}$$

$$R(i, j) \,|\, \eta(z_i^1, z_j^2) \sim \mathrm{Bernoulli}(\eta(z_i^1, z_j^2)), \tag{3}$$

[1] We focus on a 2-type ($T^1 \times T^2$) binary relationship in this paper, although several variations of relationships can be considered straightforwardly, such as discrete/continuous-valued relations and multi-type relations represented by a tensor.



(a) IRM (b) rdIRM

Fig. 1. Graphical representations of the generative models. Circle nodes denote variables, square nodes denote constants, shaded nodes denote observations, and round-edged squares indicate the dimensions of variables.

where CRP(·) is the Chinese Restaurant Process (CRP) [2], which is one of the well-known constructive algorithms of DP; Beta(·, ·) is the beta distribution; and Bernoulli(·) is the Bernoulli distribution, respectively. Figure 1a shows the IRM graphically.

We will briefly review the above process. First, the cluster assignments z_i^1 and z_j^2 are given by CRPs (Eq. (1)), where γ^1 and γ^2 are the concentration parameters of the DP that controls the number of clusters to be generated. We denote the cluster assignments for all objects other than object i as $\boldsymbol{z}_{-i}^1$. When $\boldsymbol{z}_{-i}^1$ is given, the conditional probability $P(z_i^1 = k^* \mid \boldsymbol{z}_{-i}^1, \gamma^1)$ that z_i^1 is assigned to the cluster k^* by CRP is given as follows:

$$P(z_i^1 = k^* \mid \boldsymbol{z}_{-i}^1, \gamma^1) \propto \begin{cases} m_{-i,k^*}^1 & (\text{if } m_{-i,k^*}^1 > 0), \\ \gamma^1 & (\text{if } k^* \text{ is new cluster}), \end{cases} \tag{4}$$

where m_{-i,k^*}^1 is the number of objects other than object i that are assigned to the cluster k^*. As Eq. (4) shows, the assignment z_i^1 basically depends on the probability proportional to the number m_{-i,k^*}^1 of objects that belong to each cluster. However, new clusters are generated at the rate γ^1. Assume that $K \times L$ clusters ($C^1 = \{1, 2, \cdots, K\}$ and $C^2 = \{1, 2, \cdots, L\}$) have been generated for $T^1 \times T^2$. Then, from Eq. (2), a Bernoulli parameter $\eta(k, l)$ is given according to the beta prior for each pair of clusters $C^1 \times C^2$. The parameter $\eta(k, l)$ indicates the intensity of the relationship between an object in the cluster k and an object in the cluster l. Finally, the relation $R(i, j)$ is generated from the corresponding Bernoulli trial (Eq. (3)).

3 The Relevance-Dependent Infinite Relational Model

In this section, we present our new model, called *the Relevance-Dependent Infinite Relational Model* (rdIRM).

In real-world relationships, whether each relation is intentionally generated depends on the objects related to the relation. In the case of a purchase, a customer who knows about a large number of items will have a certain opinion about whether he needs these items. As a result, this customer will generate very important relations that are relevant to decide his cluster assignment. In contrast, a customer who knows only about a few items will have vague opinions. Thus, relations that are generated by this customer would be irrelevant. That is, such an irrelevant relation should not affect the co-clustering. For the items, it is reasonable to consider that similar properties exist in terms of popularity.

To model the above situation, for each object O_i^1 and O_j^2, we introduce *relevance parameters* $\rho_i^1, \rho_j^2 \in [0, 1]$ that indicate the degree of confidence to generate the relevant relations. Then, we consider a generative mechanism in which each relation $R(i, j)$ between objects is generated from a mixture of the distribution inherent in a cluster $\eta(k, l)$ (foreground distribution) and the distribution common to the entire data η^0 (background distribution). We can construct such a mechanism as follows:

$$\begin{aligned} r_{i\to j}^1 &\sim \text{Bernoulli}(\rho_i^1), & r_{j\to i}^2 &\sim \text{Bernoulli}(\rho_j^2), \\ r_{i,j} &= f(r_{i\to j}^1, r_{j\to i}^2), & \eta_{i,j} &= r_{i,j} \times \eta(z_i^1, z_j^2) + (1 - r_{i,j}) \times \eta^0, \end{aligned}$$

where $f(\cdot, \cdot)$ is an arbitrary Boolean function that returns 1 or 0. The above mechanism enables us to embed a *relevance-dependent subset selection* into the relational model: only the informative (relevant) relations are generated from the foreground distribution $\eta(k, l)$, and the background distribution η^0 describes the non-informative (irrelevant) part of the relational data. For example, when f is a logical sum, it corresponds to that we make the mixture rate as $1-(1-\rho_i^1)(1-\rho_j^2)$. When f is a logical product, the mixture rate becomes $\rho_i^1 \times \rho_j^2$. The other logical functions work similarly.

To summarize, the generative process for the rdIRM is defined as follows:

$$\begin{aligned} z_i^1 \mid \gamma^1 &\sim \text{CRP}(\gamma^1), & z_j^2 \mid \gamma^2 &\sim \text{CRP}(\gamma^2), & (5)\\ \eta(k, l) \mid \beta &\sim \text{Beta}(\beta, \beta), & \eta^0 \mid \beta^0 &\sim \text{Beta}(\beta^0, \beta^0), & (6)\\ \rho_i^1 \mid \beta^1 &\sim \text{Beta}(\beta^1, \beta^1), & \rho_j^2 \mid \beta^2 &\sim \text{Beta}(\beta^2, \beta^2), & (7)\\ r_{i\to j}^1 \mid \rho_i^1 &\sim \text{Bernoulli}(\rho_i^1), & r_{j\to i}^2 \mid \rho_j^2 &\sim \text{Bernoulli}(\rho_j^2), & (8)\\ r_{i,j} &= f(r_{i\to j}^1, r_{j\to i}^2), & \eta_{i,j} &= r_{i,j} \times \eta(z_i^1, z_j^2) + (1 - r_{i,j}) \times \eta^0, & (9)\\ & & R(i, j) \mid \eta_{i,j} &\sim \text{Bernoulli}(\eta_{i,j}). & (10) \end{aligned}$$

Figure. 1b graphically represents this model.

Now, we will briefly explain the rdIRM process. First, the cluster assignments $\boldsymbol{z}^1$ and $\boldsymbol{z}^2$ are given as in the original IRM, (Eq. (5)). Second, the foreground distribution $\eta(k, l)$ and the background distribution η^0 are independently given from a beta prior (Eq. (6)). Third, the relevances ρ_i^1 and ρ_j^2 for O_i^1 and O_j^2, respectively, are given from beta priors (Eq. (7)). Fourth, the two switches $r_{i\to j}^1$ and $r_{j\to i}^2$ are given by a Bernoulli trial with corresponding relevances (Eq. (8)). Fifth, either the foreground $\eta(k, l)$ or the background η^0 is selected by the interaction of $r_{i\to j}^1$ and $r_{j\to i}^2$ via logical function f (Eq. (9)). Finally, the relation $R(i, j)$ is generated from the selected probability (Eq. (10)).

The difference between our rdIRM and the original IRM is that we modeled a generative process of noisy relationships by introducing objects' relevances and their interaction mechanism. That is, our rdIRM can co-cluster relational data based on a subset of relations that are relevant to underlying cluster structures.

When f is a logical sum, a relevant relation can be generated when at least one of the related objects O_i^1 or O_j^2 has high relevance. This models situations in which the relevant relationship between objects can be generated by a one-sided request, such as sending an e-mail or following a hyperlink on the Internet. When f is a logical product, the relevant relation is generated only when the objects cooperate with each other. This models situations in which an object that wants to have a relevant relation with another can be constrained from doing so. Of course, we can employ other logical functions for other interaction models.

4 Inference

We use the Collapsed Gibbs Sampler [12] to infer the parameters of the rdIRM[2]. Given $r_{i,j}$, the relational data $\boldsymbol{R}$ are separated into a foreground part and a background part; thus, the relevances ρ_i^1, ρ_j^2 and the link probabilities $\eta(k,l), \eta^0$ can be integrated out. Therefore, the inference of the rdIRM is performed by sampling the assignments $\boldsymbol{z}^1, \boldsymbol{z}^2$ and the switches $\boldsymbol{r}^1, \boldsymbol{r}^2$ one after the other. In this section, we only show the derived posteriors for running the Gibbs sampling below, because of the space limitation.

4.1 Sampling Cluster Assignments $\boldsymbol{z}^1, \boldsymbol{z}^2$

Because z_j^2 can be sampled in the same way as z_i^1, we concentrate on z_i^1. We can assume that the switch variables $\boldsymbol{r}$ ($\boldsymbol{r}^1$ and $\boldsymbol{r}^2$) have already been given before taking a sample of z_i^1, so that the cluster assignments are influenced only by the foreground part of the observations. Therefore, the conditional posterior for $z_i^1 = k^*$ is derived as follows:

$$P(z_i^1 = k^* \mid \boldsymbol{z}_{-i}^1, \boldsymbol{z}^2, \boldsymbol{r}, \boldsymbol{R}, \beta, \gamma^1) \propto \begin{cases} m_{-i,k^*}^1 \prod_{l \in C^2} \frac{B(m_r^{+i}(k^*,l)+\beta, \overline{m}_r^{+i}(k^*,l)+\beta)}{B(m_r^{-i}(k^*,l)+\beta, \overline{m}_r^{-i}(k^*,l)+\beta)} & (\text{if } m_{-i,k^*}^1 > 0), \\ \gamma^1 \prod_{l \in C^2} \frac{B(m_r^{+i}(k^*,l)+\beta, \overline{m}_r^{+i}(k^*,l)+\beta)}{B(\beta,\beta)} & (\text{if } m_{-i,k^*}^1 = 0), \end{cases} \tag{11}$$

Here, we use $B(\cdot,\cdot)$ to denote the beta function. Symbols m_r ($\overline{m}_r$) denote the numbers of links (non-links) in the foreground part of the observation, and are computed as follows:

$$m_r^{+i}(k^*,l) = \sum\sum_{\substack{s \in T^1, j \in T^2: \\ z_s^1 = k^* (z_i^1 := k^*), \\ z_j^2 = l}} (R(s,j) \times r_{i,j}), \; \overline{m}_r^{+i}(k^*,l) = \sum\sum_{\substack{s \in T^1, j \in T^2: \\ z_s^1 = k^* (z_i^1 := k^*), \\ z_j^2 = l}} ((1 - R(s,j)) \times r_{i,j}),$$

$$m_r^{-i}(k^*,l) = \sum\sum_{\substack{s \in T^1, j \in T^2: \\ z_s^1 = k^* (s \neq i), \\ z_j^2 = l}} (R(s,j) \times r_{i,j}), \; \overline{m}_r^{-i}(k^*,l) = \sum\sum_{\substack{s \in T^1, j \in T^2: \\ z_s^1 = k^* (s \neq i), \\ z_j^2 = l}} ((1 - R(s,j)) \times r_{i,j}).$$

[2] Approximative approaches such as variational inference [7] are preferable for handling large scale data. However, for the sake of accuracy, we used a sampling approach in this paper.

Note that if $r_{i,j} = 1$ for all (i, j), Eq. (11) is equivalent to the original IRM's sampler.

4.2 Sampling Switch Variables $r^1_{i\to j}, r^2_{j\to i}$

As the sampling of $r^2_{j\to i}$ is done in the same way as the sampling of $r^1_{i\to j}$, we concentrate on $r^1_{i\to j}$. Given $\boldsymbol{z}^1$ and $\boldsymbol{z}^2$, we have a finite number $K \times L$ of clusters. Thus, the conditional posterior for $r^1_{i\to j}$ is derived as follows:

$$\begin{aligned}
&P(r^1_{i\to j} \,|\, \boldsymbol{z}^1, \boldsymbol{z}^2, \boldsymbol{r}^1_{-(i\to j)}, \boldsymbol{r}^2, \boldsymbol{R}, \beta, \beta^0, \beta^1) \\
&\propto P(R(i,j) \,|\, r^1_{i\to j}, \boldsymbol{r}^1_{-(i\to j)}, \boldsymbol{r}^2, \boldsymbol{R}_{-(i,j)}, \beta^0)^{1-f(r^1_{i\to j}, r^2_{j\to i})} \\
&\times P(R(i,j) \,|\, \boldsymbol{z}^1, \boldsymbol{z}^2, r^1_{i\to j}, \boldsymbol{r}^1_{-(i\to j)}, \boldsymbol{r}^2, \boldsymbol{R}_{-(i,j)}, \beta)^{f(r^1_{i\to j}, r^2_{j\to i})} \\
&\times P(r^1_{i\to j} \,|\, \boldsymbol{r}^1_{i\to(-j)}, \beta^1),
\end{aligned} \tag{12}$$

where $\boldsymbol{R}_{-(i,j)}$ denotes the whole set of $\boldsymbol{R}$ excluding $R(i, j)$. Similarly, $\boldsymbol{r}^1_{-(i\to j)}$ denotes the whole set of $\boldsymbol{r}^1$ without $r^1_{i\to j}$, and $\boldsymbol{r}^1_{i\to(-j)}$ denotes a vector of $r^1_{i\to t}$s that are related to object i without $r^1_{i\to j}$. The terms on the right-hand side of Eq. (12) are computed as follows:

$$P(R(i,j) \,|\, r^1_{i\to j}, \boldsymbol{r}^1_{-(i\to j)}, \boldsymbol{r}^2, \boldsymbol{R}_{-(i,j)}, \beta^0) = \frac{(m_{\overline{r}}^{-(i,j)} + \beta^0)^{R(i,j)} (\overline{m}_{\overline{r}}^{-(i,j)} + \beta^0)^{1-R(i,j)}}{m_{\overline{r}}^{-(i,j)} + \overline{m}_{\overline{r}}^{-(i,j)} + 2\beta^0},$$

$$P(R(i,j) \,|\, \boldsymbol{z}^1, \boldsymbol{z}^2, r^1_{i\to j}, \boldsymbol{r}^1_{-(i\to j)}, \boldsymbol{r}^2, \boldsymbol{R}_{-(i,j)}, \beta) = \frac{(m_r^{-(i,j)}(k,l) + \beta)^{R(i,j)} (\overline{m}_r^{-(i,j)}(k,l) + \beta)^{1-R(i,j)}}{m_r^{-(i,j)}(k,l) + \overline{m}_r^{-(i,j)}(k,l) + 2\beta},$$

$$P(r^1_{i\to j} \,|\, \boldsymbol{r}^1_{i\to(-j)}, \beta^1) = \frac{(n_{\boldsymbol{r}^1_i}^{-(i,j)} + \beta^1)^{r^1_{i\to j}} (n_{\overline{\boldsymbol{r}}^1_i}^{-(i,j)} + \beta^1)^{1-r^1_{i\to j}}}{N^2 - 1 + 2\beta^1},$$

where $m_{\overline{r}}^{-(i,j)}$ and $\overline{m}_{\overline{r}}^{-(i,j)}$ denote the numbers of links and non-links, respectively, such that $r_{s,t} = 0$ for all pairs $(s, t) \neq (i, j)$; $m_r^{-(i,j)}(k, l)$ and $\overline{m}_r^{-(i,j)}(k, l)$ denote the numbers of links and non-links, respectively, such that $z^1_s = k$, $z^2_t = l$ and $r_{s,t} = 1$ for all pairs $(s, t) \neq (i, j)$; and $n_{\boldsymbol{r}^1_i}^{-(i,j)}$ and $n_{\overline{\boldsymbol{r}}^1_i}^{-(i,j)}$ denote the numbers of $r^1_{i\to t} = 1\{t \neq j\}$ and $r^1_{i\to t} = 0\{t \neq j\}$, respectively, within $\boldsymbol{r}^1_{i\to(-j)}$. Specifically, these counts are computed as follows:

$$n_{\boldsymbol{r}^1_i}^{-(i,j)} = \sum_{t\in T^2 : t\neq j} \left(r^1_{i\to t}\right), \qquad n_{\overline{\boldsymbol{r}}^1_i}^{-(i,j)} = \sum_{t\in T^2 : t\neq j} \left(1 - r^1_{i\to t}\right),$$

$$m_{\overline{r}}^{-(i,j)} = \sum\sum_{\substack{s\in T^1, t\in T^2: \\ (s,t)\neq(i,j)}} \left(R(s,t) \times (1 - f(r^1_{s\to t}, r^2_{t\to s}))\right),$$

$$\overline{m}_{\overline{r}}^{-(i,j)} = \sum\sum_{\substack{s\in T^1, t\in T^2: \\ (s,t)\neq(i,j)}} \left((1 - R(s,t)) \times (1 - f(r^1_{s\to t}, r^2_{t\to s}))\right),$$

$$m_r^{-(i,j)}(k,l) = \sum\sum_{\substack{s\in T^1, t\in T^2: \\ z^1_s = k, z^2_t = l, \\ (s,t)\neq(i,j)}} \left(R(s,t) \times f(r^1_{s\to t}, r^2_{t\to s})\right),$$

$$\overline{m}_r^{-(i,j)}(k,l) = \sum\sum_{\substack{s\in T^1, t\in T^2: \\ z^1_s = k, z^2_t = l, \\ (s,t)\neq(i,j)}} \left((1 - R(s,t)) \times f(r^1_{s\to t}, r^2_{t\to s})\right).$$

5 Experiments

In this section, we present our experimental results. To clarify the effectiveness of our subset selection mechanism, the performance of our rdIRM is compared with that of the original IRM. Through all the experiments, we assumed that the priors of all the binary variables in the generative models were uniform (Beta$(1.0, 1.0)$). In addition, we estimated the concentration parameters γ^1, γ^2 for the DPs assuming Gamma priors by sampling method presented in [8].

5.1 Experiments on Synthetic Datasets

We prepared 12 synthetic datasets. First, in accordance with the generative model of our rdIRM, we created five synthetic datasets, Data1(0.0), Data1(0.2), Data1(0.5), Data1(0.8), and Data1(1.0), where the numbers in parentheses indicate the background link probabilities η^0 for the datasets. We set the logical function f for the rdIRM to be a logical sum. The cluster assignments $\boldsymbol{z}^1$ and $\boldsymbol{z}^2$ were independently generated from fixed-dimensional multinomial distributions. The parameter values used for generating the datasets were $N^1 = N^2 = 200$, $\beta = (0.5, 0.5)$, and $\beta^1 = \beta^2 = (4.0, 3.0)$; the number of clusters were set as $K = 4$ and $L = 5$, and the parameters for the multinomials were $\boldsymbol{\pi}^1 = (0.4, 0.3, 0.2, 0.1)$ and $\boldsymbol{\pi}^2 = (0.33, 0.27, 0.20, 0.13, 0.07)$ for T^1 and T^2, respectively. Next, we also created five synthetic datasets in a similar manner (from Data2(0.0) to Data2(1.0)), except that we set the logical function f to be a logical product and we set both β^1 and β^2 to be $(4.0, 2.0)$. Finally, we created two datasets without background influences, (Data1(NULL) and Data2(NULL)). We applied the logical sum version of the rdIRM to Data1 and the logical product version to Data2.

We used three measures to evaluate clustering performance. One was the Adjusted Rand Index (ARI) [9], which is widely used for computing the similarity between true and estimated clustering results. The ARI takes a value in the range $0.0 - 1.0$, and takes a value of 1.0 when a clustering result is completely equivalent to the ground truth. Another was the number of erroneous estimated clusters (EC). We computed the average of these measures for the two sets T^1 and T^2. The rest was the test data log likelihood (TDLL), which indicates the predictive robustness of a generative model; we hid 1.0% of the observation during inference (keeping it small so that the latent cluster structure did not change), and measured the averaged log likelihood such that a hidden entry would take the actual value. A larger value is better, and a smaller one means that the model overfits the data. Finally, we repeated the experiment 10 times for each dataset using different random seeds to find an overall average.

Table 1 lists the computed measures. In the case of every dataset, except Data1(NULL) and Data2(NULL), we confirmed that the rdIRM outperformed the IRM. In particular, the rdIRM maintained good performance for sparse ($\eta^0 \approx 0.0$) or dense ($\eta^0 \approx 1.0$) data. We also list in Table 2 the maximum a posteriori (MAP) estimations of the background probability $\bar{\eta}^0$ and the estimated ratios of the foreground for synthetic datasets, except Data1(NULL) and Data2(NULL).

Table 1. ARI, EC, and TDLL on synthetic datasets

	ARI		EC		TDLL	
Dataset	IRM	rdIRM	IRM	rdIRM	IRM	rdIRM
Data1(NULL)	**1.000**	0.999	**0.000**	0.030	-0.302	**-0.261**
Data1(0.0)	0.712	**0.999**	0.678	**0.022**	-0.410	**-0.315**
Data1(0.2)	0.806	**1.000**	0.480	**0.010**	-0.432	**-0.363**
Data1(0.5)	0.868	**0.993**	0.270	**0.090**	-0.459	**-0.405**
Data1(0.8)	0.834	**0.999**	0.388	**0.013**	-0.462	**-0.385**
Data1(1.0)	0.806	**0.999**	0.435	**0.025**	-0.425	**-0.330**
Data2(NULL)	**1.000**	0.996	**0.000**	**0.000**	-0.316	**-0.232**
Data2(0.0)	0.629	**0.980**	1.053	**0.020**	-0.424	**-0.196**
Data2(0.2)	0.627	**0.913**	0.735	**0.105**	-0.576	**-0.431**
Data2(0.5)	0.759	**0.930**	0.488	**0.105**	-0.614	**-0.526**
Data2(0.8)	0.724	**0.917**	0.738	**0.097**	-0.558	**-0.438**
Data2(1.0)	0.644	**0.981**	0.910	**0.083**	-0.390	**-0.183**

Table 2. Estimated background probabilities ($\bar{\eta}^0$) and the FRs

Dataset	$\bar{\eta}^0$	FR
Data1(0.0)	0.0085	0.8484
Data1(0.2)	0.1970	0.8462
Data1(0.5)	0.4531	0.8588
Data1(0.8)	0.7674	0.8607
Data1(1.0)	0.9876	0.8611
Data2(0.0)	0.0022	0.4884
Data2(0.2)	0.2139	0.4548
Data2(0.5)	0.5033	0.4658
Data2(0.8)	0.7845	0.4397
Data2(1.0)	0.9872	0.4654

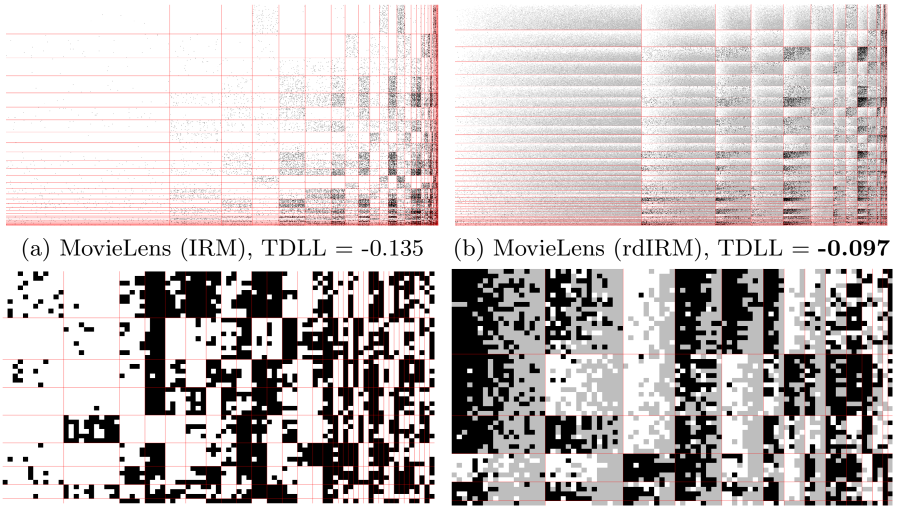
The ground truths of the foreground ratios (FRs) are 0.8197 for Data1 and 0.4622 for Data2. As the table shows, the rdIRM performs well in estimating the ground truths.

5.2 Experiments with Real-World Datasets

We applied the rdIRM to two real-world datasets. One was the "MovieLens" dataset[3], which contains a large number of user ratings of movies on a five-point scale. In our experiment, we created a binary relational dataset with a threshold that yields $R(i,j) = 1$ for ratings higher than 3 points and $R(i,j) = 0$ for all other ratings. That is, a relational value $R(i,j) = 1$ indicates that user i likes movie j. There are a total of 943 users and 1,682 movies in the dataset, and 3.5% of the relations are links. The other dataset was the "animal-feature" dataset [14], which includes relations between 50 animals and 85 features. Each feature is rated on a scale of 0–100 for each animal. We prepared the binary data with a threshold that yields $R(i,j) = 1$ for all ratings higher than the average of the entire set of ratings (20.79). That is, we used the relational value $R(i,j) = 1$ ($R(i,j) = 0$) to indicate that animal i has (does not have) feature j. In this dataset, 36.8% of the relations are links.

We used a logical sum version of the rdIRM for the MovieLens dataset and a logical product version for the animal-feature dataset. Our reason to use the former was that a user can watch any movie according to his or her preference, and similarly, movies are usually promoted independent of the users. Therefore, it seemed natural that the foreground (relevant relations) for the MovieLens dataset should be generated as per either the user's relevance ρ_i^1 or the movie's relevance ρ_j^2. On the other hand, animal features are acquired through evolution based on the specific type of animal. For example, aquatic features such as "swims" or "water" cannot be acquired by terrestrial animals. Therefore, the type of animal limits the features that it can acquire, and conversely, the type

[3] `http://movielens.umn.edu/`



(a) MovieLens (IRM), TDLL = -0.135 (b) MovieLens (rdIRM), TDLL = **-0.097**

(c) animal-feature (IRM), TDLL = -0.393 (d)animal-feature (rdIRM), TDLL = **-0.213**

Fig. 2. Clustering results for the real-world datasets. Black and white dots indicate links and non-links, respectively. In the rdIRM's results, gray dots indicate the areas that were estimated as background (irrelevant to cluster). Note that the objects within each cluster are sorted by descending order of the estimated relevances $\bar{\rho}_i^1$ and $\bar{\rho}_j^2$. "TDLL" is the computed test data log likelihood for each dataset.

of feature limits the types of animals that are related to that feature. Therefore, we used the logical product version of the rdIRM for the animal-feature dataset.

Figure 2 shows the clustering results and the computed TDLL for these real-world datasets. Figure 3 shows color maps for the estimated foreground probabilities $\bar{\eta}(k, l)$. The background probabilities η^0 that the rdIRM estimated were 0.0000 for the MovieLens dataset and 0.0036 for the animal-feature dataset. It can be seen that the original IRM organized many non-informative cluster-blocks, because the IRM considered that all the relations were relevant for cluster analysis. In contrast, the rdIRM found more vivid cluster structures owing to the use of our subset selection mechanism, which selects an informative subset of relations via the interaction of the objects' relevances. The computed TDLLs show that the rdIRM predicts hidden entries more robustly than does the original IRM for both datasets.

The left side of Table 3 lists the examples of the movie clusters produced by the rdIRM for the MovieLens dataset. In the columns for the number of links and $\bar{\rho}_j^2$, it can be seen that $\bar{\rho}_j^2$ tends to increase with the number of links. This means that we can regard the relevances as an indication of the popularity of the movies within the cluster. On the other hand, the original IRM treats all the links and non-links as relevant, so that the differences of the popularity of movies popularity affect the cluster assignment. The right side of Table 3 lists

Table 3. Examples of the clusters obtained by the rdIRM. The first column lists the object (Title/Feature). The second column lists the number of links related to the object (LNKS). The third column lists the estimated relevance ($\bar{\rho}_j^2$). The fourth column lists the cluster indices that were obtained by the original IRM (ZIRM). The left side tables are for the MovieLens dataset. The right side tables are for the animal-feature dataset.

Movie cluster 6 (contains 6 movies.)			
Title	LNKS	$\bar{\rho}_j^2$	ZIRM
Star Wars	501	0.9111	28
Return of the Jedi	379	0.5534	9
Independence Day	228	0.0921	25
Star Trek	220	0.1905	25

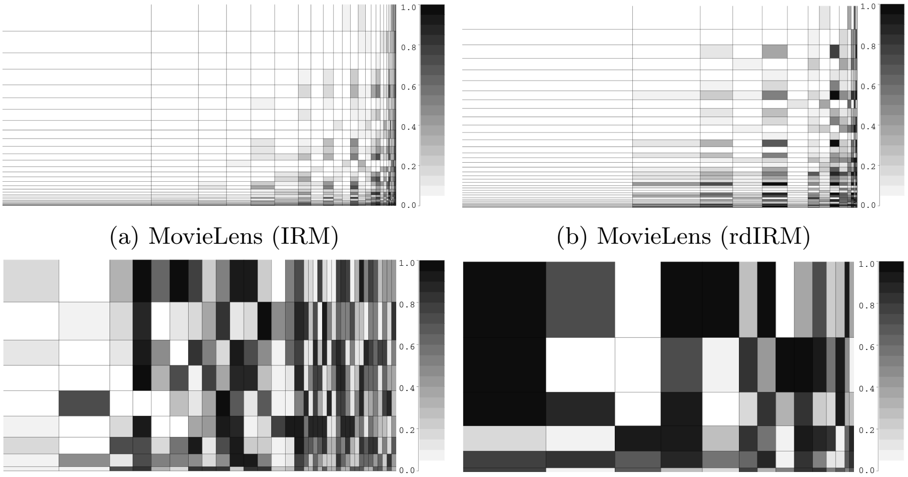
Movie cluster 7 (contains 40 movies.)			
Title	LNKS	$\bar{\rho}_j^2$	ZIRM
Silence of the Lambs	344	0.9132	26
Pulp Fiction	294	0.7598	26
Usual Suspects	232	0.6233	20
Alien	223	0.5164	20
Terminator	217	0.5608	20
Seven(Se7en)	167	0.3376	15

Movie Cluster 2 (contains 35 movies.)			
Title	LNKS	$\bar{\rho}_j^2$	ZIRM
W.W. & the Chocolate F.	189	0.7196	27
Birdcage	154	0.4762	17
Truth About Cats & Dogs	148	0.3386	17
Happy Gilmore	74	0.0360	2
Kingpin	73	0.1196	2

Feature cluster 1 (contains 10 features.)			
Feature	LNKS	$\bar{\rho}_j^2$	ZIRM
swims	10	0.9808	2
water	10	0.9808	2
coastal	8	0.9231	2
arctic	9	0.8846	2
flippers	7	0.8077	2

Feature cluster 5 (contains 15 features.)			
Feature	LNKS	$\bar{\rho}_j^2$	ZIRM
paws	19	0.9615	27
nestspot	31	0.9423	20
claws	19	0.9038	22
small	23	0.7885	21

Feature cluster 6 (contains 8 features.)			
Feature	LNKS	$\bar{\rho}_j^2$	ZIRM
meat	20	0.9808	17
fierce	21	0.9231	17
hunter	17	0.8846	17
stalker	10	0.4808	16
scavenger	6	0.1538	1
flys	1	0.0769	1



(a) MovieLens (IRM)

(b) MovieLens (rdIRM)

(c) animal-feature (IRM)

(d) animal-feature (rdIRM)

Fig. 3. The estimated foreground link probabilities $\bar{\eta}(k, l)$

the examples of the feature clusters obtained by the rdIRM for the animal-feature dataset. As with the results for the MovieLens dataset, we can see that the estimated ρ_j^2 tends to increase with the number of links. One interesting result produced by the rdIRM is that representative features such as "swims," "water," "paws," "nestspot" and "meet" were found to have high relevance in their clusters. From these results, we can say that the relevances estimated by the rdIRM indicate the popularities or representativeness of the objects. Consequently, the rdIRM finds clusters in terms of major categories by introducing the relevance-dependent subset selection mechanism.

6 Conclusions

In this paper, we proposed a new probabilistic relational model called the Relevance-Dependent Infinite Relational Model (rdIRM), which is suitable for noisy relational data analysis. The rdIRM parameterizes objects' relevances and incorporates a relevance-dependent subset selection mechanism, so that the rdIRM can estimate objects' relevances, and can co-cluster noisy relational data selecting only relevant relations that are informative for co-cluster analysis.

Our experiments with synthetic datasets confirmed that the rdIRM can find proper clusters in a noisy relational data, especially, in sparse or dense data. Moreover, our experiments on real-world datasets confirmed that the clusters obtained by the rdIRM represent major categories and that the estimated relevances can be viewed as the popularity or representativeness of the objects.

Our future research plans include extending the rdIRM so that it can also estimate the logical function f, which was given statically in this paper. We are also interested in applying our relevance-based subset selection mechanism to more advanced relational models, such as the mixed (or multiple) membership models [1,13], the hierarchical structure models [15], and the time-varying models [7].

References

1. Airoldi, E.M., Blei, D.M., Fienberg, S.E., Xing, E.P.: Mixed membership stochastic blockmodels. J. Mach. Learn. Res. 9, 1981–2014 (2008)
2. Aldous, D.: Exchangeability and related topics. In: Ecole d'Ete de Probabilities de Saint-Flour XIII, pp. 1–198 (1985)
3. Dhillon, I.S.: Co-clustering documents and words using bipartite spectral graph partitioning. In: Proc. SIGKDD, pp. 269–274 (2001)
4. Dhillon, I.S., Mallela, S., Modha, D.S.: Information-theoretic co-clustering. In: Proc. SIGKDD, pp. 89–98 (2003)
5. Ding, C., Li, T., Peng, W., Park, H.: Orthogonal nonnegative matrix t-factorizations for clustering. In: Proc. SIGKDD, pp. 126–135 (2006)
6. Ferguson, T.S.: A bayesian analysis of some nonparametric problems. The Annals of Statistics 1(2), 209–230 (1973)
7. Fu, W., Song, L., Xing, E.P.: Dynamic mixed membership blockmodel for evolving networks. In: Proc. ICML, pp. 329–336 (2009)

8. Hoff, P.D.: Subset clustering of binary sequences, with an application to genomic abnormality data. Biometrics 61(4), 1027–1036 (2005)
9. Hubert, L., Arabie, P.: Comparing partitions. J. of Classification 2(1), 193–218 (1985)
10. Ishiguro, K., Ueda, N., Sawada, H.: Subset infinite relational models. J. Mach. Learn. Res. - Proceedings Track 22, 547–555 (2012)
11. Kemp, C., Tenenbaum, J.B., Griffiths, T.L., Yamada, T., Ueda, N.: Learning systems of concepts with an infinite relational model. In: Proc. AAAI, vol. 1, pp. 381–388 (2006)
12. Liu, J.S.: The collapsed gibbs sampler in bayesian computations with applications to a gene regulation problem. J. Am. Stat. Assoc. 89(427), 958–966 (1994)
13. Mørup, M., Schmidt, M.N., Hansen, L.K.: Infinite multiple membership relational modeling for complex networks. In: Proc. MLSP, pp. 1–6 (2011)
14. Osherson, D.N., Stern, J., Wilkie, O., Stob, M., Smith, E.E.: Default probability. Cognitive Science 15(2), 251–269 (1991)
15. Roy, D., Kemp, C., Mansinghka, V., Tenenbaum, J.: Learning annotated hierarchies from relational data. In: Proc. NIPS, pp. 1185–1192 (2006)
16. Shafiei, M.M., Milios, E.E.: Latent dirichlet co-clustering. In: Proc. ICDM, pp. 542–551 (2006)

Density-Based Clustering Based on Hierarchical Density Estimates

Ricardo J.G.B. Campello*, Davoud Moulavi, and Joerg Sander**

Dept. of Computing Science, University of Alberta, Edmonton, AB, Canada
{rcampell,moulavi,jsander}@ualberta.ca

Abstract. We propose a theoretically and practically improved density-based, hierarchical clustering method, providing a clustering hierarchy from which a simplified tree of significant clusters can be constructed. For obtaining a "flat" partition consisting of only the most significant clusters (possibly corresponding to different density thresholds), we propose a novel cluster stability measure, formalize the problem of maximizing the overall stability of selected clusters, and formulate an algorithm that computes an optimal solution to this problem. We demonstrate that our approach outperforms the current, state-of-the-art, density-based clustering methods on a wide variety of real world data.

1 Introduction

Density-based clustering [1,2] is a popular clustering paradigm. However, the existing methods have a number of limitations: *(i)* Some methods (e.g., DBSCAN [3] and DENCLUE [4]) can *only* provide a "flat" (i.e. non-hierarchical) labeling of the data objects, based on a global density threshold. Using a single density threshold can often not properly characterize common data sets with clusters of very different densities and/or nested clusters. *(ii)* Among the methods that provide a clustering hierarchy, some (e.g., gSkeletonClu [5]) are not able to automatically simplify the hierarchy into an easily interpretable representation involving only the most significant clusters. *(iii)* Many hierarchical methods, including OPTICS [6] and gSkeletonClu, suggest only how to extract a flat partition by using a global cut/density threshold, which may not result in the most significant clusters if these clusters are characterized by *different* density levels. *(iv)* Some methods are limited to specific classes of problems, such as networks (gSkeletonClu), and point sets in the real coordinate space (e.g., DECODE [7], and Generalized Single-Linkage [8]). *(v)* Most methods depend on multiple, often critical input parameters (e.g., [3], [4], [7], [8], [9]).

In this paper, we propose a clustering approach that, to the best of our knowledge, is unique in that it does not suffer from any of these drawbacks. In detail, we make the following contributions: *(i)* We introduce a hierarchical clustering

[*] Currently on a sabbatical leave, he is originally from SCC/ICMC/USP, University of São Paulo at São Carlos, Brazil. He acknowledges FAPESP and CNPq.

[**] This work has been partly supported by NSERC.

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 160–172, 2013.

method, called HDBSCAN, which generates a complete density-based clustering hierarchy from which a simplified hierarchy composed only of the most significant clusters can be easily extracted. *(ii)* We propose a new measure of cluster stability for the purpose of extracting a set of significant clusters from possibly different levels of a simplified cluster tree produced by HDBSCAN. *(iii)* We formulate the task of extracting a set of significant clusters as an optimization problem in which the overall stability of the composing clusters is maximized. *(iv)* We propose an algorithm that finds the globally optimal solution to this problem. *(v)* We demonstrate the advancement in density-based clustering that our approach represents on a variety of real world data sets.

The remainder of this paper is organized as follows. We discuss related work in Section 2. In Section 3, we redefine DBSCAN, and we propose the algorithm HDBSCAN in Section 4. In Section 5, we introduce a new measure of cluster stability, propose the problem of extracting an optimal set of clusters from a cluster tree, and give an algorithm to solve this problem. Section 6 presents an extensive experimental evaluation, and Section 7 concludes the paper.

2 Related Work

Apart from methods aimed at getting approximate estimates of level sets and density-contour trees for continuous-valued p.d.f. — e.g., see [8] and references therein — not much attention has been given to hierarchical density-based clustering in more general data spaces. The works most related to ours are those in [6,9,5,10]. In [6], a post-processing procedure to extract a simplified cluster tree from the reachability plot produced by the OPTICS algorithm was proposed. This procedure did not become as popular as OPTICS itself, probably because it is very sensitive to the choice of a critical parameter that cannot easily be determined or understood. Moreover, no automatic method to extract a flat clustering solution based on local cuts in the obtained tree was described. In [9], an improved method to extract trees of significant clusters from reachability plots was proposed that is less sensitive to the user settings than the original method in [6]. However, this method is based on heuristics with embedded threshold values that can strongly affect the results, and the problem of extracting a flat solution from local cuts in the cluster tree was practically untouched; the only mentioned (ad-hoc) approach was to arbitrarily take all the leaf clusters and discard the others. In [5], the original findings from [6,9,11] were recompiled in the particular context of community discovery in complex networks. However, no mechanism to extract a simplified cluster tree from the resulting (single-linkage-like) clustering dendrogram was adopted, and only a method producing a global cut through the dendrogram was described. The algorithm AUTO-HDS [10] is, like our method, based on a principle used to simplify clustering hierarchies, which in part refers back to the work of [12]. The clustering hierarchy obtained by AUTO-HDS is typically a subset of the one obtained by our method HDBSCAN. Conceptually, it is equivalent to a sampling of the HDBSCAN hierarchical levels, from top to bottom, at a geometric rate controlled by a user-defined parameter, r_{shave}. Such a sampling can lead to an underestimation of the stability of clusters or even to missed clusters, and these side effects can only be prevented if $r_{shave} \to 0$. In this case, however, the asymptotic running time of AUTO-HDS is $O(n^3)$ [13] (in contrast to $O(n^2 \log n)$ for "sufficiently large" values of r_{shave}). In addition, the stability measure used in AUTO-HDS has the undesirable property that the stability value for a cluster in one branch of the hierarchy can be affected by the density and cardinality of other clusters lying on different branches. AUTO-HDS also attempts to perform local cuts through the hierarchy in order to extract a flat clustering solution, but it uses a greedy heuristic for selecting clusters that may give suboptimal results in terms of an overall stability.

3 DBSCAN Revisited — The Algorithm DBSCAN*

Let $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$ be a data set of n objects, and let D be an $n \times n$ matrix containing the pairwise distances $d(\mathbf{x}_p, \mathbf{x}_q)$, $\mathbf{x}_p, \mathbf{x}_q \in \mathbf{X}$, for a metric distance $d(\cdot, \cdot)$.[1] We define density-based clusters based on *core objects* alone:

Definition 1. (Core Object): An object $\mathbf{x}_p$ is called a *core object* w.r.t. ε and m_{pts} if its ε-neighborhood contains at least m_{pts} many objects, i.e., if $|\mathbf{N}_\varepsilon(\mathbf{x}_p)| \geq m_{pts}$, where $\mathbf{N}_\varepsilon(\mathbf{x}_p) = \{\mathbf{x} \in \mathbf{X} \mid d(\mathbf{x}, \mathbf{x}_p) \leq \varepsilon\}$ and $|\cdot|$ denotes cardinality. An object is called *noise* if it is not a core object.

Definition 2. (ε-Reachable): Two *core* objects $\mathbf{x}_p$ and $\mathbf{x}_q$ are *ε-reachable* w.r.t. ε and m_{pts} if $\mathbf{x}_p \in \mathbf{N}_\varepsilon(\mathbf{x}_q)$ and $\mathbf{x}_q \in \mathbf{N}_\varepsilon(\mathbf{x}_p)$.

Definition 3. (Density-Connected): Two *core* objects $\mathbf{x}_p$ and $\mathbf{x}_q$ are *density-connected* w.r.t. ε and m_{pts} if they are directly or transitively ε-reachable.

Definition 4. (Cluster): A *cluster* $\mathbf{C}$ w.r.t. ε and m_{pts} is a non-empty maximal subset of $\mathbf{X}$ such that every pair of objects in $\mathbf{C}$ is density-connected.

Based on these definitions, we can devise an algorithm DBSCAN* (similar to DBSCAN) that conceptually finds clusters as the connected components of a graph in which the objects of $\mathbf{X}$ are vertices and every pair of vertices is adjacent if and only if the corresponding objects are ε-reachable w.r.t. user-defined parameters ε and m_{pts}. Non-core objects are labeled as noise.

Note that the original definitions of DBSCAN also include the concept of *border* objects, i.e., non-core objects that are within the ε-neighborhood of a core object. Our new definitions are more consistent with a statistical interpretation of clusters as connected components of a level set of a density (as defined, e.g., in [14]), since border objects do not technically belong to the level set (their estimated density is below the threshold). The new definitions also allow a precise relationship between DBSCAN* and its hierarchical version, to be discussed next. This was only approximately possible between DBSCAN and OPTICS.

[1] The matrix D is not required if distances $d(\cdot, \cdot)$ can be computed from $\mathbf{X}$ on demand.

4 Hierarchical DBSCAN* — HDBSCAN

In this section, we introduce a hierarchical clustering method, HDBSCAN, which can be seen as a conceptual and algorithmic improvement over OPTICS. Our method has as its single input parameter a value for m_{pts}, which is a classic smoothing factor in density estimates whose behavior is well understood (methods with a corresponding parameter, e.g., [6,10,7,8], are quite robust to it). Different density levels in the resulting density-based cluster hierarchy will then correspond to different values of the radius ε.

For a proper formulation of the density-based hierarchy w.r.t. a value of m_{pts}, we define the notions of core distance and a symmetric reachability distance (following the definition used in [11]), a new notion of *ε-core objects*, as well as the notion of a *conceptual*, transformed proximity graph, which will help us to explain a density-based clustering hierarchy.

Definition 5. (Core Distance): The *core distance* of an object $\mathbf{x}_p \in \mathbf{X}$ w.r.t. m_{pts}, $d_{core}(\mathbf{x}_p)$, is the distance from $\mathbf{x}_p$ to its m_{pts}-nearest neighbor (incl. $\mathbf{x}_p$).

Definition 6. (ε-Core Object): An object $\mathbf{x}_p \in \mathbf{X}$ is called an *ε-core* object for every value of ε that is greater than or equal to the core distance of $\mathbf{x}_p$ w.r.t. m_{pts}, *i.e.*, if $d_{core}(\mathbf{x}_p) \leq \varepsilon$.

Definition 7. (Mutual Reachability Distance): The *mutual reachability distance* between two objects $\mathbf{x}_p$ and $\mathbf{x}_q$ in $\mathbf{X}$ w.r.t. m_{pts} is defined as $d_{mreach}(\mathbf{x}_p, \mathbf{x}_q) = \max\{d_{core}(\mathbf{x}_p), d_{core}(\mathbf{x}_q), d(\mathbf{x}_p, \mathbf{x}_q)\}$.

Definition 8. (Mutual Reachability Graph): It is a complete graph, $G_{m_{pts}}$, in which the objects of $\mathbf{X}$ are vertices and the weight of each edge is the mutual reachability distance (w.r.t. m_{pts}) between the respective pair of objects.

Let $G_{m_{pts},\varepsilon} \subseteq G_{m_{pts}}$ be the graph obtained by removing all edges from $G_{m_{pts}}$ having weights greater than ε. From Definitions 4, 6, and 8, it is straightforward to infer that clusters according to DBSCAN* w.r.t. m_{pts} and ε are the connected components of ε-core objects in $G_{m_{pts},\varepsilon}$; the remaining objects are noise. Consequently, all DBSCAN* partitions for $\varepsilon \in [0, \infty)$ can be produced in a nested, *hierarchical* way by removing edges in decreasing order of weight from $G_{m_{pts}}$.

Proposition 1. *Let $\mathbf{X}$ be a set of n objects described in a metric space by $n \times n$ pairwise distances. The partition of this data obtained by DBSCAN* w.r.t m_{pts} and ε is identical to the one obtained by first running Single-Linkage over the transformed space of mutual reachability distances, then, cutting the resulting dendrogram at level ε of its scale, and treating all resulting singletons with $d_{core}(\mathbf{x}_p) > \varepsilon$ as a single class representing "Noise".*

Proof. Proof sketch as per discussion above, after Definition 8. □

Proposition 1 states that we could implement a hierarchical version of DBSCAN* by an algorithm that first computes a Single-Linkage hierarchy on the space of

Algorithm 1. HDBSCAN main steps

```
1. Compute the core distance w.r.t. $m_{pts}$ for all data objects in $\mathbf{X}$.
2. Compute an MST of $G_{m_{pts}}$, the Mutual Reachability Graph.
3. Extend the MST to obtain $MST_{ext}$, by adding for each vertex a "self edge"
with the core distance of the corresponding object as weight.
4. Extract the HDBSCAN hierarchy as a dendrogram from $MST_{ext}$:
    4.1 For the root of the tree assign all objects the same label (single "cluster").
    4.2 Iteratively remove all edges from $MST_{ext}$ in decreasing order of weights
        (in case of ties, edges must be removed simultaneously):
        4.2.1 Before each removal, set the dendrogram scale value of the current
        hierarchical level as the weight of the edge(s) to be removed.
        4.2.2 After each removal, assign labels to the connected component(s)
        that contain(s) the end vertex(-ices) of the removed edge(s), to obtain
        the next hierarchical level: assign a new cluster label to a component if
        it still has at least one edge, else assign it a null label ("noise").
```

transformed distances (i.e., mutual reachability distances) and, then, processes this hierarchy to identify connected components and noise objects at each level. Here, we propose a more efficient and elegant equivalent solution.

A density-based cluster hierarchy has to represent the fact that an object o is noise below the level l that corresponds to o's core distance. To represent this in a dendrogram, we propose to include an additional dendrogram node for o at level l representing the cluster containing o at that level and higher. To directly construct such a hierarchy, we propose an extension of a Minimum Spanning Tree (MST) of the Mutual Reachability Graph $G_{m_{pts}}$, from which we then can construct the extended dendrogram by removing edges in decreasing order of weights. More precisely, we extend the MST with edges connecting each vertex o to itself (self-loops), where the edge weight is set to the core distance of o. These "self edges" will then be considered when removing edges.

Algorithm 1 shows the pseudo-code for HDBSCAN, which has as inputs a value for m_{pts} and the data set $\mathbf{X}$. It produces a clustering tree that contains all partitions obtainable by DBSCAN* (w.r.t. m_{pts}) in a hierarchical, nested way. It also contains nodes that indicate when an isolated object changes from core (i.e., dense) to noise. The result is called the "HDBSCAN hierarchy". Using an implementation of Prim's algorithm based on an ordinary list search (instead of a heap) to construct the MST, the method can be fully implemented in $O(dn^2)$ running time, where d is the number of data attributes. Also, by noticing that only the currently processed hierarchical level is needed at any point in time, the algorithm needs to keep in main memory essentially the data set $\mathbf{X}$ and the extended MST that can be constructed directly from it, which requires $O(dn)$ space. If a data matrix D is provided as input in lieu of $\mathbf{X}$, the algorithm requires $O(n^2)$ space instead, but its time complexity reduces to $O(n^2)$.

Algorithm 2. HDBSCAN step 4.2.2 with (optional) parameter $m_{clSize} \geq 1$

```
4.2.2 After each removal (to obtain the next hierarchical level), process one at a
time each cluster that contained the edge(s) just removed, by relabeling its
resulting connected subcomponent(s):
    Label spurious subcomponents as noise by assigning them the null label. If all
    subcomponents of a cluster are spurious, then the cluster has disappeared.
    Else, if a single subcomponent of a cluster is not spurious, keep its original
    cluster label (cluster has just shrunk).
    Else, if two or more subcomponents of a cluster are not spurious, assign new
    cluster labels to each of them ("true" cluster split).
```

4.1 Hierarchy Simplification

The HDBSCAN hierarchy can easily be visualized as a traditional dendrogram or related representations. However, these plots are not easy to interpret or process for large and "noisy" data sets, so it is a fundamental problem to extract from a dendrogram a summarized tree of only "significant" clusters. We propose a simplification of the HDBSCAN hierarchy based on a fundamental observation about estimates of the level sets of continuous-valued probability density functions (p.d.f.), which refers back to Hartigan's concept of *rigid clusters* [14], and which has also been employed similarly by Gupta *et al.* in [10]. For a given p.d.f., there are only three possibilities for the evolution of the connected components of a continuous density level set when increasing the density level (decreasing ε in our context) [12]: *(i)* the component shrinks but remains connected, up to a density threshold at which either *(ii)* the component is divided into smaller ones, or *(iii)* it disappears. This observation can be applied to the HDBSCAN hierarchy to select only those hierarchical levels in which new clusters arise by a "true" split of a cluster, or in which clusters disappear; these are the levels in which the most significant changes in the clustering structure occur. When decreasing ε, the ordinary removal of noise objects from a cluster is not a "true" split; a cluster only shrinks in this case, so it should keep the same label.

We can generalize this idea by setting a minimum cluster size, a commonly used practice in real cluster analysis (see, e.g., the notion of a *particle* in AUTO-HDS [10]). With a minimum cluster size, $m_{clSize} \geq 1$, components with fewer than m_{clSize} objects are disregarded, and their disconnection from a cluster does not establish a "true" split. We can adapt HDBSCAN accordingly by changing Step 4.2.2 of Algorithm 1 as shown in Algorithm 2: a connected component is deemed *spurious* if it has fewer than m_{clSize} objects or, for $m_{clSize} = 1$, if it is an isolated, non-dense object (no edges). Any spurious component is labeled as noise and its removal from a bigger component is not considered as a cluster split. In practice, this can reduce the size of the hierarchy dramatically.

The optional parameter m_{clSize} represents an independent control for the smoothing of the resulting cluster tree, in addition to m_{pts}. To make HDBSCAN more similar to previous density-based approaches and to simplify its use, we can set $m_{clSize} = m_{pts}$, which turns m_{pts} into a single parameter that acts as both a smoothing factor and a threshold for the cluster size.

5 Optimal Non-hierarchical Clustering

In many applications a user is interested in extracting a "flat" partition of the data, consisting of the prominent clusters. Those clusters, however, may have very different local densities and may not be detectable by a single, global density threshold, i.e., global cut through a hierarchical cluster representation. In this section, we describe an algorithm that provides the optimal global solution to the formal optimization problem of maximizing the overall stability of the set of clusters extracted from the HDBSCAN hierarchy.

5.1 Cluster Stability

Without loss of generality, let us initially consider that the data objects are described by a single continuous-valued attribute x. Following Hartigan's model [14], the density-contour clusters of a given density $f(x)$ on $\Re$ at a given density level λ are the maximal connected subsets of the level set defined as $\{x \,|\, f(x) \geq \lambda\}$. Most density-based clustering algorithms are to some extent based on this concept. The differences lie in the way the density $f(x)$ and the maximal connected subsets are estimated, e.g., DBSCAN* estimates the density-contour clusters for a density threshold $\lambda = 1/\varepsilon$ and a (non-normalized) K-NN estimate (for $K = m_{pts}$) of the density $f(x)$, given by $1/d_{core}(x)$.

HDBSCAN produces all possible DBSCAN* solutions w.r.t. a given value of m_{pts} and all thresholds $\lambda = 1/\varepsilon$ in $[0, \infty)$. Intuitively, when increasing λ (i.e., decreasing ε), clusters get smaller and smaller, until they disappear or break into sub-clusters; more prominent clusters will "survive" longer after they appear. To formalize this concept, we adapt the notion of *excess of mass* [15]: Imagine increasing the density level λ, and assume that a density-contour cluster $\mathbf{C}_i$ appears at level $\lambda_{min}(\mathbf{C}_i)$. The excess of mass of $\mathbf{C}_i$ is defined in Equation (1), and illustrated in Figure 1, where the darker shaded areas represent the excesses of mass of three clusters, $\mathbf{C}_3$, $\mathbf{C}_4$, and $\mathbf{C}_5$. The excess of mass of $\mathbf{C}_2$ (not highlighted in the figure) encompasses those of its descendants $\mathbf{C}_4$ and $\mathbf{C}_5$.

$$E(\mathbf{C}_i) = \int_{x \in \mathbf{C}_i} \Big(f(x) - \lambda_{min}(\mathbf{C}_i) \Big)\, dx \tag{1}$$

The excess of mass exhibits a monotonic behavior along any branch of the hierarchical cluster tree. As a consequence, this measure cannot be used to compare the stabilities of nested clusters, such as $\mathbf{C}_2$ against $\mathbf{C}_4$ and $\mathbf{C}_5$. To be able to do so, we introduce here the notion of *Relative Excess of Mass* of a cluster $\mathbf{C}_i$, which appears at level $\lambda_{min}(\mathbf{C}_i)$, as:

$$E_R(\mathbf{C}_i) = \int_{x \in \mathbf{C}_i} \Big(\lambda_{max}(x, \mathbf{C}_i) - \lambda_{min}(\mathbf{C}_i) \Big)\, dx \tag{2}$$

where $\lambda_{max}(x, \mathbf{C}_i) = \min\{f(x), \lambda_{max}(\mathbf{C}_i)\}$, and $\lambda_{max}(\mathbf{C}_i)$ is the density level at which $\mathbf{C}_i$ is split or disappears. For example, for cluster $\mathbf{C}_2$ in Figure 1 it follows that $\lambda_{max}(\mathbf{C}_2) = \lambda_{min}(\mathbf{C}_4) = \lambda_{min}(\mathbf{C}_5)$. The corresponding relative excess of mass is represented by the lighter shaded area in Figure 1.

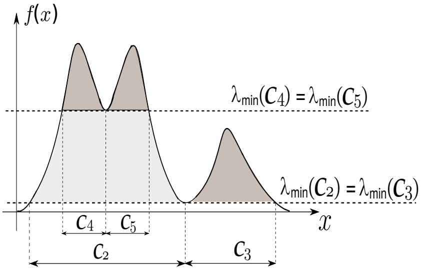


Fig. 1. Illustration of a density function, clusters, and excesses of mass

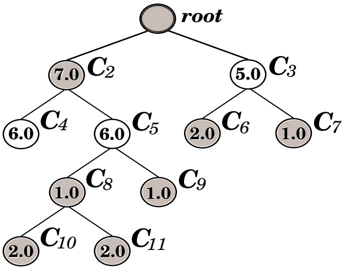


Fig. 2. Illustration of the optimal selection of clusters from a given cluster tree

For a HDBSCAN hierarchy, where we have a finite data set $\mathbf{X}$, cluster labels, and density thresholds associated with each hierarchical level, we can adapt Equation (2) to define the *stability* of a cluster $\mathbf{C}_i$ as:

$$S(\mathbf{C}_i) = \sum_{\mathbf{x}_j \in \mathbf{C}_i} \Big(\lambda_{max}(\mathbf{x}_j, \mathbf{C}_i) - \lambda_{min}(\mathbf{C}_i)\Big) = \sum_{\mathbf{x}_j \in \mathbf{C}_i} \left(\frac{1}{\varepsilon_{min}(\mathbf{x}_j, \mathbf{C}_i)} - \frac{1}{\varepsilon_{max}(\mathbf{C}_i)}\right) \quad (3)$$

where $\lambda_{min}(\mathbf{C}_i)$ is the minimum density level at which $\mathbf{C}_i$ exists, $\lambda_{max}(\mathbf{x}_j, \mathbf{C}_i)$ is the density level beyond which object $\mathbf{x}_j$ no longer belongs to cluster $\mathbf{C}_i$, and $\varepsilon_{max}(\mathbf{C}_i)$ and $\varepsilon_{min}(\mathbf{x}_j, \mathbf{C}_i)$ are the corresponding values for the threshold ε.

5.2 Optimization Algorithm

Let $\{\mathbf{C}_2, \cdots, \mathbf{C}_\kappa\}$ be the collection of all clusters in the simplified cluster hierarchy (tree) generated by HDBSCAN, except the root $\mathbf{C}_1$, and let $S(\mathbf{C}_i)$ denote the stability value of each cluster. The goal is to extract the most "prominent" clusters (plus possibly noise) as a *"flat"*, non-overlapping partition. This task can be formulated as an optimization problem with the objective of maximizing the sum of stabilities of the extracted clusters in the following way:

$$\max_{\delta_2, \dots, \delta_\kappa} \quad J = \sum_{i=2}^{\kappa} \delta_i \, S(\mathbf{C}_i)$$

$$\text{subject to} \begin{cases} \delta_i \in \{0,1\}, \;\; i = 2, \cdots, \kappa \\ \displaystyle\sum_{j \,\in \mathbf{I}_h} \delta_j = 1, \forall h \;\in\; \mathbf{L} \end{cases} \quad (4)$$

where δ_i $(i = 2, \cdots, \kappa)$ indicates whether cluster $\mathbf{C}_i$ is included in the flat solution ($\delta_i = 1$) or not ($\delta_i = 0$), $\mathbf{L} = \{h \mid \mathbf{C}_h$ is a leaf cluster$\}$ is the set of indexes of leaf clusters, and $\mathbf{I}_h = \{j \mid j \neq 1$ and $\mathbf{C}_j$ is ascendant of $\mathbf{C}_h$ (h included)$\}$ is the set of indexes of all clusters on the path from $\mathbf{C}_h$ to the root (excluded). The constraints prevent nested clusters on the same path to be selected.

Algorithm 3. Solution to Problem (4)

```
1. Initialize $\delta_2 = \cdots = \delta_\kappa = 1$, and, for all leaf nodes, set $\hat{S}(\mathbf{C}_h) = S(\mathbf{C}_h)$.
2. Starting from the deepest levels, do bottom-up (except for the root):
    2.1 If $S(\mathbf{C}_i) < \hat{S}(\mathbf{C}_{i_l}) + \hat{S}(\mathbf{C}_{i_r})$, set $\hat{S}(\mathbf{C}_i) = \hat{S}(\mathbf{C}_{i_l}) + \hat{S}(\mathbf{C}_{i_r})$ and set $\delta_i = 0$.
    2.2 Else: set $\hat{S}(\mathbf{C}_i) = S(\mathbf{C}_i)$ and set $\delta_{(\cdot)} = 0$ for all clusters in $\mathbf{C}_i$'s subtrees.
```

To solve Problem (4), we process every node except the root, starting from the leaves (bottom-up), deciding at each node $\mathbf{C}_i$ whether $\mathbf{C}_i$ or the best-so-far selection of clusters in $\mathbf{C}_i$'s subtrees should be selected. To be able to make this decision locally at $\mathbf{C}_i$, we propagate and update the total stability $\hat{S}(\mathbf{C}_i)$ of clusters selected in the subtree rooted at $\mathbf{C}_i$ in the following, recursive way:

$$\hat{S}(\mathbf{C}_i) = \begin{cases} S(\mathbf{C}_i), & \text{if } \mathbf{C}_i \text{ is a leaf node} \\ \max\{S(\mathbf{C}_i), \hat{S}(\mathbf{C}_{i_l}) + \hat{S}(\mathbf{C}_{i_r})\} & \text{if } \mathbf{C}_i \text{ is an internal node} \end{cases} \quad (5)$$

where $\mathbf{C}_{i_l}$ and $\mathbf{C}_{i_r}$ are the left and right children of $\mathbf{C}_i$ (for the sake of simplicity, we discuss the case of binary trees; the generalization to n-ary trees is trivial).

Algorithm 3 gives the pseudo-code for finding the optimal solution to Problem (4). Figure 2 illustrates the algorithm. Clusters $\mathbf{C}_{10}$ and $\mathbf{C}_{11}$ together are better than $\mathbf{C}_8$, which is then discarded. However, when the set $\{\mathbf{C}_{10}, \mathbf{C}_{11}, \mathbf{C}_9\}$ is compared to $\mathbf{C}_5$, they are discarded as $\mathbf{C}_5$ is better. Clusters $\{\mathbf{C}_4\}$ and $\{\mathbf{C}_5\}$ are better than $\mathbf{C}_2$, and $\mathbf{C}_3$ is better than $\{\mathbf{C}_6, \mathbf{C}_7\}$, so that in the end, only clusters $\mathbf{C}_3$, $\mathbf{C}_4$, and $\mathbf{C}_5$ remain, which is the optimal solution to (4) with $J = 17$.

Step 2.2 of Algorithm 3 can be implemented in a more efficient way by not setting $\delta_{(\cdot)}$ values to 0 for discarded clusters down in the subtrees (which could happen multiple times). Instead, in a simple post-processing procedure, the tree can be traversed top-down in order to find, for each branch, the shallowest cluster that has not been discarded ($\delta_{(\cdot)} = 1$). Thus, Algorithm 3 can be implemented with two traversals through the tree, one bottom-up and another one top-down. This results in an asymptotic complexity of $O(\kappa)$, both in terms of running time and memory space, where κ is the number of clusters in the simplified tree (which is typically much smaller than the number of data objects).

6 Experiments and Discussion

Data Sets. We report the performance on 9 individual data sets plus the average performance on 2 collections of data sets, representing a large variety of application domains and data characteristics (no. of objects, dimensionality, no. of clusters, and distance function). The first three data sets ("CellCycle-237", "CellCycle-384", and "YeastGalactose") represent gene-expression data. CellCycle-237 and CellCycle-384 were made public by Yeung et al. [16]; they contain 237 resp. 384 objects (genes), 4 resp. 5 known classes, and have both 17 dimensions (conditions). YeastGalactose contains a subset of 205 objects (genes) and 20 dimensions (conditions) used in [17], with 4 known classes. For these data sets we used Euclidean distance on the z-score normalized objects, which is equivalent to using Pearson correlation on the original data. The next three data sets are the Wine, Glass, and Iris from the UCI Repository [18], containing 178, 214, resp. 150 objects in 13, 9, resp. 4 dimensions, with 3, 7, resp. 3 classes. For these data sets we used Euclidean distance. The last three individual data sets consist of very high dimensional representations of text documents. In particular, "Articles-1442-5" and "Articles-1442-80", made available upon request by Naldi et al. [19], are formed by 253 articles represented by 4636 and 388 dimensions, respectively. "Cbrilpirivson" [20] is composed of 945 articles represented by 1431 dimensions and is available at http://infoserver.lcad.icmc.usp.br/infovis2/PExDownload. The number of classes in all three document data sets is 5, and we used the Cosine measure as dissimilarity function. In addition to individual data sets we also report *average* performance on two collections of data sets, which are based on the Amsterdam Library of Object Images (ALOI) [21]. Image sets were created as in [22] by randomly selecting k ALOI image categories as class labels 100 times for each $k = 2, 3, 4, 5$, then sampling (without replacement), each time, 25 images from each of the k selected categories, thus resulting in 400 sets, each of which contains 2, 3, 4, or 5 clusters and 50, 75, 100, or 125 images (objects). The images were represented using six different descriptors: color moments (144 attributes), texture statistics from the gray-level co-occurrence matrix (88 attributes), Sobel edge histogram (128 attributes), 1st order statistics from the gray-level histogram (5 attributes), gray-level run-length matrix features (44 attributes), and gray-level histogram (256 attributes). We report *average* clustering results for the texture statistics, denoted by "ALOI-TS88", which is typical for the individual descriptors. We also show results for a 6-dimensional representation combining the first principal component extracted from each of the 6 descriptors using PCA, denoted by "ALOI-PCA". We used Euclidean distance in both cases.

Algorithms. Our method, denoted here by "HDBSCAN(EOM)" (EOM refers to cluster extraction based on Excess Of Mass), is compared with: *(i)* AUTO-HDS [10]; and *(ii)* a method referred to here as "OPTICS(AutoCl)", which consists of first running OPTICS, and then using the method proposed by Sander *et al.* [9] to extract a flat partitioning. We set m_{pts} (n_{ε} in AUTO-HDS, *MinPts* in OPTICS) equal to 4 in all experiments. The speed-up control value ε in OPTICS was set to "infinity", thereby eliminating its effect. For AUTO-HDS, we set the additional parameters r_{shave} to 0.03 (following the authors' suggestion to use values between 0.01 and 0.05) and *particle size*, n_{part}, to $m_{pts} - 1$. The corresponding parameter m_{clSize} in HDBSCAN was set equivalently to m_{pts}.[2]

Quality Measures. The measures we report are the Overall F-measure [23] and Adjusted Rand Index [24], denoted by "FScore" resp. "ARI", which are measures commonly used in the literature. In addition, we also report the fraction of objects assigned to clusters (as opposed to noise), denoted by "%covered".

[2] We also tried other values of m_{pts}, r_{shave}, and n_{part}/m_{clSize}, with similar results.

Table 1. Results for all data sets

	OPTICS(AutoCl)			AUTO-HDS			HDBSCAN(EOM)		
Data Set	ARI	FScore	%covered	ARI	FScore	%covered	ARI	FScore	%covered
CellCycle-237	**0.62**	**0.71**	**0.80**	0.04	0.29	0.37	0.48	0.66	0.65
CellCycle-384	0	0.37	**1**	**0.35**	**0.50**	0.46	**0.35**	**0.50**	0.47
YeastGalactose	**0.96**	**0.97**	**0.96**	0.94	0.96	**0.96**	0.94	**0.97**	**0.96**
Wine	0.16	0.48	0.77	0.12	0.37	0.72	**0.29**	**0.62**	**0.97**
Glass	0.23	0.49	0.76	0.12	0.37	0.45	**0.24**	**0.51**	**0.79**
Iris	0.33	0.61	0.83	0.11	0.40	0.46	**0.57**	**0.78**	**1**
Articles-1442-80	0.91	0.96	0.96	0.66	0.74	0.76	**0.93**	**0.97**	**0.98**
Articles-1442-5	0.89	0.94	0.93	0.60	0.76	0.73	**0.90**	**0.95**	**0.94**
Cbrilpirivson	0.01	0.07	0.11	0.04	0.21	0.34	**0.19**	**0.47**	**0.48**
ALOI-TS88	0.45	0.67	0.74	0.50	0.70	0.78	**0.63**	**0.79**	**0.85**
ALOI-PCA	0.61	0.78	0.83	0.56	0.74	0.81	**0.72**	**0.85**	**0.91**

Clustering Results. The results obtained in our experiments are shown in Table 1. The highest obtained values for each data set are highlighted in bold. Note that HDBSCAN(EOM) outperforms the other two methods in a large majority of the data sets (in many cases by a large margin) and, in almost all cases, it covers a larger fraction of objects while having also high FScore and ARI values. A high fraction of clustered objects is only good when also the clustering quality is high. E.g., for CellCycle-384, OPTICS(AutoCl) covers 100% of the data, but with an ARI of 0, a meaningless clustering. In one of the only two cases where HDBSCAN(EOM) does not perform best, YeastGalactose, its ARI is very close to (and its FScore matches that of) the “winner”, OPTICS(AutoCl).

The collections of image data sets, ALOI-TS88 and ALOI-PCA, allowed us to perform paired t-tests with respect to ARI and FScore, confirming that the observed differences in performance between all pairs of methods is statistically significant at the $\alpha = 0.01$ significance level. This means that the methods are doing indeed different things, and, in particular, that HDBSCAN(EOM) significantly outperforms the others on these data set collections.

7 Final Remarks

A novel density-based clustering approach has been introduced that provides: *(i)* a complete density-based clustering hierarchy representing all possible DBSCAN-like solutions for an infinite range of density thresholds and from which a simplified tree of significant clusters can be extracted; and *(ii)* a flat partition composed of clusters extracted from optimal local cuts through the cluster tree. An extensive experimental evaluation on a wide variety of real world data sets has shown that our method performs significantly better and more robust than state-of-the-art methods. Our work lends itself to a number of interesting challenges for future work, which includes integration of semi-supervision and the consideration of subspaces.

References

1. Tan, P., Steinbach, M., Kumar, V.: Introduction to Data Mining. Addison-Wesley (2006)
2. Sander, J.: Density-based clustering. In: Sammut, C., Webb, G.I. (eds.) Encyclopedia of Machine Learning, pp. 270–273. Springer (2010)
3. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: Int. Conf. Knowl. Discovery and Data Mining (1996)
4. Hinneburg, A., Keim, D.A.: A general approach to clustering in large databases with noise. Knowl. and Info. Sys. 5, 387–415 (2003)
5. Sun, H., Huang, J., Han, J., Deng, H., Zhao, P., Feng, B.: gSkeletonClu: Density-based network clustering via structure-connected tree division or agglomeration. In: IEEE Int. Conf. Data Mining (2010)
6. Ankerst, M., Breunig, M.M., Kriegel, H.P., Sander, J.: Optics: ordering points to identify the clustering structure. SIGMOD Rec. 28, 49–60 (1999)
7. Pei, T., Jasra, A., Hand, D., Zhu, A.X., Zhou, C.: Decode: a new method for discovering clusters of different densities in spatial data. Data Mining and Knowl. Discovery 18, 337–369 (2009)
8. Stuetzle, W., Nugent, R.: A generalized single linkage method for estimating the cluster tree of a density. J. Comp. and Graph. Stat. 19(2), 397–418 (2010)
9. Sander, J., Qin, X., Lu, Z., Niu, N., Kovarsky, A.: Automatic extraction of clusters from hierarchical clustering representations. In: Pacific-Asia Conf. of Advances in Knowl. Discovery and Data Mining (2003)
10. Gupta, G., Liu, A., Ghosh, J.: Automated hierarchical density shaving: A robust automated clustering and visualization framework for large biological data sets. IEEE/ACM Trans. Comp. Biology and Bioinf. 7(2), 223–237 (2010)
11. Lelis, L., Sander, J.: Semi-supervised density-based clustering. In: IEEE Int. Conf. Data Mining (2009)
12. Herbin, M., Bonnet, N., Vautrot, P.: Estimation of the number of clusters and influence zones. Patt. Rec. Letters 22(14), 1557–1568 (2001)
13. Gupta, G., Liu, A., Ghosh, J.: Hierarchical density shaving: A clustering and visualization framework for large biological datasets. In: IEEE ICDM Workshop on Data Mining in Bioinf. (2006)
14. Hartigan, J.A.: Clustering Algorithms. John Wiley & Sons (1975)
15. Muller, D.W., Sawitzki, G.: Excess mass estimates and tests for multimodality. J. Amer. Stat. Association 86(415), 738–746 (1991)
16. Yeung, K.Y., Fraley, C., Murua, A., Raftery, A.E., Ruzzo, W.L.: Model-based clustering and data transformations for gene expression data. Bioinformatics 17(10), 977–987 (2001)
17. Yeung, K., Medvedovic, M., Bumgarner, R.: Clustering gene-expression data with repeated measurements. Genome Biol. 4(5) (2003)
18. Frank, A., Asuncion, A.: UCI machine learning repository (2010)
19. Naldi, M., Campello, R., Hruschka, E., Carvalho, A.: Efficiency issues of evolutionary k-means. Applied Soft Computing 11(2), 1938–1952 (2011)
20. Paulovich, F., Nonato, L., Minghim, R., Levkowitz, H.: Least square projection: A fast high-precision multidimensional projection technique and its application to document mapping. IEEE Trans. Visual. & Comp. Graphics 14(3), 564–575 (2008)

21. Geusebroek, J.M., Burghouts, G., Smeulders, A.: The Amsterdam library of object images. Int. J. of Computer Vision 61, 103–112 (2005)
22. Horta, D., Campello, R.J.: Automatic aspect discrimination in data clustering. Pattern Recognition 45, 4370–4388
23. Larsen, B., Aone, C.: Fast and effective text mining using linear-time document clustering. In: Int. Conf. Knowl. Discovery and Data Mining (1999)
24. Hubert, L., Arabie, P.: Comparing partitions. J. Classification 2(1), 193–218 (1985)

Stock Trend Prediction by Classifying Aggregative Web Topic-Opinion

Li Xue[1], Yun Xiong[1], Yangyong Zhu[1], Jianfeng Wu[2], and Zhiyuan Chen[3]

[1] School of Computer Science,Fudan University,Shanghai 200433, P.R. China
[2] Shanghai Stock Exchanges,Shanghai 200120, P.R. China
[3] Department of Information Systems,University of Maryland Baltimore County,
Baltimore, MD, 21250, USA
{xueli,yunx,yyzhu}@fudan.edu.cn, jfwu@sse.com.cn, zhchen@umbc.edu

Abstract. According to the Efficient Market Hypothesis(EMH) theory, the stock market is driven mainly by overall information instead of individual event. Furthermore, the information about hot topics is believed to have more impact on stork market than that about ordinary events. Inspired by these ideas, we propose a novel stock market trend prediction method by Classifying Aggregative Web Topic-Opinion(CAWTO), which predicts stocks movement trend according to the aggregative opinions on hot topics mentioned by financial corpus on the web. Several groups of experiments were carried out using the data of Shanghai Stock Exchange Composite Index(SHCOMP) and 287,686 financial articles released on SinaFinance[1], which prove the effectiveness of our proposed method.

Keywords: Opinion Mining,Aggregative Opinion,Stock Prediction.

1 Introduction

According to the EMH theory[1], many researchers believe it is a promising way to predict stock movement using the information appears on the web. Among current literature, most studies have tried to analyze the correlation between the time of web news release and that of stock movement. Although text mining based methods using news articles have achieved some success, the accuracy of prediction could hardly reach a satisfactory level[2].

Recently, inspired by the idea of opinion mining, a few studies[3,4,13] have appeared to predict stock market movement by mining sentiment information on the web, such as blog, twitter and editorial, which we call subjective analysis method. Although many studies of this research line have achieved higher prediction accuracy than before, some problems still exist. For example, most of previous studies focused on the opinions about individual events and evaluated them separately, which could hardly capture the overall opinion about hot issues.

Furthermore, most current studies cannot be directly applied to universal corpus covering multiple topics. However, the online financial corpus usually

[1] http://finance.sina.com.cn

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 173–184, 2013.

involves many topics, just like the corpus listed in Table 1, which covers the main topics of March 15, 2011 on SinaFinance. In addition, the trends of stock market are usually determined by the hot issues on the market, which would always be written as the key topics of financial articles. Thus, it would be a promising way to make stock prediction by the overall opinion on the key topics of daily financial articles.

Table 1. A Corpus Covering Multiple Key Topics

Number of Articles	Topic Description
36	Intellectual Property Protection & Food Security
26	Nuclear Leakage of Japanese Power Station
20	Property Purchase Restriction
10	Japanese Earthquake
8	Scandal of Shuanghui Corp. on Ractopamine
6	American Financial Crisis
4	Debate on nuclear power security in Europe
3	Flood in New Jersey,America
42	Articles about Individual Securities or Company

In this paper, we introduce a novel stock prediction method using aggregative topic-opinion of web financial corpus. Firstly, we extract article opinion by lexicon-based opinion mining method, which takes intensity and polarity as the measures of article opinion. Secondly, we transform the article opinion into a multidimensional topic-opinion vector according to our proposed topic-opinion model. Thirdly, an opinion integration method by using article weight and topic weight is used to generate the Aggregative Topic-Opinion Vectors(ATOV). Finally, the stock market trend could be predicted by these ATOVs.

The rest of this paper is organized as follows. In Section II, we review some previous works on text mining based stock prediction. Next, we present the Topic-Opinion model in Section III and show details about the weighted Topic-Opinion aggregation model in Section IV. Stock trend prediction using ATOV is introduced in Section V. Then in Section VI, we perform two groups of experiments on three datasets and make comparison between the results. Finally, we summarize the paper in Section VII.

2 Related Work

Generally, text mining based stock prediction studies mainly follow two lines: objective analysis and subjective analysis. The objective analysis focuses on mining correlation between objective information of textual articles and stock market movement trend. There have been numerous interesting attempts including the earlier works by Wthrich, et al.[5] which started to use textual news articles for financial forecasting. Later, Lavrenko et al.[6] presented a stock prediction approach by extracting most influential news. Recently, many scholars have turned to studying the market response to financial text streams. For example, Ingvaldsen et al.[7] addressed the problem of extracting, analyzing and synthesizing valuable information from continuous financial text streams. Other than news articles, Kloptchenko et al.[8] presented a technique analyzing quantitative data from annual financial reports.

In recent years, sentiment analysis and opinion mining techniques have attracted much research attention in text mining community. For example, Ahmad et al.[9] studied on extracting multi-lingual sentiment from financial news streams, which could deal with sentiment analysis on Arabic and Chinese corpus; Devitt et al.[10] explored a computable metric of positive or negative polarity in financial news text which was consistent with human judgments and could be used in a quantitative analysis of news sentiment impact on financial markets; Sehgal et al.[3] made stock prediction using web sentiment extracted from message board; Wong et al.[4] brought out a pattern-based opinion mining method for stock trend prediction. Another amazing study that predicted stock index movement by public mood and sentiment extracted from twitter, was carried out by Bollen et al.[11].

Compared with the works mentioned above, our study is more similar to some works of the second line,such as the method proposed by Mahajan et al.[12]. However, their work made stock prediction by the major events extracted from financial news, instead of by the opinion extracted from universal financial corpus.

3 Topic-Opinion Model

3.1 Definitions

Definition 1. *(Article-Opinion) For each article d_i, PS_{d_i} and NS_{d_i} are used to represent the positive intensity and negative intensity of d_i's opinion respectively.*

In this study, the Article-Opinion of article d_i, i.e.,PS_{d_i} and NS_{d_i} are evaluated by lexicon-based method. That is, for article d_i, we determine whether d_i contains any number of negative and positive terms from the sentiment lexicon. For each occurrence, we increase the score of either negative or positive by one.

Definition 2. *(Topic-Opinion) Similar to the idea of topic model,for each article d_i, two vectors, $P\text{-}TOV_{d_i}$ and $N\text{-}TOV_{d_i}$ are defined as topic-opinion of article d_i.*

$$P\text{-}TOV_{d_i} =< ps_{d_i\text{-}topic_1}, ps_{d_i\text{-}topic_2}, ..., ps_{d_i\text{-}topic_k} >$$
$$N\text{-}TOV_{d_i} =< ns_{d_i\text{-}topic_1}, ns_{d_i\text{-}topic_2}, ..., ns_{d_i\text{-}topic_k} >$$

where, $ps_{d_i\text{-}topic_k}$ and $ns_{d_i\text{-}topic_k}$ represent positive lexicon-based opinion and negative lexicon-based opinion of article d_i on $topic_k$ respectively.

3.2 Topic-Opinion Generation

Step One: Topic Extraction
It is more reasonable to predict stock movement trend according to emerging topics from new incoming corpus slice, which capture evolutions of existing topics. Thus, we apply an online LDA model[13] rather than basic LDA model in our study to extract the latest hot issues on stock market.

In this study, we assume that the articles arrive in discrete time slices, the size of which is set to a day.Thus, the topic distribution over words on day t, $\Phi_k^{(t)}$, is drawn from a Dirichlet distribution governed by the model on day $t-1$, which is presented below:

$$\begin{aligned}\Phi_k^{(t)}|\beta_k^{(t)} &\sim Dirichlet(\beta_k^{(t)}) \\ &\sim Dirichlet(\omega\hat{\Phi}_k^{(t-1)})\end{aligned}$$

where $\hat{\Phi}_k^{(t-1)}$ is the frequency distribution of a topic k over words on day $t-1$ and $0 < \omega \leq 1$ is an evolution tuning parameter introduced to control the evolution rate of the model. Thus, the generative model for day t of online LDA model can be summarized as follows:

1. For each topic $k = 1, ..., K$
   (a) Compute $\beta_k^t = \omega\hat{\Phi}_k^{t-1}$
   (b) Generate a topic $\Phi_k^t \sim Dirichlet(\cdot|\beta_k^t)$
2. For each document,$d = 1, ..., D^t$:
   (a) Draw $\theta_d^t \sim Dirichlet(\cdot|\beta^t)$
   (b) For each word,w_{d_i}, in document d:
      i Draw z_i from multinomial θ_d^t; $(p(z_i|\alpha_d^t))$
      ii Draw w_{di} from multinomial $\Phi_{z_i}; p(w_{di}|z_i, \beta_{z_i}^t)$

Similar to many other applications of topic model, there is no fixed rules for setting the value of topic number K. According to our method, it is unreasonable to set K to a large number, since we mainly focuses on opinions about the hot issues in a day. Therefore, topic number K is set to a small integer, 10 in our case.

Step Two: TOV Calculation
After applying online LDA process on corpus slice D_t, the lexicon-based Article-Opinion of each article d_i could be reformulated as TOV_{d_i} by Equations (1-2):

$$ps_{d_i\text{-}topic_k} = PS_{d_i} \cdot p_{topic_i} \tag{1}$$

$$ps_{d_i\text{-}topic_k} = PS_{d_i} \cdot p_{topic_i} \tag{2}$$

Where PS_{d_i} and NS_{d_i} are positive and negative Lexicon-based Article-Opinions of d_i respectively.

4 Weighted Topic-Opinion Aggregation Model

4.1 Weighted Topic-Opinion Aggregation

Definition 3. *(Aggregative Topic-Opinion) For corpus D, two vectors, $P\text{-}ATOV_D$ and $N\text{-}ATOV_D$ are defined as aggregative topic-opinion of corpus D.*

$$P\text{-}ATOV_D = < ps_{D\text{-}topic_1}, ps_{D\text{-}topic_2}, ..., ps_{D\text{-}topic_k} >$$
$$N\text{-}ATOV_D = < ns_{D\text{-}topic_1}, ns_{D\text{-}topic_2}, ..., ns_{D\text{-}topic_k} >$$

where, $ps_{D\text{-}topic_k}$ and $ns_{D\text{-}topic_k}$ represent positive opinion and negative opinion of corpus D on $topic_k$ respectively.

Simply speaking, the ATOV of corpus slice D_t is generated by calculating the aggregate opinions on individual topic one by one. A simple way of aggregating the opinions of corpus slice D_t on topic k is showed in Equation (3):

$$ps_{D_t\text{-}topic_k} = \sum_{d_j \in D_t} \frac{ps_{d_j\text{-}topic_k}}{|D_t|} \tag{3}$$

Obviously, the opinion aggregation approach by Equation (3) assumes every article has an equal weight. However, it is unreasonable to hold this assumption in many cases. Next, we will explain this issue by the case listed in Table 1.

Among the corpus listed in Table 1, 42 articles were written about isolated events that were related to individual stock or company, the other 113 articles mainly focused on eight topics, such as Intellectual Property Protection, Japanese Nuclear Leakage, Property Purchase Restriction etc. All these topics were hot issues on March 15, 2011, which had the major influence on the stock market or some individual securities. Intuitively, the articles written about hot issues would have more impact on stock market than ordinary ones. Thus, the importance of each article should be considered when we aggregate the opinions presented by this corpus.

In addition, the number of articles on different topics varies as shown in the first column of Table 1. For example, 36 articles were written about the topic of Intellectual Property Protection and Food Security, and only 3 articles discussed the problem about the Flood happening in New Jersey. Although, both the topics were key topics on March 15, their influences on stock market were obviously not equal, so the topic importance should also be put into consideration as we aggregate the opinions on different topics. Due to these reasons, we introduce two parameters, w_{d_j} and w_{topic_k} into Equation (3), thus the opinion aggregation method is transformed into the following way:

$$ps_{D_t\text{-}topic_k} = w_{topic_k} \cdot \sum_{d_j \in D_t} \frac{w_{d_j} \cdot ps_{d_j\text{-}topic_k}}{|D_t|} \tag{4}$$

where, w_{d_j} and w_{topic_k} represent article-weight and topic-weight respectively.

4.2 Article-Weight

In our study,we acknowledge the following two assumptions.

Assumption 1: Articles involving similar topics are likely to have similar impacts on stork market.

Assumption 2: Articles involving hot issues are likely to have large impacts on stock market.

Following the above assumptions, the article-weight of d_i could be described by Equation (5):

$$w_{d_i} \propto \frac{|S_{d_i}|}{|D_t|} \tag{5}$$

where D_t is the corpus slice consisting of all articles issued on day t, S_{d_i}=$\{d_j|$ $distance(TOV_{d_j},\ TOV_{d_i}) < \delta, d_j \in |D_t|\}$, and $distance(TOV_{d_j}, TOV_{d_i})$ indicates the dissimilarity of topic distributions of d_j and d_i.

4.3 Article-Weight Calculation

By the online LDA method, a document is represented as a vector of probabilities over K topics. Compared with Euclidean distance measure, the Kullback Leibler (KL) divergence is more suitable for computing the distance between two documents distributions, p and q[14], which is given by:

$$KL(p \parallel q) = \sum_i p(i) log \frac{p(i)}{q(i)} \tag{6}$$

Since the KL divergence is not symmetric, we regard the average of $KL(p \parallel q)$ and $KL(q \parallel p)$ as KL distance(KLD) in the rest of the paper.

Taking KL distance as the distance measure for document-topic vectors, we choose K-means as the cluster algorithm. Since the clustering task is performed on the articles issued within a time slice, the article number is relatively small, which is usually less than one thousand. Thus, we can determine the optimal value of k(the number of clusters) by minimizing the Davies-Bouldin index[15] for $k = 1, 2, ..., \sqrt{n}$, where $n = min\{|D_t|, 100\}$, and $|D_t|$ is the number of articles issued within time slice t.

By K-Means cluster algorithm, the articles inside corpus slice D_t are clustered into M clusters. Based on Equation (5), we reformulate article-weight of d_i in the following way:

$$w_{d_i} = \frac{|Cluster_m|}{|D_t|} \tag{7}$$

Where, $|Cluster_m|$ represents the size of cluster m, which is equal to the number of articles inside $Cluster_m$.

4.4 Topic-Weight

The topic importance of $topic_i$ is assumed to be in direct proportion to its lexicon-based opinion intensity, and in reverse proportion to the sum of lexicon-based opinion intensity of all topics. According to this idea, for $topic_i$, positive weight $w_{P\text{-}topic_i}$ and negative weight $w_{N\text{-}topic_i}$ can be calculated by Equations (8-9):

$$w_{P\text{-}topic\text{-}i} = \frac{PS_{topic\text{-}i} + 1}{\sum_{j=1}^{K} PS_{topic\text{-}j} + K}; \tag{8}$$

$$w_{N\text{-}topic\text{-}i} = \frac{NS_{topic\text{-}i} + 1}{\sum_{j=1}^{K} NS_{topic\text{-}j} + K}; \tag{9}$$

Where, PS_{topic_i} and NS_{topic_i} represent positive and negative lexicon-based opinion of $topic_i$, respectively.

5 Stock Prediction by CAWTO

To evaluate the effectiveness of our approach, the composite index of Shanghai Stock Exchanges(SHCOMP) is chosen as the object to be predicted. In this study, SHCOMP-Trend belongs to one of the following possible cases:

- **Up:** It means the SHCOMP of the next day will rise up above one percent than current one.
- **Down:** It means the SHCOMP of the next day will drop down more than one percent than current one.
- **Stable:** It means the fluctuation of SHCOMP will be less than one percent.

The outline of our method is illustrated in Fig.1, which mainly consists of four steps.

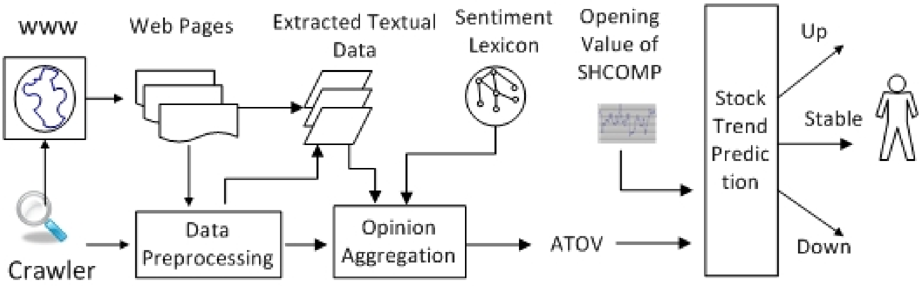


Fig. 1. Data Collection:Crawling daily financial articles from Website,such as, SinaFinance. Data Preprocessing:Performing four basic tasks, text extraction, word segmentation, stop word removal, and background word removal.Feature Extraction: Calculating ATOV according to article weight and topic weight.Making Prediction:Predicting SHCOMP movement trend by ATOV.

6 Experiments

In the experiment,directional accuracy is taken as evaluation metric.For comparison, another two groups of comparison experiments are performed:

- Making SHCOMP-Trend prediction by classifying Event-Topic-Vectors(ETVs).
- Making SHCOMP-Trend prediction by classifying Basic-ATOVs.

ETVs are generated according to the feature representation model proposed by the study[12], where each dimension represents a major event instead of an opinion. Basic-ATOV is the aggregate topic-opinion vector generated according to Equation (3), which aggregates TOVs without considering article-weight and topic-weight.

6.1 Sentiment Lexicon

A Chinese Financial Sentiment Lexicon(CFSL) is used as the sentiment lexicon for lexicon-based Article-Opinion calculation. It consists of 7409 Chinese words in total, including 631 positive words, 575 negative words, and 6203 neutral words, labeled sentiment by financial experts of SSE.

6.2 Data Setting

We crawled 287,686 financial articles issued from April 1, 2009 to September 29, 2011 from SinaFinance,a famous financial website in China,based on which we generate 912 pairs of ATOVs by applying ATOV generation method. Each pair of ATOVs consists of a P-ATOV and a N-ATOV, which represent positive aggregate topic-opinion and negative aggregate topic-opinion respectively. Among 912 pairs of ATOVs, we only select 601 pairs as the input of classifier, since only 601 days are trading days during this period time.In addition, two datasets used in comparison experiments are shown in Table 2.

Table 2. Data Sets used in comparison experiments

Data Set	Data Type	Description
Basic-ATOV	K-dimensional Vector	Contains 601 pairs of Basic-ATOVs
ETV	K-dimensional Vector	Contains 200,000 ETVs

6.3 Experiment Setting

ATOV Generation Setting

The experiment of ATOV aggregation mainly involves online LDA algorithm and k-means clustering algorithm. As to online LDA model, we employ "Matlab Topic Modeling Toolbox", authored by Mark Steyvers and Tom Griffiths in our experiment. The models were run for 200 iterations and the last sample of the Gibbs sampler was used for evaluation. As discussed in Section IV, the number of topics,

K, is set to 10 in all experiments. Following the settings in works[17], a and b are set to $50/K$, and 0.1, respectively. As to k-means clustering algorithm, the clusters number k is determined dynamic by minimizing the Davies-Bouldin index.

Classification Setting

According to current literatures, many studies take Support Vector Machine(SVM) as classifier. For comparison with the prediction accuracy by using other data features, i.e., Basic-ATOV, ETV, a SVM classifier, namely LibSVM[2] is adopted in our experiments.Besides, a classifier based on multiple data domain description(MDDD) is also adopted in the experiments, which was claimed more suitable for multi-class classification task[16].

For the SVM classifier, we use a linear kernel with default parameters (C=1). As to the MDDD classifier, the parameter β is set to 0.2 according to the settings of study[16].

6.4 Result

Result by ATOV

Table 3 shows the directional accuracy of SHCOMP-Trend prediction obtained by classifying P-ATOV and N-ATOV in two-round experiments. From the results, we can clearly find that the N-ATOV dataset achieves higher directional accuracy than P-ATOV in both rounds. In the first round experiments, the directional accuracy of N-ATOV reaches 75.1% when we take MDDD as the classifier, which is the highest directional accuracy. Judging by the two-fold cross-validation method, the average directional accuracy of N-ATOV are 70.3% and 74.7% achieved by SVM and MDDD respectively. This is a clear improvement over 65.2% and 69.5% when the P-ATOV dataset is used, which suggests the N-ATOV has higher predictive ability than the P-ATOV. In other words, our findings imply that negative opinions of hot topics usually have greater impact on stock market than positive ones. Besides, we can also find that all of the directional accuracies achieved by MDDD are higher than the corresponding ones achieved by SVM.

Table 3. Directional Accuracy Using ATOV

Data	Training Range	Testing Range	SVM	MDDD
P-ATOV	Apr,1,2009-Aug,1,2010	Aug,1,2010-Sep,29,2011	65%	68.3%
N-ATOV	Apr,1,2009-Aug,1,2010	Aug,1,2010-Sep,29,2011	69.5%	**75.1%**
P-ATOV	Jun,1,2010-Sep,29,2011	Apr,1,2009-May,31,2010	65.4%	70.7%
N-ATOV	Jun,1,2010-Sep,29,2011	Apr,1,2009-May,31,2010	71.1%	74.3%

Comparison Result

For comparison, the results of experiments using ETV and Basic-ATOV are shown in Table 4 and Table 5 respectively. According to Table 4, the overall directional accuracy achieved by Basic-ATOV(i.e., Basic-P-ATOV, Basic-N-ATOV) is lower than that achieved by ATOV(i.e., P-ATOV, N-ATOV), and the

[2] `http://www.csie.ntu.edu.tw/cjlin/libsvm`

Table 4. Directional Accuracy Using ETV

Data	Training Range	Testing Range	SVM	MDDD
ETV	Apr,1,2009-Aug,1,2010	Aug,1,2010-Sep,29,2011	59.1%	59.8%
ETV	Jun,1,2010-Sep,29,2011	Apr,1,2009-May,31,2010	59.7%	**61.9%**

Table 5. Directional Accuracy Using Basic-ATOV

Data	Training Range	Testing Range	SVM	MDDD
Basic-P-ATOV	Apr,1,2009-Aug,1,2010	Aug,1,2010-Sep,29,2011	59.1%	59.5%
Basic-N-ATOV	Apr,1,2009-Aug,1,2010	Aug,1,2010-Sep,29,2011	**62.2%**	62%
Basic-P-ATOV	Jun,1,2010-Sep,29,2011	Apr,1,2009-May,31,2010	57.5%	60.1%
Basic-N-ATOV	Jun,1,2010-Sep,29,2011	Apr,1,2009-May,31,2010	60%	61.6%

best result is 62.2%, which is attained by SVM on Basic-N-ATOV. Furthermore, negative opinions(i.e.,Basic-N-ATOV) also display higher predictive ability than positive ones(i.e., Basic-P-ATOV) according to Table 4.

In Table 5, the average directional accuracies achieved by SVM and MDDD on ETV are 59.4% and 60.8% respectively, which are very close to the result claimed by the study[14].From Fig.2, the difference between the directional accuracies obtained by Basic-ATOV and ETV is indistinct. Thus, we can hardly conclude that opinion-based information(i.e., ATOV, Basic-ATOV) has higher predictive ability than event-based information(i.e., ETV), which was claimed by Wong et al.[4].However, both types of ATOV outperform Basic-ATOV and ETV in predicting SHCOMP-Trend.

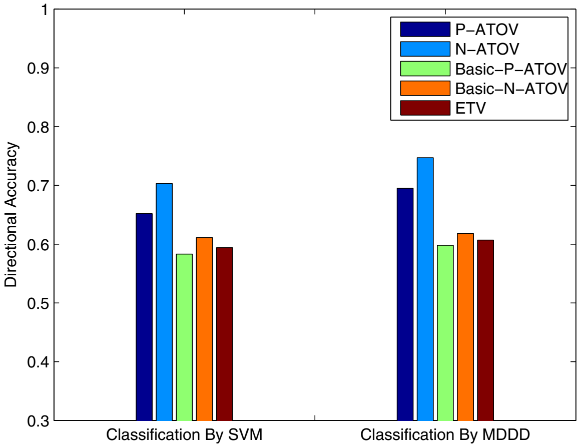


Fig. 2. Prediction Results Obtained By SVM and MDDD

7 Conclusion

In this study, we have explored a new way of predicting stock trend according to the overall opinions on hot topics discussed in web financial corpus. To achieve this goal, a weighted topic-opinion aggregation method is first proposed, by which the aggregative topic-opinion vector(ATOV) can be generated according to article weight and topic weight. By classifying such ATOVs, the stock market movement trend can be predicted with high accuracies. To prove the effectiveness of this method, several groups of experiments on real world data have been carried out, among which the highest directional accuracy of SHCOMP-Trend prediction is up to 75.1%. Furthermore, based on the outcomes of comparison experiments, the ATOV gains a notable advantage over Basic-ATOV and ETV, which are the aggregative opinion generated by a basic integration method and the event-based information extracted by the topic model respectively. In addition, the negative aggregative topic-opinions are found to have higher predictive ability than positive ones. Finally, we also find that different classifiers could lead to relatively large variations of prediction accuracy. Consequently, how to select a suitable classifier according to the type of specific prediction task, i.e., binary classification, multiple classification, becomes one issue of our future works.

Acknowledgment. The research was supported by the National Science Foundation Project of China under Grant.No.61170096 and National Science and Technology Support Program of China under Grant. No.2012BAH13F02.

References

1. Fama, E.F.: The behavior of stock-market prices. The Journal of Busines 38(1), 34–105 (1965)
2. Koppel, M., Shtrimberg, I.: Good news or bad news? Let the market decide. In: Proceedings of the AAAI Spring Symposium on Exploring Attitude and Affect in Text: Theories and Applications, pp. 86–88 (2004)
3. Sehgaland, V., Song, C.: SOPS: Stock Prediction Using Web Sentiment. In: Proceedings of the 7th IEEE International Conference on Data Mining 2007, Data Mining in Web 2.0 Environments Workshop, Omaha, U.S, pp. 21–26 (2007)
4. Wong, K.-F., Xia, Y., Xu, R., Wu, M., Li, W.: Pattern-based Opinion Mining for Stock Market Trend Prediction. International Journal of Computer Processing of Oriental Languages 21(4), 347–362 (2008)
5. Wthrich, B., Cho, V., Leung, S., Peramunetilleke, D., Sankaran, K., Zhang, J., Lam, W.: Daily Prediction of Major Stock Indices from Textual WWW Data. In: Proceedings of the 4th ACM SIGKDD, NY, pp. 364–368 (1998)
6. Lavrenko, V., Schmill, M., Lawrie, D., Ogilvie, P., Jensen, D., Allan, J.: Language models for financial news recommendation. In: Proceedings of the 9th International Conference on Information and Knowledge Management (2000a)
7. Ingvaldsen, J., Gulla, V., Læreid, T., Sandal, P.: Financial News Mining: Monitoring Continuous Streams of Text. In: Proc. IEEE/WIC/ACM International Conference on Web Intelligence (2006)

8. Kloptchenko, A., Eklund, T., Back, B., Karlsson, J., Vanharanta, H., Visa, A.: Combining data and text mining techniques for analyzing financial reports. In: Proc. Eighth Americas Conference on Information Systems (2002)
9. Ahmad, K., Cheng, D., Almas, Y.: Multi-lingual sentiment analysis in financial news streams. In: Proc. of the 1st Intl. Conf. on Grid in Finance, Italy (2006)
10. Devitt, A., Ahmad, K.: Sentiment Polarity Identification in Financial News: A Cohesion-based Approach. In: Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, Prague, pp. 984–991 (June 2007)
11. Bollen, J., Mao, H., Zeng, X.: Twitter mood predicts the stock market. Journal of Computational Science (2011)
12. Mahajan, A., Dey, L., Haque, S.M.: Mining Financial News for Major Events and Their Impacts on the Market. Presented at the WI-IAT 2008. IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, pp. 423–426 (2008)
13. AlSumait, L., Barbar, D., Domeniconi: Online LDA: Adaptive topic model for mining text streams with application on topic detection and tracking. In: Proceedings of the IEEE International Conference on Data Mining (2008)
14. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer (2006)
15. Davies, D.L., Bouldin, D.: A cluster separation measure. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1:224-1:227 (1979)
16. Xue, L., Chen, M., Xiong, Y., Zhu, Y.: User Navigation Behavior Mining Using Multiple Data Domain Description. In: IEEE/WIC/ACM, International Conference on Web Intelligence and Intelligent Agent Technology (2010)
17. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. The Journal of Machine Learning Research 3, 993–1022 (2003)

The Role of Hubs in Cross-Lingual Supervised Document Retrieval

Nenad Tomašev, Jan Rupnik, and Dunja Mladenić

Institute Jožef Stefan
Artificial Intelligence Laboratory
Jamova 39, 1000 Ljubljana, Slovenia
{nenad.tomasev,jan.rupnik,dunja.mladenic}@ijs.si

Abstract. Information retrieval in multi-lingual document repositories is of high importance in modern text mining applications. Analyzing textual data is, however, not without associated difficulties. Regardless of the particular choice of feature representation, textual data is high-dimensional in its nature and all inference is bound to be somewhat affected by the well known *curse of dimensionality*. In this paper, we have focused on one particular aspect of the dimensionality curse, known as *hubness*. *Hubs* emerge as influential points in the k-nearest neighbor (kNN) topology of the data. They have been shown to affect the similarity based methods in severely negative ways in high-dimensional data, interfering with both retrieval and classification. The issue of hubness in textual data has already been briefly addressed, but not in the context that we are presenting here, namely the multi-lingual retrieval setting. Our goal was to gain some insights into the cross-lingual hub structure and exploit it for improving the retrieval and classification performance. Our initial analysis has allowed us to devise a hubness-aware instance weighting scheme for canonical correlation analysis procedure which is used to construct the common semantic space that allows the cross-lingual document retrieval and classification. The experimental evaluation indicates that the proposed approach outperforms the baseline. This shows that the hubs can indeed be exploited for improving the robustness of textual feature representations.

Keywords: hubs, curse of dimensionality, document retrieval, cross-lingual, canonical correlation analysis, common semantic space, k-nearest neighbor, classification.

1 Introduction

Text mining has always been one of the core data mining tasks, not surprisingly, as we use language to express our understanding of the world around us, encode knowledge and ideas. Analyzing textual data across a variety of sources can lead to some deep and potentially useful insights.

The use of internet has spawned vast amounts of textual data, even more so now with the advent of Web 2.0 and the increased amount of user-generated content. This data, however, is expressed in a multitude of different languages. There is a high demand for effective and efficient cross-language information retrieval tools, as they allow the users to access potentially relevant information that is written in languages they are not familiar with.

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 185–196, 2013.

Nearest neighbor approaches are common both in text classification [1][2][3] and document retrieval [4][5][6], which is not surprising given both the simplicity and the effectiveness of most kNN methods. Nearest neighbor methods can be employed both at the document level or at the word level.

The *curse of dimensionality* is known to affect the k-nearest neighbor methods in clearly negative ways. The distances concentrate [7] and uncovering relevant examples becomes more difficult. Additionally, some examples have a tendency to become *hubs*, i.e. very frequent nearest neighbors [8]. Though this may not in itself sound like a severe limitation, it turns out to be quite detrimental in practice. Namely, the *hubness* of particular documents depends more on data preprocessing, feature selection, normalization and the similarity measure than on the actual perceived semantic correlation between the document and its reverse nearest neighbors. In other words, the semantics of similarity is often either completely broken or severely compromised around hubs.

Textual data is high-dimensional and the impact of hubness on various text mining tasks involving nearest neighbor reasoning needs to be closely evaluated.

In this paper we examined the hub structure of an aligned bi-lingual document corpus, over a set of 14 different binary categorization problems. We will show that there is a high correlation between the hub structure in different language representations, but this correlation vanishes when using the common semantic representation. This similarity in the kNN graph topology can be exploited for improving the system performance and we demonstrate this by proposing a hubness-aware instance weighting scheme for the canonical correlation analysis [9].

2 Related Work

2.1 Emergence of Hubs

The concept of hubs is probably most widely known from network analysis [10] and the hubs-and-authorities (HITS) algorithm [11] which was a precursor to PageRank in link analysis. However, hubs arise naturally in other domains as well, as for instance the protein interaction networks [12]. Hubness is a common property of high-dimensional data which has been correlated with the distance concentration phenomenon. Any intrinsically high-dimensional data with meaningful distribution centers ought to exhibit some degree of hubness [8][13][14]. The phenomenon has been most thoroughly examined in the music retrieval community [15][16][17]. The researchers had noticed that some songs were constantly being retrieved by the system, even though they were not really relevant for the queries. The hubness in audio data is still an unresolved issue. Similar phenomena in textual data have received comparatively little attention.

Denote by $N_k(x_i)$ the total number of occurrences of a neighbor point x_i. If the $N_k(x_i)$ is very much higher than k, we will say that x_i is a hub, and if it is much lower than k, we will say that x_i is an *orphan* or *anti-hub*. In case of labeled data, we can further decompose the total occurrence frequency as follows: $N_k(x_i) = GN_k(x_i) + BN_k(x_i)$, where $GN_k(x_i)$ and $BN_k(x_i)$ represent the number of *good* and *bad* k-occurrences, respectively. An occurrence is said to be *good* if the labels of neighbor points match and *bad* if there is a mismatch. Bad hubness is, obviously, closely related to the misclassification rates in kNN methods.

In intrinsically high-dimensional data, the entire distribution of k-neighbor occurrences changes and becomes highly skewed.[1] Not only does this result in some examples being frequently retrieved in kNN sets, but also in that most examples never occur as neighbors and are in fact unintentionally ignored by the system. Only a subset of the original data actually participates in the learning process. This subset is not a carefully selected one, so such implicit data reduction usually induces an information loss.

Furthermore, it is advisable to consider not only bad hubness but also the detailed neighbor occurrence profiles, by taking into account the class-specific neighbor occurrences. The occurrence frequency of a neighbor point x_i in neighborhoods of points from class $c \in C$ is denoted by $N_{k,c}(x_i)$ and will be referred to as *class hubness*.

Several hubness-aware classification methods have recently been proposed in order to reduce the negative influence of bad hubs on kNN classification (hw-kNN [8], h-FNN [18], NHBNN [19], HIKNN [20]).

Apart from classification, data hubness has also been used in clustering [21], metric learning [22] and instance selection [23].

2.2 Canonical Correlation Analysis (CCA)

A common approach to analyzing multilingual document collections is to find a common feature representation, so that the documents that are written in different languages can more easily be compared. One way of achieving that is by using the canonical correlation analysis.

Canonical Correlation Analysis (CCA) [9] is a dimensionality reduction technique somewhat similar to Principal Component Analysis (PCA) [24]. It makes an additional assumption that the data comes from two sources or views that share some information, such as a bilingual document corpus [25] or a collection of images and captions [26]. Instead of looking for linear combinations of features that maximize the variance (PCA) it looks for a linear combination of feature vectors from the first view and a linear combination for the second view, that are maximally correlated.

Formally, let $S = (x_1, y_1), \ldots, (x_n, y_n)$ be the sample of paired observations where $x_i \in \mathbb{R}^p$ and $y_i \in \mathbb{R}^q$ represent feature vectors from some p and q-dimensional feature spaces. Let $X = [x_1, \ldots, x_n]$ and let $Y = [x_1, \ldots, x_n]$ be the matrices with observation vectors as columns, interpreted as being generated by two random vectors $\mathcal{X}$ and $\mathcal{Y}$. The idea is to find two linear functionals (row vectors) $\alpha \in \mathbb{R}^p$ and $\beta \in \mathbb{R}^q$ so that the random variables $\alpha \cdot \mathcal{X}$ and $\beta \cdot \mathcal{Y}$ are maximally correlated. The α and β map the random vectors to random variables, by computing the weighted sums of vector components. This gives rise to the following optimization problem:

$$\underset{\alpha \in \mathbb{R}^p, \beta \in \mathbb{R}^q}{\text{maximize}} \quad \frac{\alpha C_{XY} \beta'}{\sqrt{\alpha C_{XX} \alpha'} \sqrt{\beta C_{YY} \beta'}}, \tag{1}$$

where C_{XX} and C_{YY} are empirical estimates of the variances of $\mathcal{X}$ and $\mathcal{Y}$ respectively and C_{XY} is an estimate of the covariance matrix. Assuming that the observation vec-

[1] *Skewness* of the k-occurrence distribution is defined as $SN_k(x) = \frac{m_3(N_k(x))}{m_2^{3/2}(N_k(x))} = \frac{1/n \sum_{i=1}^{N}(N_k(x_i)-k)^3}{(1/n \sum_{i=1}^{N}(N_k(x_i)-k)^3)^{3/2}}$. High positive skewness which is encountered in intrinsically high-dimensional data indicates that the distribution tail is longer on the right distribution side.

tors are centered, the matrices are computed in the following way: $C_{XX} = \frac{1}{n-1}XX'$, $C_{YY} = \frac{1}{n-1}YY'$ and $C_{XY} = \frac{1}{n-1}XY'$.

This optimization task can be reduced to an eigenvalue problem and includes inverting the variance matrices C_{XX} and C_{YY}. In case of non-invertible matrices, it is possible to use a regularization technique by replacing C_{XX} with $(1-\kappa)C_{XX}+\kappa I$, where $\kappa \in [0,1]$ is the regularization coefficient and I is the identity matrix.

A single canonical variable is usually inadequate in representing the original random vector and typically one looks for k projection pairs $(\alpha_1, \beta_1), \ldots, (\alpha_k, \beta_k)$, so that α_i and β_i are highly correlated and α_i is uncorrelated with α_j for $j \neq i$ and analogously for β.

The problem can be reformulated as a symmetric eigenvalue problem for which efficient solutions exist. If the data is high-dimensional and the feature vectors are sparse, iterative methods can be used, such as the well known Lanczos algorithm [27]). If the size of the corpus is not prohibitively large, it is also possible to work with the dual representation and use the "kernel trick" [28] to yield a nonlinear version of CCA.

3 Data

For the experiments, we examined the Acquis aligned corpus data (http://langtech.jrc.it/JRC-Acquis.html), which comprise a set of more than 20000 documents in many different languages. To simplify the initial analysis, we focused on the bi-lingual case and compared the English and French aligned document sets. We will consider more language pairs in our future work. The documents were labeled and associated with 14 different binary classification problems.

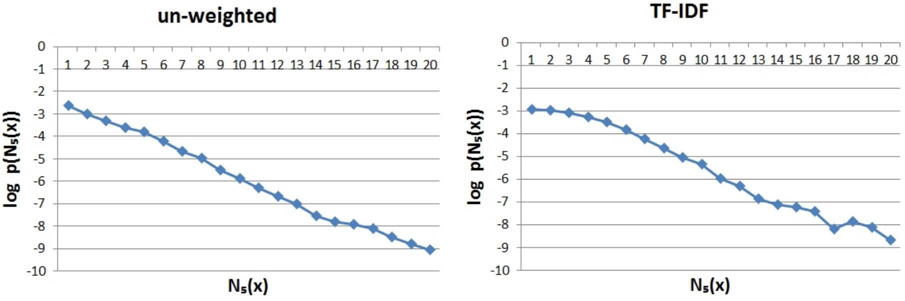
The documents were analyzed in the standard bag-of-words representation after tokenization, lemmatization and stop word removal. Only nouns, verbs, adjectives and adverbs were retained, based on the part-of-speech tags. The inter-document similarity was measured by the cosine similarity measure.

Common semantic representation for the two aligned document sets was obtained by applying CCA. Both English and French documents were then mapped onto the common semantic space (CS:E, CS:F). The used common semantic representation was 300-dimensional, as we wanted to test our assumptions in the context of dimensionality reduction and slight information loss. Longer representations would be preferable in practical applications.

The Acquis corpus exhibits high hubness. This is apparent from Figure 1. The data was normalized by applying TF-IDF, which is a standard preprocessing technique. The normalization only slightly reduces the overall hubness.

The common semantic projections exhibit significantly lower hubness than the original feature representations, which already suggests that there might be important differences in the hub structure. The outline of the data is given in Table 1. The two languages exhibit somewhat different levels of hubness.

If the hubness information is to be used in the multi-lingual context, it is necessary to understand how it maps from one language representation to another. Both the quantitative and the qualitative aspects of the mapping need to be considered. The quantitative aspect refers to the the correlation between the total document neighbor occurrence counts and provides the answer to the general question of whether the same documents



(a) No feature weighting

(b) TF-IDF

Fig. 1. The logarithmic plots of the 5-occurrence distribution on the set of English Acquis documents with or without performing TF-IDF feature weighting. The straight line in the un-weighted case shows an exponential law in the decrease of the probability of achieving a certain number of neighbor occurrences. Therefore, frequent neighbors are rare and most documents are anti-hubs. Note that $N_5(x)$ is sometimes much more than 20, both charts are cut-off there for clarity. Performing TF-IDF somewhat reduces the overall hubness, even though it remains high.

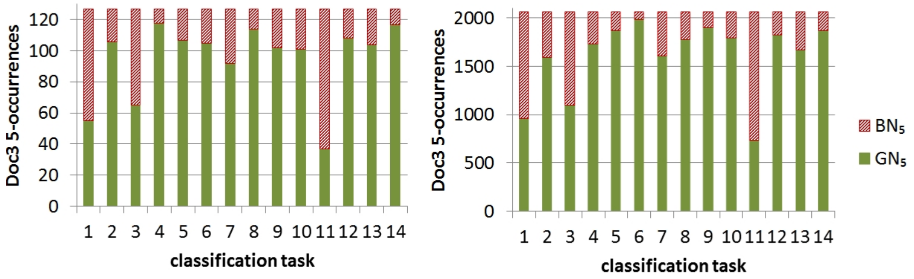
Table 1. Overview of the k-occurrence skewness (S_{N_k}) for all four document corpus representations. To further illustrate the severity of the situation, the degree of the major hub ($\max N_k$) is also given. Both quantities are shown for $k = 1$ and $k = 5$.

Data set	size	d	S_{N_1}	$\max N_1$	S_{N_5}	$\max N_5$
ENG	23412	254963	16.13	95	19.45	432
FRA	23412	212955	80.98	868	54.22	3199
CS:E	23412	300	5.20	38	1.99	71
CS:F	23412	300	4.90	38	1.99	62

become hubs in different languages. The qualitative aspect is concerned with characterizing the type of influence expressed by the hubs in correlating the good and bad hubness (label mismatch percentages) in both languages.

Let us consider one randomly chosen hub document from the corpus. Figure 2 shows its occurrence profiles in both English and French over all 14 binary classification problems. The good/bad occurrence distributions for this particular document appear to be quite similar in both languages, even though the total hubness greatly differs. From this we can conclude that, even though the overall occurrence frequency depends on the language, the semantic nature of the document determines the type of influence it will exhibit if and when it becomes a hub. On the other hand, this particular document is an anti-hub in both projections onto the common semantic space, i.e. it never occurs as a neighbor there. This illustrates how the CCA mapping changes the nature of the k-nearest neighbor structure, which is what Table 1 also confirms.

The observations from examining the influence profiles of a single document are easily generalized by considering the average Pearson correlation between bad hubness ratios over the 14 binary label assignments, as shown in Table 2(a). There is a quite strong positive correlation between document influence profiles in all considered representations and it is strongest between the projections onto the common semantic space,



(a) English version of the text

(b) French version of the text

Fig. 2. Comparing the 5-occurrences of one randomly chosen document (Doc-3) across various classification tasks (label arrays) in English and French language representations. The hubness of Doc-3 differs greatly, but the type of its influence (good/bad hubness ratio) seems to be preserved.

which was to be expected. As for the total number of neighbor occurrences (Table 2(b) and Table 2(c)), the Pearson product-moment gives positive correlation between the hubness of English and French texts, as well as between the projected representations. In all other cases there is no linear correlation. We measured the non-linear correlation by using the Spearman correlation coefficient (Table 2(c)). It seems that there is some positive non-linear correlation between hubness in all the representations.

The results of correlation comparisons can be summarized as follows: frequent neighbor documents among English texts are usually also frequent neighbors among the French texts and the nature of their influence is very similar. Good/bad neighbor documents in English texts are expected to be good/bad neighbor documents in French

Table 2. Correlations of document hubness and bad hubness between different language representations: English, French, and their projections onto the common semantic space

(a) Pearson correlation between bad hubness ratios of documents $(BN_k(x)/N_k(x))$

ENG	FRA	CS:E	CS:F	
	0.68	0.61	0.58	ENG
		0.56	0.58	FRA
			0.76	CS:E
				CS:F

(b) Pearson correlation between total hubness (occurrence frequencies)

ENG	FRA	CS:E	CS:F	
	0.47	0.08	0.06	ENG
		0.01	0.01	FRA
			0.64	CS:E
				CS:F

(c) Spearman correlation between total hubness (occurrence frequencies)

ENG	FRA	CS:E	CS:F	
	0.67	0.29	0.25	ENG
		0.25	0.29	FRA
			0.70	CS:E
				CS:F

texts and vice-versa. We will exploit this apparent regularity for improving the neighbor structure of the common semantic space, as will be discussed in Section 4.

4 Towards a Hubness-Aware Common Semantic Representation

In the canonical correlation analysis, all examples contribute equally to the process of building a common semantic space. However, due to hubness, not all documents are to be considered equally relevant or equally reliable. Documents that become bad hubs exhibit a highly negative influence. Furthermore, as shown in Figure 2, a single hub-document can act both as a bad hub and as a good hub at the same time, depending on the specific classification task at hand. Therefore, instance selection doesn't seem to be a good approach, as we cannot both accept and reject an example simultaneously.

What we propose instead is to introduce instance weights to the CCA procedure in order to control the influence of hubs on forming the common semantic representation in hope that this would in turn improve the cross-lingual retrieval and classification performance in the common semantic space.

The weights introduce a bias in finding the canonical vectors: the search for canonical vectors is focused on the spaces spanned by the instances with high weights.

Given a document sample S, let $u_1, \ldots, u_n$ be the positive weights for the examples $x_i \in X$ and $v_1, \ldots, v_n$ be the positive weights for the examples $y_i \in Y$. We propose to compute the modified covariance and variance matrices as follows:

$$\tilde{C}_{XX} := \frac{1}{n-1}\sum_{i=1}^{n} u_i^2 x_i x_i', \qquad \tilde{C}_{YY} := \frac{1}{n-1}\sum_{i=1}^{n} v_i^2 y_i y_i'$$
$$\tilde{C}_{XY} := \frac{1}{n-1}\sum_{i=1}^{n} u_i v_i x_i y_i' \tag{2}$$

These matrices are input for the standard CCA optimization problem. By modifying them, we are able to directly influence the outcome of the process. The weighting approach is equivalent to performing over-sampling of the instances based on their specified weights and then computing the covariances and variances.

Let $h(x_i, k)$ and $h_B(x_i, k)$ be the standardized hubness and standardized bad hubness scores respectively, i.e. $h(x_i, k) = \frac{N_k(x_i) - \mu_{N_k(x_i)}}{\sigma_{N_k(x_i)}}$ and $h_B(x_i, k) = \frac{BN_k(x_i) - \mu_{BN_k(x_i)}}{\sigma_{BN_k(x_i)}}$. A high standardized hubness score means that the document is very influential and relevant for classification and retrieval, while a high bad hubness score indicates that the document is unreliable.

We have experimented with several different weighting schemes. We will focus on two main approaches. The first approach would be to increase the influence of relevant points (*hubs*) in the CCA weighting. The second meaningful approach is to reduce the influence of unreliable points (*bad hubs*). Additionally, for comparisons, we will also consider the opposite of what we propose, i.e. reducing the influence of hubs

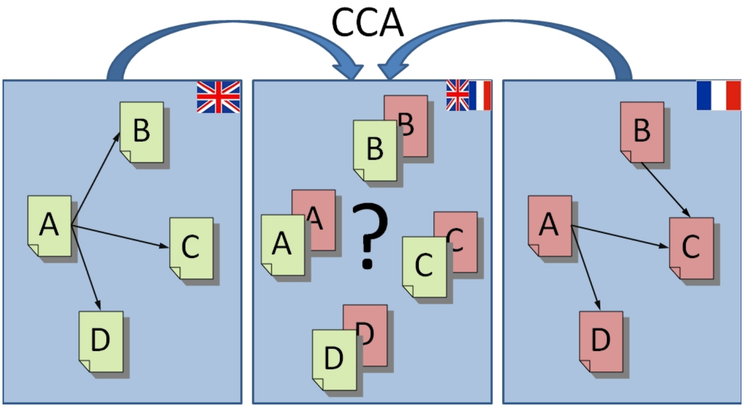


Fig. 3. The CCA procedure maps the documents written in different languages onto the common semantic space. According to the analysis given in Table 2, this changes the kNN structure significantly, which has consequences for the subsequent document retrieval and/or classification. By introducing instance weights we can influence the mapping so that we preserve certain aspects of the original hub-structure and reject the unwanted parts of it.

and increasing the influence of bad hubs. Therefore, the considered weighting schemes are given as follows: un-weighted, $v_i := 1$, emphasized hubs, $v_i := e^{h(x_i,k)}$, de-emphasized hubs, $v_i := e^{-h(x_i,k)}$, emphasized bad hubs, $v_i := e^{h_B(x_i,k)}$, and de-emphasized bad hubs, $v_i := e^{-h_B(x_i,k)}$.

5 Experimental Evaluation

In the experimental protocol, we randomly selected two disjoint subsets of the aligned corpus: 2000 documents were used for training ad 1000 for testing. For each of the 14 binary classification problems we computed five common semantic spaces with CCA on the training set: the non-weighted variant (CS:N), emphasized hubs (CS:H), de-emphasized hubs (CS:h), emphasized bad hubs (CS:B) and de-emphasized bad hubs (CS:b). The training and test documents in both languages were then projected onto the common semantic space. In each case, we evaluated the quality of the common semantic space by measuring the performance of both classification and document retrieval. The whole procedure was repeated 10 times, hence yielding the repeated random sub-sampling validation. We have measured the average performance and its standard deviation.

Many of the binary label distributions were highly imbalanced. This is why the classification performance was measured by considering the Matthews Correlation Coefficient (MCC) [29].

Comparing the classification performance on the original (non-projected) documents with the performance on the common semantic space usually reveals a clear degradation in performance, unless the dimensionality of the projected space is high enough to capture all the relevant discriminative information.

The overview of the classification experiments is given in Table 3. We only report the result on the English texts and projections, as they are basically the same in the French part of the corpus. We have used the kNN classifier with $k = 5$, as we are primarily interested in capturing the change of the neighbor-structure in the data. It is immediately apparent that the weights which emphasize document hubness (CS:H) achieve the best results among the common semantic document representations. Reducing the influence of bad hubs (CS:b) is in itself not enough to positively affect the classification performance. This might be because many hubs reside in borderline regions, so they might carry some relevant disambiguating feature information. It seems that emphasizing the relevance by increasing the preference for all hub-documents gives the best classification results.

Table 3. The Matthews correlation coefficient (MCC) values achieved on different projected representations. The symbols •/○ denote statistically significant worse/better performance ($p < 0.01$) compared to the non-weighted projected representation (CS:N).

Label	Original	CS:N	CS:H	CS:h	CS:B	CS:b
lab1	73.0 ± 3.3	34.2 ± 4.6	**69.2** ± **2.8** ○	66.0 ± 3.3 ○	52.8 ± 4.8 ○	46.6 ± 10.2 ○
lab2	69.2 ± 3.0	52.3 ± 4.4	**65.1** ± **3.9** ○	38.3 ± 3.8 •	45.8 ± 7.0	35.7 ± 8.6 •
lab3	50.2 ± 3.3	27.6 ± 3.8	44.1 ± 3.0 ○	42.2 ± 5.0 ○	**44.8** ± **3.6** ○	33.7 ± 3.0 ○
lab4	32.2 ± 4.4	18.8 ± 6.4	**28.1** ± **2.8** ○	21.1 ± 3.9	20.6 ± 3.7	20.3 ± 6.5
lab5	28.9 ± 12.4	16.8 ± 12.9	17.7 ± 11.7	**21.9** ± **14.4**	10.2 ± 5.5	15.7 ± 6.0
lab6	38.1 ± 6.2	31.2 ± 6.0	29.3 ± 8.2	**33.6** ± **5.4**	23.5 ± 5.8 •	26.2 ± 6.6
lab7	54.5 ± 3.2	38.9 ± 4.0	**48.4** ± **4.2** ○	45.7 ± 3.0 ○	42.3 ± 6.3	36.5 ± 6.8
lab8	44.6 ± 6.3	31.5 ± 6.9	**40.4** ± **6.4** ○	33.5 ± 5.7	23.0 ± 5.0 •	19.6 ± 8.7 •
lab9	76.2 ± 3.4	32.0 ± 5.4	**74.4** ± **3.4** ○	61.8 ± 3.7 ○	45.7 ± 5.2 ○	37.7 ± 7.6
lab10	41.4 ± 4.2	26.1 ± 3.8	34.0 ± 3.8 ○	31.6 ± 5.5	**34.4** ± **4.6** ○	26.6 ± 5.2
lab11	53.5 ± 2.5	27.9 ± 2.8	**48.6** ± **4.0** ○	42.0 ± 3.5 ○	44.9 ± 3.8 ○	33.7 ± 3.8 ○
lab12	39.2 ± 4.0	31.5 ± 3.4	35.4 ± 5.9	**35.6** ± **6.6**	22.8 ± 4.9 •	20.3 ± 5.7 •
lab13	45.4 ± 3.4	29.9 ± 5.2	**38.5** ± **6.0** ○	37.1 ± 4.6 ○	32.6 ± 5.4	28.0 ± 4.9
lab14	49.9 ± 4.5	35.4 ± 7.1	**44.8** ± **7.6**	44.1 ± 7.4	22.4 ± 5.9 •	23.4 ± 11.7
AVG	49.7	31.0	**44.1**	39.6	33.3	28.9

In evaluating the document retrieval performance, we will focus on the k-neighbor set purity as the most relevant metric. The inverse mate rank is certainly also important, but the label matches are able to capture a certain level of semantic similarity among the fetched results. A higher purity among the neighbor sets ensures that, for instance, if your query is about the civil war, you will not get results about gardening, regardless of whether the aligned mate was retrieved or not. This is certainly quite useful. The comparisons are given in Table 4.

Once again, the CS:H weighting proves to be the best among the evaluated hubness-aware weighting approaches, as it retains the original purity of labels among the document kNNs. It is significantly better than the un-weighted baseline (CS:N).

The CS:H weighting produces results most similar to the ones in the original English corpus and we hypothesized that it is because this particular document weighting scheme best helps to preserve the kNN structure of the original document set. We examined the relevant correlations and it turns out that this is indeed the case, as shown

Table 4. The average purity of the k-nearest document sets in each representation. The symbols ●/○ denote significantly lower/higher purity ($p < 0.01$) compared to the non-weighted case (CS:N). The best result in each line is in bold.

Label	Original	CS:N	CS:H	CS:h	CS:B	CS:b
lab1	84.5 ± 1.3	80.7 ± 1.6	**84.1 ± 1.1** ○	83.3 ± 1.5 ○	83.7 ± 1.5 ○	81.7 ± 2.1
lab2	90.5 ± 1.2	84.5 ± 3.2	**90.1 ± 1.2** ○	88.2 ± 2.0 ○	89.6 ± 1.5 ○	84.9 ± 3.7
lab3	74.4 ± 0.9	71.3 ± 1.0	74.4 ± 1.0 ○	73.6 ± 0.9 ○	**74.6 ± 1.2** ○	72.6 ± 1.1
lab4	85.8 ± 1.6	84.6 ± 4.4	85.9 ± 1.5	**85.9 ± 1.8**	85.1 ± 1.5	84.1 ± 3.6
lab5	96.0 ± 0.6	95.9 ± 1.3	95.9 ± 0.8	**96.3 ± 0.8**	95.3 ± 1.0	94.5 ± 3.0
lab6	91.7 ± 0.9	90.2 ± 3.4	**91.6 ± 1.1**	91.6 ± 1.5	90.8 ± 1.5	89.5 ± 3.5
lab7	79.7 ± 0.8	78.0 ± 2.2	**79.7 ± 1.0**	79.0 ± 1.6	79.5 ± 0.6	77.8 ± 1.7
lab8	89.1 ± 1.3	87.0 ± 3.4	**89.0 ± 1.2**	88.5 ± 1.6	88.0 ± 1.3	85.6 ± 3.2
lab9	91.8 ± 1.1	84.7 ± 3.1	**92.0 ± 1.1** ○	89.6 ± 1.5 ○	90.9 ± 1.3 ○	83.9 ± 3.1
lab10	84.3 ± 0.7	**84.5 ± 1.4**	84.4 ± 0.6	84.4 ± 0.8	83.7 ± 0.7	83.4 ± 1.6
lab11	77.0 ± 0.9	73.5 ± 1.1	77.1 ± 0.8 ○	75.5 ± 0.9 ○	**77.3 ± 0.6** ○	74.7 ± 1.2
lab12	88.7 ± 1.2	**88.7 ± 3.3**	88.6 ± 1.3	88.7 ± 1.9	87.6 ± 1.5	87.9 ± 3.5
lab13	82.3 ± 1.5	81.9 ± 2.1	**82.4 ± 1.5**	82.2 ± 1.8	82.0 ± 1.4	80.7 ± 2.5
lab14	92.7 ± 0.8	92.1 ± 2.8	92.3 ± 0.7	**92.7 ± 1.2**	91.7 ± 1.3	91.7 ± 3.1
AVG	86.3	84.1	**86.3**	85.7	85.7	83.8

Table 5. The correlations of document hubness between some of the different common semantic representations, as well as the original English documents. CS:H (emphasize hubness when building the rep.) best preserves the original kNN structure, which is why it leads to similar classification performance, despite the dimensionality reduction.

(a) Pearson correlation between total hubness on the **training** set (occurrence frequencies)

ENG	CS:N	CS:H	CS:h	
	0.05	**0.42**	0.02	ENG
		0.03	0.05	CS:N
			0.02	CS:H
				CS:h

(b) Pearson correlation between total hubness on the **test** set (occurrence frequencies)

ENG	CS:N	CS:H	CS:h	
	0.65	**0.88**	0.75	ENG
		0.68	0.93	CS:N
			0.80	CS:H
				CS:h

in Table 5. By preserving the original structure, it compensates for some of the information loss which would have resulted due to the dimensionality reduction during the CCA mapping.

6 Conclusions and Future Work

We have examined the impact of hubness on cross-lingual document retrieval and classification, from the perspective of calculating the common semantic document representation. Hubness is an important aspect of the dimensionality curse which plagues the similarity-based learning methods.

Our analysis shows that the hub-structure of the data remains preserved across different languages, but is radically changed by the canonical correlation analysis mapping onto the common semantic space. The dimensionality reduction also results in some information loss. We have proposed to overcome the information loss by introducing the *hubness-aware* instance weights into the CCA optimization problem, which have helped in preserving the original kNN structure of the data during the CCA mapping.

The experimental evaluation shows that increasing the influence of hubs on spanning the common semantic space results in an increased kNN classification performance and the higher neighbor set purity.

These initial experiments were performed on an aligned bi-lingual corpus and we intend to expand the analysis by comparing more languages. Additionally, we intend to examine the unsupervised aspects of the problem, like clustering.

Acknowledgments. This work was supported by ICT Programme of the EC under XLike (ICT-STREP-288342) and LTWeb (ICT-CSA-287815).

References

1. Tan, S.: An effective refinement strategy for knn text classifier. Expert Syst. Appl. 30, 290–298 (2006)
2. Jo, T.: Inverted index based modified version of knn for text categorization. JIPS 4(1), 17–26 (2008)
3. Trieschnigg, D., Pezik, P., Lee, V., Jong, F.D., Rebholz-Schuhmann, D.: Mesh up: effective mesh text classification for improved document retrieval. Bioinformatics (2009)
4. Chau, R., Yeh, C.H.: A multilingual text mining approach to web cross-lingual text retrieval. Knowl.-Based Syst., 219–227 (2004)
5. Peirsman, Y., Padó, S.: Cross-lingual induction of selectional preferences with bilingual vector spaces. In: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, HLT 2010, pp. 921–929. Association for Computational Linguistics (2010)
6. Lucarella, D.: A document retrieval system based on nearest neighbour searching. J. Inf. Sci. 14, 25–33 (1988)
7. Aggarwal, C.C., Hinneburg, A., Keim, D.A.: On the surprising behavior of distance metrics in high dimensional space. In: Van den Bussche, J., Vianu, V. (eds.) ICDT 2001. LNCS, vol. 1973, p. 420. Springer, Heidelberg (2000)
8. Radovanović, M., Nanopoulos, A., Ivanović, M.: Nearest neighbors in high-dimensional data: The emergence and influence of hubs. In: Proc. 26th Int. Conf. on Machine Learning (ICML), pp. 865–872 (2009)
9. Hotelling, H.: The most predictable criterion. Journal of Educational Psychology 26, 139–142 (1935)
10. David, E., Jon, K.: Networks, Crowds, and Markets: Reasoning About a Highly Connected World. Cambridge University Press, New York (2010)
11. Kleinberg, J.M.: Hubs, authorities, and communities. ACM Comput. Surv. 31(4es) (December 1999)
12. Ning, K., Ng, H., Srihari, S., Leong, H., Nesvizhskii, A.: Examination of the relationship between essential genes in ppi network and hub proteins in reverse nearest neighbor topology. BMC Bioinformatics 11, 1–14 (2010)

13. Radovanović, M., Nanopoulos, A., Ivanović, M.: Hubs in space: Popular nearest neighbors in high-dimensional data. Journal of Machine Learning Research 11, 2487–2531 (2011)
14. Radovanović, M., Nanopoulos, A., Ivanović, M.: On the existence of obstinate results in vector space models. In: Proc. 33rd Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, pp. 186–193 (2010)
15. Aucouturier, J., Pachet, F.: Improving timbre similarity: How high is the sky? Journal of Negative Results in Speech and Audio Sciences 1 (2004)
16. Flexer, A., Gasser, M., Schnitzer, D.: Limitations of interactive music recommendation based on audio content. In: Proceedings of the 5th Audio Mostly Conference: A Conference on Interaction with Sound, AM 2010, pp. 13:1–13:7. ACM, New York (2010)
17. Schnitzer, D., Flexer, A., Schedl, M., Widmer, G.: Using mutual proximity to improve content-based audio similarity. In: ISMIR 2011, pp. 79–84 (2011)
18. Tomašev, N., Radovanović, M., Mladenić, D., Ivanović, M.: Hubness-based fuzzy measures for high dimensional k-nearest neighbor classification. In: Machine Learning and Data Mining in Pattern Recognition, MLDM Conference (2011)
19. Tomasev, N., Radovanović, M., Mladenić, D., Ivanović, M.: A probabilistic approach to nearest-neighbor classification: naive hubness bayesian kNN. In: Proceedings of the 20th ACM International Conference on Information and Knowledge Management, CIKM 2011, Glasgow, Scotland, UK, pp. 2173–2176. ACM, New York (2011)
20. Tomašev, N., Mladenić, D.: Nearest neighbor voting in high dimensional data: Learning from past occurrences. Computer Science and Information Systems 9(2) (June 2012)
21. Tomašev, N., Radovanović, M., Mladenić, D., Ivanović, M.: The role of hubness in clustering high-dimensional data. In: Huang, J.Z., Cao, L., Srivastava, J. (eds.) PAKDD 2011, Part I. LNCS, vol. 6634, pp. 183–195. Springer, Heidelberg (2011)
22. Tomašev, N., Mladenić, D.: Hubness-aware shared neighbor distances for high-dimensional *k*-nearest neighbor classification. In: Corchado, E., Snášel, V., Abraham, A., Woźniak, M., Graña, M., Cho, S.-B. (eds.) HAIS 2012, Part II. LNCS, vol. 7209, pp. 116–127. Springer, Heidelberg (2012)
23. Buza, K., Nanopoulos, A., Schmidt-Thieme, L.: INSIGHT: Efficient and effective instance selection for time-series classification. In: Huang, J.Z., Cao, L., Srivastava, J. (eds.) PAKDD 2011, Part II. LNCS, vol. 6635, pp. 149–160. Springer, Heidelberg (2011)
24. Pearson, K.: On lines and planes of closest fit to systems of points in space. Philos. Mag. 2(6), 559–572 (1901)
25. Fortuna, B., Cristianini, N., Shawe-Taylor, J.: A Kernel Canonical Correlation Analysis For Learning The Semantics Of Text. In: Kernel Methods in Bioengineering, Communications and Image Processing, pp. 263–282. Idea Group Publishing (2006)
26. Hardoon, D.R., Szedmák, S., Shawe-Taylor, J.: Canonical correlation analysis: An overview with application to learning methods. Neural Computation 16(12), 2639–2664 (2004)
27. Cullum, J.K., Willoughby, R.A.: Lanczos Algorithms for Large Symmetric Eigenvalue Computations, vol. 1. Society for Industrial and Applied Mathematics, Philadelphia (2002)
28. Jordan, M.I., Bach, F.R.: Kernel independent component analysis. Journal of Machine Learning Research 3, 1–48 (2001)
29. Powers, D.M.W.: Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness & Correlation. Technical Report SIE-07-001, School of Informatics and Engineering, Flinders University, Adelaide, Australia (2007)

Text Document Topical Recursive Clustering and Automatic Labeling of a Hierarchy of Document Clusters

Xiaoxiao Li[1], Jiyang Chen[2], and Osmar Zaiane[1]

[1] Department of Computing Science, University of Alberta,
Edmonton, Alberta, Canada
{xl16,zaiane}@cs.ualberta.ca
[2] Google Canada, Kitchener, Ontario, Canada
jiyang@google.com

Abstract. The overwhelming amount of textual documents available nowadays highlights the need for information organization and discovery. Effectively organizing documents into a hierarchy of topics and subtopics makes it easier for users to browse the documents. This paper borrows community mining from social network analysis to generate a hierarchy of topically coherent document clusters. It focuses on giving the document clusters descriptive labels. We propose to use betweenness centrality measure in networks of co-occurring terms to label the clusters. We also incorporate keyphrase extraction and automatic titling in cluster labeling. The results show that the cluster labeling method utilizing KEA to extract keyphrases from the documents generates the best labels overall comparing to other methods and baselines.

Keywords: Text Mining and Web Mining, Cluster Labeling, Document Clustering.

1 Introduction

In this information-explosion era, the retrieval and representation of information is vital for people′s information needs. For textual documents, two main types of information needs are: (1) finding a specific piece of information and (2) browsing the topics and structure of a given document collection [6].

Search engines are effective information retrieval tools for finding specific information. Most search engines return a long list of ranked results to users in response to a query. This presentation works well when the query is non-ambiguous and straight-forward. However, about 16% of user queries are estimated to be Ambiguous Queries, that is to say, they have multiple meanings [10]. For example, the query “jaguar” could mean “jaguar the car”, “jaguar the animal” or “jaguar Mac OS” etc. There are even more queries that are Broad Queries that have multiple aspects [10]. In these situations, documents on different aspects of the query, and even irrelevant documents are mixed together. Even an experienced user would waste time and energy in sifting through the long list of

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 197–208, 2013.

results to locate the ones that they need. The second kind of information need is to browse a document collection without a well-defined goal of searching [6]. A user may just want to browse the structure and topics of a certain document collection [5]. For example, a reader may want to know what topics a blog web site covers to see whether it is of interest; an executive may want to monitor the company emails to have an overview of the subjects discussed. For this need, the long list of documents is not effective either.

One of the solutions to the above problems is document clustering and labeling. This procedure aims at clustering a document collection into smaller groups where each group is on a different topic. It can be done recursively until the topics are specific enough. This will generate a hierarchy of document clusters with labels. This representation allows users to effectively zoom in and locate the documents of interest. It has been proved to facilitate the searching and browsing process [1]. This paper borrows Community Mining from Social Network Analysis to discover different topical coherent document groups and gives each document cluster descriptive labels. Our experiments have shown that our method have strong disambiguation ability and the labeling method utilizing a keyphrase extraction tool KEA gives overall good labels for document clusters.

2 Related Work

2.1 Attempts in Improving the Ranked List

Three major methods to help users to focus on the partition of documents that they may be interested in are: query refinement recommendation, pre-retrieval classification and post-retrieval clustering.

Popular search engines such as Google, Yahoo! and Bing give query refinement recommendations in the form of “Related Searches” besides the search results. Shortcomings of this method are: 1. it utilizes user query logs which may not be available on all document collections; 2. the recommendations do not have a hierarchical structure; 3. it does not group similar topics; and 4. query senses that are not as popular are left out. Classification can also bring documents into order. It classifies each document into one of the pre-defined classes. The categories are well-defined and distinctive in an ontology. However, due to its manual nature, such an ontology covers only a limited number of topics and it is expensive to build and maintain [23].

Another way to solve this problem is by document clustering and labeling, also known as Automatic Taxonomy Generation (ATG) [23]. Researchers have used document clustering to re-organize and represent retrieved documents and observed superior results than ranked lists [5,23]. Document clustering attempts to group documents of the same topic together. The clusters are then labeled with labels that indicate their topics. A user can browse the clusters and select the topic of interest and be led to relevant documents. ATG can generate the taxonomy fully automatically with no external knowledge. Some commercial systems that use ATG to represent web search results are yippy.com and carrotsearch.com. In this paper, we focus on using ATG to generate a hierarchy of topics to improve the presentation of a ranked list of documents.

2.2 A Review of ATG Approaches

In this section we briefly review three major ATG categories including document-based, word-based, and co-clustering based methods.

Document-based ATG methods represent each document as an N-dimensional vector of features with the Vector Space Model (VSM). A feature is a wordphrase called a term and the value can be the document frequency. Conventional clustering methods can be used to cluster documents[23]. Some examples are Scatter/Gather [5], STC (Suffix Tree Clustering) [26], and SnakeT [8]. These methods use snippets that are short parts of the documents to cut down running time but snippets do not contain all the information and it is hard to get good labels from them. Our method can work on full texts with reasonable running time.

Word-based ATG methods such as the subsumption algorithm [20], DisCover [14] and J-Walker that uses a concept ontology WordNet [4] aim at organizing words by theasural relationships [12]. They first generate a concept hierarchy where each concept is a single feature, and then assign documents to the concepts. Labels generated by these methods may not be meaningful by general users and one feature is not enough to conclude the topics in the clusters.

Co-clustering based ATG methods select terms from the documents as keywords, cluster the keywords, and at the same time generate document clusters. FCoDoK [13] and FSKWIC [9] represent keywords as M-dimensional vectors and group keywords with similar document distribution together. Dhillon developed an co-clustering ATG algorithms based on bipartite graph partitioning [7]. Chen et al. proposed a method that builds a keyword graph based on the document co-occurrences of the keywords [2]. They use the K Nearest Neighbor (KNN) algorithm to find keyword clusters and then form document clusters by their similarity with each keyword cluster but they do not have statistical analysis on cluster labeling. Scaiella et al. use a Wikipedia annotator TAGME to find the Wikipedia page titles associated with each document snippet [21]. In their keyword graph, a node is a Wikipedia page title (*topic*), the edge weights are the topic-to-topic similarities computed based on the Wikipedia linked-structure. Then they bi-section the keyword graph into clusters. This method only works on snippets and TAGME can introduce errors to the graph.

3 Methodology

The basic idea of our approach is to reformulate the document clustering problem into a topical community mining problem. Rather than clustering the documents, we extract keywords from them to build a graph indicating the sentence co-occurrences of the keywords. We do community mining on this graph to get communities of highly co-occurring keywords. Then, we map the documents back to the keyword communities to form document clusters. Our approach belongs to the category of co-clustering. Our method maintains the important information while avoiding problems caused by the high-dimensionality in document-based ATG methods. By choosing labels from a group of keywords we are able to describe a cluster with multiple aspects while word-based ATG methods only generate one feature for each cluster.

The process of our approach is shown in Fig 1. The user sends a query to a search engine and sees a taxonomy of the query senses and subtopics along with the documents. Our method applies to any document collection with mixed topics besides search results. We discuss the major phases below.

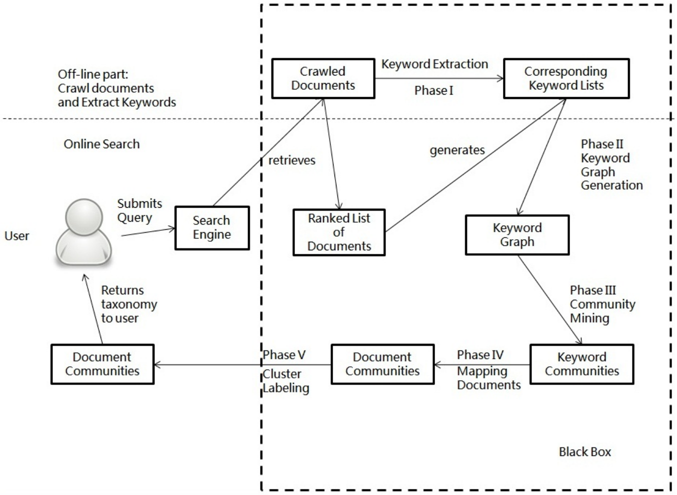


Fig. 1. General procedure of our approach

3.1 Phase I: Keyword Extraction

We first extract keywords from each document. This step is usually time-consuming and can be done off-line along with crawling. We choose Noun Phrases as keywords because they are grammatically consistent and meaningful to users [18]. We first do Part of Speech (POS) Tagging to tag each single word from the document with its part-of-speech, then we lemmatize all the words to reduce the inflectional forms. The next step is pruning where we convert the first word in a sentence to lower case, and remove stop words and words that contain non-alphabetic characters. Finally, we extract Noun Phrases based on a lexical heuristic (Adjective).*(Noun).+. We consider a word or phrase with zero or more Adjectives with one or more Nouns following them as a Noun Phrase [15].

3.2 Phase II: Keyword Graph Generation

We use the keyword pair lists corresponding to the documents to generate a keyword graph. A node in this graph is a keyword. An edge is formed when two nodes have co-occurred in at least one sentence. The edge weight is the sentence co-occurrences. The key assumption in forming edges using co-occurrences is that words describing the same topic are often used together. Co-occurrences have been shown to carry useful correlation information and be able to identify different topical groups [2]. We select nodes based on Document Frequency (DF) to reduce noise. Only keywords with DF higher than a threshold t_{df} remain nodes in the keyword graph. Moreover, terms that are exactly the same, or are contained in the query are removed from the nodes set.

3.3 Phase III: Community Mining

We do community mining on the keyword graph to detect different topical communities. Community mining is the grouping of nodes such that nodes in the same community are more connected with each other than with nodes outside the community. We use the Fast Modularity clustering algorithm ($O(nlog^2n)$) which is based on one of the most well-known community mining metrics: Modularity Q [3]. Modularity Q measures the quality of a graph partitioning. The Fast Modularity algorithm greedily optimizes the modularity score in the graph partitioning in an agglomerative manner [3]. This algorithm automatically detects the number of communities and generates compact taxonomies. It has an advantage over existing commercial systems such as carrotsearch.com and Yippy, and also some most recent works since these methods partition the document collection to about 10 clusters which is not always the real number of topics [2][21]. While many state-of-the-art search result clustering algorithms are flat, our method applies the Fast Modularity algorithm recursively in a top-down manner until certain conditions are reached. We use a Modularity threshold t_Q to determine the need to further split the communities.We further refine the communities by deleting noisy communities and merging similar communities.

3.4 Phase IV: Mapping Documents to Keyword Communities

In this phase we assign the documents to the keyword communities to generate document clusters as illustrated in Fig 2 on some examples from the query "jaguar". A dashed line represents the connection between a document and a keyword. The solid lines are the edges in the keyword graph. For each document on the left, we calculate its overall TFIDF score in each of the keyword communities and assign it to the keyword communities accordingly. Given a document d and a keyword community c, d's overall TFIDF score in c is the sum of the TFIDF scores of all the keywords that are both in d and in c. We assign d to the community that has the highest overall TFIDF score s. Besides, since a document may have multiple topics, we assign d to another community c' as well if its overall TFIDF score in c' is higher than $0.9 * s$.

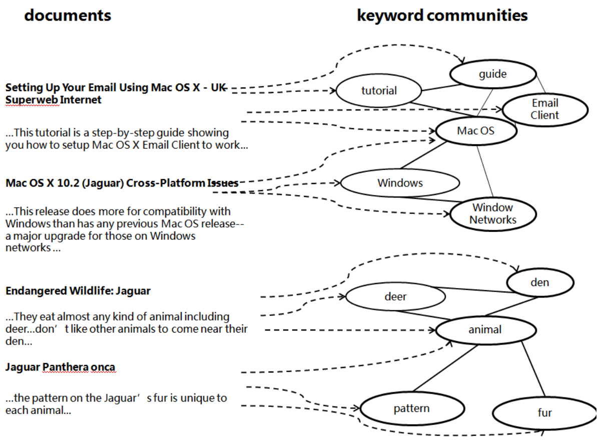

documents
keyword communities
Setting Up Your Email Using Mac OS X - UK Superweb Internet
...This tutorial is a step-by-step guide showing you how to setup Mac OS X Email Client to work...
Mac OS X 10.2 (Jaguar) Cross-Platform Issues
...This release does more for compatibility with Windows than has any previous Mac OS release-- a major upgrade for those on Windows networks...
Endangered Wildlife: Jaguar
...They eat almost any kind of animal including deer...don' t like other animals to come near their den...
Jaguar Panthera onca
...the pattern on the Jaguar' s fur is unique to each animal...
guide
tutorial
Email Client
Mac OS
Windows
Window Networks
den
deer
animal
pattern
fur

Fig. 2. Illustration of Phase IV

3.5 Phase V: Cluster Labeling

We treat document cluster labeling as a ranking problem of the keywords in each community. The most common cluster labeling method is to use the most frequent or central phrases in a document cluster as labels [17]. In our experiments we use the Degree Centrality labeling method as a baseline BL1. We use the "Frequent and Predict Words" method that detects terms that are more likely to appear in a cluster than in other clusters as labels as another baseline BL2 [19]. We come up with four labeling methods (LM1, LM2, LM3, LM4) that select labels based on the keyword cluster, the document cluster, and the connection between the co-clusters.

LM1 finds important terms from the keyword communities as labels based on the betweenness centrality that reflects a node′s influence on the communications between other nodes in the community. Betweenness centrality measures the number of shortest paths between other nodes that goes through a certain node. The intuition of using betweenness centrality for labeling is that sometimes terms of the same topic may not directly co-occur in a sentence and may be connected by terms that play a vital role in connecting terms.

We also try to select labels based on the document clusters. Keyphrases and titles both introduce the topics of the documents. We propose to incorporate a famous and effective keyphrase extraction algorithm KEA (LM2) [24] and an automatic titling method (LM3) [16] to identify important terms from the documents. We use an updated KEA tool[1] to extract keyphrases from the documents. LM3 extracts title words from each document by an automatic titling method. We take the Noun Phrases (NP) from the first two sentences and the titles from the documents as title words. The terms are ranked by the number of documents where they serve as important terms. The top 20 are cluster labels.

[1] `http://www.nzdl.org/Kea/description.html`

The last labeling method, LM4, looks at the connections between a keyword community and its corresponding document community (the dashes lines in Fig 2). It ranks the terms by the sum of their TF-IDF scores in each document cluster. The top 20 forms a label list.

After getting the label lists, we do post-processing based on lemmas, abbreviations, synonyms and hypernyms to make them more readable. The top five labels in the post-processed list is the final label list for each cluster.

4 Experiments and Discussions

4.1 Data Collection and Pre-processing

We constructed our data sets using Google. For each query, we searched for some of its senses in Google and gathered the top results. We merged these pages together as the document collection under the query. We experimented with a multitude of queries with ambiguous meanings. For illustration purposes, we show some examples here. A list of queries, their query senses and the subtopics along with the size of the document collections is shown in Table 1. For example, the *tiger* data set is merged by the search results of *tiger aircraft*, *tiger woods*, *tiger animal*, and *tiger hash*. For the query sense *jaguar car* in the *jaguar* data set, we selected two subtopics: *jaguar car history* and *jaguar car dealer*. We feed them to Google search engine and merged their results as the documents of the query sense *jaguar car*.

Table 1. List of queries, query senses, subtopics of query senses (shown in parentheses) with the number of documents

Query	Query Senses and subtopics	Document Set Size
jaguar	animal (animal facts, animal rescue), car (car history, car dealer), Mac OS, guitar	180
penguin	Pittsburgh hockey team, publisher, kids club, algorithm	150
avp	Volleyball, antivirus software, Avon, movie, airport	150
tiger	Aircraft, Woods, animal, hash	120
michael jordan	basketball player (career, quotes), Berkeley researcher	90

4.2 Evaluation Metrics

We compare our results with the ground truth gained from Google to evaluate the clustering performance. We adapt two evaluation metrics: ARI [25] and Cluster Contamination (CC) [6]. ARI measures how close the clusters generated by our system (P) matches the the ground truth (R). CC measures the purity of the clusters. The clustering is good if it has a high ARI and a low CC. We conducted a user survey with 11 volunteers to obtain the labeling ground truth. The label with most votes is the ground truth label S of a document cluster. Given S and its parent label P, a system label L is a correct label if L is identical with S, S P, or P S [22]. Four evaluation metrics are used, namely match@N, P@N, MRR@N and MTRR@N [14,22]. N is the number of labels presented to the users. Good labels have high metric scores and small N.

4.3 Experimental Results

In our experiments, we set $t_{df} = 0.04$, and $t_Q = 0.3$ (details are omitted due to lack of space). We evaluate the top levels and the lower level clusters in the taxonomy separately. The top levels contain clusters that are directly under the root. They reflect the disambiguation ability of our method. The lower levels show how well our method discovers different aspects (subtopics) of the same topic. While all the levels are important in the browsing process, the top levels carry more responsibility because they are the ones that users see first.

Document Clustering Performances. We compare our method with an effective variation of K-Means [11] to examine the document clustering quality. We use the number of clusters found by our method as the parameter k, the keywords extracted by our method as the features and the TFIDF scores as the feature values for K-means. The clustering performance on the top levels is listed in Table 2. Our method gets higher ARI score on all queries and less contamination on all but one queries than K-Means. We found that K-Means tend to generate one big and highly contaminated cluster with several small and pure clusters. Our method does not generate highly polluted clusters thus is more desirable for browsing. Besides, K-Means requires the number of clusters k in advance whereas our method automatically detects k. Overall, our clustering method outperforms K-Means on the top levels. Table 3 shows the clustering performance on the lower levels. Our method has higher ARI scores on 2 out of 3 query senses but it generates more contaminated clusters on lower levels. It is maybe due to the fact that there is no clear separation between the vocabularies used by different subtopics of the same topic.

Table 2. ARI and average CC score of our method and K-means on the top levels

Query	ARI score		average CC score	
	our method	K-Means	our method	K-means
jaguar	0.968	0.521	0.053	0.169
penguin	0.842	0.319	0.152	0.352
avp	0.802	0.615	0.239	0.178
tiger	0.771	0.325	0.193	0.261
michael jordan	1	0.022	0	0.439

Cluster Labeling Performances. The cluster labeling performances on the four evaluation metrics of our four methods and two baselines on the top levels are presented in Fig 3 with N ranging from 1 to 5. We can see that LM2 which is the labeling method utilizing KEA achieves the highest average score over all queries on all the metrics. In Table 4 we show the top 5 labels picked by LM2 for each query sense. Beside each label is the number of users who chose it as the cluster label in the user survey. We also show the ground truth labels, each with the number of users who have picked it. The labeling performances on the lower levels are shown in Fig 4. We can see that BL1, which is the worst method on the top levels, is among the best methods on the lower levels. LM2 which is

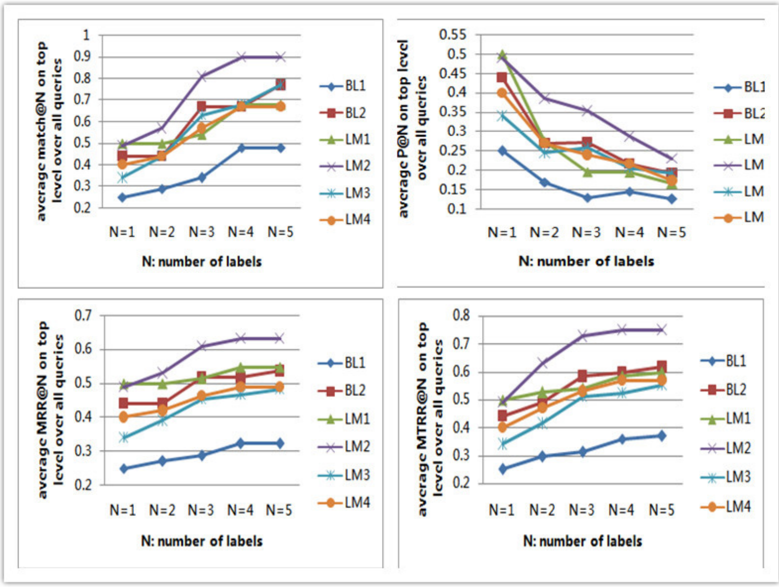


Fig. 3. Average match@N, P@N, MRR@N and MTRR@N on the top level over all queries of different labeling methods

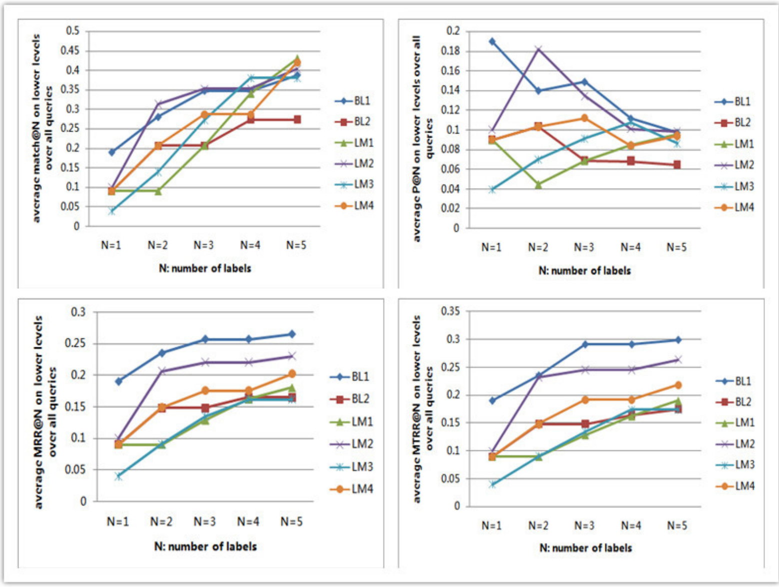


Fig. 4. Average match@N, P@N, MRR@N and MTRR@N on lower levels over all queries of different labeling methods

Table 3. ARI and average CC score of our method and K-means on lower levels

Query sense	subtopics	ARI score		CC score	
		our method	K-Means	our method	K-means
jaguar/animal	facts, rescue	0.428	0.321	0.288	0.261
jaguar/car	history, dealer	0.122	-0.0003	0.512	0.400
Michael Jordan/basketball player	career, quotes	0.107	0.328	0.658	0.550

Table 4. Top cluster labels by the users, and by our KEA method for each query sense with the number of users who picked each label

Query	Query Sense	Ground Truth Labels	Our labels by LM2
jaguar	animal	animal-7	animals (7), cat (1), habitats (1), species (1), leopard (0)
	car	jaguar car-7	cars (3), jaguar car (7), dealer (1), history (1), sport car (3)
	Mac OS	mac os-7	OS (5), mac (3), mac OS (7), apples (2), window (0)
	guitar	guitar-9	guitars (9), fenders (2), pickup (0), neck (0), bridges (0)
penguin	Pittsburgh hockey team	pittsburgh penguin-7	pittsburgh (3), pittsburgh penguin (7) , teams (1), hockey (5), league (1)
	publisher	book-6 publisher-6	books (6), publishers (6), penguin book (3), imprints (1), penguin group(2)
	kids club	club penguin-7	clubs (1), club penguin (7), kid (1), games (0), online (0)
	algorithm	google penguin algorithm-6	google (3), algorithms (3), updates (2), google penguin (3), algorithm update (2)
AVP	volleyball	volleyball-8	volleyball (8), beaches (0), tours (1), beach volleyball (6), sport (2)
	antivirus software	antivirus-8	kaspersky (2), viruses (2), software (5), antivirus (8), kaspersky lab (1)
	Avon	avon product-7	avon (5), avon product (7), product (2), market (0), stock (3)
	movie	movie-6	predators (2), movies (6), alien (2), weyland (0), predator movie (3)
	airport	international airport-4	airports (1), scranton (1), international airport (4), avoca (2), hotel (1)
tiger	aircraft	tiger airway-8	aircraft (7), pilot (0), tiger moth (2), tiger airway (8), markets (0)
	Woods	golf-6	woods (1), tiger wood (1), golf (6), opens (1), tournament (1)
	animal	animal-8	animal (8), cubs (0), habitat (0), species (2), cat (0)
	hash	tiger hash algorithm-7	hash (4), tiger hash (6), hash function (6), function (1), file (0)
Michael Jordan	basketball player	basketball-7	basketball (7), player (1), NBA[national basketball association] (5), basketball player (5), games (0)
	Berkeley researcher	machine learning-11	research (3), university (3), machine learning (11), learning (1), berkeley (1)

the best on the top levels is the second best on the lower levels. Overall, LM2 is the best labeling method both on the top and the lower levels. Note that the metric scores on the lower levels are less than those of the top levels. In our user survey we have found that even for humans it is harder to agree on the labels of the lower levels than of the top levels. One reason is that the subtopics are difficult to differentiate. Another reason might be that similar terms are used when covering subtopics.

5 Conclusions and Future Work

In this work we use a co-clustering ATG method based on the co-occurrences of frequent keywords to generate a taxonomy for a document collection that performs well in disambiguating topics but not as well in separating subtopics of the same topic. We propose four different labeling methods and found that the labeling method utilizing KEA generates the best overall cluster labels. Future works include ways to improve the performance on the lower levels. One possibility is to explore other ways to build the keyword graph so that it is sensitive to different subtopics. Another future work is to combine different labeling methods to improve the labeling performance by reflecting the strength of each method.

References

1. Berendsen, R., Kovachev, B., Nastou, E.-P., de Rijke, M., Weerkamp, W.: Result disambiguation in web people search. In: Baeza-Yates, R., de Vries, A.P., Zaragoza, H., Cambazoglu, B.B., Murdock, V., Lempel, R., Silvestri, F. (eds.) ECIR 2012. LNCS, vol. 7224, pp. 146–157. Springer, Heidelberg (2012)
2. Chen, S.Y., Chang, C.N., Nien, Y.H., Ke, H.R.: Concept extraction and clustering for search result organization and virtual community construction. Computer Science and Information Systems 9(1), 323–355 (2012)
3. Clauset, A., Newman, M.E.J., Moore, C.: Finding community structure in very large networks. Physical Review E 70(6), 66111 (2004)
4. Cui, H., Zaiane, O.R.: Hierarchical structural approach to improving the browsability of web search engine results. In: Proceedings of the12th International Workshop on Database and Expert Systems Applications, pp. 956–960. IEEE (2001)
5. Cutting, D.R., Karger, D.R., Pedersen, J.O., Tukey, J.W.: Scatter/gather: a cluster-based approach to browsing large document collections. In: Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 1992, pp. 318–329. ACM, New York (1992)
6. Dawid, W.: Descriptive Clustering as a Method for Exploring Text Collections. PhD thesis, Poznan University of Technology, Poznań, Poland (2006)
7. Dhillon, I.S.: Co-clustering documents and words using bipartite spectral graph partitioning. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 269–274. ACM (2001)
8. Ferragina, P., Gulli, A.: A personalized search engine based on web-snippet hierarchical clustering. Software: Practice and Experience 38(2), 189–225 (2008)
9. Frigui, H., Nasraoui, O.: Simultaneous categorization of text documents and identification of cluster-dependent keywords. In: Proceedings of the 2002 IEEE International Conference on Fuzzy Systems, FUZZ-IEEE 2002, vol. 2, pp. 1108–1113. IEEE (2002)
10. Jansen, B.J., Booth, D.L., Spink, A.: Determining the user intent of web search engine queries. In: Proceedings of the 16th International Conference on World Wide Web, pp. 1149–1150. ACM, New York (2007)
11. Kanungo, T., Mount, D.M., Netanyahu, N.S., Piatko, C.D., Silverman, R., Wu, A.Y.: An efficient k-means clustering algorithm: Analysis and implementation. Pattern Analysis and Machine Intelligence 24(7), 881–892 (2002)

12. Krishnapuram, R., Kummamuru, K.: Automatic taxonomy generation: Issues and possibilities. In: Fuzzy Sets and Systems IFSA 2003, pp. 184–184 (2003)
13. Kummamuru, K., Dhawale, A., Krishnapuram, R.: Fuzzy co-clustering of documents and keywords. In: The 12th IEEE International Conference on Fuzzy Systems, vol. 2, pp. 772–777. IEEE (2003)
14. Kummamuru, K., Lotlikar, R., Roy, S., Singal, K., Krishnapuram, R.: A hierarchical monothetic document clustering algorithm for summarization and browsing search results. In: Proceedings of the 13th International Conference on World Wide Web, pp. 658–665. ACM (2004)
15. Liu, Z., Huang, W., Zheng, Y., Sun, M.: Automatic keyphrase extraction via topic decomposition. In: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, pp. 366–376. Association for Computational Linguistics (2010)
16. Lopez, C., Prince, V., Roche, M.: Automatic titling of electronic documents with noun phrase extraction. In: Soft Computing and Pattern Recognition (SoCPaR), pp. 168–171. IEEE (2010)
17. Manning, C.D., Raghavan, P., Schutze, H.: Introduction to information retrieval, vol. 1. Cambridge University Press, Cambridge (2008)
18. Mei, Q., Shen, X., Zhai, C.X.: Automatic labeling of multinomial topic models. In: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 490–499. ACM (2007)
19. Popescul, A., Ungar, L.H.: Automatic labeling of document clusters. Unpublished Manuscript (2000)
20. Sanderson, M., Croft, B.: Deriving concept hierarchies from text. In: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 206–213. ACM (1999)
21. Scaiella, U., Ferragina, P., Marino, A., Ciaramita, M.: Topical clustering of search results. In: Proceedings of the fifth ACM International Conference on Web Search and Data Mining, pp. 223–232. ACM (2012)
22. Treeratpituk, P., Callan, J.: Automatically labeling hierarchical clusters. In: Proceedings of the 2006 International Conference on Digital Government Research, pp. 167–176. ACM (2006)
23. Wang, X., Bramer, M.: Exploring web search results clustering. In: Research and Development in Intelligent Systems XXIII, pp. 393–397 (2007)
24. Witten, I.H., Paynter, G.W., Frank, E., Gutwin, C., Nevill-Manning, C.G.: Kea: Practical automatic keyphrase extraction. In: Proceedings of the Fourth ACM Conference on Digital Libraries, pp. 254–255. ACM (1999)
25. Yip, K.Y., Cheung, D.W., Ng, M.K.: Harp: A practical projected clustering algorithm. IEEE Transactions on Knowledge and Data Engineering 16(11), 1387–1397 (2004)
26. Zamir, O., Etzioni, O.: Grouper: a dynamic clustering interface to Web search results. In: Proceedings of the Eighth International Conference on World Wide Web, WWW 1999, pp. 1361–1374. Elsevier North-Holland, Inc., New York (1999)

Query-Document Relevance Topic Models

Meng-Sung Wu[1], Chia-Ping Chen[2], and Hsin-Min Wang[3]

[1] ITRI, Hsinchu, Taiwan
[2] National Sun Yat-Sen University, Kaohsiung, Taiwan
[3] Institute of Information Science, Academia Sinica, Taipei, Taiwan
wums@itri.org.tw, cpchen@mail.cse.nsysu.edu.tw, whm@iis.sinica.edu.tw

Abstract. In this paper, we aim to deal with the deficiency of current information retrieval models by integrating the concept of relevance into the generation model from different topical aspects of the query. We study a series of relevance-dependent topic models. These models are adapted from the latent Dirichlet allocation model. They are distinguished by how the notation of query-document relevance, which is critical in information retrieval, is introduced in the modeling framework. Approximate yet efficient parameter estimation methods based on the Gibbs sampling technique are employed for parameter estimation. The results of experiments evaluated on the Text REtrieval Conference Corpus in terms of the mean average precision (mAP) demonstrate the superiority of the proposed models.

Keywords: latent Dirichlet allocation, query-document relevance, topic model, information retrieval.

1 Introduction

Language model, which captures the statistical regularities of language generation, (LM) has been successfully applied in information retrieval (IR) [13,22]. However, the LM-based IR approaches often suffer from the problem of the word usage variety. Using topic models to address the above issue has been an area of interesting and exciting research. Topic model refers to the language model that is commonly used for extracting and analyzing the semantic information in a collection of documents. Probabilistic latent semantic analysis (PLSA) [7] and latent Dirichlet allocation (LDA) [2] are two well-known topic models for documents. In PLSA, a document model is a mixture of multinomials, where each mixture component corresponds to a topic. The parameters in the mixture of multinomials, e.g., weights and multinomial parameters, can be easily estimated via the maximum likelihood principle. In LDA, weights and multinomial parameters are treated as random variables with the (conjugate) Dirichlet prior distributions. The maximum a posterior estimates for these variables are used for document models. Topic model and its variants have been applied to applications such as language modeling and language model adaptation [4,6,20], information retrieval [16,18,19], tag-based music retrieval [9,17], and social network analysis [10].

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 209–220, 2013.

For IR applications, the state-of-the-art topic models can be somewhat deficient. The main issue here is that they often fail to exploit the valuable information conveyed in the queries while focusing only on document contents. Chemudugunta et al. [3] propose a probabilistic topic model which assumes that words are generated either from a specific aspect distribution or a background distribution. Wei and Croft [19] linearly combine the LDA model with document-specific word distributions to capture both general as well as specific information in documents. Another interesting topic modeling approach gives users the ability to provide feedback on the latent topic level and reformulate the original query [1,14]. In addition, Tao et al. [15] construct a method to expand every document with its neighborhood information. As described in [12], query association is one of the most important forms of document context, which could improve the effectiveness of IR systems. In this paper, we aim to deal with this deficiency by integrating the concept of relevance into the generation model from different topical aspects of the query rather than expanding a query from an initially retrieved set of documents [24]. That is, we design IR systems with emphasis on the degree of matchedness between the user's information needs and the relevant documents.

In this paper, we propose a novel technique called relevance-dependent topic model (RDTM). The main contribution of this work is modeling the generation of a document and its relevant past queries with topics for information retrieval. Relevant past queries are incorporated to obtain a more accurate model for the information need. The model assumes that relevant information about the query may affect the mixture of the topics in the documents and the topic of each term in a document may be sampled from either using the normal document specific mixture weights in LDA or using query specific mixture weights. The parameter estimation of the proposed RDTM is implemented by the Gibbs sampling method [5].

The remainder of this paper is organized as follows. The background of this research work is surveyed in Section 2, with emphasis on the review of stochastic methods for information retrieval. Proposed relevance-dependent topic models and the corresponding learning and inference algorithms based on Gibbs sampling are introduced and explained in details in Section 3. The experimental results are presented and discussed in Section 4. Lastly, summarization and the concluding remarks are given in Section 5.

2 Review and Related Works

2.1 LDA-Based Document Model

In a **topic model**, the probability of a word in a document depends on the topic of the document. Without loss of generality, a word is denoted by $w \in \{1, 2, \ldots, V\}$, where V is the number of distinct words/terms in a vocabulary. A **document**, represented by $\mathbf{d} = w_1, \ldots, w_{n_d}$, is a sequence of words. A **collection** of documents is denoted by $\mathcal{D} = \{\mathbf{d}_1, \ldots, \mathbf{d}_D\}$. The number of topics is assumed to be K, so a topic is denoted by $z \in \{1, \ldots, K\}$. A *latent* topic model is a topic model but the topics are not observed. Mathematically, a latent topic model is equivalent to a convex combination of a set of topic models. In this paper, the relevance-based topic model is an extension of the latent Dirichlet allocation. Thus, we briefly review LDA as follows.

Latent Dirichlet Allocation. In LDA [2], the weights and multinomial parameters are random variables $\varPhi$ and $\varTheta^{(d)}$ with conjugate Dirichlet priors. LDA can be represented by a graphical model (GM) as shown in Fig. 1 (a). The generation of $\mathcal{D}$ encoded in this graph is as follows.

- Start
- Sample from a Dirichlet prior with parameter β for the multinomial ϕ_z over the words for each topic z;
- For each document $d \in \{1, \ldots, D\}$ [1]
    - Sample from a Dirichlet prior with parameter α for the multinomial $\theta^{(d)}$ over the topics;
    - For $n = 1 \ldots n_d$
        - ∗ Sample from $\theta^{(d)}$ for the topic z_n;
        - ∗ Sample from ϕ_{z_n} for the word w_n;
- End

For $\mathcal{D} \triangleq \{w_1, \ldots, w_\nu\} = \mathbf{w}$, the joint probability is

$$\begin{aligned} &P(\mathbf{w}, \mathbf{z}, \theta, \phi|\alpha, \beta) \\ &= P(\phi|\beta) \prod_{d=1}^{D} \Big(P(\theta^{(d)}|\alpha) \prod_{n=1}^{n_d} P(w_n|z_n, \phi) P(z_n|\theta^{(d)}) \Big) \end{aligned} \tag{1}$$

Marginalizing over θ, ϕ, and $\mathbf{z}$, we have

$$P(\mathbf{w}|\alpha, \beta) = \int\int P(\phi|\beta) \prod_{d=1}^{D} \Big(P(\theta^{(d)}|\alpha) \cdot \prod_{n=1}^{n_d} \sum_{z_n} P(w_n|z_n, \phi) P(z_n|\theta^{(d)}) \Big) d\theta d\phi. \tag{2}$$

Note that the posterior distribution for $\varTheta^{(d)}$ varies from document to document.

Parameter Estimation of LDA via Gibbs Sampling. In LDA, the prior distributions $P(\theta^{(d)}|\alpha)$ and $P(\phi|\beta)$ of the latent variables $\varTheta^{(d)}$ and $\varPhi$ are different from the posterior distributions $P(\theta^{(d)}|\alpha, \mathcal{D})$ and $P(\phi|\beta, \mathcal{D})$. Using the maximum a posterior (MAP) estimates $\hat{\theta}^{(d)}(\alpha, \mathcal{D})$ and $\hat{\phi}(\beta, \mathcal{D})$ of the posterior distributions of $\varTheta^{(d)}$ and $\varPhi$, the model for the nth word w in a given document d can be approximated by a multinomial mixture model as follows

$$\hat{P}(w_n|\hat{\theta}^{(d)}, \hat{\phi}) = \sum_{z_n} P(w_n|z_n, \hat{\phi}) P(z_n|\hat{\theta}^{(d)}). \tag{3}$$

[1] Note that d is document index and $\mathbf{d}$ is document representation.

That is, $\hat{\theta}^{(d)}(\alpha, \mathcal{D})$ is the multinomial parameter for topics and $\hat{\phi}(\beta, \mathcal{D})$ is the multinomial parameter for words.

Recall that $\mathcal{D}$ is represented by $\mathbf{w} = \{w_1, \ldots, w_\nu\}$. In principle, given samples of $\mathbf{z}$ drawn from $P(\mathbf{z}|\mathbf{w})$, we can estimate $\hat{\theta}(\mathbf{w})$ and $\hat{\phi}(\mathbf{w})$ simply by their relative frequencies. The key inferential problem is how to compute the posterior distribution $P(\mathbf{z}|\mathbf{w})$, which is directly proportional to the joint distribution

$$P(\mathbf{z}|\mathbf{w}) = \frac{P(\mathbf{z}, \mathbf{w})}{P(\mathbf{w})} = \frac{P(\mathbf{z}, \mathbf{w})}{\sum_{\mathbf{z}} P(\mathbf{z}, \mathbf{w})} \tag{4}$$

In practice, however, it is obvious that the denominator in (4) is an enormous discrete distribution with K^ν parameters, and sampling directly from $P(\mathbf{z}|\mathbf{w})$ is not feasible [5]. Alternative methods have been used to estimate the parameters of topic models [2,5,11]. Therefore, we use the stochastic methods for the estimation problem.

In the Gibbs sampling method, z_n is sequentially sampled using the so-called full-conditional distribution $P(z_n|\mathbf{z}_{-n}, \mathbf{w})$, where $\mathbf{z}_{-n}$ denotes $\mathbf{z}$ excluding z_n. According to the graphical model depicted in Fig. 1 (a), we have

$$\begin{aligned}
&P(z_n = k|\mathbf{z}_{-n}, \mathbf{w}) \\
&= \frac{P(\mathbf{z}, \mathbf{w})}{P(\mathbf{z}_{-n}, \mathbf{w})} \\
&= \frac{P(\mathbf{z}_{-n}, \mathbf{w}_{-n}) P(z_n = k, w_n|\mathbf{z}_{-n}, \mathbf{w}_{-n})}{P(\mathbf{z}_{-n}, \mathbf{w}_{-n}) P(w_n|\mathbf{z}_{-n}, \mathbf{w}_{-n})} \\
&\propto P(w_n, z_n = k|\mathbf{z}_{-n}, \mathbf{w}_{-n}) \\
&= P(w_n|z_n = k, \mathbf{z}_{-n}, \mathbf{w}_{-n}) P(z_n = k|\mathbf{z}_{-n}, \mathbf{w}_{-n}) \\
&\approx \hat{\phi}_{-n}^{(k,w_n)} \hat{\theta}_{-n}^{(d_n,k)} \\
&= \frac{n_{-n}^{(k,w_n)} + \beta^{(w_n)}}{n_{-n}^{(k,\cdot)} + V\beta^{(w_n)}} \frac{n_{-n}^{(d_n,k)} + \alpha^{(k)}}{n_{-n}^{(d_n,\cdot)} + K\alpha^{(k)}},
\end{aligned} \tag{5}$$

where $n_{-n}^{(k,w_n)}$ is the number of instances of w_n in $\mathbf{w}$ assigned to the topic k excluding the current instance; $n_{-n}^{(k,\cdot)}$ is the sum of $n_{-n}^{(k,w_n)}$ over $w_n = 1, \ldots, N$; $n_{-n}^{(d_n,k)}$ is the number of words in d_n (the document that term n belongs to) assigned to topic k excluding the current instance; and $n_{-n}^{(d_n,\cdot)}$ is the sum of $n_{-n}^{(d_n,k)}$ over $k = 1, \ldots, K$.

Prior parameters α's and β's are used to balance the prior knowledge and the observation of data. Once a set of samples is available, the estimates $\hat{\theta}$ and $\hat{\phi}$ are simply given by

$$\hat{\phi}^{(k,w)} = \frac{n^{(k,w)} + \beta^{(w)}}{n^{(k,\cdot)} + V\beta^{(w)}}, \quad \hat{\theta}^{(d,k)} = \frac{n^{(d,k)} + \alpha^{(k)}}{n^{(d,\cdot)} + K\alpha^{(k)}}. \tag{6}$$

The symbols in (6) have the same meaning as in (5) except that the current instance is *not* excluded.

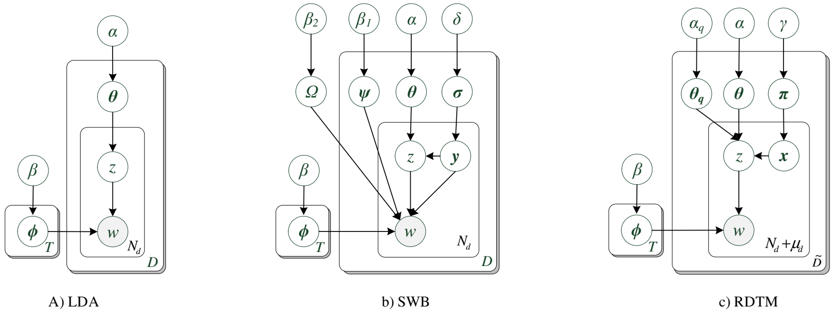


Fig. 1. Graphical models studied in this paper: (a) latent Dirichlet allocation (LDA); (b) special words with background (SWB); (c) Relevance-dependent topic model (RDTM)

2.2 Topic Model with Background Distribution

Topic models are unsupervised probabilistic models for the document collection and are generally used for extracting coarse-grained semantic information from the collection [2,7]. It assumes that words of a document are drawn from a set of topic distributions. Chemudugunta et al. [3] proposed SWB (special words with background) models for different aspects of a document. In SWB, special words are incorporated into a generative model. Each document is represented as a combination of three kinds of multinomial word distributions. Fig. 1 (b) shows the graphical model of SWB. A hidden switch variable y is used to control the generation of a word. $y = 0$ means that the word is sampled from a mixture distribution θ_z over general topics z, $y = 1$ means that the word is drawn from the document-specific multinomial distribution ψ with symmetric Dirichlet prioris parametrized by β_1, and $y = 2$ means that the word is a background word and sampled from the corpus-level multinomial distribution Ω with symmetric Dirichlet prioris parametrized by β_2.

The conditional probability of a word w given a document d can be written as:

$$\begin{aligned} P(w|d) = P(y=0|d) \sum_{z} P(w|z,\phi) P(z|\theta^{(d)}) \\ + P(y=1|d) P'(w|d,\psi) \\ + P(y=2|d) P''(w|\Omega). \end{aligned} \tag{7}$$

The model has been applied in information retrieval, and it has been showed that the model can match documents both at a general level and at a specific word level.

3 Relevance-Dependent Topic Model

3.1 LDA with Model Expansion

In the RDTM, we introduce a word-level *switch variable* x_n for a topic z_n in the graphical model of LDA. For each word position, the topic z is sampled from the distribution over topics associated with a latent variable x. It is used to determine whether to generate the word from a document specific distribution or a query specific distribution. If the word w is seen in the past relevant queries, then $x = 1$, and the word is sampled from the general topic z specific to the query $\theta_q^{(d)}$. Otherwise, then $x = 0$, and the word is sampled from the general topic z specific to the document $\theta^{(d)}$. In RDTM, observed variables include not only the words in a document but also the words in the set of queries that are relevant to the document.

The generation of $\tilde{\mathcal{D}} = \tilde{\mathbf{w}}$ is stated as follows.

- Start
- Sample from a Dirichlet prior with parameter β for the multinomial ϕ_z for each topic z;
- Sample from a Beta prior with parameter γ for the Bernoulli π;
- For each document $d \in \{1, \ldots, D\}$
  - Sample from a Dirichlet prior with parameter α for the multinomial $\theta^{(d)}$ over the topics;
  - Sample from a Dirichlet prior with parameter α_q for the multinomial $\theta_q^{(d)}$ over the topics;
  - For each word position $n = 1, \ldots, n_d, n_d + 1, \ldots, n_d + \mu_d$
    - sample from π for x_n;
    - if $x_n = 1$, sample from $\theta_q^{(d)}$ for the topic z_n; else ($x_n = 0$), sample from $\theta^{(d)}$ for the topic z_n;
  - Sample from ϕ_{z_n} for the word w_n;
- End

Fig. 1 (c) depicts the graphical model expansion. Again, for each $\tilde{\mathbf{d}} \in \tilde{\mathcal{D}}$, the observed variables consist of $\mathbf{d}$ and $\mathbf{q}(\mathbf{d})$. Given hyperparameters α, α_q, β, and γ, the joint distribution of all observed and hidden variables can be factorized as follows

$$\begin{aligned} &P(\tilde{\mathbf{d}}, \mathbf{z}, \mathbf{x}, \theta, \theta_q, \phi, \pi | \alpha, \alpha_q, \beta, \gamma) = P(\pi|\gamma)P(\phi|\beta) \\ &\times \prod_{d=1}^{D} \Big(P(\theta^{(d)}|\alpha) P(\theta_q^{(d)}|\alpha_q) \prod_{n=1}^{n_d+\mu_d} P(\tilde{w}_n|z_n, \phi) P(z_n|x_n, \theta^{(d)}, \theta_q^{(d)}) P(x_n|\pi) \Big). \end{aligned} \tag{8}$$

Recall that in Section 2.1, the generation model for the word w in a given document d is approximated by

$$\hat{P}(w|\hat{\theta}^{(d)}, \hat{\phi}) = \sum_{z=1}^{K} P(w|z, \hat{\phi}) P(z|\hat{\theta}^{(d)}), \tag{9}$$

where $\hat{\theta}$ and $\hat{\phi}$ are estimated by the Gibbs samples drawn from the posterior distribution of the hidden variables $P(\mathbf{z}|\mathbf{w})$. With RDTM, it is still infeasible to compute $P(\mathbf{z}, \mathbf{x}|\tilde{\mathbf{w}})$ directly, so we use the Gibbs sampling technique again to sample $\mathbf{z}$ and $\mathbf{x}$ from the full conditional $P(z_n, x_n|\tilde{\mathbf{w}}, \mathbf{z}_{-n}, \mathbf{x}_{-n})$ sequentially. z_n can be sampled from the following probabilities

$$\begin{aligned}
&P(z_n = k|\mathbf{z}_{-n}, \mathbf{x}, \tilde{\mathbf{w}}) \\
&\propto \; P(z_n = k, x_n|\mathbf{z}_{-n}, \mathbf{x}_{-n}, \tilde{\mathbf{w}}) \\
&\propto \; P(z_n = k|x_n, \theta^{(d)}, \theta_q^{(d)})P(\tilde{w}_n|z_n = k) \\
&\propto \begin{cases} \tilde{\theta}^{(d_n,k)}_{-n}\tilde{\phi}^{(k,\tilde{w}_n)}_{-n}, & x_n = 0, \\ \tilde{\theta}^{(\tilde{d}_n,k)}_{q,-n}\tilde{\phi}^{(k,\tilde{w}_n)}_{-n}, & x_n = 1, \quad k = 1, \ldots, K. \end{cases}
\end{aligned} \tag{10}$$

From a Gibbs sample, the approximation of $\tilde{\theta}, \tilde{\theta}_q$ and $\tilde{\phi}$ can be obtained as follows

$$\tilde{\phi}^{(k,\tilde{w})} = \frac{n^{(k,\tilde{w})} + \beta^{(w)}}{n^{(k,\cdot)} + V\beta^{(w)}}, \quad \tilde{\theta}_q^{(\tilde{d},k)} = \frac{n^{(\tilde{d},k)} + \alpha_q^{(k)}}{n^{(\tilde{d},\cdot)} + K\alpha_q^{(k)}}, \quad \tilde{\theta}^{(d,k)} = \frac{n^{(d,k)} + \alpha^{(k)}}{n^{(d,\cdot)} + K\alpha^{(k)}}. \tag{11}$$

x_n can be sampled from the odds

$$\begin{aligned}
&\frac{P(x_n = 0|\mathbf{z}, \mathbf{x}_{-n}, \tilde{\mathbf{w}})}{P(x_n = 1|\mathbf{z}, \mathbf{x}_{-n}, \tilde{\mathbf{w}})} \\
&= \frac{P(x_n = 0, z_n|\mathbf{z}_{-n}, \mathbf{x}_{-n}, \tilde{\mathbf{w}})}{P(x_n = 1, z_n|\mathbf{z}_{-n}, \mathbf{x}_{-n}, \tilde{\mathbf{w}})} \\
&= \frac{P(x_n = 0|\mathbf{z}_{-n}, \mathbf{x}_{-n}, \tilde{\mathbf{w}})P(z_n|x_n = 0, \mathbf{z}_{-n}, \mathbf{x}_{-n}, \tilde{\mathbf{w}})}{P(x_n = 1|\mathbf{z}_{-n}, \mathbf{x}_{-n}, \tilde{\mathbf{w}})P(z_n|x_n = 1, \mathbf{z}_{-n}, \mathbf{x}_{-n}, \tilde{\mathbf{w}})} \\
&= \frac{\tilde{\pi}_0 \cdot \tilde{\theta}^{(d_n,z_n)}_{-n}}{\tilde{\pi}_1 \cdot \tilde{\theta}^{(\tilde{d}_n,z_n)}_{q,-n}},
\end{aligned} \tag{12}$$

where $\tilde{\pi}_0 = \frac{n^{(d)}_{-n}+\gamma}{n^{(D)}_{-n}+2\gamma}$ and $\tilde{\pi}_1 = \frac{n^{(\tilde{d})}_{-n}+\gamma}{n^{(D)}_{-n}+2\gamma}$.

3.2 RDTM for Information Retrieval

When the corpus-level topic models are directly applied to the ad-hoc retrieval tasks, the average precision is often very low [18], due to the fact that the corpus-level topic distribution is too coarse [3,19]. Significant improvements can be achieved through a linear combination with the document model [3,18,19]. In the language-model approaches for information retrieval, the query likelihoods given the document models, $P_{\mathrm{LM}}(\mathbf{q}|\mathcal{M}_{\mathbf{d}})$ are used to rank the documents. By the bag-of-words assumption, the query likelihood can be expressed by [13]

$$P_{\mathrm{LM}}(\mathbf{q}|\mathcal{M}_{\mathbf{d}}) = \prod_{w \in \mathbf{q}} P(w|\mathcal{M}_{\mathbf{d}}). \tag{13}$$

where $\mathcal{M}_{\mathbf{d}}$ is the language model estimated based on document $\mathbf{d}$. The probability $P(w|\mathcal{M}_{\mathbf{d}})$ is defined as follows [23],

$$P(w|\mathcal{M}_{\mathbf{d}}) = \frac{n_d}{n_d + \sigma}P_{\mathrm{ML}}(w|\mathbf{d}) + (1 - \frac{n_d}{n_d + \sigma})P_{\mathrm{ML}}(w|\mathcal{D}) \quad , \tag{14}$$

with $P_{\mathrm{ML}}(w|\mathcal{D})$ (resp. $P_{\mathrm{ML}}(w|\mathbf{d})$) being the maximum likelihood estimate of a query term w generated in the entire collection $\mathcal{D}$ (resp. $\mathbf{d}$). n_d is the length of document $\mathbf{d}$. Note that (14) is a Bayesian learning of the word probability with a Dirichlet prior σ [23]. In this paper, σ is set to $1,000$ since it achieves the best results in [19].

Compared to the standard query likelihood document model, RDTM offers a new and interesting framework to model documents. Motivated by the significant improvements obtained by Wei and Croft [19], we formulate our model as the linear combination of the original query likelihood document model and RDTM

$$P(\mathbf{q}|\mathcal{M}_{\mathbf{d}}) = \lambda \tilde{P}_{\mathrm{LM}}(\mathbf{q}|\mathcal{M}_{\mathbf{d}}) + (1-\lambda) P_{\mathrm{RDTM}}(\mathbf{q}|\mathcal{M}_{\mathbf{d}}), \quad 0 \le \lambda \le 1, \tag{15}$$

The RDTM model facilitates a new representation for a document based on topics. Given the posterior estimators (11), the query likelihood $P_{\mathrm{RDTM}}(\mathbf{q}|\mathcal{M}_{\mathbf{d}})$ can be calculated as follows:

$$\begin{aligned} P_{\mathrm{RDTM}}(\mathbf{q}|\mathcal{M}_{\mathbf{d}}) &= \prod_{w \in \mathbf{q}} P_{\mathrm{RDTM}}(w|\mathcal{M}_{\mathbf{d}}) \\ &= \prod_{w \in \mathbf{q}} \sum_{z=1}^{K} P(w|z, \hat{\phi}) \Big(P(x=1|\tilde{\pi}) P(z|\hat{\theta}^{(q)}) + P(x=0|\tilde{\pi}) P(z|\hat{\theta}^{(d)}) \Big). \end{aligned} \tag{16}$$

4 Experiments

In this section we empirically evaluate RDTM in ad hoc information retrieval and compare it with other state-of-the-art models.

4.1 Data and Setting

We perform experiments on two TREC testing collections: namely the Associated Press Newswire (AP) 1988-90 on disk 1-3 with topics 51-150 as test queries, and the Wall Street Journal (WSJ) with topics 151-200 as test queries. Queries are taken from the “title” field of TREC topics only (i.e., short queries). The remaining TREC topics are used as the historical queries together with their corresponding relevant documents to learn the document models in the training phase. In other words, topics 151-300 are used as the historical queries for the AP task, while topics 51-150 and 201-300 are used as the historical queries for the WSJ task. The preprocessing steps include stemming and stop word removal.

Several parameters need to be determined in the experiments. We use symmetric Dirichlet prior with $\alpha = \alpha_q = 50/K$, $\beta = \beta_1 = 0.01$, $\beta_2 = 0.0001$, $\delta = 0.3$ and $\gamma = 0.5$, which are common settings in the literature. The number of topics K are set to 200. The interpolation parameter λ is selected by cross validation, and it is finally set to 0.7.

The retrieval performance is evaluated in terms of the mean average precision (mAP) and 11-point recall/precision. To evaluate the significance of performance difference between two methods, we employ the Wilcoxon test [8] for the outcomes. All the statistically significant performance improvements with a 95% confidence according to the Wilcoxon test are marked by stars in the results.

Table 1. The results for the query likelihood (QL) model, the LDA-based document model (LBDM), the special words with the background (SWB) model, and the relevance-dependent topic models (RDTM0 and RDTM) evaluated on the WSJ data set. The evaluation measure is the average precision.

recall	QL	LBDM	SWB	RDTM0	RDTM
0.00	0.7359	0.7501	0.7431	0.7579	0.7813* **
0.10	0.5774	0.6016	0.6072	0.6032 *	0.6044
0.20	0.4766	0.5068	0.5176	0.5500* **	0.5576* **
0.30	0.4272	0.4570	0.4745	0.4950* **	0.4919* **
0.40	0.3779	0.3843	0.4095	0.4260* **	0.4288* **
0.50	0.3265	0.3429	0.3639	0.3771* **	0.3732* **
0.60	0.2457	0.2742	0.2892	0.3016* **	0.2919* **
0.70	0.2046	0.2209	0.2321	0.2270*	0.2228*
0.80	0.1702	0.1754	0.1706	0.1703	0.1673
0.90	0.1064	0.1071	0.0993	0.0911	0.1070**
1.00	0.0551	0.0401	0.0375	0.0400 **	0.0391

4.2 Results

We compare the effectiveness of our relevance-dependent topic model (RDTM) with the query likelihood (QL) model [23], LDA-based document model (LBDM) [19] and special words with the background (SWB) model [3]. In addintion, we also adds the query terms into the relevant documents when training the LDA-based model. That is, we expand each document in the training set with the queries known to be relevant, and then learn the document language model based on the augmented text data. This method is referred to as RDTM0. For the query likelihood model, we use the Dirichlet model described in (14). Retrieval results on the WSJ collection are presented in Table 1. We can see that both RDTM0 and RDTM achieves better results than QL, LBDM and SWB. This shows that incorporating query-document relevance into the document model by using the relevant past queries is helpful to IR. From Table 1, it is obvious that both RDTM0 and RDTM significantly outperform QL. To evaluate the significance of improvements over LBDM and SWB, we employ the Wilcoxon test [8] with a 95% confidence. Statistically significant improvements of RDTM0 and RDTM over both LBDM (marked by *) and SWB (marked by **) are observed at many recall levels.

Table 2 compares the results of QL, LBDM, and RDTM0 on two data sets. We can see that both LDA-based models (LBDM and RDTM0) improve over the query likelihood (QL) model. The mAP of RDTM0 is 0.2305, which is better than those obtained by LBDM (0.2162) and QL (0.1939) on the AP collection. The relative improvement in mAP of RDTM0 over LBDM is 6.61%. In the same measure, the mAP of RDTM0 is 0.3489, which is better than those obtained by LBDM (0.3347) and QL (0.3162) on the WSJ collection. In the table, "*" and "**" mean that a significant improvement is achieved over QL and LBDM, respectively.

Table 2. The results of QL, LBDM, and RDTM0 in mean average precision. % diff indicates the relative improvement of RDTM0 over LBDM.

	QL	LBDM	RDTM0	% diff
AP	0.1939	0.2162	0.2305	6.61* **
WSJ	0.3162	0.3347	0.3489	4.24* **

Table 3. The results of LBDM, SWB, and RDTM in mean average precision. % diff indicates the relative improvement of RDTM over LBDM and SWB.

	LBDM	SWB	RDTM	% diff over LBDM	% diff over SWB
AP	0.2162	0.2274	0.2316	7.12*	1.85**
WSJ	0.3347	0.342	0.3536	5.65*	3.39**

In Table 3, we compare the retrieval results of RDTM with the LBDM and SWB on two data sets. Obviously, RDTM achieves improvements over both LBDM and SWB, and the improvements are significant. Considering that SWB has already obtained significant improvements over LBDM, the significant performance improvements of RDTM over SWB are in fact very encouraging. The mAP of RDTM is 0.3536, which is better than those obtained by SWB (0.342) and LBDM (0.3347), with a 3.39% and 5.65% improvement in mean average precision, respectively, on the WSJ collection. In the same measure, the relative improvements of mAP of RDTM over SWB and LBDM are 1.85%, and 7.12%, respectively, on the AP collection. In the table, "*" and "**" mean that a significant improvement is achieved over LBDM and SWB, respectively.

Several comments can be made based on the results. First, IR performance can be improved by using topic models for document smoothing, as it is observed that RDTM, SWB, and LBDM achieve higher mAP than QL. Second, the document representation with known relevant queries works well, as both data expansion and model expansion lead to improvements over the baseline methods. This new representation could be applied to other retrieval, classification, and summarization tasks.

5 Conclusion

In this paper, we investigate the relevance dependent generative model for text. The new methods for ad hoc information retrieval simultaneously model document contents and query information into the topic model based on latent Dirichlet allocation. One implementation is a data expansion approach that directly adds query terms into the related documents for the training of the LDA-based model (RDTM0), and the other is a model expansion approach that assumes relevant information about the query may affect the mixture of the topics in the documents (RDTM). Model expansion leads to a larger graph for which the parameter estimation is realized by the method of Gibbs sampling. Experimental results on the TREC collection show that our proposed approach achieves significant improvements over the baseline methods using the query-likelihood (QL) model and the general LDA-based document model (LBDM and SWB).

In the future, it would be interesting to explore other ways of incorporating relevance into the topic-model framework for text. As in [21], we will try to explore the utility of different types of topic models for IR. In addition, we can test our approach on large corpora (such as the World Wide Web) or train our model in a semi-supervised manner. Alternatively, we can try to add more information to extend the existing model.

Acknowledgments. This work was supported by Ministry of Economic Affairs, Taiwan, R.O.C. project under No. B3522P1200.

References

1. Andrzejewski, D., Buttler, D.: Latent Topic Feedback for Information Retrieval. In: Proceedings of ACM KDD Conference on Knowledge Discovery and Data Mining, pp. 600–608 (2011)
2. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet Allocation. Journal of Machine Learning Research 3(4-5), 993–1022 (2003)
3. Chemudugunta, C., Smyth, P., Steyvers, M.: Modeling general and specific aspects of documents with a probabilistic topic model. Advances in Neural Information Processing Systems, 241–248 (2007)
4. Chien, J.T., Wu, M.S.: Adaptive Bayesian latent semantic analysis. IEEE Transactions on Audio, Speech, and Language Processing 16(1), 198–207 (2008)
5. Griffiths, T.L., Steyvers, M.: Finding scientific topics. Proceedings of the National Academy of Sciences, 5228–5235 (2004)
6. Heidel, A., Chang, H.A., Lee, L.S.: Language Model Adaptation Using Latent Dirichlet Allocation and an Efficient Topic Inference Algorithm. In: Proceedings of INTERSPEECH, pp. 2361–2364 (2007)
7. Hofmann, T.: Probabilistic latent semantic indexing. In: Proceedings of ACM SIGIR International Conference on Research and Development in Information Retrieval, pp. 50–57 (1999)
8. Hull, D.: Using statistical testing in the evaluation of retrieval experiments. In: Proceedings of ACM SIGIR International Conference on Research and Development in Information Retrieval, pp. 329–338 (1993)
9. Levy, M., Sandler, M.: Learning latent semantic models for music from social tags. Journal of New Music Research 2(37), 137–150 (2008)
10. Mei, Q., Cai, D., Zhang, D., Zhai, C.: Topic Modeling with Network Regularization. In: Proceeding of the 17th International Conference on World Wide Web, pp. 101–110 (2008)
11. Minka, T., Lafferty, J.D.: Expectation-propagation for the generative aspect model. In: Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence, pp. 352–359 (2002)
12. Scholer, F., Williams, H.E.: Query association for effective retrieval. In: Proceedings of the ACM CIKM International Conference on Information and Knowledge Management, pp. 324–331 (2002)

13. Song, F., Croft, W.B.: A general language model for information retrieval. In: Proceedings of ACM SIGIR International Conference on Research and Development in Information Retrieval, pp. 279–280 (1999)
14. Song, W., Yu, Z., Liu, T., Li, S.: Bridging topic modeling and personalized search. In: Proceedings of COLING, pp. 1167–1175 (2010)
15. Tao, T., Wang, X., Mei, Q., Zhai, C.: Language Model Information Retrieval with Document Expansion. In: Proceedings of HLT/NAACL, pp. 407–414 (2006)
16. Wallach, H.: Topic Modeling: Beyond Bag-of-Words. In: Proceedings of the 23rd International Conference on Machine Learning, pp. 977–984 (2006)
17. Wang, J.C., Wu, M.S., Wang, H.M., Jeng, S.K.: Query by Multi-tags with Multilevel Preferences for Content-based Music Retrieval. In: IEEE International Conference on Multimedia and Expo (ICME) (2011)
18. Wang, X., McCallum, A., Wei, X.: Topical N-Grams: phrase and topic discovery, with an application to information retrieval. In: Seventh IEEE International Conference on Data Mining (ICDM), pp. 697–702 (2007)
19. Wei, X., Croft, W.B.: LDA-based document models for ad-hoc retrieval. In: Proceedings of ACM SIGIR International Conference on Research and Development in Information Retrieval, pp. 178–185 (2006)
20. Wu, M.S., Lee, H.S., Wang, H.M.: Exploiting semantic associative information in topic modeling. In: Proceedings of the IEEE Workshop on Spoken Language Technology, pp. 384–388 (2010)
21. Yi, X., Allan, J.: A Comparative Study of Utilizing Topic Models for Information Retrieval. In: Boughanem, M., Berrut, C., Mothe, J., Soule-Dupuy, C. (eds.) ECIR 2009. LNCS, vol. 5478, pp. 29–41. Springer, Heidelberg (2009)
22. Zhai, C.: Statistical Language Models for Information Retrieval: A Critical Review. Foundations and Trends in Information Retrieval 3(2), 137–213 (2008)
23. Zhai, C., Lafferty, J.D.: A study of smoothing methods for language models applied to Ad Hoc information retrieval. In: Proceedings of ACM SIGIR International Conference on Research and Development in Information Retrieval, pp. 334–342 (2001)
24. Zhai, C., Lafferty, J.D.: Model-based feedback in the language modeling approach to information retrieval. In: Proceedings of the CIKM International Conference on Information and Knowledge Management, pp. 403–410 (2001)

A Two-Stage Approach for Generating Topic Models

Yang Gao, Yue Xu, Yuefeng Li, and Bin Liu

School of Electrical Engineering and Computer Science,
Queensland University of Technology, Brisbane, Australia
{y10.gao,b5.liu}@student.qut.edu.au, {yue.xu,y2.li}@qut.edu.au

Abstract. Topic modeling has been widely utilized in the fields of information retrieval, text mining, text classification etc. Most existing statistical topic modeling methods such as LDA and pLSA generate a term based representation to represent a topic by selecting single words from multinomial word distribution over this topic. There are two main shortcomings: firstly, popular or common words occur very often across different topics that bring ambiguity to understand topics; secondly, single words lack coherent semantic meaning to accurately represent topics. In order to overcome these problems, in this paper, we propose a two-stage model that combines text mining and pattern mining with statistical modeling to generate more discriminative and semantic rich topic representations. Experiments show that the optimized topic representations generated by the proposed methods outperform the typical statistical topic modeling method LDA in terms of accuracy and certainty.

Keywords: Topic modeling, Topic representation, Tf-idf, Frequent pattern mining, Entropy.

1 Introduction

The statistical topic modeling technique has attracted big attention due to its more robust and interpretable topic representations and wide applications in the fields of information retrieval, text mining, text classification, scientific publication topic analysis and prediction[1-4] etc. It starts from Latent Semantic Analysis (LSA) [5] that can capture most significant feature of collection based on semantic structure of relevant documents. Probabilistic LSA (pLSA) [6] and Latent Dirichlet Allocation (LDA) [7] are variations to improve the interpretation of results from statistical view of LSA. These techniques are more effective on document modeling and topic extraction, which are represented by topic-document and word-topic distribution, respectively. Many topic models not only automatically extract topics from text, but also detect the evolution of topics over time [8], discover the relationship among the topics [9], supervise the topics [10] with other information (authorship, citations, et al.) for extensional applications, such as recommendation [11] and so on.

Basically, the existing statistical topic modeling approaches generate multinomial distributions over words to represent topics in a given text collection. The word distributions are derived based on word frequency in the collection. Therefore, popular words are very often chosen to represent topics. For instance, Table 1 shows an

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 221–232, 2013.

example of multinomial word distributions used to represent four topics of a scientific publication collection. It can be seen from Table 1 that word “method” dominantly occurs across all four topics with high probability. It is obvious that “method” is a general word and very popularly used in describing research works in almost all different areas. It actually will not contribute much to uniquely represent distinctive features of any research area or topic. These kind of popular words bring a lot of confusion to the topic representation other than distinctively representing the topics.

Table 1. An example of topic representation using word distributions

Topic 0	Topic 11	Topic 12
method 0.04 , sample 0.04	**method** 0.07 , predict 0.06	classification 0.13, feature 0.08
distribute 0.04, dimension 0.03	linear 0.03, weight 0.03	accuracy 0.04, class 0.04
parameter 0.03	kernel 0.03	**method** 0.04

Except for the ambiguity problem produced by popular words, another fundamental problem is that topics are represented by multinomial distribution of isolated words which lack semantic and interpretable meaning. Although topic models can supply much information and annotate documents with the discovered topics and also supply word distribution for each topic, users still have difficulties to interpret the semantic meanings of the topics only based on the distribution of words, especially for those who are not very familiar with the related area. Mei et al. [12] and Lau et al. [13] developed automatic labeling methods for interpreting the semantics of topics by phrases. But, they heavily depend on candidate resources for labeling topics. If the topics themselves are diverse or novel to the candidate dataset, the systems will mislabel the topics. Although Lau et al. [14] labeled a topic by selecting a single term from the known distribution of words rather than candidate resources, the selected word can hardly represent the whole topic well.

In order to solve the problems of word ambiguity and semantic coherence that exist in almost all topic models, we need new model to update the topic representations. The new method should extract more distinctive representations and discover the hidden associations under multinomial words distributions. In text mining, many methods have been developed to generate text representation for a collection of documents. Most text mining methods are keyword-based approaches which use single words to represent documents. Based on the hypothesis that phrases may carry more semantic meaning than keywords, approaches to use phrases instead of keywords have also been proposed. However, investigations have found that phrase-based methods were not always superior to keyword based methods [15-17]. Recently, data mining based methods have been proposed to generate patterns to represent documents which have achieved promising results [18]. Topic modeling has the advantage of classification from large collections, while text mining is good at extracting interesting features to represent collections. So, it leads us to improve the accuracy and coherence of topic representations by utilizing text mining techniques, especially term weighting and pattern mining methods.

In this paper, a two-stage approach is proposed to combine the statistical topic modeling technique with the classical data mining techniques with the hope to improve the accuracy of topic modeling in large document collections. In stage 1, the most recognized topic modeling method Latent Dirichlet Allocation (LDA) is used to generate initial topic models. In stage 2, the most popular used term weighting method tf-idf and the frequent pattern mining method are used to derive more discriminative terms and patterns to represent topics of the collections. Moreover, the frequent patterns reveal structural information about the associations between terms that make topics more understandable, semantically relevant and cover broaden meanings.

2 Stage 1 – Topic Representation Generation

Latent Dirichlet Allocation [7] is a typical statistical topic modeling technique and the most common topic modeling tool currently in use. It can discover the hidden topics in collections of documents with the appearing words. Let $D = \{d_1, \cdots, d_M\}$ be a collection of documents, called documents database. The total number of documents in corpus is *M*. The idea behind LDA is that every document is considered involving multiple topics and each topic can be defined as a distribution over fixed vocabulary of terms that appear in documents. Specifically, LDA models a document as a probabilistic mixture of topics and treats each topic as a probability distribution over words. For the *i*th word in document *d*, denoted as $w_{d,i}$, the probability of $w_{d,i}$, $P(w_{d,i})$ is defined as:

$$P\left(w_{d,i}\right) = \sum_{j=1}^{V} P(\, w_{d,i}|z_{d,i} = Z_j) \times P(z_{d,i} = Z_j) \tag{1}$$

where $z_{d,i}$ is the topic assignment for $w_{d,i}$, $z_{d,i} = Z_j$ means that the word $w_{d,i}$ is assigned to topic *j*, Z_j represents topic *j* and the *V* represents the total number of topics. Let $\boldsymbol{\phi}_j$ be the multinomial distribution over words for Z_j , $\boldsymbol{\phi}_j = \left(\varphi_{j,1}, \varphi_{j,2}, \cdots, \varphi_{j,n}\right)$, $\sum_{k=1}^{n} \varphi_{j,k} = 1$. $\varphi_{j,k}$ indicates the proportion of the *k*th word in ic Z_j, that is, $\varphi_{j,i} = P(\, w_{d,i}|z_{d,i} = Z_j)$. $\boldsymbol{\theta}_d$ refers to multinomial distribution over topics in document *d*, which is *P*(*Z*). $\boldsymbol{\theta}_d = (\vartheta_{d,1}, \vartheta_{d,2}, \cdots, \vartheta_{d,V})$, $\sum_{j=1}^{V} \vartheta_{d,j}$ =1. $\vartheta_{d,j}$ indicates the proportion of topic *j* in document *d*. LDA is generative model that only observed variable is $w_{d,i}$, while $\boldsymbol{\phi}_j$, $\boldsymbol{\theta}_d$, $z_{d,i}$ are all latent variables that need to be estimated. Blei et al. [7] introduce Dirichlet to the posterior probability $\boldsymbol{\phi}_j$ and $\boldsymbol{\theta}_d$, which optimize the topics and documents distributions.

Among many available algorithms for estimating hidden variables, the Gibbs sampling method is a very effective strategy for parameter estimation [19, 20]. The results of LDA are at two levels, corpus level and document level. At corpus level, *D* is represented by a set of topics each of which is represented by a probability distribution over word, $\boldsymbol{\phi}_j$ for topic *j*. Overall, we have $\Phi = \{\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \cdots, \boldsymbol{\phi}_V\}$ for all topics. For illustrating the results derived by LDA, let's look at a simple example depicted in Table 2 to Table 4. Let $D = \{d_1, d_2, d_3, d_4\}$ be a small set of four documents and there are 12 words appearing in the documents. Assuming the documents in *D* involve 3 topics, Z_1, Z_2, and Z_3. Table 2 illustrates the word distribution for each of the topics. At document level, each document d_i is represented by topic distributions $\boldsymbol{\theta}_{d_i}$. For the simple example mentioned above, the document representation is illustrated in Table 3. Apart from these two level outcomes, LDA also generates word – topic assignment, that is, the word occurrence is considered related to the topics by LDA. Table 4 illustrates an example of the word-topic assignments.

Table 2. Example results of LDA: Topic representation – probability distribution over words

Topic	Φ
$\boldsymbol{\phi}_1$	w_2: $\frac{1}{3}$, w_1: $\frac{1}{5}$, w_4: $\frac{2}{15}$, w_7: $\frac{2}{15}$, w_3: $\frac{1}{15}$, w_5: $\frac{1}{15}$, w_6: $\frac{1}{15}$
$\boldsymbol{\phi}_2$	w_8: $\frac{1}{3}$, w_1: $\frac{4}{15}$, w_7: $\frac{2}{15}$, w_9: $\frac{2}{15}$, w_2: $\frac{1}{15}$
$\boldsymbol{\phi}_3$	w_{10}: $\frac{4}{13}$, w_{11}: $\frac{3}{13}$, w_1: $\frac{2}{13}$, w_7: $\frac{2}{13}$, w_4: $\frac{1}{13}$, w_{12}: $\frac{1}{13}$

Table 3. Example results of LDA: Document representation – probability distribution over topics

Document	Z_1 ($\vartheta_{d_i,1}$)	Z_2 ($\vartheta_{d_i,2}$)	Z_3 ($\vartheta_{d_i,3}$)
d_1	0.6	0.2	0.2
d_2	0.2	0.5	0.3
d_3	0.3	0.3	0.4
d_4	0.3	0.4	0.3

Table 4. Example results of LDA: word – topic assignments

Docu-ment	Z_1		Z_2		Z_3	
	$\vartheta_{d,1}$	words	$\vartheta_{d,2}$	words	$\vartheta_{d,3}$	words
d_1	0.6	w_1,w_2,w_3,w_2,w_1	0.2	w_1,w_9,w_8	0.2	w_7,w_{10},w_{10}
d_2	0.2	w_2,w_4 ,w_4	0.5	w_7, w_8,w_1,w_8, w_8	0.3	w_1,w_{11},w_{12}
d_3	0.3	w_2,w_1,w_7,w_5	0.3	w_7,w_1,w_3,w_2	0.4	w_4,w_7,w_{10},w_{11}
d_4	0.3	w_2,w_7,w_6	0.4	w_9,w_8,w_1	0.3	w_1,w_{11},w_{10}

The topic representation using word distribution and the document representation using topic distribution are the most important contributions provided by LDA. The topic representation indicates which words are important to which topic and the document representation indicates which topics are important for a particular document. These representations have been widely used in various application domains such as information retrieval, document classification, text mining etc. On the other hand, the word-topic assignments also indicate which words are important to which topics, which is similar to the topic representation. However, the topic representation is at corpus level, while the word-topic assignments are at document level, which implicate more detailed or more specific association between topics and words. In this paper, we propose to mine word-topic assignments generated by LDA for more accurate or more discriminative topic representations for a given collection of documents.

3 Stage 2 – Topic Representation Optimization

For most LDA based applications, the words with high probabilities in topics' word distributions are usually chosen to represent topics. For example, the top 4 words for the 3 topics, as showed in Table 2, are: w_2, w_1, w_4, w_7 for topic 1, w_8, w_1, w_7, w_9 for topic 2 and w_{10}, w_{11}, w_1, w_7 for topic 3. From the simply example we can see that words w_1 and w_7 have relatively high probabilities for all the three topics. That means, they most likely represent general concepts or common concepts of the three topics and cannot distinctively represent the three topics. Moreover, the words in topic representations generated by LDA are individual single words. These single words provide too limited information about the relationships between the words and too limited semantic meaning to make the topics understandable. In this section, we propose two methods based on text mining and pattern mining techniques, which are detailed in the following sub sections, aiming at alleviating the mentioned problems.

3.1 Tf-idf Weighting Based Topic Modeling

The first method is based on the well-known term weighting method tf-idf (term frequency – inverse document frequency). The distinct feature of the tf-idf method is that it chooses discriminative terms to represent a document or a topic rather than popular terms. As we illustrated in the above example, there exist general or common terms in the topics' word distributions generated by LDA. We propose to utilize the tf-idf technique to process the topics' word distributions in order to generate more discriminative words to represent topics. As illustrated in Table 4, LDA generates word-topic assignments for each document, which reveal word importance to topics for that document. The basic idea of the proposed tf-idf based method is to find the discriminative words from the words which are assigned to a topic by LDA to represent that topic. There are two steps in the proposed method. The first step is to construct a collection called *topical document collection*, denoted as D_{topic}. Each document in the collection consists of all the word-topic assignments to a topic in the original document collection D. The second step is to generate a set of words for representing each document in D_{topic} by applying the tf-idf method to the collection.

(1) Construct Collection D_{topic}

Let R_{d_i,Z_j} represent the word-topic assignment to topic Z_j in document d_i. $R_{d_i,Z_j}$is a sequence of words assigned to topic Z_j in document d_i. For the example illustrated in Table 4, for topic Z_1 in document d_1, R_{d_1,Z_1} = <w_1, w_2, w_3, w_2, w_1 >, or simply R_{d_1,Z_1} = w_1 w_2 w_3 w_2 w_1. Each document d_i' in D_{topic} is defined as

$$d_i' = \{R_{d_i,Z_j} | d_i \in D\} \tag{2}$$

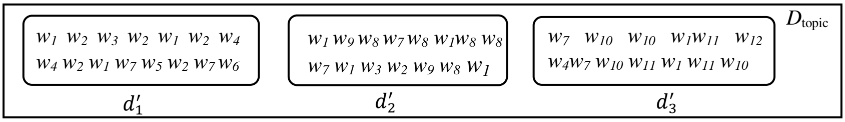

D_{topic}
w_1 w_2 w_3 w_2 w_1 w_2 w_4
w_4 w_2 w_1 w_7 w_5 w_2 w_7 w_6
d_1'
w_1 w_9 w_8 w_7 w_8 w_1w_8 w_8
w_7 w_1 w_3 w_2 w_9 w_8 w_1
d_2'
w_7 w_{10} w_{10} w_1w_{11} w_{12}
w_4w_7 w_{10} w_{11} w_1 w_{11} w_{10}
d_3'

Fig. 1. Dtopic with three topical documents

d_i' consists of the word-topic assignments R_{d_i,Z_j} to topic Z_j, each word-topic assignment R_{d_i,Z_j} can be treated as a sentence in the document d_i'. d_i' is called a *topical document* since it consists of the words for a particular topic. Assuming that the original document collection D has V number of topics, the collection D_{topic} is defined as $D_{\text{topic}} = \{d'_1, d'_2, \cdots, d'_V\}$. For the example given in Table 4, a topical document collection can be constructed as showed in Fig.1.

(2) Generate Document Representation for Collection D_{topic}

For the topical document, the word distribution over topic j, denoted as $(\boldsymbol{\phi}_j)_{\text{tf-idf}}$, is generated based on their tf-idf scores, which are calculated by equation (3). $tf(t_{i,j})$ is the frequency of term $t_{i,j}$ in the ith topical document, where $|d'_i|$ is the count of terms in d'_i, $N(t_{i,j})$ is the count of $t_{i,j}$ appearing in d'_i. Inverse document frequency (idf) reflects the popularity of term $t_{i,j}$ across topical documents in D_{topic}, where V is the total number of topical documents and $df(t_{i,j})$ is the document frequency. Thus, high tf-idf term weighting indicates high term frequency but low overall collection frequency.

$$tfidf(t_{i,j}) = tf(t_{i,j}) \times idf(t_{i,j}) = \frac{N(t_{i,j})}{|d'_i|} \times log\frac{V+1}{df(t_{i,j})} \tag{3}$$

Table 5 provides an example of the results which shows that, the tf-idf method weakens the effect of the common words w_1 and w_7, in the meanwhile, increases the weights for the distinctive words in each topic.

Table 5. Example results of tf-idf: Topic representation – probability distribution over words

Topic	$\Phi_{\text{tf-idf}}$
$\boldsymbol{\phi}_1$	w_2: 0.1 , w_4: 0.04 , w_5: 0.04, w_6: 0.04, w_1: 0.02 , w_3: 0.02, w_7: 0.017
$\boldsymbol{\phi}_2$	w_8: 0.2 , w_9: 0.08 , w_1: 0.03, w_2: *0.02,* w_7: 0.017
$\boldsymbol{\phi}_3$	w_{10}: 0.19 , w_{11}: 0.14, w_{12}: 0.046 ,$w4$: 0.023, w_1: 0.019 , w_7: 0.019

3.2 Pattern-Based Topic Modeling

A pattern is usually defined as a set of related terms or words. As discussed in Section 1, patterns carry more semantic meaning and are more understandable than isolated words. The idea of the pattern based representations starts from the knowledge of frequent patterns mining. It plays an essential role in many data mining tasks that try to find interesting patterns from datasets. We believe that pattern based representations can be more meaningful and more accurate to represent topics. Moreover, pattern based representations contain structural information which can reveal the association between the terms.

(1) Construct Transactional Dataset

The purpose of the proposed pattern based method is to discover associated words (i.e., patterns) from the words assigned by LDA to topics. With this purpose in mind, we construct a set of words from each word-topic assignment R_{d_i,z_j} instead of using the sequence of words in R_{d_i,z_j}, because for pattern mining, the frequency of a word within a transaction is insignificant. Let I_{ij} be a set of words which occur in R_{d_i,Z_j}, I_{ij} $=\{w | w \in R_{d_i,Z_j}\}$, i.e., I_{ij} contains the words which are in document d_i and assigned to topic Z_j by LDA. I_{ij} is called a *topical document transaction*, is a set of words without any duplicates. From all the word-topic assignments R_{d_i,Z_j} to topic Z_j, we can construct a transactional dataset $\mathcal{T}_j$. Let $D = \{d_1, \cdots, d_M\}$ be the original document collection, the transactional dataset $\mathcal{T}_j$ for topic Z_j is defined as $\mathcal{T}_j = \{I_{1j}, I_{2j}, \dots I_{Mj}\}$. For the topics in D, we can construct V transactional datasets. An example of the transactional datasets is illustrated in Fig.2, which is generated from the example in Table 4.

Transactional datasets

transaction	topic document transaction
1	$\{w_1, w_2, w_3\}$
2	$\{w_2, w_4\}$
3	$\{w_1, w_2, w_5, w_7\}$
4	$\{w_2, w_6, w_7\}$

$\mathcal{T}_1$

transaction	topic document transaction
1	$\{w_1, w_8, w_9\}$
2	$\{w_1, w_7, w_8\}$
3	$\{w_1, w_2, w_3, w_7\}$
4	$\{w_1, w_8, w_9\}$

$\mathcal{T}_2$

transaction	topic document transaction
1	$\{w_7, w_{10}\}$
2	$\{w_1, w_{11}, w_{12}\}$
3	$\{w_4, w_7, w_{10}, w_{11}\}$
4	$\{w_1, w_{11}, w_{10}\}$

$\mathcal{T}_3$

Fig. 2. Transactional datasets generated from Table 4

(2) Generate Pattern-based Representation

Frequent itemsets are the most widely used patterns generated from transactional datasets to represent useful or interesting patterns. The basic idea of the proposed pattern based method is to use the frequent patterns generated from each transactional dataset $\mathcal{T}_j$ to represent topic Z_j. For a given minimal support threshold σ, and itemset p in $\mathcal{T}_j$ is frequent if $supp(p) >= \sigma$, where $supp(p)$ is the support of p which is the number of transactions in $\mathcal{T}_j$ that contain p. Take $\mathcal{T}_2$ as an example, which is the transactional dataset for topic Z_2. For a minimal support threshold $\sigma = 2$, all the frequent patterns generated from $\mathcal{T}_2$ are given in Table 6. $\{w_8\}$ and $\{w_1, w_8\}$ are the dominant patterns for topic 2. Comparing with the term based topic representation, patterns represent the associated words that carry more concrete and identifiable meaning. For instance, "data mining" is more concrete than just one word "mining" or "data".

Table 6. The frequent patterns discovered from the Z_2 topical transaction database. $\sigma = 2$

Patterns	*supp*
$\{w_8\}$,$\{w_1,w_8\}$	3
$\{w_9\}$,$\{w_8,w_9\}$,$\{w_1,w_9\}$,$\{w_1,w_8,w_9\}$,$\{w_1,w_7\}$	2

4 Experiments and Evaluation

We have conducted experiments to evaluate the performance of the proposed two topic modeling methods. In this section, we present the results of the evaluation.

4.1 Datasets

Four datasets are used in the experiments, which contain the abstracts of the papers published in the proceedings of KDD, SIGIR, CIKM and HT from 2002 to 2011. The four datasets contain 1227, 1722, 2048 and 483 abstracts, respectively. The abstracts are crawled from the ACM digital library[1], and stemmed by using Porter's stemmer package[2] in the Apache's Lucene Java.

4.2 Experiment Procedure

The whole procedure taken in the experiments is depicted in Fig. 3. The first step is dataset preparation to construct the datasets described in Section 4.1. Then in the step of topic generation, we utilize the sampling-based LDA tool provided in MALLET[3] to generate LDA topic models. The number of topics $V = 20$, the number of iterations of Gibbs sampling is 1000, the hyperparameters of LDA $\alpha = 50/V = 2.5$, $\beta = 0.01$ in this experiment [20]. Step 3 is to construct the topical document datasets and the transactional datasets for optimizing topic representations, and the final step is to generate the discriminative terms based and the frequent pattern based topic representations using the pro-posed methods introduced in Section 3. We divide each dataset into training set and testing set, 90% of the documents in each dataset are used as the training set for generating topic models, while the other 10% of the documents in each dataset are left for evaluation.

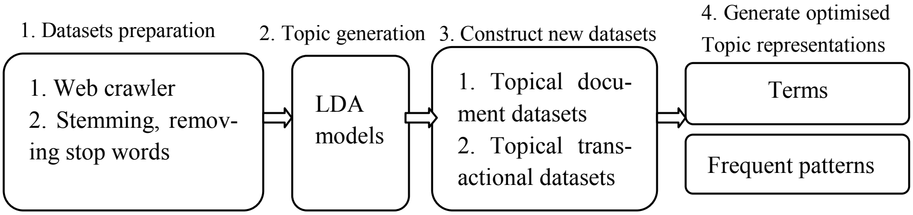

1. Datasets preparation
1. Web crawler
2. Stemming, removing stop words
2. Topic generation
LDA models
3. Construct new datasets
1. Topical document datasets
2. Topical transactional datasets
4. Generate optimised Topic representations
Terms
Frequent patterns

Fig. 3. Four steps taken for optimizing topic representation

4.3 Experiment Result Analysis

LDA is chosen as the baseline model to compare with the two proposed methods in the experiments. Table 7 demonstrates some examples of the topic representations generated by using the three models, i.e., the LDA model, the tf-idf based model, and the pattern based model. The top 12 words or patterns in each of the topic representations generated by the three models are displayed in Table 7 for two topics, topic 4 and topic 0, of dataset KDD.

[1] http://dl.acm.org/

[2] http://tartarus.org/martin/PorterStemmer

[3] http://mallet.cs.umass.edu/index.php. MALLET is a Java-based package for topic modeling and other machine learning applications to text.

Table 7. Examples of topics representations (topic 4 and topic 0 for dataset KDD)

Topic 4			Topic 0		
Baseline	Tf-idf	Patterns	Baseline	Tf-idf	Patterns
large	large	large	method	sample	method
algorithm	scale	algorithm	sample	dimension	distribution
compute	algorithm	compute	distribution	parameter	high
efficient	efficient	efficient	dimension	gene	sample
scale	highly	scale	parameter	distance	dimension
number	fast	number	estimate	outlier	estimate
size	size	size	distance	method	parameter
order	number	large scale	high	low	high dimension
correlate	pair	large algorithm	gene	distribution	number
highly	million	order	paper	high	sample method
local	memory	large compute	random	component	distribution method
fast	faster	large efficient	outlier	random	component

Table 8. Sample patterns in 5 topic representations for dataset KDD

Topic	Patterns
1	Probabilistic model, Information model, Text document, Topic model, Makov model
9	Clustering based algorithm, Result algorithm, Algorithm quality, Hierarchical cluster
10	Data mining, Data set, Data analysis, Data application, Data method, Data set mining
14	Web user, User search, Query search, User query, User recommendation,
18	Pattern mining, Frequent mining, Frequent patterns, Rule mining, Association mining,

From the results we can see that the top 12 words or patterns have a large overlap between each pair of the three methods, which could indicate that all the three methods can derive similar representations. But, when taking a close look, we can find that the results generated by the pattern based method provide much more concrete and specific meaning. For example, for topic 4, all the three methods rank ‘large’ as the top 1 word which is a general term. However, the pattern based method generates more specific patterns ‘large algorithm’, ‘large scale’, and ‘large compute’ which make the topic representation much easier to understand, while the other two methods cannot. Similar evidence can be seen for topic 0 as well. We have showed an example in Table 1 that the word ‘method’ was chosen by LDA for representing three topics including topic 0. In Table 7, the topic representations for topic 0 generated by the three methods are listed, from which we can see that, the ranking of the word ‘method’ was decreased by the tf-idf based method. This indicates that the word ‘method’ is not a discriminative word for uniquely representing topic 0. Moreover, the pattern based representations enrich the content of the topic representations generated by existing models such as LDA by discovering hidden associations among words, which makes the topics more detailed and comprehensive. Just for illustrating the usefulness of the pattern based method, we display in Table 8 some other patterns contained in the topic representations for dataset KDD. From the results we can see that patterns supply meaningful and semantic topic representations.

4.4 Evaluation

The ultimate goal of the proposed methods as well as other existing topic modeling methods is to represent the topics of a given collection of documents as accurately as possible. For the existing topic modeling methods and the proposed methods, the topic representations are word or pattern distributions with probabilities. The more certain the chosen words or patterns are in the topic representations, the more accurate the topic representations become. By taking this view, in this paper, we use *information entropy*, a well known certainty measurement developed in information theory, as the merit to evaluate the generalization performance of the proposed methods. Using the documents in the testing set, we compute the entropy of the topic models generated from the training set to evaluate the performance of the proposed models. The lower the entropy, the more certain the topic models to represent the topics and therefore the more predictable the documents' topics are. Formally, for a testing set D_{test}, the entropy of the topic models is defined as:

$$\text{entropy}(D_{\text{test}}) = -\sum_{z\in Z}\sum_{\mathrm{d}\in D_{\text{test}}}\sum_{\mathrm{w}\in d} p(w|z)p(z)\log[p(w|z)p(z)] \tag{4}$$

where $p(w|z)$ is the topic representation $\boldsymbol{\phi}_z$for a topic derived by LDA, the tf-idf based, and the pattern based methods. $p(z)$ is the document representation $\boldsymbol{\theta}_d$ generated from LDA. For the evaluation, both the tf-idf weighting and patterns supports have been normalized into probabilities. The evaluation result is presented in Table 9.

Table 9. Evaluation results on 4 datasets

Datasets	Baseline(LDA)	Tf-idf	Patterns
KDD	32.6	31.8	12.4
SIGIR	42.5	40.4	20.1
CIKM	49.7	47.7	26.6
HT	10.9	10.2	4.5

The evaluation clearly indicates that the tf-idf based model fairly achieved lower entropy values than the baseline model, meaning that, it has better performance when interpreting the meaning of the topics. Furthermore, the pattern based method achieved even much lower entropy values than any of the other two. Based on the results, we can conclude that the pattern based method apparently can generate more certain and more accurate representations for the topics of a document collection.

5 Related Work

Topic models have been extended to capture more interesting properties [7-10,19-20], but most of them represent topics by multinomial word distributions. Topic labeling [12-14] is a prevalent method to express semantic meaning of topics as mentioned in Introduction. For another example, Magatti et al. [21] present a method to calculate the similarities between given topics and known hierarchies, then choose the most agreed labels to represent the topics. However, the drawback of the existing methods of topic labeling is that they are heavily restricted to candidate resources and limited on semantic coverage. Topical *n*-gram (TNG) [22] model discovers topically-relevant phrases by Markov dependencies in word sequences based on the structure of LDA, which is relevant to our work. Except for the method of generating topic phrases, Zhao et al. [23] proposed a principled probabilistic phrase ranking algorithm for extracting top keyphrases as topic representations from the candidate phrases. The results provided in [22] and [23] show that the topics represented by the phrases are more interpretable than that of its LDA counterpart. But comparing with the pattern based representations proposed in this paper, the phrases may share low occurrences in documents, which can't achieve effective retrieval performance.

6 Conclusion

This paper proposed a two stage model to generate more discriminative and semantic rich representations for modeling the topics in a given collection of documents. The main contribution of this paper is the novel approach of combining data mining techniques and statistical topic modeling techniques to generate pattern based representations and discriminative term based representations for modeling topics. In the first stage of the proposed approach, any topic modeling method, as long as it can generate words distributions over topics, can be used to generate the initial topic representations for documents in the collection. In the second stage, we proposed to mine the initial topic representations generated in the first stage for more accurate topic representations by using the term weighting method tf-idf and the pattern mining method. Our experiment results show that the pattern based representations and the discriminative term based representations generated in the second stage are more accurate and more certain than the representations generated by the typical statistical topic modeling method LDA. Another strength provided by the pattern based representations is the structural information carried within the patterns. In the future, we will further study the structure of the patterns and discover the relationship between words which will represent the topics at a more detailed level.

References

1. Mei, Q., Zhai, C.X.: Discovering Evolutionary Theme Patterns from Text: An Exploration of Temporal Text Mining. In: KDD 2005, pp. 198–207 (2005)
2. Zhai, C.X., Velivelli, A., Yu, B.: A Cross-collection Mixture Model for Comparative Text Mining. In: KDD 2004, pp. 1285-129 (2004)
3. Wei, X., Croft, W.B.: LDA-based Document Models for Ad-hoc Retrieval. In: SIGIR 2006, pp. 178–185 (2006)
4. Wang, X., McCallum, A., Wei, X.: Topical N-grams: Phrase and Topic Discovery, with an Application to Information Retrieval. In: ICDM 2007, pp. 697–702 (2007)
5. Deerwester, S., et al.: Indexing by Latent Semantic Analysis. Journal of the American Society for Information Science 41(6), 391–407 (1990)

6. Hofmann, T.: Unsupervised Learning by Probabilistic Latent Semantic Analysis. Machine Learning 42(1), 177–196 (2001)
7. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet Allocation. Journal of Machine Learning Research 3, 993–1022 (2003)
8. Blei, D.M., Lafferty, J.D.: Dynamic Topic Models. In: Proceedings of the 23rd International Conference on Machine Learning, pp. 113–120 (2006)
9. Blei, D.M., Lafferty, J.D.: A Correlated Topic Model of Science. Annals of Applied Statistics 1(1), 17–35 (2007)
10. Blei, D.M., McAuliffe, J.D.: Supervised Topic Models. In: Adv. NIPS (2007)
11. Wang, C., Blei, D.M.: Collaborative Topic Modeling for Recommending Scientific articles. In: KDD 2011, pp. 448–456 (2011)
12. Mei, Q., Shen, X., Zhai, C.: Automatic Labeling of Multinomial Topic Models. In: KDD 2007, pp. 490–499 (2007)
13. Lau, J.H., et al.: Automatic Labelling of Topic Models. In: Proceeding of the 49th Annual Meeting of the Association for Computational Linguistics, pp. 1536–1545 (2011)
14. Lau, J.H., et al.: Best Topic Word Selection for Topic Labelling. In: Proceedings of the 23rd International Conference on Computional Linguistics, pp. 605–613 (2010)
15. Lewis, D.D.: An Evaluation of Phrasal and Clustered Representations on a Text Categorization task. In: SIGIR 1992, 15th ACM International Conference on Research and Development in Information Retrieval, pp. 37–50 (1992)
16. Scott, S., Matwin, S.: Feature Engineering for Text Classification. In: The 16th International Conference on Machine Learning, pp. 379–388 (1999)
17. Sebastiani, F.: Machine Learning in Automated Text Categorization. ACM Computing Surveys (CSUR) 34(1), 1–47 (2002)
18. Wu, S.-T., Li, Y., Xu, Y.: Deploying Approaches for Pattern Refinement in Text Mining. In: IEEE International Conference on Data Mining, pp. 1157–1161 (2006)
19. Steyvers, M., Griffiths, T.L.: Finding Scientific Topics. Proceedings of the National Academy of Sciences 101, 5228–5235 (2004)
20. Steyvers, M., Griffiths, T.: Probabilistic Topic Models. Handbook of Latent Semantic Analysis 427(7), 424–440 (2007)
21. Magatti, D., et al.: Automatic Labeling of Topics. In: Ninth International Conference on Intelligent Systems Design and Applications, pp. 1227–1232 (2009)
22. Wang, X., McCallum, A., Wei, X.: Topical N-grams: Phrase and Topic Discovery, with an Application to Information Retrieval. In: ICDM 2007, pp. 697–702 (2007)
23. Zhao, W.X., et al.: Topical keyphrase extraction from Twitter. In: Proceedings of 49th Annual Meeting of the Assocation for Computational Linguistics: Human Language Technologies (HLT 2011) (2011)

Effective Top-Down Active Learning for Hierarchical Text Classification

Xiao Li[1,2,⋆], Charles X. Ling[1], and Huaimin Wang[2]

[1] Department of Computer Science, The University of Western Ontario
[2] National Laboratory for Parallel & Distributed Processing,
National University of Defense Technology
{xli485,cling}@csd.uwo.ca, whm_w@163.com

Abstract. Hierarchical text classification is an important task in many real-world applications. To build an accurate hierarchical classification system with many categories, usually a very large number of documents must be labeled and provided. This can be very costly. Active learning has been shown to effectively reduce the labeling effort in traditional (flat) text classification, but few works have been done in hierarchical text classification due to several challenges. A major challenge is to reduce the so-called out-of-domain queries. Previous state-of-the-art approaches tackle this challenge by simultaneously forming the unlabeled pools on all the categories regardless of the inherited hierarchical dependence of classifiers. In this paper, we propose a novel top-down hierarchical active learning framework, and effective strategies to tackle this and other challenges. With extensive experiments on eight real-world hierarchical text datasets, we demonstrate that our strategies are highly effective, and they outperform the state-of-the-art hierarchical active learning methods by reducing 20% to 40% queries.

Keywords: Active Learning, Hierarchical Text Classification.

1 Introduction

Given documents organized in a meaningful hierarchy (such as a topic hierarchy), it is much easy for users to browse and search the desired documents. Thus, hierarchical text classification is an important task in many real-world applications, which include, for example, news article classification [8], webpage topic classification [3,2,10] and patent classification [5]. In hierarchical text classification, a document is assigned with multiple suitable categories from a predefined hierarchical category space. Different from traditional flat text classification, the assigned categories for each document in the hierarchy have inherited hierarchical relations. For example, in the hierarchy of the Open Directory Project (ODP), one path of the hierarchy includes *Computers* (Comp.) → *Artificial Intelligence* (A.I.) → *Machine Learning* (M.L.). Any webpage belonging to *M.L.* also belongs to *A.I.* and *Comp.*.

[⋆] Partial work was done when Xiao Li was an intern at National Laboratory for Parallel & Distributed Processing, Changsha, China.

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 233–244, 2013.

In this paper, we study machine learning approaches for building hierarchical classification system. According to [13], the most effective and appropriate approach for building hierarchical classification system is to train a binary classifier on each category of the hierarchy. To train an accurate hierarchical classification system with many categories, usually a very large number of labeled documents must be provided for a large number of classifiers. However, labeling a large amount of documents in a large hierarchy is very time-consuming and costly. This severely hinders the training of accurate hierarchical classification systems.

Active learning has been studied and successfully applied to reduce the labeling cost in binary text classification [15,11,17]. In active learning, in particular the pool-based active learning, the learner intelligently selects the most informative unlabeled example from the unlabeled pool to query an oracle (e.g., human expert) for the label. This can lead a good classification model with a much smaller number of labeled examples, compared to traditional passive learning. Several works have extended binary active learning to multi-class and multi-label text classification [1,4,19]. Basically, they use the one-VS-rest approach to decompose the learning problem to several binary active learning tasks.

However, active learning has not been widely studied for hierarchical text classification. The key question is how to effectively select the most useful unlabeled examples for a *large* number of *hierarchically* organized classifiers. Many technical challenges exist. For example, how should the unlabeled pool be formed for each category in the hierarchy? If not formed properly, the classifier may select many so-called *out-of-domain* examples from the pool. For example, a classifier on *A.I.* is trained under the category of *Comp.*. These examples are called *in-domain* examples for *A.I.*. Examples not belonging to *Comp.* are the so-called *out-of-domain* examples for *A.I.*. If an unlabeled example selected by the classifier for *A.I.* is an *out-of-domain* example, such as a document belonging to *Society*, the oracle will always answer “no”, and such a query will be virtually useless, and thus wasted in training the classifier for *A.I.*. Thus, avoiding the *out-of-domain* examples for hierarchical classifiers is very important.

As far as we know, only one work [9] has been published previously on hierarchical active learning. To solve the *out-of-domain* problem, the authors use the prediction of higher-level classifiers to refine the unlabeled pools for lower-level classifiers. In their approach, the quality of the lower-level unlabeled pools depends critically on the classification performance of the higher-level classifiers. However, the authors seemed not to pay enough attention to this important fact, and their methods allow all classifiers to simultaneously select examples to query oracles (see Section 2 for a review). This still leads to a large number of *out-of-domain* queries, as we will show in Section 4.4.

As the hierarchical classifiers are organized based on the top-down tree structure, we believe that a natural and better way to form the unlabeled pools is also in the top-down fashion. In this paper, we propose a novel top-down active learning framework, to effectively form the unlabeled pools, and select the most informative, *in-domain* examples for the hierarchical classifiers. Under our top-down active learning framework, we discuss effective strategies to tackle various challenges encountered. With extensive experiments on eight real-world hierarchical text datasets, including the RCV1-V2 and ODP datasets, we demonstrate that our method is highly effective, and it outperforms the state-of-the-art hierarchical active learning methods including [9] by reducing 20% to 40% queries.

2 Previous Works

To our best knowledge, only one work [9] has been published previously in active learning for hierarchical text classification. We call it the *parallel* active learning framework. In their approach, at each iteration of active learning (see Figure 1), the classifiers for all categories *independently* and *simultaneously* query the oracles for the corresponding labels. To avoid selecting the *out-of-domain* examples, they use the prediction of higher-level classifiers to refine the unlabeled pools for lower-level classifiers. Specifically, an unlabeled example will be added into the lower-level unlabeled pools only if its predictions from all the ancestor classifiers are positive.

A drawback of their approach is that they do not consider the hierarchical dependence of classification performance of the classifiers in their framework but allow all classifiers to *simultaneously* form the pools and select examples to query oracles. Considering a typical running iteration of their approach (see Figure 1). If the quality of the unlabeled pool $\mathcal{U}_{comp}$ (formed by the classifier for *Comp.*) is not good, possibly many *out-of-domain* examples (e.g., examples from *Society*) may still be selected by the classifiers for *A.I.*. This will lead to a large number of *out-of-domain* (wasted) queries, as we will show in Section 4.4.

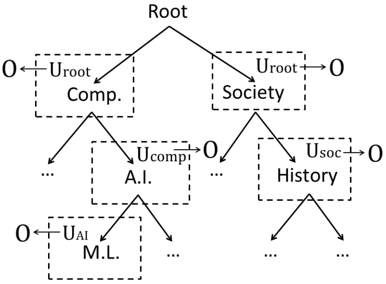
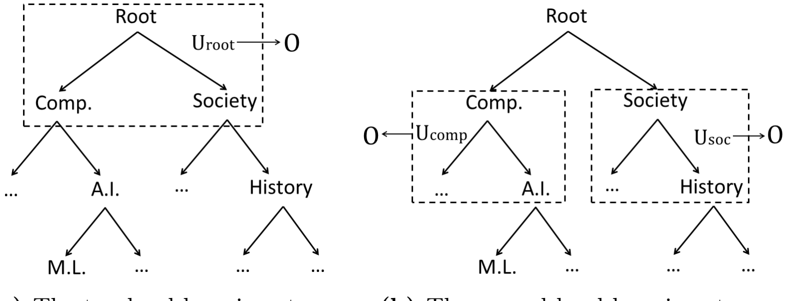


Fig. 1. A typical iteration of the parallel active learning framework. Multiple active learning processes (represented by dashed windows) are simultaneously conducted. U denotes the unlabeled pool and O denote the oracle. The horizontal arrows mean querying the oracle while the down arrows mean building the unlabeled pools.

How can we effectively solve the *out-of-domain* problem and the other challenges to improve active learning in hierarchical text classification? As the hierarchical classifiers are organized based on the top-down tree structure, we believe that a natural and better way to do active learning in hierarchical text classification is also in the top-down fashion. In the next section, we propose a new top-down active learning framework for hierarchical text classification to effectively tackle these challenges.

3 Top-Down Hierarchical Active Learning Framework

In this section, we propose our top-down hierarchical active learning framework. Different to the parallel framework which simultaneously forms the unlabeled pools for all categories, our top-down approach forms the unlabeled pools in the top-down fashion. We use Figure 2 to describe our basic idea.



(a) The top level learning stage. **(b)** The second level learning stage.

Fig. 2. Examples of two typical active learning stages in the top-down active learning framework. Only partial nodes in the hierarchy are allowed to do active learning. The notations in this figure follow Figure 1.

In Figure 2a, we start active learning at the top level of hierarchy. The top-level classifiers for *Comp.* and *Society* select examples from the global unlabeled pool $\mathcal{U}_{root}$ to query the oracle for the labels of top-level categories. The answered examples from the oracle will be used to form the unlabeled pools $\mathcal{U}_{comp}$ and $\mathcal{U}_{soc}$. After the top-level classifiers are well trained (estimated by our stopping strategy. see in Section 3.2), we start active learning in the second level. In Figure 2b, the second-level classifiers for A.I. (or History) and its sibling categories select examples from the unlabeled pool $\mathcal{U}_{comp}$ (or $\mathcal{U}_{soc}$) to query the oracle. As the examples in $\mathcal{U}_{comp}$ (or $\mathcal{U}_{soc}$) have true labels of *Comp.* (or *Society*) which are answered by the oracle, we can ensure that the second-level classifiers will not select *any out-of-domain* examples.

Comparing Figure 1 and Figure 2, we can see that the main difference between the parallel framework and our top-down framework is which nodes are chosen to do active learning (the dashed windows in both figures) at each iteration. The parallel framework chooses all the nodes while our top-down framework only chooses a subset of appropriate nodes in the top-down fashion. We call the set of those nodes as *working set*, denoted by $\mathcal{W}$. We present the pseudo code of our top-down active learning framework.

```
Input: Query budget $B$
Output: Classifiers for all nodes
repeat
    Add the root nodes $n_0$ into $\mathcal{W}$;
    repeat
        Select examples from $\mathcal{U}_n$ to query oracles and update children classifiers
        for each node $n$ in $\mathcal{W}$ until its stopping criteria is satisfied;
        Form the unlabeled pools for the children nodes of the finished nodes;
        Replace the finished nodes in $\mathcal{W}$ with their children nodes;
    until $\mathcal{W}$ is empty;
until $B = 0$;
```

For our top-down active learning framework, two critical challenges need to be resolved for effective active learning. The first challenge is that the unlabeled pools may be too small. We use the examples answered by the oracle to form the unlabeled pools, and they can be too small for lower-level classifiers to learn effectively. The problem may become worse when active learning is applied to even lower-level categories. The second challenge is how do we stop learning as it is critical for the effective scheduling of active learning in different levels. We will tackle the two challenges in the following subsections.

3.1 Dual-Pool Strategy

For the second and lower-level nodes, we need to form the unlabeled pools that are large enough but have few *out-of-domain* examples. In this section, we propose a novel dual-pool strategy to enlarge the unlabeled pools. Two different unlabeled pools will be built: the *answered pool* and the *predicted pool.*

Answered Pool. Our top-down active learning framework schedules the nodes to query the oracle from the top level to the bottom level. For a node (category) in the working set, we ask oracles for the labels of its children categories. For a child category c, among the answered examples from the oracle, the *positive* examples of c will be used to form the unlabeled pool for the children categories of c. The *negative* examples will not be used as they are already *out-of-domain.* By doing so, we can ensure that no *out-of-domain* examples will be selected into the unlabeled pools of children categories. We call such a pool the *answered pool* and use $\mathcal{U}^a$ to denote it.

Predicted Pool. The quality of the answered pool $\mathcal{U}^a$ is perfect. However, as the size of $\mathcal{U}^a$ depends on the positive class ratio of the ancestor nodes, it could be very slow to accumulate enough examples. Thus, we can also use the prediction of the higher-level classifiers to enlarge the unlabeled pools. Although this method is also used in the parallel framework (see Section 2), it should be noted that when we build the lower-level unlabeled pools, the higher-level classifiers are already assumed to be well-trained. The prediction of higher-level classifiers would be accurate. Thus, the risk of introducing *out-of-domain* examples would be much smaller than the parallel framework. We call the pool built by this method as *predicted pool*, denoted by $\mathcal{U}^p$.

Refiltering Dual Pools. We have two unlabeled pools for each node n_i in our top-down framework, i.e., the answered pool $\mathcal{U}_i^a$ and the predicted pool $\mathcal{U}_i^p$. When we select a batch of examples to query the oracle, a natural question is how do we allocate the batch of queries to each pool? On one hand, the quality of $\mathcal{U}_i^a$ is perfect but the uncertain (useful) examples maybe be too few due to the small pool size; on the other hand, more useful examples may exist in the larger predicted pool $\mathcal{U}_i^p$ but we may take the risk of selecting the *out-of-domain* examples. To balance the tradeoff, we propose a *refiltering strategy* for allocating the queries to both $\mathcal{U}_i^a$ and $\mathcal{U}_i^p$.

Our basic idea is to filter out the certain examples from the pools before we allocate the batch of queries. Specifically, given the batch size M, we firstly filter out the certain examples from both $\mathcal{U}_i^a$ and $\mathcal{U}_i^p$ to generate two small candidate pools $\mathcal{C}_i^a$ and $\mathcal{C}_i^p$. The filtering threshold will be empirically tuned in our experiments (See Section 4.4). As the examples in $\mathcal{C}_i^a$ are all perfect (answered by oracle) and uncertain (worthy to learn), we put more queries into the perfect candidate pool $\mathcal{C}_i^a$ by allocating $\min\{|\mathcal{C}_i^a|, M\}$ queries. The rest of queries will be allocated to $\mathcal{C}_i^p$.

3.2 Stopping Strategy

An important factor of our top-down hierarchical active learning framework is knowing when to stop learning for the nodes in the working set. In other words, how to estimate if the classifiers are well-trained or not? A heuristic approach is to estimate the classification performance by cross-validation. However, from our pilot experiments, such method is quite unstable due to the small size of the labeled examples in active learning.

In this paper, we adopt a simple yet effective approach to stop learning. Simply speaking, if no uncertain examples can be further selected from the candidate pools, we stop learning. This is reasonable as querying very certain examples can not improve the classification performance [20]. In our top-down framework, this strategy can be implemented by checking the size of the two candidate pools $\mathcal{C}^a$ and $\mathcal{C}^p$. If both pools are empty, that means all the examples in the unlabeled pools are very certain, we stop learning.[1]

To summarize, in this section, we propose our top-down hierarchical active learning framework with several strategies to tackle the *out-of-domain* problem and the other challenges encountered. In the next section, we will conduct extensive experiments to verify the effectiveness of our framework.

4 Experiments

In this section, we conduct extensive empirical studies to evaluate our top-down hierarchical active learning framework compared to the state-of-the-art hierarchical active learning approaches.

4.1 Datasets

We use eight real-world hierarchical text datasets in our experiments. The first three datasets (20 Newsgroup, OHSUMED and RCV1-V2) are common benchmark datasets for evaluation of text classification methods. The other five datasets are webpages collected from Open Directory Project (ODP).

The first dataset is *20 Newsgroups*[2], a collection of news articles partitioned evenly across 20 different newsgroups. We manually group these categories into a

[1] For the root node which selects examples from the very large global unlabeled pool, this stopping strategy could be very slow. Thus, we empirically set 25% remained budget as the query limit for the root node.

[2] `http://people.csail.mit.edu/jrennie/20Newsgroups/`

meaningful three-level hierarchy. The second dataset is *OHSUMED*[3], a clinically-oriented MEDLINE dataset. We use the subcategory *heart diseases* which is also used by [7,12]. The third dataset is *RCV1-V2* [8], a news archive from Reuters. We use the 23,149 documents from the topic classification task in our experiments.[4] The other five datasets are webpages collected from ODP. ODP is a web directory with a complex topic hierarchy. In our experiments, we focus on a subset of the webpages extracted from the Science subtree.[5] The original Science subtree has more than 50 subcategories. We choose five subcategories closely related to the academic disciplines.[6] They are *Astronomy*, *Biology*, *Chemistry*, *Earth Sciences* and *Math*.

For each dataset, we use bag-of-words model to represent documents. Each document is represented by a vector of term frequency. We use Porter Stemming to stem each word and remove the rare words occurring less than three times. Small categories which have less than ten documents are also removed. After the preprocessing, we give the detailed statistics of the datasets in Table 1.

Table 1. The statistics of the datasets. Cardinality is the average categories per example (multi-label).

Dataset	Examples	Features	Nodes	Cardinality	Height
20 Newsgroup	18,774	61,188	27	2.20	3
OHSUMED	16,074	12,427	86	1.92	4
RCV1-V2	23,149	47,152	96	3.18	4
Astronomy	3,308	54,632	34	1.91	4
Biology	17,450	148,644	108	3.03	4
Chemistry	4,228	56,767	34	1.44	4
Earth Sciences	5,313	71,756	58	2.16	4
Math	11,173	108,559	107	1.93	4

4.2 Performance Measure

In this paper, we use the hierarchical F-measure [16,13,9], a popular performance measure in hierarchical text classification, to evaluate the performance of hierarchical classification methods. It is defined as,

$$hF = \frac{2 \times hP \times hR}{hP + hR} \quad where \quad hP = \frac{\sum_i |\hat{P}_i \bigcap \hat{T}_i|}{\sum_i |\hat{P}_i|}, \quad hR = \frac{\sum_i |\hat{P}_i \bigcap \hat{T}_i|}{\sum_i |\hat{T}_i|} \tag{1}$$

where hP is the hierarchical precision and hR is the hierarchical recall. $\hat{P}_i$ is the hierarchical categories predicted for test example x_i while $\hat{T}_i$ is the true categories of x_i.

[3] http://ir.ohsu.edu/ohsumed/

[4] http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/

[5] It can be freely downloaded at http://olc.ijs.si/dmozReadme.html.

[6] Most of the other subcategories are A-Z index lists and non-academic topics (e.g., Publications and Conferences).

4.3 Experiment Configuration

We adopt the hierarchical SVMs [10,14,18] as the base learner. On each category, a linear SVM classifier is trained to distinguish its sibling categories under the same parent category. We use LIBLINEAR [6] as the implementation of linear SVM. Following the configuration of [9], we set up the penalty C as 1,000 and the cost coefficient w as the ratio of negative examples in the training set. Other parameters of LIBLINEAR are set to the default values.

We compare our top-down framework with the parallel framework [9] and the baseline random approach. We use the *average uncertainty* [4] as the informativeness measure. It measures the example based on the average uncertainty among all children classifiers under the same parent. For the parallel framework, we choose the *uncertainty sampling* [15] which is also used in their experiments. For both approaches, the uncertainty of example is measured by the absolute SVM margin score. For the random approach, we simply select the examples randomly from the global unlabeled pool.

We set up the total query budget as 1000. The active learning experiment is decomposed into several iterations. In each iteration, each node in the working set selects M examples to query the oracle. Similar to [9], the batch size M is set as the logarithm of the unlabeled pool size on each category. We use the simulated oracle in our experiments. When receiving an query, the oracle replies the true labels for all its subcategories. It should be noted that in [9], each query only returns one label. To make a fair comparison, we also return the labels of all the subcategories for the parallel framework and the random approach.

To avoid the impact of randomness, we use 10-fold cross validation to evaluate the performance of active learning approaches. Specifically, when conducting active learning experiments on each dataset, we randomly split the dataset into 10 subsets with equal size. Of the 10 subsets, one set is retained as testing data. For the remain nine sets, we randomly sample 0.1% data as the labeled set. The remaining examples will be used as the unlabeled pool. The active learning experiments are then repeated 10 times. The final results are averaged over the 10 runs and accompanied by the error margins with 95% confidence intervals.

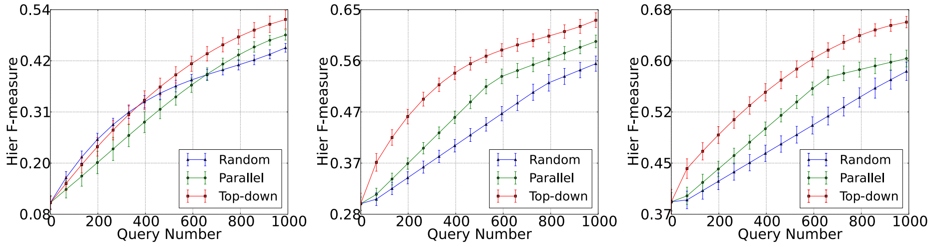
4.4 Experimental Results on Benchmark Datasets

Before the experiments, we setup the parameters for our top-down framework. We need to decide a proper uncertainty threshold to filtering out the certain examples (see dual-pool strategy in Section 3.1). As the SVM margin score based uncertainty is not comparable, we normalize it by the function $g(f) = exp(-\frac{f^2}{0.01})$ $(0 < g \leq 1)$ where f is the SVM margin score. We compare the uncertainty thresholds in different values from 0.1, 0.2, to 0.9 on the RCV1-V2 dataset. We find that generally the larger the threshold is, the better the performance is.[7] Thus, we use 0.9 as the uncertainty threshold in our experiments.

Firstly, we discuss the experimental results on the three benchmark datasets (20 Newsgroup, OHSUMED and RCV1-V2). Figure 3 shows the performance

[7] Due to page limit, the figure is omitted.

curves of hierarchical F-measure averaging over 10 runs. We can see that our top-down approach (framework) outperforms the parallel approach and the random approach significantly on all datasets. Specifically, on the OHSUMED and RCV1-V2 datasets, the performance curves of our top-down approach dominate the parallel approach and the random approach throughout the whole iterations. On the 20 Newsgroup dataset, surprisingly, during the earlier stage of active learning (before 400 queries), we observe the overlap of performance curves of our top-down approach and the random approach. The parallel approach performs even worse. This could be due to the poor initial classification performance (smaller than 0.1). However, after around 500 queries, our approach starts to outperform the random approach and the parallel approach and keeps the dominant margin till the end.



(a) 20 Newsgroup **(b)** OHSUMED **(c)** RCV1-V2

Fig. 3. Hierarchical F-measure on the 20 Newsgroup, OHSUMED and RCV1-V2 datasets

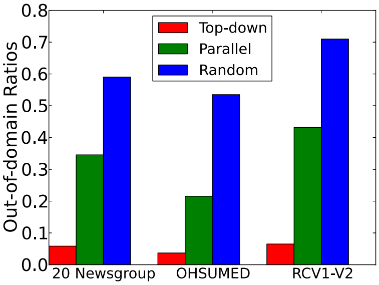


Fig. 4. The *out-of-domain* ratios of the queries on the 20 Newsgroup, OHSUMED and RCV1-V2 datasets

We examine the ratio of *out-of-domain* queries. Figure 4 shows the average *out-of-domain* ratios on the three datasets. We can see that our top-down approach has a huge reduction of the *out-of-domain* queries. Among the three datasets, our top-down approach issues less than 10% *out-of-domain* queries. By analyzing the experiment logs of our top-down approach, we discovery that for the second and the lower-level, on average about 40% queries are allocated to the answered pools (see Section 3.1). As the labels in the answered pools are given by the oracle, the quality of the selected examples is perfect. Thus, no *out-of-domain* examples will be selected. The observed few *out-of-domain* examples only occur in the predicted pools. The low ratio also indicates that the predicted pools built by our dual-pool strategy are much more accurate than the parallel framework. This explains why our top-down active learning approach is more effective than the parallel approach.

We also study how many queries can be saved by our top-down approach. For the three approaches, we record their best performance and the queries needed in Table 2. We find that to achieve the best performance of the parallel approach, our top-down approach needs much fewer queries. About 20% to 37% queries can be saved. For example, on the RCV1-V2 dataset, the parallel approach needs 1,000 queries to achieve 0.606 hierarchical F-measure, while our top-down approach only requires 630 queries. Thus, (1000 – 630)/1000 = 37% queries are saved. Compared to the random approach, the query reduction is even more significant (about 30% to 56%). It clearly indicates that our top-down approach is more effective in reducing the queries than the parallel approach and the baseline random approach.

Table 2. The best hierarchical F-measure with needed queries on the 20 Newsgroup, OHSUMED and RCV1-V2 datasets. The value in the bracket is the relative query reduction.

	Method	Hier F1	Random	Parallel	Top-down
20 Newsgroup	Random	0.455	1000	850 (15%)	700 (30%)
	Parallel	0.483		1000	800 (20%)
	Top-down	0.518			1000
OHSUMED	Random	0.552	1000	720 (28%)	440 (56%)
	Parallel	0.591		1000	680 (33%)
	Top-down	0.630			1000
RCV1-V2	Random	0.587	1000	660 (34%)	490 (51%)
	Parallel	0.606		1000	630 (37%)
	Top-down	0.661			1000

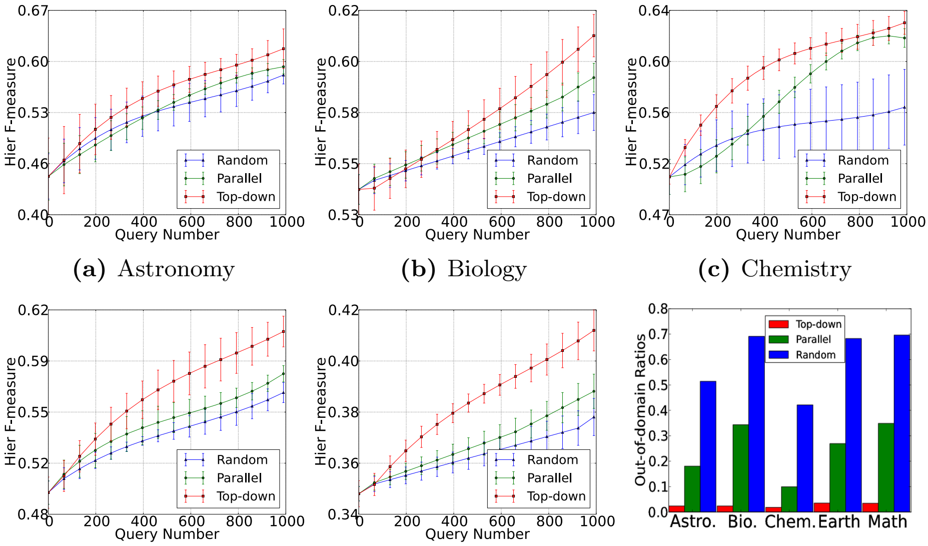
4.5 Experimental Results on ODP Datasets

In the following experiments, we compare the performance of the three approaches on five ODP datasets. From Figure 5, we find that on all datasets, our top-down approach performs consistently better than both the parallel approach and the random approach. The largest improvement occurs on the Math dataset where our top-down approach saves 40% queries to achieve the best performance of the parallel approach.[8] By analyzing the *out-of-domain* ratio in Figure 5f, we find that our top-down approach reduces the ratio of *out-of-domain* queries by 32% on the Math dataset compared to the parallel approach. The similar pattern can also be observed on the Biology and Earth Sciences datasets where about 32% and 23% *out-of-domain* queries can be saved. For the Astronomy and Chemistry datasets, we can see that the parallel approach makes less than 20% *out-of-domain* ratios. This can explain why our top-down approach performs only slightly better than the parallel approach on the Astronomy and Chemistry datasets. However, on some of the ODP datasets, the performance curves of the parallel approach have an obvious large overlap with

[8] The top-down approach requires 600 queries to achieve 0.4 hierarchical F-measure of the parallel approach which requires 1,000 queries. The saving is (1000 − 600)/1000=40%.

the random approach, while our top-down approach always outperforms the two approaches at the end of active learning.

To summarize, from our extensive experiments on eight real-world hierarchical text datasets, we empirically demonstrate that our top-down active learning framework is more effective than the state-of-the-art active learning approaches for hierarchical text classification.



(a) Astronomy **(b)** Biology **(c)** Chemistry

(d) Earth Sciences **(e)** Math **(f)** Out-of-domain Ratios

Fig. 5. Hierarchical F-measure and the ratios of *out-of-domain* queries on the ODP datasets

5 Conclusion and Future Work

In this paper, we study the problem of active learning for hierarchical text classification. A major challenge for effective hierarchical active learning is to form the unlabeled pools to avoid the so-called out-of-domain queries. Previous state-of-the-art approaches tackle this challenge by simultaneously forming the unlabeled pools on all the categories of the hierarchy regardless of the inherited hierarchical dependence of classifiers. In this paper, we propose a novel top-down hierarchical active learning framework which utilizes the top-down tree structure to form the unlabeled pools. Under our framework, we propose several effective strategies to tackle the out-of-domain problem and the other challenges encountered. With extensive experiments on eight real-world hierarchical text datasets, we demonstrate that our top-down framework is highly effective, and it outperforms the state-of-the-art hierarchical active learning methods by reducing 20% to 40% queries.

In our future work, we plan to use crowdsourcing for hierarchical active learning in real-world applications, such as constructing the hierarchical classifiers for search engines with hierarchy.

Acknowledgments. We thank the anonymous reviewers for their valuable suggestions. This work was partially supported by the National High Technology Research and Development Program of China (Grant No. 2012AA011201).

References

1. Brinker, K.: On active learning in multi-label classification. In: From Data and Information Analysis to Knowledge Engineering, pp. 206–213 (2006)
2. Ceci, M., Malerba, D.: Classifying web documents in a hierarchy of categories: a comprehensive study. J. Intell. Inf. Syst. 28, 37–78 (2007)
3. Dumais, S., Chen, H.: Hierarchical classification of web content. In: SIGIR 2000, pp. 256–263. ACM (2000)
4. Esuli, A., Sebastiani, F.: Active learning strategies for multi-label text classification. In: Boughanem, M., Berrut, C., Mothe, J., Soule-Dupuy, C. (eds.) ECIR 2009. LNCS, vol. 5478, pp. 102–113. Springer, Heidelberg (2009)
5. Fall, C.J., Törcsvári, A., Benzineb, K., Karetka, G.: Automated categorization in the international patent classification. SIGIR Forum 37(1), 10–25 (2003)
6. Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X.R., Lin, C.J.: Liblinear: A library for large linear classification. J. Mach. Learn. Res. 9, 1871–1874 (2008)
7. Lam, W., Ho, C.Y.: Using a generalized instance set for automatic text categorization. In: SIGIR 1998, pp. 81–89 (1998)
8. 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)
9. Li, X., Kuang, D., Ling, C.X.: Active learning for hierarchical text classification. In: Tan, P.-N., Chawla, S., Ho, C.K., Bailey, J. (eds.) PAKDD 2012, Part I. LNCS, vol. 7301, pp. 14–25. Springer, Heidelberg (2012)
10. Liu, T.Y., Yang, Y., Wan, H., Zeng, H.J., Chen, Z., Ma, W.Y.: Support vector machines classification with a very large-scale taxonomy. SIGKDD Explor. Newsl. 7, 36–43 (2005)
11. Roy, N., McCallum, A.: Toward optimal active learning through sampling estimation of error reduction. In: ICML 2001, pp. 441–448 (2001)
12. Ruiz, M.E., Srinivasan, P.: Hierarchical neural networks for text categorization (poster abstract). In: SIGIR 1999, pp. 281–282 (1999)
13. Silla Jr., C.N., Freitas, A.A.: A survey of hierarchical classification across different application domains. Data Min. Knowl. Discov. 22, 31–72 (2011)
14. Sun, A., Lim, E.P.: Hierarchical text classification and evaluation. In: ICDM 2001, pp. 521–528 (2001)
15. Tong, S., Koller, D.: Support vector machine active learning with applications to text classification. J. Mach. Learn. Res. 2, 45–66 (2002)
16. Verspoor, K., Cohn, J., Mniszewski, S., Joslyn, C.: Categorization approach to automated ontological function annotation. In: Protein Science, pp. 1544–1549 (2006)
17. Xu, Z., Yu, G., Tresp, V., Xu, X., Wang, J.: Representative sampling for text classification using support vector machines. In: Sebastiani, F. (ed.) ECIR 2003. LNCS, vol. 2633, pp. 393–407. Springer, Heidelberg (2003)
18. Xue, G.R., Xing, D., Yang, Q., Yu, Y.: Deep classification in large-scale text hierarchies. In: SIGIR 2008, pp. 619–626 (2008)
19. Yang, B., Sun, J.T., Wang, T., Chen, Z.: Effective multi-label active learning for text classification. In: KDD 2009, pp. 917–926 (2009)
20. Zhu, J., Wang, H., Hovy, E., Ma, M.: Confidence-based stopping criteria for active learning for data annotation. ACM Trans. Speech Lang. Process 6(3), 3:1–3:24 (2010)

Forgetting Word Segmentation in Chinese Text Classification with $L1$-Regularized Logistic Regression[*]

Qiang Fu, Xinyu Dai[**], Shujian Huang, and Jiajun Chen

National Key Laboratory for Novel Software Technology,
Nanjing University, Nanjing 210023, China
{fuq,dxy,huangsj,chenjj}@nlp.nju.edu.cn

Abstract. Word segmentation is commonly a preprocessing step for Chinese text representation in building a text classification system. We have found that Chinese text representation based on segmented words may lose some valuable features for classification, no matter the segmented results are correct or not. To preserve these features, we propose to use character-based N-gram to represent the Chinese text in a larger scale feature space. Considering the sparsity problem of the N-gram data, we suggest the $L1$-regularized logistic regression ($L1$-LR) model to classify Chinese text for better generalization and interpretation. The experimental results demonstrate our proposed method can get better performance than those state-of-the-art methods. Further qualitative analysis also shows that character-based N-gram representation with $L1$-LR is reasonable and effective for text classification.

Keywords: Text classification , Text representation , Chinese Character-based N-gram , $L1$-regularized logistic regression.

1 Introduction

Text classification is a task which automatically assigns an appropriate category for a text according to its content. The task formally can be defined as follows. We have a set of training pairs as $\{(d_1, l_1), (d_2, l_2), ..., (d_n, l_n)\}$, where d_i indicates a text and l_i is the corresponding label drawn from a set of discrete values indexed by $\{1, 2, ..., k\}$. The training data is used to build a classification model h. Then for a given test text t whose class label is unknown, the training model is used to predict the class label l for this text. In recent years, with the rapid explosion of information, text in digital form comes from everywhere. In order to handle a large amount of text, automatically text classification has become not only an important research area, but also a urgent need in different kinds of applications.

[*] National Natural Science Foundation of China (No. 61003112, 61073119), the National Fundamental Research Program of China (2010CB327903), and the Jiangsu Province Natural Science Foundation(No.BK2011192).

[**] Corresponding author.

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 245–255, 2013.

As a result, automatically text classification has been widely studied in machine learning and information retrieval community. When Chinese becomes more and more popular these years, Chinese text classification will be an important way to deal with the large amount of Chinese text.

The common procedure of text classification can be divided into three parts, text representation, feature selection and classification. Generally speaking, before using a classifier, we should present each text as a vector in a high dimensional Euclidean space. Commonly, each word or character in text can be viewed as a feature. And all of these features compose of a feature space. Because the feature space is too large and redundant, for better generalization and performance, we usually apply some dimension reduction or feature selection methods to represent the data in a reasonable scale feature space. The last step is to build a classification model which will be used to predict the class label given new text.

Chinese text classification is a little different from English text classification. We usually need one more step before the common procedure—word segmentation, because Chinese sentences do not delimit words by spaces. Though word segmentation seems a necessary step, we think it may bring some potential problems for classification. Obviously, segmentation errors may bring bad influence to the classification. And even the segmented results are totally correct, some useful information may also be lost which incarnate in our experiments later. Another problem is that word segmentation cannot recognize new words very well. For example, the word-based classification performance is not so good in Social network message data where so many new words exists [7]. So, a natural idea is to use character-based N-gram instead of words.

In this paper, we propose a framework that adopts a character-based N-gram approach to Chinese text classification and uses regularized logistic regression classifier to solve the sparse problem of the N-gram data. There are two main contributions of our work. Firstly, we demonstrate words segmentation will lose some valuable information for classification, no matter the segmented results are correct or not. We also show that character-based N-gram text representation is more suitable for Chinese text classification. Secondly, for better generalization and interpretation, we introduce the $L1$ regularized logistic regression ($L1$-LR) model to classify Chinese text which can get better classification performance.

The rest of this paper is organized as follows. In the next section, we will discuss the background. In section 3, we discuss our works on Chinese text classification, including the detail of proposed classification method and analysis of doing so. In section 4, we present several experimental results and qualitative analysis of N-gram based regularized logistic regression in Chinese text classification followed by conclusions in section 5.

2 Background

In this section, we review some basic information of Chinese text classification, including text representation, feature selection and classifier. In Chinese text representation, one way is to use word segmentation. For Chinese word segmentation, there are two mostly used word segmentation tools: ICTCLAS (Institute of Computing Technology, Chinese Lexical Analysis System) [2] and Stanford Word Segmenter [3]. Additionally, Stanford Word Segmenter has two certain specifications, that is PeKing University criterion (pku) and Chinese Treebank criterion (ctb). Different criterions will produce different results. Another way to represent text is N-gram based approach, which William and John [5] firstly used in English text classification and received good effects. After preprocessing of text content, we should turn the text into feature vectors next. The commonly used approach is $tf \cdot idf$ [6], which is a weighted technology for text representation. And a high value means the feature (word, character etc.) is important for one text in corpus. Another way is to use 0-1 weight vector, indicating whether one feature appearance in a document or not.

After the generation of the feature vectors, feature selection methods will be used to reduce the feature space. In text classification, commonly used feature selection approaches are Gini Index, Information Gain, Mutual Information and χ^2-Statistic [4]. All of these four methods aim to find the relationship between features and class labels. According to some criterion, these methods give each feature a value that indicate the importance of this feature in classification. But these methods only pay attention to the relation between features and labels, and ignore the relationship between the features which is also important to classification.

After the feature selection is performed, we can use it to training a classifier. In text classification, commonly used classifier are SVM, Logistic regression, Decision Tree, Naive Bayes Classifiers and Neural Network Classifiers. Among these, SVM and Logistic regression are basically linear classifier and do well in text classification. Compared with SVM, the loss function of logistic regression is closer to a linear classifier. So, when we add a regularization term to the classifier, logistic regression is more able to reflect the difference among the difference regularized term than SVM. Furthermore, in large-scale sparse case, logistic regression can perform well compared with SVM [8]. Besides, logistic regression does well in many natural language processing applications [9]. In this paper, we will use logistic regression as our classifier.

3 N-Gram Based Regularized Logistic Regression

Automatic word segmentation error may influence the performance of Chinese text classification. Even if the segmented results are totally correct, some useful information will also be lost. Luo and Ohyama [1] have studied the impact of word segmentation on Chinese text classification. They compared text classification performance on automatic word segmentation, manual word segmentation and character-based N-gram approach, respectively. And they used support vector machine (SVM) with linear kernel, polynomial kernel and radial basis function as classifiers, respectively. The results show that the manual word segmentation gets the best performance, and character-based N-gram also gets the better performance than automatic word segmentation. But in real application, it's almost impossible to get manual word segmentation for each text. From their work, they don't explain why N-gram features work or not, and what kinds of N-gram feature are most valuable for classification. In this paper, we will totally forget word segmentation and focus on how to effectively extract valuable N-gram features for classification.

Moreover, Zhang and OLES [10] have compared performance of several linear model in text classification, including regularized logistic regression. The results show that regularized logistic regression has a performance that is comparable with other state-of-the-art methods, especially when we have a large scale feature space. And as is well-known, $L1$-regularized logistic regression is an outstanding method to generate a sparse model for classification. The sparse model can give us a chance to dig the huge potential of character-based N-gram for classification. Our work can be divided into two parts. First, we use character-based N-gram approach to represent Chinese text classification. Second, we use regularized logistic regression to train Chinese text classifier.

3.1 Text Representation

Using character-based N-gram to represent text are dramatically simpler, we could avoid the complicated process for the word segmentation. We just extract a sequence of N consecutive characters. In practice, we usually use $N \leq 3$. Table. 1 shows an example of N-gram features.

Table 1. An example of N-gram features

Original Text	南京市长江大桥
1-gram	南;京;市;长;江;大;桥
2-gram	南京;京市;市长;长江;江大;大桥
3-gram	南京市;京市长;市长江;长江大;江大桥

We use $tf \cdot idf$ as the feature weight. A $tf \cdot idf$ value consists of two parts. One is term frequency (tf), another is inverse document frequency (idf). Term frequency counts the relevant frequency of one feature in a given text. Higher $tf \cdot idf$ value of one feature means that it is more important than others. Inverse document frequency indicates whether one feature appears in most of the documents or not. If one feature appears in most of the documents, it is useless for classification and its idf value will be lower. A $tf \cdot idf$ value for one feature in a text is computed as follows:

$$\frac{n_{i,j}}{\sum_k n_{k,j}} \times \log \frac{|D|}{|\{j : t_i \in d_j\}|} \tag{1}$$

where $n_{i,j}$ is the frequency of feature t_i in text d_j, $|D|$ is the total number of documents.

By using N-gram, we can transfer text into feature vectors easily, without any complex word segmentation or other language specific techniques. So, the main measures we adopt are as follows: We use unigram and bigram or unigram, bigram and trigram to represent the text. And $tf \cdot idf$ is used as the weight.

3.2 Regularized Logistic Regression

In logistic regression [10], we model the conditional probability as follows:

$$P\left(y = 1 \left| w, x \right.\right) = \frac{1}{1 + e^{-w^T x}} \tag{2}$$

$$P\left(y = -1 \left| w, x \right.\right) = \frac{e^{-w^T x}}{1 + e^{-w^T x}} \tag{3}$$

Commonly, we use maximum likelihood estimation (MLE) to obtain an estimate of w, which minimizes the following equation:

$$w = \arg\min_{w} \sum_{i=1}^{N} \log\left(1 + e^{-y_i w^T x_i}\right) \tag{4}$$

Equation 4 may be ill-conditioned numerically. One way to solve this problem is to use regularization. In regularized approximation, by adding a regularizer to the loss function, it can limit model complexity and tune parameters for better generalization. General form of regularization is as follows:

$$\min_{f} \frac{1}{N} \sum_{i=1}^{N} L\left(y_i, f\left(x_i\right)\right) + \lambda R\left(f\right) \tag{5}$$

Where $L\left(y_i, f\left(x_i\right)\right)$ is the loss function, and $R\left(f\right)$ is the regularizer, $\lambda \geq 0$ is a coefficient, which aims to adjust the relationship between the two terms. The goal is to find a model f that is uncomplicated (if possible), and also makes the loss function L small.

In this paper, we use logistic regression with 1-norm regularizer for classification, denote as $L1$-regularized ($L1$-LR) logistic regression. And for comparing, we also use logistic regression with 2-norm regularizer for classification, denote as $L2$-regularized ($L2$-LR) logistic regression. The objective function of $L1$-LR and $L2$-LR are shown as follows, respectively:

$$\min_{w} \left\|w\right\|_1 + C \sum_{i=1}^{N} \log\left(1 + e^{-y_i w^T x_i}\right) \tag{6}$$

$$\min_{w} \frac{1}{2} \left\|w\right\|_2 + C \sum_{i=1}^{N} \log\left(1 + e^{-y_i w^T x_i}\right) \tag{7}$$

3.3 Analysis of Using *L*1-Regularized Logistic Regression

As mentioned before, we choose $L1$-regularized logistic regression as the classifier. The first reason is that, in character-based N-gram case, the feature space is too large to analysis. And the data sparsity is very serious. The second reason is that most of text classification tasks are linear separable [14]. A simple linear model may perform well compared to complicated models. When compared with SVM, the loss function of logistic regression is closer to a linear classifier. If we plus a regularized term to the classifier, logistic regression is more able to reflect the difference among the difference regularized term than SVM. Therefore, we choose to use regularizer logistic regression as classifier, especially, the $L1$-regularized logistic regression. Since $L1$-regularization is a sparse model, the feature vector produced by $L1$-regularization has fewer none-zero features. Due to the properties of 1-norm and 2-norm when searching in the hypothesis space, $L1$-regularization encourage less features which may be important in classification to be nonzero and the rest features are zero. On the other hand, $L2$-regularization is more like a kind of average, it encourage more features to be a small value. With this property, $L1$-regularized logistic regression will select some key features from N-gram based feature space. These selected features may seem a bit weird by human, but are definitely valuable for classification. Those selected features can help to interpret the significance of character-based N-gram approach.

Moreover, Andrew Y. Ng [11] proved that using $L1$-regularization, the sample complexity (i.e., the number of training examples required to learn well) grows only logarithmically in the number of irrelevant features. But any rotationally invariant algorithm (e.g., $L2$-regularized logistic regression) exist a worst case that the sample complexity grows at least linearly in the number of irrelevant features. According to this theorem when the irrelevant features are much more than the training text, $L1$-regularization will also achieve a good results. Moreover, text classification is the case that much more irrelevant features come from limited amount of text.

4 Experiments and Results

4.1 Setup

To compare with the previous Chinese text classification, we also implement some common approaches in this paper. As mentioned above, we use ICTCLAS and Stanford Word Segmenter (pku and ctb) as word segmenter, respectively. Top 80 percent word features are selected with four feature selection methods, Gini Index, Information Gain, Mutual Information and χ^2-Statistic. Then, we use SVM as a baseline which is a state-of-the-art classifier in text classification. We use libsvm [12] as tools to train a SVM classifier.

In N-gram based text classification, we use $(1 + 2)$-gram (use unigram and bigram for text representation) and $(1 + 2 + 3)$-gram (use unigram, bigram and trigram for text representation) in experiment. And we use liblinear [13] as tools to train the regularized logistic regression. For solving $L1$-LR, the liblinear use newGLMNET algorithm, see [15] for computational complexity and other details. Additionally, we also use regularized logistic regression on segmented text. At last, we use 10-fold cross validation in our experiments.

4.2 Experiment on Chinese Corpus

Fudan Chinese text classification corpus was used in our experiment (it is released on `http://www.nlpir.org/download/tc-corpus-answer.rar`). We select 9 classes from this corpus. The total number of documents is 9330. The categories include art, history, space, computer, environment, agriculture, economy, politics and sports. For a better comparison with these three different word segmenter, we list the results of N-gram three times. Table. 2 shows the result of text classification on Fudan corpus. Where the $X2$ means that use some word segmenter and χ^2-Statistic as feature selection. The *Gini* means that use some word segmenter and Gini Index as feature selection. The IG means that use some word segmenter and Information Gain as feature selection. The MI means that use some word segmenter and Mutual Information as feature selection. The $L1$-LR means the $L1$-regularized logistic regression. And $L2$-LR means the $L2$-regularized logistic regression.

Table 2. Results of text classification on Fudan corpus

	N-gram		Stanford Word Segmenter(pku)			
	1+2 gram	1+2+3 gram	X2	Gini	IG	MI
$L1$-LR	95.44	**95.57**	94.84	92.27	89.49	86.75
$L2$-LR	95.34	95.31	95.31	92.90	92.52	88.55
SVM	95.35	95.38	95.08	89.48	87.72	81.20

	N-gram		Stanford Word Segmenter(ctb)			
	1+2 gram	1+2+3 gram	X2	Gini	IG	MI
$L1$-LR	95.44	**95.57**	94.88	92.09	88.93	86.79
$L2$-LR	95.34	95.31	95.21	93.00	92.65	88.47
SVM	95.35	95.38	95.16	86.93	86.40	81.16

	N-gram		ICTCLAS			
	1+2 gram	1+2+3 gram	X2	Gini	IG	MI
$L1$-LR	95.44	**95.57**	94.81	92.40	89.43	87.37
$L2$-LR	95.34	95.31	95.29	92.14	91.58	88.10
SVM	95.35	95.38	95.24	88.70	87.20	80.90

From Table. 2, it is obvious that character-based N-gram with $L1$-regularized logistic regression does best in Chinese text classification regardless of which word segmenter is used. Our regularized classifier really does better than traditional features selections methods. And in large scale data classification, regularized logistic regression is more effective than SVM. Additionally, the result shows that dealing with the large and sparse text data, $L1$-regularized logistic regression is better than $L2$-regularized logistic regression.

4.3 Experiment on English Corpus

Moreover, in order to further validate the generality of N-gram based regularized logistic regression approach. We experiment this method on English text classification. 20-News English text classification corpus was used in our experiment (it is released on `http://qwone.com/~jason/20Newsgroups/`). We use 10 classes selected from 20-News corpus. The total number of documents is 10000 (1000 documents for each class). Then, we repeat the previous steps. Note that for English text, we use word-based N-gram instead of character-based N-gram. The results are shown in Table. 3.

Table 3. Results of text classification on 20-News corpus

	1+2 gram	1+2+3 gram	X2	Gini	IG	MI
$L1$-LR	93.22	**93.73**	92.01	90.94	86.02	83.00
$L2$-LR	91.87	91.87	91.69	91.93	87.26	84.72
SVM	92.05	92.21	91.63	89.99	84.26	81.01

From Table. 3, it is obvious that word-based N-gram with regularized logistic regression does best in English text classification. The result shows that N-gram based regularized logistic regression also performs better than the state-of-the-art approach, in English text classification.

4.4 Accuracy Changes over Nonzero Features

Furthermore, we present the variety curve of text classification accuracy over the number of the nonzero features. We use Fudan corpus as training data and use spline interpolation to reflect the variation trend. The result is shown in Figure. 1. Note that we omit the curve of $(1+2+3)$-gram, since they are almost the same.

In Figure. 1, we can find that text classification accuracy grow rapidly with rising of nonzero features and reach the maximum at around 2000 nonzero features. Then, as the nonzero features continued to increase, the accuracy comes down slightly. This also shows that the sparsity of text data from another side. And a small number of features selected by sparse model are enough to achieve good classification accuracy.

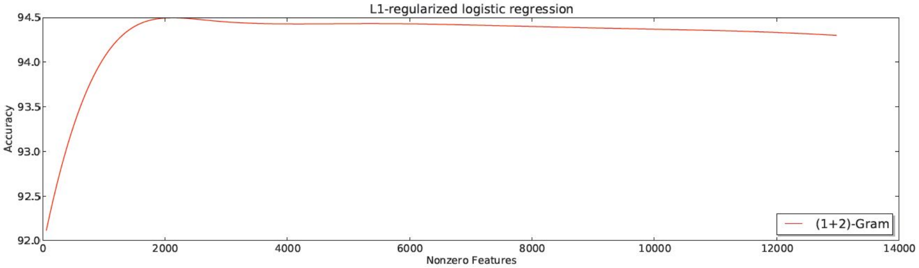


Fig. 1. Accuracy changes over nonzero features

4.5 Qualitative Analysis

At last, in order to qualitative analysis that N-gram based $L1$-regularized logistic regression can select some key features which we cannot obtain through word segmentation. We experiment on a binary-class Chinese text classification problem. We select two class from Fudan Chinese corpus, which composed by 1357 documents labeled as 'computer' and 1601 documents labeled as 'economy'. In order to reflect the importance of each feature more directly, we use 0-1 vector instead of $tf \cdot idf$. Thus, the importance of one feature is totally depend on the weight vector w. Then, we use $(1 + 2 + 3)$-gram and $L1$-regularized logistic regression. After the classifier has been trained, we select some typical features from top ranked features. These features are shown in Table. 4.

Table 4. An example of top ranked training features. #occur in Comp is the number of occurrences of the given ngram in documents labeled as 'Computer'. #doc in Comp is the number of documents labeled as 'Computer' which contain the given ngram. #occur in Econ is the number of occurrences of the given ngram in documents labeled as 'Economy'. #doc in Econ is the number of documents labeled as 'Economy' which contain the given ngram.

	Total	#occur in Comp	#doc in Comp	#occur in Econ	#doc in Econ
向对象	757	757	137	0	0
分类名	1489	0	0	1489	1488
o.	1866	1761	1296	105	44
化学报	697	697	511	0	0
经济发	7866	16	11	7850	1220
【原	5556	0	0	5556	1477
原刊地	1275	0	0	1275	1274
政	42714	207	80	42507	1453
识经	3870	12	6	3858	296
期V	752	752	752	0	0
”。	3710	122	68	3588	1028
主义市	2395	0	0	2395	489

From Table. 4 we can find that most of the top features are N-gram form which cannot be generated by word segmenter. These N-gram features can be divided into three categories. The first category of features presents two separate words. These two words will appear in two classes of text, respectively. But, in one of the two classes, they appear sequentially. Taking '向对象' as an example, this trigram is the suffix of '面向对象' (object-oriented). In segmented text, this feature is segmented as '面向' (oriented) and '对象' (object). The feature '面向' (oriented) appears in 264 computer documents and 230 economy documents, respectively. The feature '对象' (object) appears in 476 computer documents and 406 economy documents, respectively. The number of times they appear in these two classes are similar. As a result, these two features are not useful in classifying the two classes. But the feature '向对象' appears in 137 computer documents and 0 economy documents, respectively. Obviously, this trigram is more useful than ’面向’ (oriented) and ’对象’ (object) for classifying the two classes, significantly. Take ’经济发’ as another example which is the prefix of ’经济发展’ (economic development). In segmented text, this feature is segmented as ’经济’ (economic) and ’发展’ (development). The feature ’ 经济’ (economic) appears in 155 computer documents and 1548 economy documents, respectively. The feature ’发展’ (development) appears in 534 computer documents and 1500 economy documents, respectively. But the feature ’经济发’ appears in 11 computer documents and 1220 economy documents, respectively, which is a much stronger indicator for classification.

The second category of features consists of only one Chinese character. But this character is usually a prefix or suffix of a group of words. This group of words usually appears in only one of the two classes. Using just one Chinese character to represent a group of words is more effective. Take ’政’ as an example. In economy text, there are lots of words containing the character ’政’. (such as, ’政治’ (politics), ’政策’ (policy), ’政府’ (government), etc.) But, from Table. 4, the feature ’政’ appears in 80 computer documents and 1453 economy documents, respectively. The number of times they appear in these two classes are different very much. It is obviously that one character will suffice.

The third category of features presents the beginning or end of one sentence. Take ’” 。’ as an example. In segmented text, they are segmented as ’” ’ and ’。’. But, from Table. 4, the feature ’” 。’ appears in 68 computer documents and 1028 economy documents, respectively. It is obviously that ’” 。’ is useful in classification. As an explanation, we find that, in economy text, there are lots of quotes in the end of the sentence.

The above analysis shows that N-gram based $L1$-regularized logistic regression can get some key features which cannot be generated by word segmenter. But these information are useful in text classification, indeed.

5 Conclusions

In this paper, we demonstrate the drawbacks of using word segmentation in Chinese text classification. We propose a framework that use character-based N-gram with regularized logistic regression on Chinese text classification. And, we made experiments on Fudan Chinese text classification corpus. Results of different word segmenters and different feature selection methods are compared. The proposed method gets the best performance. Though quantitative and qualitative analysis, we further discussed for experiments why the N-gram features work better than word features, and what kinds of N-gram features are valuable for classification.

But there are still some limitations of our method. For example, the regularizer we used doesn’t consider the relationship between features. We just use 1-norm or 2-norm as the regularizer. In the future work, we will consider adding structure information in the regularizer for better performance hopefully.

References

1. Luo, X., Ohyama, W., Wakabayashi, T., Kimura, F.: Impact of Word Segmentation Errors on Automatic Chinese Text Classification. In: 10th IAPR International Workshop on Document Analysis Systems, pp. 271–275 (2012)
2. Zhang, H., Yu, H., Xiong, D., Liu, Q.: HHMM-based Chinese Lexical Analyzer ICTCLAS. In: Proceedings of the Second SIGHAN Workshop on Chinese Language Processing, Sapporo, Japan, pp. 184–187 (2003)
3. Tseng, H., Chang, P., Andrew, G., Jurafsky, D., Manning, C.: A Conditional Random Field Word Segmenter. In: Fourth SIGHAN Workshop on Chinese Language Processing, Jeju Island, Korea, pp. 168–171 (2005)
4. Aggarwal, C.C., Zhai, C.: A survey of text classification algorithms. In: Mining Text Data, pp. 163–213. Springer (2012)
5. Cavnar, W.B., Trenkle, J.M.: Ngram-based text categorization. In: Proceedings of SDAIR 1994, 3rd Annual Symposium on Document Analysis and Information Retrieval, Las Vegas, US, pp. 161–175 (1994)
6. Salton, G., Fox, E.A., Wu, H.: Extended Boolean information retrieval. Communications of the ACM 26(11), 1022–1036 (1983)
7. Phan, X.H., Nguyen, L.M., Horiguchi, S.: Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In: Proceedings of the 17th International Conference on World Wide Web, pp. 91–100. ACM Press, New York (2008)
8. Komarek, P., Moore, A.: Fast Robust Logistic Regression for Large Sparse Datasets with Binary Outputs. Artificial Intelligence and Statistics (2003)
9. Berger, A.L., Della Pietra, S.A., Della Pietra, V.J.: A Maximum Entropy Approach to Natural Language Processing. Computational Linguistics 22(1), 39–71 (1996)
10. Zhang, T., Oles, F.: Text categorization based on regularized linear classification methods. Information Retrieval, 5–31 (2001)
11. Andrew, Y., Ng: Feature selection, l1 vs. l2 regularization, and rotational invariance. In: Proceedings of the Twenty-First International Conference on Machine learning (ICML), pp. 78–85. ACM Press, New York (2004)
12. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. ACM Transactions on Intelligent Systems and Technology 2, 27:1–27:27 (2011), http://www.csie.ntu.edu.tw/~cjlin/libsvm
13. Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X.R., Lin, C.J.: LIBLINEAR: A library for large linear classification. Journal of Machine Learning Research 9, 1871–1874 (2008)
14. Joachims, T.: Text categorization with support vector machines: Learning with many relevant features. In: Proceedings of the 10th European Conference on Machine Learning, Heidelberg, Germany, pp. 137–142 (1998)
15. Yuan, G.X., Ho, C.H., Lin, C.J.: An improved glmnet for l1-regularized logistic regression. The Journal of Machine Learning Research, 1999–2030 (2012)

Crest: Cluster-based Representation Enrichment for Short Text Classification[*]

Zichao Dai[1], Aixin Sun[2], and Xu-Ying Liu[1]

[1] MOE Key Laboratory of Computer Network and Information Integration,
School of Computer Science and Engineering, Southeast University, Nanjing, China
daixiaodai.geek@gmail.com, liuxy@seu.edu.cn

[2] School of Computer Engineering, Nanyang Technological University, Singapore
axsun@ntu.edu.sg

Abstract. Text classification has gained research interests for decades. Many techniques have been developed and have demonstrated very good classification accuracies in various applications. Recently, the popularity of social platforms has changed the way we access (and contribute) information. Particularly, short messages, comments, and status updates, are now becoming a large portion of the online text data. The shortness, and more importantly, the sparsity, of the short text data call for a revisit of text classification techniques developed for well-written documents such as news articles. In this paper, we propose a cluster-based representation enrichment method, namely Crest, to deal with the shortness and sparsity of short text. More specifically, we propose to enrich a short text representation by incorporating a vector of topical relevances in addition to the commonly adopted *tf-idf* representation. The topics are derived from the knowledge embedded in the short text collection of interest by using hierarchical clustering algorithm with purity control. Our experiments show that the enriched representation significantly improves the accuracy of short text classification. The experiments were conducted on a benchmark dataset consisting of Web snippets using Support Vector Machines (SVM) as the classifier.

Keywords: Short text classification, Representation enrichment, Clustering.

1 Introduction

The prevalence of Internet-enabled devices (e.g., laptops, tablets, and mobile phones) and the increasing popularity of social platforms are changing the way we consume and produce information online. A large portion of the data accessible online is user-generated content in various forms, such as status updates, micro-blog posts, comments, and short product reviews. In other words, much

[*] This work was partially done while the first author was visiting School of Computer Engineering, Nanyang Technological University, supported by MINDEF-NTU-DIRP/2010/03, Singapore.

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 256–267, 2013.

user-generated textual content is in the form of *short text*. The unique characteristics (e.g., shortness, noisiness, and sparsity) distinguish short text from the well written documents such as news articles and most Web pages. These unique characteristics call for a revisit of the techniques developed for text analysis and understanding, including text classification.

Text classification refers to the task of automatically assigning a textual document one or more predefined categories. It has been heavily studied for decades and many techniques have been proposed and have demonstrated good classification accuracies in various application domains [13,16]. Nevertheless, most text classification techniques take advantage of the information redundancy naturally contained in the well-written documents (or long documents in contrast to short text). When facing with short text, the shortness, noisiness, and sparsity, adversely affect the classifiers from achieving good classification accuracies. To improve short text classification accuracy has since attracted significant attention from both the industries and academia.

To deal with the shortness and sparsity, most solutions proposed for short text classification aim to enrich short text representation by bringing in additional semantics. The additional semantics could be from the short text data collection itself (e.g., named entities, phrases) [7] or be derived from a much larger external knowledge base like Wikipedia and WordNet [4,7,10]. The former requires shallow Natural Language Processing (NLP) techniques while the later requires a much larger and "appropriate" dataset. Very recently, instead of enriching short text representation, another approach known as search-and-vote is proposed to improve short text classification [15]. The main idea is to mimic human judging processing by identifying a few topical representative keywords from each short text and use the identified topical keywords as queries to search for similar short texts from the labeled collection. Very much similar to k-nearest-neighbor classifier, the category label of the short text for classification is voted by using the search results. Note that, the aforementioned different approaches deal with the shortness and sparsity of short text from very different perspectives and are mostly orthogonal to each other. In other words, on the one hand, these different approaches could be combined to potentially achieve much better classification accuracies than any of the approaches alone; on the other hand, this calls for further research to improve each individual researches.

In this paper, we focus on improving short text classification accuracy by enriching the text representation, by not only using its raw words (e.g., bag-of-words) but also topical representations. Our approach naturally falls under the *representation enrichment* approach. However, our approach is different from the earlier works in representation enrichment because of two reasons. First, we do not use shallow NLP techniques to extract phrases or any specific patterns because most short texts are noisy preventing many existing NLP toolkits from achieving good accuracy. Second, we do not use external knowledge base like Wikipedia because some of the short text data collection might be from very specific or niche areas where it is hard to find an "appropriate" and large dataset. In other words, we consider that if we can discover internally useful knowledge solely from the training dataset when an "appropriate" large external dataset is not available. More specifically, we propose a generic method named Crest to first discover "high-quality" topic clusters from the training data by grouping similar (but not necessary from the same category) training examples together to form clusters. Each short text instance is then represented using the topical similarities between the short text and the topic clusters in addition to its words feature vector. The main advantages of Crest include the following:

- *Low-cost in knowledge acquisition.* As we mentioned above, Crest does not rely on any external knowledge source. It mines topic clusters solely from the training examples.
- *Reduction in data sparsity.* The topic clusters discovered from the training data define a new feature space that each short text instance can be mapped to. In this new space, the dimensionality is the number of "high-quality" clusters discovered from the training data, which is much smaller than the number of words in the bag-of-words representation.
- *Easy in implementation and combination.* The Crest framework is easy to implement and can be easily combined with other approaches dealing with short text classification.

The rest of the paper is organized as follows. Section 2 surveys the related work in short text classification. Section 3 describes the Crest method. Section 4 reports the experimental results and Section 5 concludes this paper.

2 Related Work

Short text processing has attracted research interests for a long time, particularly in the meta-search applications to group similar search results into meaningful topic clusters. Nevertheless, the key research problem in search snippet clustering is to automatically generate meaningful cluster labels [3]. Another direction of research in short text processing is to evaluate the similarity of a pair of short texts using external knowledge obtained from search engines [11,17]. In [1], semantic similarity between words is obtained by leveraging page counts and text snippets returned by search engine.

For short text classification, the work on query classification is more related as each query can be treated as a piece of short text. In [14], the authors use titles and snippets to expand the Web queries and achieve better classification accuracy on query classification task compared to using the queries alone. However, the efficiency and the reliability issues of using search engine limit the employment of search-based method, especially when the set of short text under consideration is large. To address these issues, researchers turn to utilize explicit taxonomy/concepts or implicit topics from external knowledge source. These corpora (e.g., Wikipedia, Open Directory) have rich predefined taxonomy and human labelers assign thousands of Web pages to each node in the taxonomy. Such information can greatly enrich the short text. These research has shown positive improvement though they only used the man-made categories and concepts in those repositories. Wikipedia is used in [6] to build a concept thesaurus to enhance traditional content similarity measurement. Similarly, in [8], the authors use Wikipedia concept and category information to enrich document representation to address semantic information loss caused by bag-of-words representation. A weighted vector of Wikipedia-based concepts is also used for relatedness estimation of short text in [5]. However, lack of adaptability is one possible shortcoming of using predefined taxonomy in the above ways because the taxonomy may not be proper for certain classification tasks. To overcome this shortcoming, the authors in [10] derived latent topics from a set of documents from Wikipedia and then used the topics as additional features to expand the short text. The idea is further extended in [4], to explore the possibility of building classifier by learning topics at multi-granularity levels. Experiments show that the methods above using the discovered latent topics achieve the state-of-the-art performance. In summary, these methods try to *enrich* the representation of a short text using additional semantics from an external collection of documents. However, in some specific domain (e.g., military or healthcare) it might be difficult to get such high quality external corpora due to privacy or confidentiality reasons.

Most germane to this work is the approach proposed in [2] which applies probabilistic latent semantic analysis (pLSA) on text collection and enriches document representation using the latent factors identified. However, pLSA becomes less reliable in identifying latent topics when applying to very short texts, due to the difficulties of sparsity and shortness. In this paper, we use a different approach to find the topics embedded in the short text collection by clustering the documents in the collection.

3 The CREST Method

Most existing topic-based methods rely on large external sources (such as Wikipedia or search engines). However, there exist tough situations in some specific domains (e.g., military or healthcare) where lack of reliable high quality external knowledge repositories. This limits the employment of these methods. In this scenario, the only available resource is the collection of labeled short texts. How to exploit the limited collection at utmost becomes crucial in short text classification.

The good performance of topic-based methods shows latent topics can be very useful to short text classification. Since the document collection is the only available resource in our scenario, we derive latent topics from the document collection itself by exploiting clustering. Then, we use the topic clusters to enrich the representation for short texts. The general process of CREST (***C****luster-based* ***R****epresentation* ***E****nrichment for* ***S****hort* ***T****ext Classification*) method is illustrated in Figure 1.

Suppose a document collection $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$ has n short text documents, where $\mathbf{x}$ is pre-processed short text document and $\mathbf{x} \in X = R^d$. In this paper, we adopt *tf-idf* [12] representation. And y is category label, $y \in Y = \{1, 2, \ldots, k\}$. L is a learning algorithm, training a classifier $h : X \to Y$.

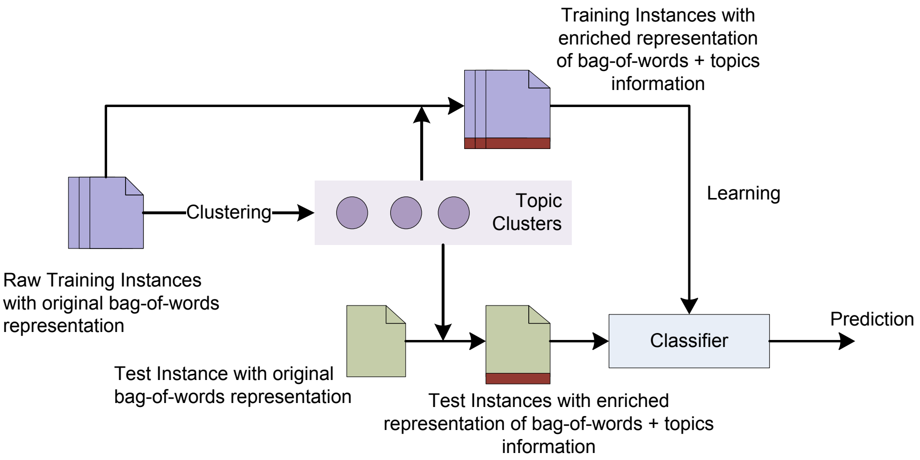

Training Instances with enriched representation of bag-of-words + topics information
Clustering
Topic Clusters
Learning
Raw Training Instances with original bag-of-words representation
Prediction
Classifier
Test Instance with original bag-of-words representation
Test Instances with enriched representation of bag-of-words + topics information

Fig. 1. Procedure of CREST

3.1 Topic Clusters Generation

Clustering is good at finding knowledge structure inside data. CREST exploits clustering to find topics. Intuitively, for each high-level category, for example "Business", it has its a few sub-topics, such as "Accounting","Finance". The sub-topics could have different topical words, especially when the text is very short. In other word, each cluster contains terms and concepts mainly in one sub-topic which we could take advantage of to enrich short texts and reduce their sparsity.

However, due to the sparsity of short text, the similarity of a pair of short text instances may not be reliable enough when it is reflected by distance in a clustering method. Thus, the resulting clusters may not be qualified as topics. The challenge here is to select "high-quality" clusters as *topic clusters*. Note that, even though there exist many clustering methods, not all clusters generated by a clustering method is useful. For instance, a cluster containing very few documents (say, only one) or a large number of documents from many different categories are not useful clusters. The clusters with very few documents fail to cover enough concepts in a sub-topic while the clusters containing too many documents are not topically specific.

In summary, CREST selects "high-quality" clusters as topic clusters with two criteria: (i) *high support*, i.e., the number of documents in a cluster is large; and (ii) *high purity*, i.e., the percentage of dominant category of the short texts in a cluster is high.

Suppose a cluster Q contains a set of short text instances, $Q = \{(\mathbf{x}_i, y_i)\}_{i=1}^{q}$, then the *support* of Q is the number of instances in it, i.e.,

$$support(Q) = |Q|. \tag{1}$$

And the *purity* of Q is the percentage of dominant category of the short texts in it, which is defined as:

$$purity(Q) = \frac{\max_y \sum_{\mathbf{x}_i \in Q} I(y_i = y)}{|Q|}, \tag{2}$$

where, $I(x)$ is indicator function, $I(x) = 1$ if $x = 1$ and 0 otherwise.

More specifically, Crest uses a clustering method, such as $EfficientHAC$ [9], to group short texts into clusters. When a cluster's purity is low, it does not represent a sub-topic even if its support is high. Therefore, we select the clusters whose purity values are larger than a pre-defined threshold. We then get a set of candidate-clusters $\mathcal{C} = \{C_1, C_2, \ldots, C_{|C|}\}$. To select clusters with high "support" and "purity", we assign a weight to each cluster in $\mathcal{C}$ indicating the quality to be a topic cluster of each cluster. Let w_i be C_i's weight:

$$w_i = support(C_i) \times purity(C_i), \tag{3}$$

Then the top N clusters with the highest weights are selected as topic clusters $\mathcal{T}$, which are rich of representative terms or concepts in particular sub-topics, and are later used to enrich short text's representation.

In most cases, the weights of candidate-clusters in $\mathcal{C}$ are influenced more by their support values. It is reasonable, since the purity values of candidate-clusters in $\mathcal{C}$ are all larger than a purity threshold, which is often a relatively high value to assure all clusters in $\mathcal{C}$ be of high purity.

3.2 Representation Enrichment Using Topic Clusters

Crest enriches representation of short text by combining a short text instance's original feature vector, i.e., *tf-idf* vector, and the additional information from the topic clusters. To extract knowledge from topic clusters, a good choice is to use the similarity between a short text instance $\mathbf{x}$ and each of the topic cluster T_i in $\mathcal{T}$, which contains the common terms or concepts of a sub-topic. So the similarity between a short text instance $\mathbf{x}$ and a topic cluster T_i reflects how likely the common terms or concepts of the sub-topic represented by T_i would appear in the text if the "short" text were longer.

For example, a short text (taken from the benchmark dataset used in our experiments) is "manufacture manufacturer directory directory china taiwan products manufacturers directory- taiwan china products manufacturer directory exporter directory supplier directory suppliers business". And there are two topic clusters: cluster 1 represents a sub-topic of "business" category, and cluster 2 represents a sub-topic of "health" category. Cluster 1 contains concepts like "relation", "produce", "machine", and so on. Cluster 2 contains concepts like "symptoms", "treatment", "virus", "diet". Obviously, the short text is more similar to cluster 1. And if it were longer, the word "produce", "machine" have a larger chance to appear in the text.

Algorithm 1. The Crest Algorithm

```
Input : Training set $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, Learning algorithm $L$ to train a
        classifier $h : X \to Y$, Purity threshold $p \in [0, 1]$, Hierarchical clustering
        algorithm $EfficientHAC$, The number of topic clusters $N$
Training phrase:
%Generate topic clusters
Use $EfficientHAC$ algorithm to generate raw cluster set $R$
Candidate-cluster set $C = \{r | r \in R \wedge purity(r) \geq p\}$
for $i = 1$ to $|C|$ do
    $w_i = support(C_i) \times purity(C_i)$ %cluster weight
Select top $N$ clusters from $C$ with highest weights into topic cluster set $T$
%Enrich representation
for $i = 1$ to $n$ do
    for $j = 1$ to $N$ do
        Calculate similarity $sim(\mathbf{x}_i, T_j)$ according to Eq.4
    $\mathbf{x}'_i = (\mathbf{x}_i, sim_1(\mathbf{x}), \ldots, sim_N(\mathbf{x}))$
New data set $D' = \{(\mathbf{x}'_i, y_i)\}_{i=1}^n$
Output: A classifier $h = L(D')$

Test phrase: for a test instance $\mathbf{x}$
for $j = 1$ to $N$ do
    Calculate the similarity $sim(\mathbf{x}, T_j)$ according to Eq.4
$\mathbf{x}' = (\mathbf{x}, sim_1(\mathbf{x}), \ldots, sim_N(\mathbf{x}))$
Prediction $\hat{y} = h(\mathbf{x}')$
```

Define the similarity between a short text $\mathbf{x}$ and a topic cluster T as:

$$sim(\mathbf{x}, T) = \frac{\mathbf{x} \cdot T}{\|\mathbf{x}\| \, \|T\|} \tag{4}$$

In $sim(\mathbf{x}, T)$, the dot product is used to compute the initial similarity value between short text and topic cluster. Since the lengths of topic clusters are varying, to reduce their influence, we normalize the lengths of both short text and topic cluster to get final similarity, i.e., cosine similarity.

Let $\mathbf{s} = (sim(\mathbf{x}, T_1), \ldots, sim(\mathbf{x}, T_N))$ be the similarity vector, then the enriched representation of $\mathbf{x}$ is:

$$\mathbf{x}' = (\mathbf{x}, \mathbf{s}) \tag{5}$$

The pseudo code of Crest is shown in Algorithm 1, in which the clustering algorithm $EfficientHAC$ can be replaced by another hierarchical clustering algorithm.

4 Experiments

Since the problem setting of this paper is that there is no external knowledge sources, it is inappropriate to compare Crest with methods relying on some

Table 1. Basic Statistics of Experiment Dataset

Category	# training instances	# test instances
Business	1200	300
Computer	1200	300
Culture	1880	330
Education	2360	300
Engineering	220	150
Health	880	300
Politics	1200	300
Sports	1120	300
Total	10060	2280

external knowledge source. We compare CREST with original representation of short text (i.e., *tf-idf* vectors, denoted by "Raw"). In CREST, the clustering strategies EfficientHAC [9] is single-link, and the purity threshold is set to be 0.9. We test different values $10, 30, 50, 70, 100, 120$ for the number of topic clusters N. We use SVM as learning algorithm for both CREST and Raw representations using SVM^{light} with default parameter settings[1]. We run experiments on the benchmark dataset of search snippets collected by [10] and the statistics of the dataset is shown in Table 1.

For each parameter settings, we run the experiment for 20 times, then compute the average value. We record the F_1 measurement. Table 2 shows the F_1 results, where the tabular in boldface means that CREST's result is significantly better than Raw by pairwise t-test with significance level at 0.95, "*best*" is the best F_1 value among CREST with different N's, "*avg.*" is the average F_1 value over all categories. The results are plotted in Fig. 2.

Table 2. F_1 Results (%)

Method	busin.	compu.	cultu.	educa.	engin.	healt.	polit.	sport.	*avg.*
Raw	50.23	67.64	66.41	67.49	29.37	59.69	33.32	78.24	56.55
CREST $N = 10$	**58.79**	**68.92**	**68.01**	**69.57**	25.58	**63.78**	**37.09**	**80.23**	**59.00**
CREST $N = 30$	**55.87**	**69.60**	**68.38**	**68.78**	15.95	**62.49**	**38.51**	**80.23**	**57.48**
CREST $N = 50$	**53.97**	**68.63**	**66.91**	**69.48**	25.58	**61.20**	**39.68**	**80.15**	**58.20**
CREST $N = 70$	**56.37**	**70.54**	**66.91**	**69.48**	**31.11**	**60.16**	**39.78**	**80.31**	**59.33**
CREST $N = 100$	**55.65**	**69.72**	**67.15**	**70.48**	**33.14**	**60.47**	**39.36**	**78.89**	**59.36**
CREST $N = 120$	**54.91**	**68.90**	**68.36**	**69.62**	**33.14**	**60.9**	**38.95**	**78.65**	**59.18**
best	**58.79**	**70.54**	**68.38**	**70.48**	**33.14**	**63.78**	**39.78**	**80.31**	

These results show that CREST improves the classification performance considerably compared to Raw in every category with almost all parameter settings. Especially, in some specific categories such as "business" and "politics", the improvement is as large as 17.13% and 19.51%, respectively. The results show that CREST method utilizing topic clusters extracted from limited training examples

[1] `http://svmlight.joachims.org/`

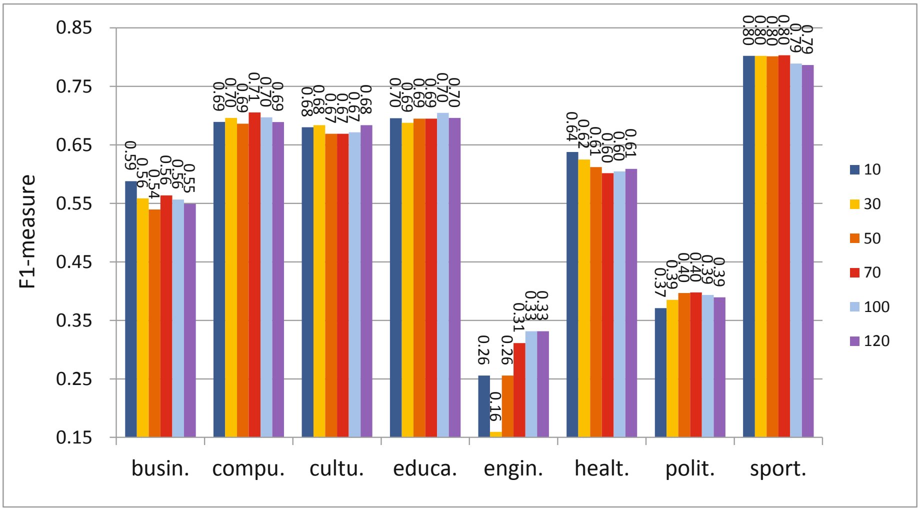


Fig. 2. Comparison among Different Embedded Number of Topic Clusters

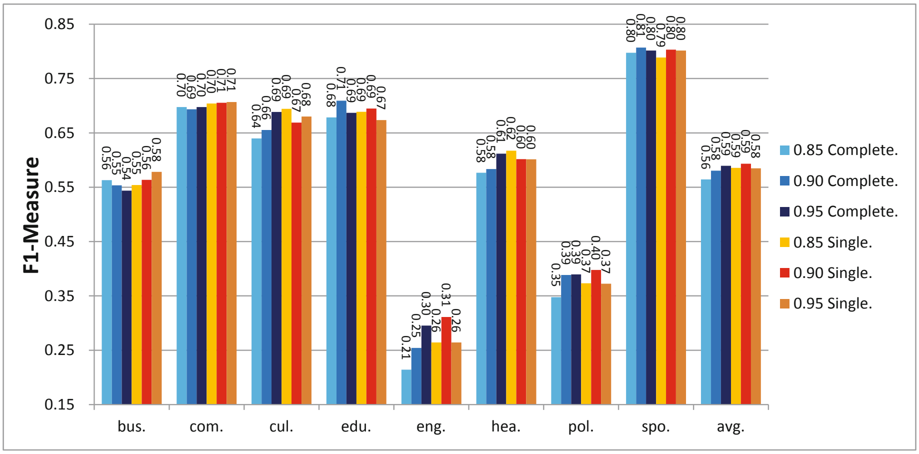


Fig. 3. Comparison among Different Clustering Strategies and Purity Thresholds

to enrich short texts is a useful way to overcome the shortness and sparsity of short texts. From Fig. 2 we can see that Crest is very robust to the change of N, the number of topic clusters. Even when N is very small, Crest improves the performance largely in almost all categories. This shows the power of the enriched representation by exploring topic clusters. The only exception is that in category "engineering", only when the number of topic clusters N is greater than 70 can Crest improves the performance. One possible reason is that "engineering" category has fewer instances than other categories but covers relatively a large topic. The instances in this category are harder to be gathered together by a clustering method. Crest manages to improve the performance of this category by increasing the number of topic clusters in N.

To further study how parameters will affect Crest, we record the F_1 results of Crest with different clustering strategies (single-link or complete-link) and different purity thresholds (0.85, 0.90, 0.95) while fixing $N = 70$. The results are shown in Fig. 3. Generally speaking, Crest is very robust to the change of these parameters when purity threshold is above 0.90. Since the topic clusters with higher purity would be more topic-specific, higher purity threshold leads to more helpful critical terms or concepts. On the other hand, clustering strategy doesn't affect the performance significantly. Crest is slightly more sensitive to purity threshold when using the single-link strategy than using the complete-link strategy.

The above experimental results lead to the following conclusions: (1) Crest can greatly improve the short text classification performance in term of F_1 measure by enriching the representation with topic information; and (2) Crest is robust to parameter settings.

5 Conclusion

Short text classification problem attracts much attention from information retrieval field recently. In order to handle its shortness and sparsity, various approaches have been proposed to enrich short text to get more features like latent topics or other information. However, most of them rely on large external knowledge sources more or less. These methods solve the problem to some extent, but still leave large space for improvement, especially under the hard condition that no external knowledge source can be acquired. We proposed Crest method to handle the short text classification in such tough situation. Crest generates "high-quality" clusters as topic clusters from training data by exploiting clustering method, and then uses the topic information to extend representation for short text. The experimental results showed that compared to the original representation, Crest can significantly improves the classification performance.

Though we see positive improvement brought by Crest, there are still room for further consideration to boost the performance. For example, we can try to combine Crest with other methods for short text classification, such as methods relying on external knowledge sources. And organizing "high-quality" clusters in multi-granularity way to investigate whether it can further improve Crest is another interesting problem worth exploring.

Acknowledgement. This work was supported by NSFC (No. 61105046), SRFDP (Specialized Research Fund for the Doctoral Program of Higher Education, by Ministry of Education, No. 20110092120029), and Open Foundation of National Key Laboratory for Novel Software Technology of China (KFKT2011B01). The work of the second author was supported by MINDEF-NTU-DIRP/2010/03, Singapore.

References

1. Bollegala, D., Matsuo, Y., Ishizuka, M.: Measuring semantic similarity between words using web search engines. In: Proceedings of the 16th International Conference on World Wide Web, New York, pp. 757–766 (2007)
2. Cai, L., Hofmann, T.: Text categorization by boosting automatically extracted concepts. In: Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval, New York, pp. 182–189 (2003)
3. Carpineto, C., Osiński, S., Romano, G., Weiss, D.: A survey of web clustering engines. ACM Computing Surveys (CSUR) 41(3), 17:1–17:38 (2009)
4. Chen, M., Jin, X., Shen, D.: Short text classification improved by learning multi-granularity topics. In: Proceedings of the 22nd International Joint Conference on Artificial Intelligence, pp. 1776–1781 (2011)
5. Gabrilovich, E., Markovitch, S.: Computing semantic relatedness using wikipedia-based explicit semantic analysis. In: Proceedings of the 20th International Joint Conference on Artifical Intelligence, San Francisco, CA, pp. 1606–1611 (2007)
6. Hu, J., Fang, L., Cao, Y., Zeng, H.-J., Li, H., Yang, Q., Chen, Z.: Enhancing text clustering by leveraging wikipedia semantics. In: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, New York, NY, pp. 179–186 (2008)
7. Hu, X., Sun, N., Zhang, C., Chua, T.-S.: Exploiting internal and external semantics for the clustering of short texts using world knowledge. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management, pp. 919–928 (2009)
8. 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 International Conference on Knowledge Discovery and Data Mining, New York, NY, pp. 389–396 (2009)
9. Manning, C., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, Cambridge (2008)
10. Phan, X.-H., Nguyen, L.-M., Horiguchi, S.: Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In: Proceedings of the 17th International Conference on World Wide Web, New York, NY, pp. 91–100 (2008)
11. Sahami, M., Heilman, T.D.: A web-based kernel function for measuring the similarity of short text snippets. In: Proceedings of the 15th International Conference on World Wide Web, New York, NY, pp. 377–386 (2006)
12. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Information Processing and Management 24(5), 513–523 (1988)
13. Shen, D., Chen, Z., Yang, Q., Zeng, H., Zhang, B., Lu, Y., Ma, W.: Web-page classification through summarization. In: Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 242–249 (2004)

14. Shen, D., Pan, R., Sun, J.-T., Pan, J.J., Wu, K., Yin, J., Yang, Q.: Query enrichment for web-query classification. ACM Transactions on Information Systems 24(3), 320–352 (2006)
15. Sun, A.: Short text classification using very few words. In: Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2012, New York, NY, pp. 1145–1146 (2012)
16. Tang, J., Wang, X., Gao, H., Hu, X., Liu, H.: Enriching short text representation in microblog for clustering. Frontiers of Computer Science in China 6(1), 88–101 (2012)
17. Yih, W.-T., Meek, C.: Improving similarity measures for short segments of text. In: Proceedings of the 22nd National Conference on Artificial Intelligence, pp. 1489–1494 (2007)

Cross Language Prediction of Vandalism on Wikipedia Using Article Views and Revisions

Khoi-Nguyen Tran and Peter Christen

Research School of Computer Science
The Australian National University, Canberra, ACT 0200, Australia
{khoi-nguyen.tran,peter.christen}@anu.edu.au

Abstract. Vandalism is a major issue on Wikipedia, accounting for about 2% (350,000+) of edits in the first 5 months of 2012. The majority of vandalism are caused by humans, who can leave traces of their malicious behaviour through access and edit logs. We propose detecting vandalism using a range of classifiers in a monolingual setting, and evaluated their performance when using them across languages on two data sets: the relatively unexplored hourly count of views of each Wikipedia article, and the commonly used edit history of articles. Within the same language (English and German), these classifiers achieve up to 87% precision, 87% recall, and F1-score of 87%. Applying these classifiers across languages achieve similarly high results of up to 83% precision, recall, and F1-score. These results show characteristic vandal traits can be learned from view and edit patterns, and models built in one language can be applied to other languages.

1 Introduction

Wikipedia is the largest free and open access online encyclopedia. It is written by millions of volunteers and accessed by hundreds of millions of people each month. These kinds of large open collaborative environments are naturally attractive to vandals. A malicious modification to a Wikipedia article is available instantly to millions of potential readers. Vandalism comes in many forms, where we adopt the definitions of Priedhorsky et al. [1], repeated here for convenience: misinformation, mass delete, partial delete, offensive, spam, nonsense, and other.

Vandalism is a key issue on Wikipedia, despite the majority of vandalism being caught and repaired very quickly [1–3]. Finding and repairing these vandalisms distracts Wikipedia editors from writing articles and other important work. To lighten the burden of finding and resolving vandalism, anti-vandalism bots have been created and are operating since 2006. Although these bots use simple rules and word lists, they find the majority of obvious vandalism cases [4].

As Wikipedia grows larger and vandals adapt to anti-vandalism bots, new techniques are needed to combat vandalism. Many machine learning techniques (see Sect. 2) offer potential automated solutions. Vandalism is commonly identified from user comments in Wikipedia data dumps of the complete edit history,

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 268–279, 2013.

where patterns in language, content, metadata, users, and others can be modelled. Various features, ranging from simple metadata to complex word analyses, are constructed for machine learning algorithms. These vandalism studies often use the English Wikipedia, but rarely the other 280+ language editions.

In this paper, we explore crosslingual vandalism detection by using a relatively unexplored data set, the hourly article view count, and the commonly used complete edit history of Wikipedia. We also combine these two data sets to observe any benefits from additional language independent features. We look at two language editions, English and German, and compare and contrast the performance of standard classifiers in identifying vandalism within a language and applied across language.

We hypothesise vandalism can be characterised by the view patterns of a vandalised articles. Vandals may be eliciting behavioural patterns before, during, and after a vandalised edit. We further hypothesise that behaviour of vandals is similar across language domains. This means models developed in one language can be applied to other languages. This can potentially reduce the cost of training classifiers for each language. We find this cross language application of vandalism models produces similarly high results as for a single language.

Our contributions are (1) novel use of the hourly article view data set for vandalism detection; (2) creation and combination of data sets with language independent features; and (3) showing the cross language applicability of vandalism models built for one language.

The rest of this paper is organised as follows. Section 2 reviews the related work. Section 3 provides statistics of the Wikipedia data sets and how to create the combined data set. Section 4 details the machine learning algorithms and their parameters. Section 5 summarises the results, providing precision, recall, F1-score, and execution times. Section 6 discusses the significance, quality, and limitations of this data set and approach. Finally, we conclude this paper in Section 7 with outlook to future work.

2 Related Work

We survey some of the most related research on vandalism detection. Vandalism is a prominent issue on Wikipedia, which arise in many research looking the dynamics of Wikipedia. One increasingly popular approach of finding vandalism is to use machine learning techniques. This approach and others are applied to a Wikipedia vandalism detection competition at the PAN workshop[1].

The complex open collaborative environment of Wikipedia has seen many studies trying to comprehend the interactions that lead to developing content. By its open nature, vandalism or more general malicious edits have occurred on every Wikipedia article [2]. Vandalism is a burden on Wikipedia, where its occurrence and work in identifying and reverting it are increasing [3]. The time spent on maintenance work, such as reverting vandalism, by Wikipedians (registered users) are increasing, which leave less time for writing articles [3].

[1] http://pan.webis.de/

Wikipedians have a variety of ways to deal with vandalism, which including developing and using tools to identify vandalism, such as bots [5]. Many types of vandalism can be identified clearly from visualisations of the edit history using flow diagrams [2]. Other types of vandalism require more complex analysis of the article content. Although many cases of vandalism are repaired almost immediately [1–3], the probability that an article will be vandalised is increasing over time [1].

Vandalism often has many characteristics, where use of machine learning is becoming increasingly common [6]. These machine learning techniques require building features from the Wikipedia data sets, which can range from simple metadata to more complex analysis of content, semantics, authors, and interactions. Anti-vandalism bots have been constantly monitoring Wikipedia since 2006, but the simple features and constructed rules and word lists used by the bots can be easily deceived and leave room for improvement [4].

Analysing the words used in the content of articles can provide evidence of vandalism. When comparing revisions of an article, word level features can determine whether the use of certain words will be rejected and reverted in later revisions [7]. The revision history of an article offers a distribution of words relevant to that article. This word distribution allows machines to find use of unexpected words, which is a common type of vandalism [8]. More general analyses of words and content often use natural language processing techniques, which can provide models that well surpass rule based approaches and other machine learning approaches [9]. Linguistic features from applying natural language processing can characterise vandalism and be learned by machines [10].

By combining content analyses with other information about authors and objective measures of edit quality, reputation systems can be developed to identify vandalism [11]. Without these features, spatio-temporal properties of metadata can be sufficient for machine learning algorithms to detect vandalism [12]. However, machine learning algorithms can be improved by using many features, to which some research use a range of features identified from past research studies to train algorithms [12].

In recent years, the task of identifying vandalism on Wikipedia has been turned into a competition. The PAN Workshop hosted Wikipedia vandalism detection competitions as part of its workshops in 2010 and 2011. In 2010, a vandalism corpus was created using the Amazon's Mechanical Turk to label its data set [13]. This crowdsourcing of vandalism identification proved to be successful and a larger crowdsourced corpus of over 30,000 Wikipedia edits was released in 2011, and in three languages: English, German, and Spanish [14]. This multilingual vandalism corpus uses 65 features to quantify characteristics of an edit to capture vandalism. The 2010 winner explored metadata features from edits and expanded word list features for a Random Forest classifier [15]. A post 2010 competition study combined spatio-temporal analysis of metadata [12], reputation system [11], and natural language processing features to further improve on the winning system. The 2011 winner focused on language independent features and constructed 65 features for an alternating decision tree classifier [16].

Table 1. Basic statistics of edit history data set. All revisions until start of June 2012.

Language	Content articles	Article revisions	Distinct usernames	Distinct IP addresses
English	4,000,264	305,821,091	4,020,470	25,669,884
German	1,419,217	65,732,032	447,603	5,565,475

Table 2. Basic statistics of article view data set. From January 2012 to May 2012.

Language	Articles viewed	Total views
English	2,261,593	4,567,904,954
German	805,964	1,493,732,111

3 Wikipedia Data Sets

In this section, we describe the process of generating the data sets used for vandalism classification. We use two data sets: the complete edit history of Wikipedia in English and German[2], and the hourly article view count[3]. We describe data with language codes "en" for English and "de" for German. These two raw data sets are processed as described in the subsections below.

We use the edit history data dump of 1 June 2012 for the English Wikipedia, and 3 June for the German Wikipedia. Table 1 summarises the number of articles and revisions, and distinct usernames. Content articles are strictly encyclopedic articles and do not include articles for redirects, talk, user talk, help, and other auxiliary article types. We provide count of usernames and IP addresses in Table 1 to give indication of activity in the two Wikipedias.

The raw article view data set contains all of MediaWiki projects (including Wikipedia). As of writing this paper, we have obtained all data from January to May 2012. We filter only revisions made in this time period from the edit history data. Table 2 provides some basic statistics on the raw data set filtered to view counts of English and German articles. Accordingly, we filtered the edit history data set to revisions made between January and May 2012.

3.1 Vandalised Revisions

From the raw revision data, every revision is reduced to a vector of features described in Table 3. These features are selected for their language independence and simplicity. For each revision, we analyse its comment for keywords of "vandal" and "rvv" (revert due to vandalism), indicating the occurrence of vandalism in the previous revision(s). The appropriate revisions are then marked as an occurrence of vandalism.

To align the timestamp of revisions to the corresponding article view data set, we round up the revision time to the next hour. This ensures that the hourly

[2] `http://dumps.wikimedia.org/backup-index.html`

[3] `http://dumps.wikimedia.org/other/pagecounts-raw/`

Table 3. Description of edit history data set

Attribute	Description
Article title	Unique identifier of a Wikipedia article.
Hour timestamp	The timestamp of this revision. In the format of YYYYMMDD-HH0000. The minutes and seconds are used to round up to the next hour.
Anonymous edit	The editor of this revision is considered to be anonymous if an IP address is given. 0 for an edit by a registered user, and 1 for an edit by an anonymous user.
Minor revision	Revisions can be flagged as minor edits. 0 for normal revision, and 1 for minor revision.
Size of comment (bytes)	The size of the given comment of this revision.
Size of article text (bytes)	The size of the complete article of this revision.
Vandalism	This revision is marked as vandalism by analysing the comment of the following revision(s). 0 for not vandalism, and 1 for vandalism.

Table 4. Description of article view data set

Attribute	Description
Project name	The name of the MediaWiki project, where we are interested in Wikipedia projects in English ("en") and German ("de").
Hour timestamp	In the format of YYYYMMDD-HH0000, where YYYY for year; MM for month; DD for day of the month; HH for 24-hour time (from 00 to 23); and minutes and seconds are not given.
Article title	The title of the Wikipedia article. Article may not exist as the data set is derived from Web server request logs.
Number of requests	The number of requests in that hour. Not unique visits by users.
Bytes transferred	The total number of bytes transferred from the requests.

article views references the correct revision when combining the two data sets. The alignment is performed on all revisions and should not affect classification.

We emphasise that user labelling of Wikipedia vandalism is noisy and incomplete. Some research provides solutions to this problem such as active learning [8], but a fully automated approach have inherent limitations as human involvement is necessary for some cases of vandalism [17]. We find about 2% of revisions between January to May 2012 contain vandalism. This is consistent with studies looking at these keywords [3], but less than the 4-7% reported in other studies looking at vandalism beyond user labelling [1, 11, 13].

3.2 Article Views

The raw article view data set is structured by views of article aggregated by hour. We perform a simple transformation and filtering of articles seen in the revisions data set above. The resulting features are summarised in Table 4.

We also extract the redirect articles from the revisions data set and change all access to redirect articles to the canonical article. These extra view counts are aggregated accordingly.

These article views are important to seeing the impact of vandalism on Wikipedia [1]. With the average survival time of vandalism being 2.1 days [3], this leaves many hours for unsuspecting readers to encounter vandalised content. However, the behaviour of vandals may also be seen in a change in access patterns, which may be from vandals checking on their work, or that article drawing attention from readers and their peers.

A previous research study [1] (before the release of this data set) derived article views from the full Wikipedia server logs. This provides a much finer time unit for analysis, but with a huge increase in data to process. With the time unit of hours, this data set may provide coarse patterns of behaviours, but with manageable data size.

There are few research studies that use this data set. Most research has developed tools for better access to this huge data resource and to provide simple graphs for topic comparison. One relevant study [18] use this data set to compare access to medical information on seasonal diseases like the flu. Access patterns in this data set reflect the oncoming of seasonal diseases. Wikipedia is accessed more than other online health information providers, and is a prominent source of online health information. Although vandalism is not covered, the seasonal access patterns elude to potential targets of vandalism.

To determine whether these article views occurred when articles are in a vandalised state, we scan the edit history data set and label all article views of observed vandalised or non-vandalised revisions. The unknown views from revisions made before January 2012, or articles without revisions in this 5 month period under study, are discarded. Thus, we have an article view data set labelled with whether the views are of vandalised revisions. The resulting size of the data is identical to the combined data set in the following subsection. This labelled article view data set allows us to determine whether view patterns can be used to predict vandalism.

From this resulting combined set, we split the “Hour timestamp” attribute into an “hour” attribute. This allows the machine learning algorithm to learn daily access patterns. In future work, we intend to experiment with monthly and yearly access patterns when we have obtained enough raw data.

3.3 Combined Data Set

The combined data set is the result of merging of two time series data sets for each language. The data set is constructed by adding features from the labelled revisions data set to the labelled article view data set by repeating features of the revisions. Thus for every article view, we have information on whether a vandalised revision was viewed and what the properties of that revision are.

We use the “hour” attribute split from the timestamp in the article views data set. Thus, we have the following 8 features in our combined data set: **hour, size**

Table 5. Statistics of the various data sets. With percentage of vandalism.

Data set	Vandalised revisions	Article views	Combined (train)	Combined (test)
English	17,159,583	525,382,429	271,584,092	99,611,391
(vandal)	(2.08%) 356,618	-	(2.34%) 6,367,602	(2.04%) 2,033,838
German	3,731,714	284,932,083	139,967,644	55,010,679
(vandal)	(0.10%) 3,889	-	(0.06%) 86,534	(0.07%) 40,143

of comment, size of article, anonymous edit, minor revision, number of requests, bytes transferred, and **vandalism** (class label).

These features are language independent and capture the metadata of revisions commonly used, and access patterns. Note that we remove the article name as they are not necessary in evaluating the quality of classification. For example, access patterns of vandalised articles may be similar to other vandalised articles, regardless of the name of articles. For future work, we may identify the articles classified and further analyse to determine genuine cases of vandalism unlabelled or overlooked by editors.

To apply the classification algorithms, we split the combined data set by date into a training set (January to April) and a test set (May). The statistics of the data sets in this section are shown in Table 5 for comparison.

4 Cross Language Vandalism Prediction

We use the Scikit-learn toolkit [19], which provides many well-known machine learning algorithms for science and engineering. We selected the following supervised machine learning algorithms from the toolkit:

- Decision Tree (DT)
- Random Forest (RF)
- Gradient Tree Boosting (GTB): binomial deviance as the loss function.
- Stochastic Gradient Descent (SGD): logistic regression as the loss function.
- Nearest Neighbour (NN): KDTree data structure.

We experimented with different settings available for the classifiers above, but we found there is little to no variance in the results. This is likely because all classifiers converged with the already large number of observations given.

From Table 5, we see the data set is highly unbalanced, which is unsuitable for some of our classifiers. We resolved this problem by undersampling the non-vandalism observations to match the number of vandalism observations. We apply this to all three data sets. Thus, we built a balanced subset of the training and testing data.

We repeated the application of the classifiers to the balanced data to observe any effects from the random samples of non-vandalism observations. We found all classifiers seem to have converged with the already large number of observations in the balanced subset.

Table 6. Classification results of the revisions data set

Model-Language	Precision					Recall					F1-Score				
	DT	RF	GTB	SGD	NN	DT	RF	GTB	SGD	NN	DT	RF	GTB	SGD	NN
en-en	0.78	**0.84**	**0.84**	**0.84**	0.69	0.78	0.83	**0.84**	**0.84**	0.69	0.78	0.83	**0.84**	**0.84**	0.69
de-de	0.75	**0.84**	**0.84**	0.69	0.69	0.74	0.83	**0.84**	0.51	0.68	0.74	0.83	**0.84**	0.36	0.68
de-en	0.70	0.81	**0.82**	0.64	0.59	0.70	0.80	**0.82**	0.51	0.57	0.70	0.80	**0.81**	0.35	0.56
en-de	0.76	0.82	**0.83**	**0.83**	0.58	0.76	0.82	**0.83**	**0.83**	0.56	0.76	0.82	**0.83**	**0.83**	0.54

Table 7. Classification results of the article views data set

Model-Language	Precision					Recall					F1-Score				
	DT	RF	GTB	SGD	NN	DT	RF	GTB	SGD	NN	DT	RF	GTB	SGD	NN
en-en	**0.82**	0.55	0.78	0.62	0.69	**0.80**	0.53	0.73	0.50	0.69	**0.80**	0.48	0.72	0.35	0.69
de-de	**0.81**	0.69	0.70	0.25	0.69	**0.74**	0.69	0.70	0.50	0.68	**0.72**	0.69	0.70	0.33	0.68
de-en	0.55	0.63	**0.68**	0.25	0.59	0.50	0.63	**0.68**	0.50	0.57	0.35	0.62	**0.68**	0.33	0.56
en-de	0.60	0.51	**0.62**	0.54	0.58	0.55	0.50	**0.62**	0.50	0.56	0.48	0.42	**0.62**	0.34	0.54

Table 8. Classification results of the combined data set

Model-Language	Precision					Recall					F1-Score				
	DT	RF	GTB	SGD	NN	DT	RF	GTB	SGD	NN	DT	RF	GTB	SGD	NN
en-en	**0.86**	**0.87**	**0.85**	0.84	0.69	**0.84**	**0.87**	**0.85**	0.84	0.69	**0.83**	**0.87**	**0.85**	0.84	0.69
de-de	0.81	0.84	**0.88**	**0.72**	0.69	0.74	0.82	**0.87**	0.51	0.68	0.72	0.82	**0.87**	0.35	0.68
de-en	0.65	0.73	**0.83**	0.60	0.59	0.53	0.68	0.82	0.50	0.57	0.42	0.66	**0.82**	0.34	0.56
en-de	0.70	0.77	0.82	0.83	0.58	0.58	0.75	0.82	0.83	0.56	0.51	0.75	0.82	0.83	0.54

We also tried to train a Support Vector Machine (SVM) classifier, but we are unable to obtain results because of the different order in magnitude of training time. We experimented with very few number of samples (0.1-1% of the data set) to obtain results for SVM within a reasonable time frame. However, we found all classifiers above and including the SVM performed poorly with the small number of observations.

For cross language vandalism prediction, we first train classification models for our two languages: English and German. These models are then evaluated on the testing set for the same language, then to the testing set of the other language. This may seem odd with the independent nature of language domains. However, our data sets capture language independent features of Wikipedia. This cross language application of models allows a generalisation of editing and viewing behaviour across Wikipedia.

This cross language application of models has seen successful applications in the research area of cross language text categorisation [20, 21]. When considering text, cultural knowledge of the target language is needed to inform classifiers. The advantage of cross language application of models is that one model can be used for multiple languages, saving resources developing models for each language. This is particularly relevant to Wikipedia with its large range of languages. This research allows the potential generalisation of the concentration of vandalism research in English to other languages without additional inputs.

Table 9. Approximate execution time of classifiers in seconds

Time Taken (s)	DT	RF	GTB	SGD	NN
Training (en)	750	550	800	5	20
Training (de)	3	4	15	1	1
Testing (en-en)	5	16	3	0.5	150
Testing (de-de)	0.5	0.5	0.5	0.5	2
Testing (de-en)	2	7	5	2	90
Testing (en-de)	0.5	0.5	0.5	0.5	4

5 Experimental Results

The classification results are presented in Tables 6, 7, and 8. These are the total obtained scores from classification of the two classes: vandalism and non-vandalism. They present the classification results of a classifier trained in one language and applied to another. For example, "en-de" means the classification model is trained on the English training set, then applied to the German testing set. The highest classification scores of the classifier group are highlighted in bold font in Tables 6 and 7. For the combined data set, the highest scores and scores that outperformed the individual data sets are highlighted in bold font in Table 8. The approximate execution times, gathered and rounded from multiple runs, are summarised in Table 9.

For the monolingual application of classification models in the single data sets, the tree based methods generally have better performance. In particular GTB and RF for the revisions data set, and DT for the views data set. They are also the most expensive models to train.

The crosslingual application showed similar, but generally weaker, performance across all measures. GTB and RF continue to show generally better performance than the other classifiers. Interestingly, SGD performed best in the monolingual and crosslingual cases when trained on the English revisions data, suggesting English may offer more patterns to detect vandalism. This is encouraging because SGD is the fastest algorithm to train. The crosslingual application of models is not detrimental in most cases for all data sets, but with similar performance to the monolingual case. This suggests cross language classification of vandalism is feasible with a variety of data sets.

In the combined data set, we see improvements to the classification scores, but mainly in the monolingual case. GTB continues to show high performance with improvements from the additional features. In general the combination of the data sets does not provide a significant advantage to the classifiers. The classifiers seem to do as well on the combined data set compared to individual data sets, but not much better. This suggests the classifiers are learning the best models from each data set, but improvements are not common.

The monolingual classification scores of the revisions data set in Table 6 are comparable and better than many state-of-the-art systems. Note that the data sets used in various research studies are often constructed differently, and so care is needed when comparing different studies. From overviews of the PAN Wikipedia Vandalism Detection competition [14, 22], our results show better performance than many of entries, while using fewer features. The competition showcased multilingual entries in 2011, but no cross language application of models is seen. White and Maessen [23] presents an entry into the 2010 PAN vandalism competition and collated results from other Wikipedia vandalism research. We find our results for monolingual classification to generally have higher precision, recall, and F1-score.

6 Discussion

Vandalism is an important cross language issue on Wikipedia as more people contribute to and use Wikipedia as a resource in many different languages. The current research on vandalism shows promising technologies to automatically detect and repair vandalism. However, these research studies largely concentrate on the English Wikipedia. The generalisation of these studies to other languages may not always be possible because of the independence of language domains, and the peculiarities in languages. Multilingual vandalism research is appearing, aided by construction of multilingual vandalism data sets, such as those by the PAN workshop. The cross language vandalism detectors are ideal as models develop in one language can be applied to other languages.

The advantages of the presented data sets are the simple to extract language independent features. These few features with the application of baseline classification algorithms outperform many past research studies. The combination of editing and viewing patterns shows some increase in performance, but generally allows classifiers to adapt to the best predictive features from both data sets individually. The article view data set may be too coarse to predict vandalism at the hourly level, but we found some classifiers can find patterns of vandalism as well, or better than the revisions data set in some cases.

Some limitations of our approach include using few features, not analysing the content, and the necessity of the revisions data set to label the article views data set. The rich number of features used in other studies allows classifiers to learn more patterns of vandalism. This can often improve performance, but we find these data sets can be difficult to generate, especially when deploying solutions in bots. We have ignored the content of revisions, where word analysis may show the clear cases of unlabelled cases of vandalism. However, this is simply not feasible on a large scale required for Wikipedia and its many languages.

Our data set offers indications of vandalism that can be investigated with more complex techniques. The article views data set alone is not sufficient for vandalism detection and requires labelling from the revisions data set. However, by building labelled article views data sets, unlabelled articles can be incorporated and learned in a semi-supervised setting. Despite these limitations, we have shown cross language application of vandalism models is feasible, and view patterns can be used to predict vandalism and may offer improvements to classifiers.

7 Conclusion

We have presented data sets for vandalism detection and demonstrated the application of various machine learning algorithms to detect vandalism within one language and across languages. We developed three data sets from the hourly article view count data set, complete edit history of Wikipedia, and their combination. We looked at two language editions of Wikipedia: English and German. Within the same language, these baseline classifiers achieve up to 87% precision, 87% recall, and an F1-score of 87%. The cross language application of these classifiers achieved similarly high results of up to 83% precision, recall, and F1-score. We find Gradient Tree Boosting showed generally best performance in predicting vandalism, despite being the most time consuming algorithm. These results show the view and edit behaviour of vandals is similar across different languages. The implication of this result is that vandalism models can be trained in one language and applied to other languages.

In future work, we could extend the time span of the data set and apply to other languages. This would provide further evidence for the general applicability of classification models cross language to detect vandalism using this combined data set. We may add further features to enrich the data set and explore other balancing techniques. We could improve the baseline classifiers by building classifiers more suited to this data set. In the long term, we plan to have this system able to generate the data set in near real time and predict possible cases of vandalism for closer analysis.

References

1. Priedhorsky, R., Chen, J., Lam, S.T.K., Panciera, K., Terveen, L., Riedl, J.: Creating, destroying, and restoring value in wikipedia. In: Proceedings of the 2007 International ACM Conference on Supporting Group Work, GROUP 2007, pp. 259–268. ACM, New York (2007)
2. Viégas, F.B., Wattenberg, M., Dave, K.: Studying cooperation and conflict between authors with history flow visualizations. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI 2004, pp. 575–582. ACM, New York (2004)
3. Kittur, A., Suh, B., Pendleton, B.A., Chi, E.H.: He says, she says: conflict and coordination in wikipedia. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI 2007, pp. 453–462. ACM, New York (2007)
4. Smets, K., Goethals, B., Verdonk, B.: Automatic vandalism detection in wikipedia: Towards a machine learning approach. In: AAAI Workshop on Wikipedia and Artificial Intelligence: An Evolving Synergy, pp. 43–48 (2008)
5. Panciera, K., Halfaker, A., Terveen, L.: Wikipedians are born, not made: a study of power editors on wikipedia. In: Proceedings of the ACM 2009 International Conference on Supporting Group Work, GROUP 2009, pp. 51–60. ACM, New York (2009)
6. Potthast, M., Stein, B., Gerling, R.: Automatic vandalism detection in wikipedia. In: Macdonald, C., Ounis, I., Plachouras, V., Ruthven, I., White, R.W. (eds.) ECIR 2008. LNCS, vol. 4956, pp. 663–668. Springer, Heidelberg (2008)

7. Rzeszotarski, J., Kittur, A.: Learning from history: predicting reverted work at the word level in wikipedia. In: Proc. of the ACM 2012 Conf. on Computer Supported Cooperative Work, CSCW 2012, pp. 437–440. ACM, New York (2012)
8. Chin, S.C., Street, W.N., Srinivasan, P., Eichmann, D.: Detecting wikipedia vandalism with active learning and statistical language models. In: Proc. of the 4th Workshop on Information Credibility, WICOW 2010, pp. 3–10. ACM (2010)
9. Wang, W.Y., McKeown, K.: ”got you!”: Automatic vandalism detection in wikipedia with web-based shallow syntactic-semantic modeling. In: Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), Beijing, China. Coling 2010 Organizing Committee, pp. 1146–1154 (August 2010)
10. Harpalani, M., Hart, M., Singh, S., Johnson, R., Choi, Y.: Language of vandalism: Improving wikipedia vandalism detection via stylometric analysis. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers, vol. 2, pp. 83–88 (2011)
11. Adler, B., de Alfaro, L., Pye, I.: Detecting wikipedia vandalism using wikitrust. Notebook Papers of CLEF 1, 22–23 (2010)
12. West, A.G., Kannan, S., Lee, I.: Detecting wikipedia vandalism via spatio-temporal analysis of revision metadata? In: Proceedings of the Third European Workshop on System Security, EUROSEC 2010, pp. 22–28. ACM, New York (2010)
13. Potthast, M.: Crowdsourcing a wikipedia vandalism corpus. In: Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2010, pp. 789–790. ACM, New York (2010)
14. Potthast, M., Holfeld, T.: Overview of the 2nd international competition on wikipedia vandalism detection. In: Notebook for PAN at CLEF (2011)
15. Velasco, S.: Wikipedia vandalism detection through machine learning: Feature review and new proposals. In: Lab Report for PAN-CLEF 2010 (2010)
16. West, A.G., Lee, I.: Multilingual vandalism detection using language-independent & ex post facto evidence - notebook for pan at clef 2011. In: Petras, V., Forner, P., Clough, P.D. (eds.) CLEF (Notebook Papers/Labs/Workshop) (2011)
17. Wu, Q., Irani, D., Pu, C., Ramaswamy, L.: Elusive vandalism detection in wikipedia: a text stability-based approach. In: Proceedings of the 19th ACM International Conference on Information and Knowledge Management, CIKM 2010, pp. 1797–1800. ACM, New York (2010)
18. Laurent, M., Vickers, T.: Seeking health information online: does wikipedia matter? Journal of the American Medical Informatics Association 16(4), 471–479 (2009)
19. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research 12, 2825–2830 (2011)
20. Rigutini, L., Maggini, M., Liu, B.: An em based training algorithm for cross-language text categorization. In: Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence, pp. 529–535 (September 2005)
21. Liu, Y., Dai, L., Zhou, W., Huang, H.: Active learning for cross language text categorization. In: Tan, P.-N., Chawla, S., Ho, C.K., Bailey, J. (eds.) PAKDD 2012, Part I. LNCS, vol. 7301, pp. 195–206. Springer, Heidelberg (2012)
22. Potthast, M., Stein, B., Holfeld, T.: Overview of the 1st international competition on wikipedia vandalism detection. In: Braschler, M., Harman, D., Pianta, E. (eds.) CLEF (Notebook Papers/LABs/Workshops) (2010)
23. White, J., Maessen, R.: Zot! to wikipedia vandalism - lab report for pan at clef 2010. In: CLEF (Notebook Papers/LABs/Workshops) (2010)

An Optimized Cost-Sensitive SVM for Imbalanced Data Learning

Peng Cao[1,2], Dazhe Zhao[1], and Osmar Zaiane[2]

[1] Key Laboratory of Medical Image Computing of Ministry of Education,
Northeastern University, China
[2] University of Alberta, Canada
{cao.p,zhaodz}@neusoft.com, zaiane@ualberta.ca

Abstract. Class imbalance is one of the challenging problems for machine learning in many real-world applications. Cost-sensitive learning has attracted significant attention in recent years to solve the problem, but it is difficult to determine the precise misclassification costs in practice. There are also other factors that influence the performance of the classification including the input feature subset and the intrinsic parameters of the classifier. This paper presents an effective wrapper framework incorporating the evaluation measure (AUC and G-mean) into the objective function of cost sensitive SVM directly to improve the performance of classification by simultaneously optimizing the best pair of feature subset, intrinsic parameters and misclassification cost parameters. Experimental results on various standard benchmark datasets and real-world data with different ratios of imbalance show that the proposed method is effective in comparison with commonly used sampling techniques.

1 Introduction

Recently, the class imbalance problem has been recognized as a crucial problem in machine learning and data mining [1]. This problem occurs when the training data is not evenly distributed among classes. This problem is also especially critical in many real applications, such as credit card fraud detection when fraudulent cases are rare or medical diagnoses where normal cases are the majority. In these cases, standard classifiers generally perform poorly. Classifiers usually tend to be overwhelmed by the majority class and ignore the minority class examples. Most classifiers assume an even distribution of examples among classes and assume an equal misclassification cost. Moreover, classifiers are typically designed to maximize accuracy, which is not a good metric to evaluate effectiveness in the case of imbalanced training data. Therefore, we need to improve traditional algorithms so as to handle imbalanced data and choose other metrics to measure performance instead of accuracy. We focus our study on imbalanced datasets with binary classes.

Much work has been done in addressing the class imbalance problem. These methods can be grouped in two categories: the data perspective and the algorithm perspective [2]. The methods with the data perspective re-balance the class distribution by re-sampling the data space either randomly or deterministically. The main disadvantage

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 280–292, 2013.

of re-sampling techniques are that they may cause loss of important information or the model overfitting, since that they change the original data distribution.

A cost-sensitive classifier tries to learn more characteristics of samples with the minority class by setting a high cost to the misclassification of a minority class sample. It does not modify the data distribution. Weiss [3] left the questions "why doesn't the cost-sensitive learning algorithm perform better given the known drawbacks with sampling; and are there ways to improve the effectiveness of cost-sensitive learning algorithms." We need to improve the effectiveness of cost sensitive learning algorithms by optimizing factors which influence the performance of cost sensitive learning.

There are two challenges with respect to the training of cost sensitive classifier. The misclassification costs play a crucial role in the construction of a cost sensitive learning model for achieving expected classification results. However, in many contexts of imbalanced dataset, the misclassification costs cannot be determined. Beside the cost, the feature set and intrinsic parameters of some sophisticated classifiers also influence the classification performance. Moreover, these factors influence each other. This is the first challenge. The other is the gap between the measure of evaluation and the objective of training on the imbalanced data [4]. Indeed, for evaluating the performance of a cost-sensitive classifier on a skewed data set, the overall accuracy is irrelevant. It is common to employ other evaluation measures to monitor the balanced classification ability, such as G-mean [5] and AUC [6]. However, these cost-sensitive classifiers measured by imbalanced evaluation are not trained and updated with the objective of the imbalanced evaluation. To achieve good prediction performance, learning algorithms should train classifiers by optimizing the concerned performance measures [7].

In order to solve the challenges above, we design a novel framework for training a cost sensitive classifier driven by the imbalanced evaluation criteria. The training scheme can bridge the gap between the training and the evaluating of cost sensitive learning, and it can learn the optimal factors associated with the cost sensitive classifier automatically. The significance of the scheme has two questions to fix: how to optimize these factors simultaneously; and using what evaluation criteria for guiding their optimization. These two issues are our key steps for improving the cost sensitive learning in the context of the class imbalance problem without cost information. Our main contributions in this paper are centered around the questions above.
The contributions of this work can be listed as follows:

1) Optimizing the factors (ratio misclassification cost, feature set and intrinsic parameters of classifier) simultaneously for improving the performance of cost-sensitive SVM.

2) Imbalanced data classification is commonly evaluated by measures such as G-mean and AUC instead of accuracy. However, for many classifiers, the learning process is still largely driven by error based objective functions. We use the measure directly to train the classifier and discover the optimal parameter, ratio cost and feature subset based on different evaluation functions like the G-mean or AUC. Different metrics can reflect different aspect performance of classifiers.

2 Related Works

The common methods to solve data imbalance are data re-sampling perspective and algorithm perspective. Re-sampling methods are attractive under most imbalanced circumstances. This is because re-sampling adjusts only the original training dataset, instead of modifying the learning algorithm; therefore it provides a convenient and effective way to deal with imbalanced learning problems using standard classifiers by balancing the instances of the classes. Weiss and Provost observed that the naturally occurring distribution is not always optimal [8]. Therefore, one needs to modify the original data distribution. The idea of sampling is to purposefully manipulate the class distributions by under-sampling and over-sampling.

The methods with the algorithm perspective adapt existing common classifier learning algorithms to bias towards the small class, such as cost-sensitive learning. Cost-sensitive learning is one of the most important topics in machine learning and data mining, and attracted significant attention in recent years. Cost-sensitive learning methods consider the costs associated with misclassifying examples. The objective of cost-sensitive methods is to minimize the expected cost of misclassifications without changing the class distribution [9]. A closely related idea to cost-sensitive learners is shifting the bias of a machine to favor the minority class so as to obtain better recognition ability by adjusting the costs associated with misclassification rather than to seek the minimum of total misclassification cost [4, 10-12]. In the construction of cost sensitive learning, the parameter of misclassification cost plays an indispensable role.

There is another issue in the class imbalance problem. The importance of feature selection to class imbalance problems, in particular, was realized and has attracted increasing attention from machine learning and data mining communities. Wrappers and embedded methods are feature subset selection methods that consider feature interaction in the selection process. Some authors have conducted studies on using feature selection to combat the class imbalance problem [13, 14]. Zheng and Srihari [14] suggest that existing measures used for feature selection are not appropriate for imbalanced datasets. The wrapper feature selection seems a good approach.

3 Cost-Sensitive SVM

Support Vector Machines (SVM), which has strong mathematical foundations based on statistical learning theory, has been successfully adopted in various classification applications. SVM maximizes a margin in a hyperplane separating classes. However, it is overwhelmed by the majority class instances in the case of imbalanced datasets because the objective of regular SVM is to maximize the accuracy. In order to provide different costs associated with the two different kinds of errors, cost-sensitive SVM (CS-SVM) [15] is a good solution. CS-SVM is formulated as follows:

$$\begin{aligned} &Min\ \frac{1}{2}\|\boldsymbol{w}\|^2 + C_+\sum\nolimits_{i:y_i=+1}\xi_i + C_-\sum\nolimits_{j:y_j=-1}\xi_j \\ &s.t.\ \ y_i[(\boldsymbol{w}^T\boldsymbol{x}_i)+b] \ge 1-\xi_i \quad i=1,\cdots,n \\ &\qquad \xi_i \ge 0 \end{aligned} \tag{1}$$

where the C_+ is the higher misclassification cost of the positive class, which is the primary interest, while C_- is the lower misclassification cost of the negative class. Using the different error cost for the positive and negative classes, the hyperplane could be pushed away from the positive instances. In this paper, we fix $C_- = C$ and $C_+ = C \times C_{rf}$, where C and C_{rf} are respectively the regularization parameter and the ratio misclassification cost factor. In the construction of cost sensitive SVM, the misclassification cost parameter plays an indispensable role. For the cost information, Veropoulos et al. have not suggested any guidelines for deciding what the relative ratios of the positive to negative cost factors should be.

In general, the Radial Basis Function (RBF kernel) is a reasonable first choice for the classification of the nonlinear datasets, as it has fewer parameters (γ).

4 Optimized Cost Sensitive SVM by Measure of Imbalanced Data

SVM tries to minimize the regularized hinge loss; it is driven by an error based objective function. However, the overall accuracy is not an appropriate evaluation measure for imbalanced data classification. As a result, there is an inevitable gap between the evaluation measure by which the classifier is to be evaluated and the objective function based on which the classifier is trained. The classifier for imbalanced data learning should be driven by more appropriate measures. We inject the appropriate measures into the objective function of the classifier in the training with PSO. The common evaluation for imbalanced data classification is G-mean and AUC. However, for many classifiers, the learning process is still driven by error based objective functions. In this paper we explicitly treat the measure itself as the objective function when training the cost sensitive learning. We designed a measure oriented training framework for dealing with imbalanced data classification issues. Chalwa et al. [6] propose a wrapper paradigm that discovers the amount of re-sampling for a dataset based on optimizing evaluation functions like the f-measure, and AUC. To date, there is no research about training the cost sensitive classifier with measure based objective functions. This is one important issue that hinders the performance of cost-sensitive learning.

Another important issue of applying the cost-sensitive learning algorithm to the imbalanced data is that the cost matrix is often unavailable for a problem domain. The misclassification cost, especially the ratio misclassification cost, plays a crucial role in the construction of a cost sensitive approach; the knowledge of misclassification costs is required for achieving expected classification result. However, the values of costs are commonly given by domain experts. They remain unknown in many domains where it is in fact difficult to specify the precise cost ratio information. It is not exact to set the cost ratio to the inverse of the imbalance ratio (the number of majority instances divided by the number of minority instances); especially it is not accurate for some classifier such as SVM. Some cost sensitive learning use a heuristic approach to search the optimal cost matrix, such as Genetic Algorithm [10] or grid search to find the optimal cost setup [12].

Apart from the ratio misclassification cost information, feature subset selection and the intrinsic parameters of the classifier have a significant bearing on the performance. Both factors are not only important for imbalanced data classification, but also for any classification. Feature selection is the technique of selecting a subset of discriminative features for building robust learning models by removing most irrelevant and redundant features from the data. Optimal feature selection can concurrently achieve good accuracy and dimensionality reduction. Unfortunately, the imbalanced data distributions are often accompanied by high dimensionality in real-world datasets such as text classification, bioinformatics, and computer aided detection. It is important to select features that can capture the high skew in the class distribution [1]. Moreover, proper intrinsic parameter setting of classifiers, such as regularization cost parameter and the kernel function parameter for SVM, can improve the classification performance. It is necessary to use the grid search to optimize the regulation parameter and kernel parameters. Moreover, these three factors influence each other. Therefore, obtaining the optimal ratio misclassification cost, feature subset and intrinsic parameters must occur simultaneously.

Based on the reasons above, our specific goal is to devise a strategy to automatically determine the optimal factors during training of the cost sensitive classifier oriented by the imbalanced evaluation criteria (G-mean and AUC).

In this paper, for the multivariable optimization, especially the hybrid multivariable, the best methods are swarm intelligence techniques. We choose the particle swarm optimization as our optimization method because it is mature and easy to implement. Particle swarm optimization (PSO) is a population-based global stochastic search method [16]. PSO optimizes an objective function by a population-based search. The population consists of potential solutions, named particles. These particles are randomly initialized and move across the multi-dimensional search space to find the best position according to an optimization function. During optimization, each particle adjusts its trajectory through the problem space based on the information about its previous best performance (personal best, *pbest*) and the best previous performance of its neighbors (global best, *gbest*). Eventually, all particles will gather around the point with the highest objective value.

The position of individual particles is updated as follows:

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{2}$$

With *v*, the velocity calculated as follows:

$$v_{id}^{t+1} = w \times v_{id}^t + c_1 \times r_1 \times (pbest_{id}^t - x_{id}^t) + c_2 \times r_2 \times (gbest^t - x_{id}^t) \tag{3}$$

Where v_i^t indicates velocity of particle *i* at iteration *t*; w indicates the inertia factor; C_1 and C_2 indicate the cognition and social learning rates, which determine the relative influence of the social and cognition components. r_1 and r_2 are uniformly distributed random numbers between 0 and 1, x_i^t is current position of particle *i* at iteration *t*, $pbest_i^t$ indicates best of particle *i* at iteration *t*, $gbest^t$ indicates the best of the group.

Evaluation measures play a crucial role in both assessing the classification performance and guiding the classifier modeling. The purpose of cost-sensitive learning is usually to build a model with total minimum misclassification costs. However, it should be based on the known cost matrix condition. The purpose of our cost sensitive learning is to get a best AUC or G-mean evaluation metric. We train the cost sensitive learning using performance measures as the objective functions directly. Through training the cost sensitive classifier with measure based objective functions, we can discover the best factors in terms of the different evaluation. The evaluation metrics value is taken as the fitness function to adjust the position of a particle. These two different evaluations reflect different aspect of the classifier. AUC affects the ranking ability and G-mean involves the accuracies of both classes at the same time.

For binary class classification, the cost parameter is only one parameter, which means the relative cost information, ratio misclassification cost factor C_{rf}. Since the RBF kernel is selected for the cost sensitive SVM, γ and C are the parameters to be optimized. We need to combine the discrete and continuous values in the solution representation since the costs and parameters we intend to optimize are continuous while the feature subset is discrete. Each feature is represented by a 1 or 0 for whether it is selected or not. The major difference between the discrete PSO [17] and the original version is that the velocities of the particles are rather defined in terms of probabilities that a bit will change to one. Using this definition a velocity must be restricted within the range [0, 1], to which all continuous values of velocity are mapped by a sigmoid function:

$$v'^{t}_{i} = sig(v^{t}_{i}) = \frac{1}{1+e^{-v^{t}_{i}}} \tag{4}$$

Equation 4 is used to update the velocity vector of the particle while the new position of the particle is obtained using Equation 5.

$$x^{t+1}_{i} = \begin{cases} 1 & if \quad r_i < v'^{t}_{i} \\ 0 & otherwise \end{cases} \tag{5}$$

Where r_i is a uniform random number in the range [0,1] .

Algorithm 1. MOCSSVM (optimized cost sensitive SVM by imbalanced data measure)

```
Input: Training set $D$; termination condition $T$; population size $SN$; metric $E$; $NumFolds$ =5
Randomly initialize particle population positions and velocities (including cost matrix, intrinsic parameters, and feature subset)
repeat
  foreach particle $i$
    Construct the $D_i$ with the feature selected by the particle $i$
    for $k$=1 to $NumFolds$
      Separate $D_i$ randomly into $Trt^k_i$ (80%) for training and $Trv^k_i$ (20%) for validation
      Train CS-SVM with cost matrix and intrinsic parameters optimized by the particle $i$ on the $Trt^k_i$
      Evaluate the cost sensitive classifier on the $Trv^k_i$ , and obtain the value $M^k_i$ based on $E$
    end for
    $M_i$=average($M^k_i$); Assign the fitness of particle $i$ with $M_i$
    if fitness ($pbest_i$) <= fitness ($x_i$)
        then $pbest_i = x_i$
    end if
  end foreach
  set gbest as best pbest
  foreach particle $i$
      update $velocity_i$ and $position_i$ with Eq. 2 and 3.
  end foreach
until termination condition
output optimal parameters, cost ratio and feature subset of gbest
```

The solution (i.e. particle) includes three parts: the ratio misclassification cost, the intrinsic parameters of classifier, and the feature subsets. Figure 1 illustrates the mixed solution representation in the PSO.

Ratio cost	Intrinsic parameters		Feature subset				
C_{rf}	C	γ	f_1	f_2	…	f_{n-1}	f_n

Fig. 1. Solution representation

The detailed algorithm MOCSSVM to optimize cost sensitive SVM by imbalanced data measure is shown in Algorithm 1. It is a wrapper framework for empirically discovering the potential misclassification cost ratio, feature subset, and intrinsic parameters for CSL oriented by the imbalanced evaluation criteria (G-mean and AUC).

5 Experimental Study

5.1 Dataset Description

To evaluate the classification performance of our proposed method in different classification tasks, and to compare with other methods specifically devised for imbalanced data, we tried several datasets from the UCI database. We used all available datasets from the combined sets used in [4]. This also ensures that we did not choose only the datasets on which our method performs better. The minority class label (+) is indicated in Table 1. The datasets chosen have diversity in the number of attributes and imbalance ratio. Moreover, the datasets used have both continuous and categorical attributes. All the experiments are conducted by 10-fold cross-validation.

Table 1. The data sets used for experimentation

The dataset name is appended with the label of the minority class (+)

Dataset (+)	Instances	Features	Class balance
Hepatitis (1)	155	19	1:4
Glass (7)	214	9	1:6
Segment (1)	2310	19	1:6
Anneal (5)	898	38	1:12
Soybean (12)	683	35	1:15
Sick (2)	3772	29	1:15
Car (3)	1728	6	1:24
Letter (26)	20000	16	1:26
Hypothyroid(3)	3772	29	1:39
Abalone (19)	4177	8	1:130

5.2 Experiment I

In this experiment, the comparison is conducted between our method and the intermediate method or basic method, such as basic SVM with and without the feature selection, cost sensitive SVM, cost sensitive SVM with grid search and our method MOCSSVM with/without the feature selection. For the basic SVM with feature selection, it is a common wrapper feature selection method with evaluation by classification performance. As for CSSVM, the misclassification cost ratio is searched iteratively to maximize the measure score within a range of cost value. CSSVM uses a grid search for optimization. We also need to treat this misclassification cost ratio as a hyperparameter, and locally optimize this parameter. However, it is not feasible to use a triple circulation for optimizing the best parameters, so we optimize the best parameter pair(C and γ) firstly, then locally optimize the cost ratio parameter based on the best parameter pair(C and γ). All SVM models in this experiment use the same kernel, RBF, and for basic SVM and CSSVM, the intrinsic parameters are fixed with default values (C=1 and γ =1).

For the PSO setting of our method MOCSSVM, the initial parameter values of it in our proposed method were set according to the conclusion drawn in [18]. The parameters were used: C_1=2.8, C_2=1.3, w=0.5. To empirically provide good performance while at the same time keeping the time complexity tractable, the particle number was set dynamically according to the amount of the variables optimized (=1.5×|variables to be optimized|), and the termination condition could be a certain number of iterations (500 cycles) or other convergence condition (no changes any more within 2× |variables to be optimized| cycles). Besides these parameters in PSO, the other parameters are the upper and lower of limit parameter of model to be optimized. For Grid-CSSVM and MOCSSVM, the ranges for C and γ are based on a grid search for SVM parameters as recommended in [19]. The range of C is (2^{-5}, 2^{15}), and the range of γ is (2^{-15}, 2^{3}). The range of ratio misclassification cost factor C_r was empirically set between 1 and 10×*ImbaRatio* (ratio between the instance amounts of two classes).

In this experiment, we assess the overall quality of classifiers with only the AUC evaluation metric. From the result in Table 2, we found that simultaneously optimizing the feature subset, parameter and cost ratio generally help the base classifiers learned on the different data sets, regardless of feature selecting or not.

Table 2. Experimental results between all the methods based on the SVM

Dataset	Basic SVM		CS-SVM	Grid-CSSVM	MOCSSVM	
	without *FS*	*FS*	without *FS*	without *FS*	without *FS*	*FS*
Hepatitis	0.632	0.714	0.707	0.801	**0.861**	0.855
Glass	0.952	0.957	0.953	0.955	0.994	**1**
Segment	1	1	1	1	1	1
Anneal	0.876	0.925	0.957	**1**	**1**	**1**
Soybean	1	1	1	1	1	1
Sick	0.728	0.761	0.788	0.848	0.908	**0.975**
Car	0.990	0.987	0.990	0.999	**1**	**1**
Letter	0.898	0.895	0.909	0.983	0.980	**0.999**
Hypothyrid	0.830	0.855	0.887	0.945	0.973	**0.988**
Abalone	0.638	0.712	0.722	0.839	0.867	0.893
Average	0.854	0.881	0.892	0.937	0.957	0.971

Under the condition where the feature selection is not carried out, we found that the simultaneous optimization for all the factors using PSO outperforms the optimization using grid search, which optimizes the intrinsic parameters first, then searches the optimal misclassification cost parameter based on the best intrinsic parameters.

It lacks many potential parameter pairs not searched in the parameter space. Hence, it shows that the parameters need to be search at the same time. Moreover, in MOCSSVM, the use of feature selection was found to improve the AUC for each dataset except the Hepatitis dataset.

Although, we take some dynamic strategies for improving the efficiency of the PSO algorithm, the average running iterations for PSO-based approach is slightly inferior to that of the grid search algorithm. However, it significantly improves the classification accuracy and obtains fewer input features for the classifiers. Therefore, we can draw the conclusion that by simultaneously optimizing the intrinsic, misclassification cost parameter and feature selection with the imbalanced evaluation measure guiding improves the classification performance of the cost sensitive SVM on different datasets.

5.3 Experiment II

The comparison is conducted between our method and the other state-of-the-art imbalanced data classifiers, such as the random under-sampling (RUS), SMOTE [20], SMOTEBoost [21], and SMOTE combined with asymmetric cost classifier [5]. For the under-sampling algorithm, the SMOTE and SMOTEBoost, the re-sampling rate is unknown. In our experiments, in order to compare equally, no matter under-sampling or over-sampling method, we also use the evaluation measure as the optimization objective of the re-sampling method to search the optimal re-sampling level. The increment step and the decrement step are both set at 10%. This is a greedy search, which process repeats, greedily, until no performance gains are observed. The optimal re-sampling rate is decided in an iterative fashion according to the evaluation metrics. Thus, in each fold, the training set is separated into training subset and validating subset for searching the appropriate rate parameters. The evaluation metrics are also used with the G-mean and AUC. For the CS-SVM with SMOTE, for each re-sampling rate searched, the optimal misclassification cost ratio is determined by searching under the evaluation measure guiding under the current over-sampling level of SMOTE.

As shown in bold in Table 3, our MOCSSVM outperforms all the other approaches on the great majority of datasets. It did not get the best result only on the Glass dataset. From the results, we can see that the random under-sampling has the worst performance. This is because it is possible to remove certain significant examples and under-sampling the majority class causes larger angles between the ideal and learned hyperplane, and also reduces the total number of training instances which also contributes to increasing angles [5]. Both the SMOTE and SMOTEBoost improve the classification on the imbalanced data. The over-sampling algorithm that tries to improve on it inevitably sacrifices some specificity in order to improve the sensitivity; but the degree of sensitivity improved is larger than the lost specificity. However, they have a potential disadvantage of distorting the class distribution. SMOTE combined with a different cost classifier is better than only SMOTE over-sampling, and it is the method that shares most of the second best results. In the majority of cases, the G-mean value from the G-mean wrapper is higher than the one of the AUC wrapper, but in some cases, the G-mean value from the AUC wrapper is higher, such as Hepatitis and Abalone datasets for MOCSSVM and Glass. Even for MOCSSVM, the average G-mean from AUC optimization is better than the one from G-mean optimization. From this, we believe that by using AUC as the wrapper evaluation function we get better performances, which is the similar conclusion as in [6]. We believe that employing the AUC evaluation measure as optimization objective could lead to more generalized performances. Similarly, the two evaluation metrics wrapper optimizations for the same classifier result in different misclassification cost, feature subset and intrinsic parameters, since they optimize different properties of the classifier.

The feature selection is as important as the re-sampling in the imbalanced data classification, especially with high dimensional datasets. However, feature selection is often ignored. Our method does feature selection in the wrapper paradigm, hence improves the classification performance on the datasets which have higher dimensionality, such as Anneal, Sick and Hypothyroid.

Table 3. Experimental comparison between MOCSSVM method and other imbalanced data methods

Dataset		**RUS**		**SMOTE**		**SMOTE Boost**		**SMOTE-CSSVM**		**MOCSSVM**	
		wrapper metric		**wrapper metric**		**wrapper metric**		**wrapper metric**		**wrapper metric**	
		AUC	**GM**	**AUC**	**GM**	**AUC**	**GM**	**AUC**	**GM**	**AUC**	**GM**
Hepatitis	**AUC**	0.663	0.528	0.754	0.721	0.788	0.759	0.813	0.783	**0.855**	0.823
	GM	0.598	0.487	0.672	0.667	0.558	0.592	0.628	0.729	**0.805**	0.801
	Fea.	19								7	8
Glass	**AUC**	0.955	0.948	0.988	0.986	0.981	0.978	0.992	0.975	**1**	0.995
	GM	0.817	0.803	0.844	0.858	0.874	0.862	0.965	**0.988**	0.986	0.971
	Fea.	9								5	4
Segment	**AUC**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
	GM	0.993	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0.998	**1**
	Fea.	19								10	11
Anneal	**AUC**	0.882	0.866	0.912	0.876	0.891	0.889	0.957	0.934	**1**	**1**
	GM	0.616	0.535	0.758	0.821	0.761	0.784	0.819	0.835	0.999	**1**
	Fea.	38								14	12
Soybean	**AUC**	**1**	0.992	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
	GM	0.876	0.953	0.947	0.965	0.992	0.997	**1**	0.997	**1**	**1**
	Fea.	35								12	12
Sick	**AUC**	0.784	0.742	0.822	0.799	0.841	0.824	0.931	0.874	**0.975**	0.954
	GM	0.206	0.141	0.452	0.528	0.508	0.512	0.811	0.825	0.893	**0.915**
	Fea.	29								9	7
Car	**AUC**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
	GM	0.964	0.964	0.962	0.958	0.979	0.981	0.995	**0.998**	0.996	**0.998**
	Fea.	6								4	4
Letter	**AUC**	0.907	0.896	0.966	0.956	0.987	0.965	0.988	0.980	**0.999**	0.995
	GM	0.925	0.933	0.947	0.954	0.934	0.922	0.965	0.961	0.983	**0.985**
	Fea.	16								12	10
Hypothyroid	**AUC**	0.876	0.843	0.971	0.915	0.967	0.955	0.973	0.971	0.988	**0.989**
	GM	0.482	0.612	0.853	0.894	0.876	0.903	0.876	0.901	0.964	**0.968**
	Fea.	29								9	14
Abalone	**AUC**	0.781	0.613	0.822	0.754	0.799	0.780	0.846	0.812	**0.893**	0.855
	GM	0.618	0.687	0.712	0.814	0.645	0.744	0.698	0.817	**0.853**	0.785
	Fea.	8								4	5
Average	**AUC**	0.885	0.843	0.924	0.900	0.925	0.915	0.950	0.933	**0.971**	0.961
	GM	0.710	0.711	0.815	0.814	0.813	0.830	0.876	0.910	**0.948**	0.943
win/tie/lose	**AUC**	0/3/7	0/2/8	0/3/7	0/3/7	0/3/7	0/3/7	0/3/7	0/3/7	base	1/4/5
	GM	0/0/10	0/1/9	0/1/9	0/1/9	0/1/9	0/1/9	0/2/8	1/2/7	3/1/6	base

We use the MOCSSVM method as a baseline and compare the other methods against it. Although all methods are optimized under the evaluation measure oriented, we can clearly see that MOCSSVM is almost always equal to, or better than other methods. What is most important is that our method does not change the data distribution, while the re-sampling may make the generalization not as good as the training, since that the data distribution are different between the training set and test set.

Many papers conclude that there is no consistent clear winner between the sampling approaches and the cost-sensitive technique. However, the conclusions were based on the default condition without sufficient search in the parameters space. In this paper, we have empirically shown that under the evaluation measure guiding, the performances of cost sensitive SVM with cost, feature subset and intrinsic parameter optimized are better than the re-sampling methods with sampling level optimized.

5.4 Experiment III

Computer aided detection provides a computer output in order to assist radiologists in the diagnosis of Lung Cancer on medical images. It can be divided into initial nodule identification step and false-positive reduction step. The purpose of false-positive reduction is to remove false positives (FPs) as much as possible while retaining a relatively high sensitivity. It is a typical class imbalance issue since the two classes are typically skewed and have unequal misclassification costs. Our database consists of 98 thin section CT scans with 106 solid nodules, obtained from Guangzhou hospital in China. We obtained the appropriate candidate nodule samples objectively using a candidate nodule detection algorithm, which identifies 95 true nodules as positive class and 592 non-nodules as negative class from the total CT scans; the class imbalance ratio is 1:6. The imbalance level is not extremely high, but the misclassification costs of each class are very different. The imbalance level is dependent on reliability and accuracy of the initial detection process. Our feature extraction process generated 43 features from multiple views. Using these features, we construct the input space for our classifiers. Our method outperforms the other common approach (Table 4). It means that our method can be applied on the nodule or other lesion detection. The measure optimization used is the AUC metric.

Table 4. Experiment result of candidate nodule classification

metric	SVM	CSSVM	RUS	SMOTE	SMOTE-Boost	SMOTE-CSSVM	MO CSSVM
AUC	0.681	0.785	0.603	0.948	0.948	0.956	**0.969**
GM	0.208	0.662	0.590	0.826	0.818	0.867	**0.937**

6 Conclusion

Learning with class imbalance is a challenging task. We propose a wrapper paradigm oriented by the evaluation measure of imbalanced dataset as objective function with respect to misclassification cost, feature subset and intrinsic parameters of SVM. Our measure oriented framework could wrap around an existing cost-sensitive classifier. The proposed method has been validated on some benchmark imbalanced data and real application. The experimental results presented in this study have demonstrated that the proposed framework provides a very competitive solution to other existing state-of-the-arts methods, in optimization of G-mean and AUC for combating imbalanced classification problems. These results confirm the advantages of our approach, showing the promising perspective and new understanding of cost sensitive learning. In the future research, we will extend the framework to the imbalanced multiclass data classification.

References

1. Chawla, N.V., Japkowicz, N., Kolcz, A.: Editorial: special issue on learning from imbalanced data sets. SIGKDD Explorations Special Issue on Learning from Imbalanced Datasets 6(1), 1–6 (2004)
2. Kotsiantis, S., Kanellopoulos, D., Pintelas, P.: Handling imbalanced datasets: A review. GESTS International Transactions on Computer Science and Engineering, 25–36 (2006)
3. Weiss, G., McCarthy, K., Zabar, B.: Cost-sensitive learning vs. sampling: Which is Best for Handling Unbalanced Classes with Unequal Error Costs? In: IEEE ICDM, pp. 35–41 (2007)
4. Yuan, B., Liu, W.H.: A Measure Oriented Training Scheme for Imbalanced Classification Problems. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining Workshop on Biologically Inspired Techniques for Data Mining, pp. 293–303 (2011)
5. Akbani, R., Kwek, S., Japkowicz, N.: Applying support vector machines to imbalanced datasets. In: European Conference on Machine Learning (2004)
6. Chawla, N.V., Cieslak, D.A., Hall, L.O., Joshi, A.: Automatically countering imbalance and its empirical relationship to cost. Utility-Based Data Mining: A Special issue of the International Journal Data Mining and Knowledge Discovery (2008)
7. Li, N., Tsang, I., Zhou, Z.: Efficient Optimization of Performance Measures by Classifier Adaptation. IEEE Transactions on Pattern Analysis and Machine Intelligence PP(99), 1 (2012)
8. Weiss, G., Provost, F.: Learning when training data are costly: the effect of class distribution on tree induction. J. Artif. Intel. Res., 19:315–19:354 (2003)
9. Zhou, Z.H., Liu, X.Y.: Training Cost-Sensitive Neural Networks with Methods Addressing the Class Imbalance Problem. IEEE Transactions on Knowledge and Data Engineering 18(1), 63–77 (2006)
10. Sun, Y., Kamel, M.S., Wang, Y.: Boosting for Learning Multiple Classes with Imbalanced Class Distribution. In: Proc. Int'l Conf. Data Mining, pp. 592–602 (2006)
11. Wang, B.X., Japkowicz, N.: Boosting support vector machines for imbalanced data sets. Journal of Knowledge and Information Systems 4994, 38–47 (2008)
12. Thai-Nghe, N.: Cost-Sensitive Learning Methods for Imbalanced Data. In: Intl. Joint Conf. on Neural Networks (2010)
13. Forman, G.: An Extensive Empirical Study of Feature Selection Metrics for Text Classification. J. Machine Learning Research 3, 1289–1305 (2003)
14. Zheng, Z., Wu, X., Srihari, R.: Feature selection for text categorization on imbalanced data. SIGKDD Explorations 6(1), 80–89 (2004)

15. Veropoulos, K., Campbell, C., Cristianini, N.: Controlling the sensitivity of support vector machines. In: International Joint Conference on AI, pp. 55–60 (1999)
16. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: IEEE Int. Conf. Neural Networks, pp. 1942–1948 (1995)
17. Khanesar, M.A., Teshnehlab, M., Shoorehdeli, M.A.: A novel binary particle swarm optimization. In: Mediterranean Conference on Control & Automation, pp. 1–6 (2007)
18. Carlisle, A., Dozier, G.: An Off-The-Shelf PSO. In: PSO Workshop, pp. 1–6 (2001)
19. Hsu, C.W., Chang, C.C., Lin, C.J.: A Practical Guide to Support vector Classification, National Taiwan UniversityTechnical Report (2003)
20. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: Synthetic minority over-sampling technique. J. Artif. Intell. Res. 16, 321–357 (2002)
21. Chawla, N.V., Lazarevic, A., Hall, L.O., Bowyer, K.W.: SMOTEBoost: Improving Prediction of the Minority Class in Boosting. In: Lavrač, N., Gamberger, D., Todorovski, L., Blockeel, H. (eds.) PKDD 2003. LNCS (LNAI), vol. 2838, pp. 107–119. Springer, Heidelberg (2003)

A Positive-biased Nearest Neighbour Algorithm for Imbalanced Classification

Xiuzhen Zhang and Yuxuan Li[*]

School of Computer Science and IT, RMIT University
GPO Box 2476, Melbourne 3001, Australia
{xiuzhen.zhang}@rmit.edu.au, yuxli@csse.unimelb.edu.au

Abstract. The k nearest neighbour (kNN) algorithm classifies a query instance to the most frequent class among its k nearest neighbours in the training instance space. For imbalanced class distribution where positive training instances are rare, a query instance is often overwhelmed by negative instances in its neighbourhood and likely to be classified to the negative majority class. In this paper we propose a Positive-biased Nearest Neighbour (PNN) algorithm, where the local neighbourhood of query instances is dynamically formed and classification decision is carefully adjusted based on class distribution in the local neighbourhood. Extensive experiments on real-world imbalanced datasets show that PNN has good performance for imbalanced classification. PNN often outperforms recent kNN-based imbalanced classification algorithms while significantly reducing their extra computation cost.

Keywords: imbalanced classification, nearest neighbour classification, kNN.

1 Introduction

The k nearest neighbour (kNN) algorithm [1,2,6] applies a simple and intuitive rule to make classification decisions: instances close in the input space are likely to belong to the same class. Typically a kNN classifier classifies a query instance to the class that appears most frequently among its k nearest neighbours, where k is a parameter tuning the classification performance. In contrast to the maximum-generality bias of most concept learning systems [22], kNN adopts the maximum-specificity bias for classification and does not formulate a generalised conceptual model from the training instances at the training stage.

In many applications, training instances for a class form several clusters in the training instance space. With most concept learning systems (e.g. the decision tree), the model for classification usually is disjunction of several component subconcepts, where each disjunctive subconcept describes a cluster of training instances called a disjunct [9,23]. Small disjuncts refer to clusters of a small number of instances. The class imbalance problem often presents itself as a small disjunct

[*] Currently with The University of Melbourne.

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 293–304, 2013.

problem, where the positive minority class comprises small disjuncts [10]. In our discussions we call the minority class the positive class and the majority class the negative class.

Re-sampling and cost-sensitive learning are common strategies to combat class imbalance [22]. Our experiments show that however, unlike decision trees, re-sampling and cost-sensitive learning do not significantly improve the performance of kNN for imbalanced classification. This may be partly explained by that kNN makes classification decision by examining the local neighbourhood of query instances while re-sampling and cost-sensitive learning are global strategies. Although re-sampling can achieve overall even class distribution in the training instance space, it may not have significant effect on the local neighbourhood of every instance. As a result, given a query instance, its neighbourhood is likely overwhelmed by instances from the majority class, and as a result the instance is more likely to be classified to the majority class.

To improve the performance of kNN for imbalanced classification, we propose a positive-biased nearest neighbour (PNN) algorithm to prudently formulate positive subconcepts from small disjuncts of positive training instances, so as to increase the sensitivity of kNN to the positive class while not introducing too many false positives. Given a query instance and parameter k, if positive instances are scarce in the *local neighbourhood* of the query instance, we enlarge the neighbourhood for classification decision. Moreover we estimate the probability that a query instance belongs to the positive class based on comparing the positive frequency in its local neighbourhood with the overall positive frequency in the training instance space; intuitively without any prior knowledge of class prior, any query instance has a 50% probability of being positive, and query instances falling into regions with higher positive frequency than the overall positive frequency in the training space are more likely, i.e., with $> 50\%$ probability, to be positive.

Our experiments show that the simple yet effective decision bias of PNN leads to more accurate decision for the minority class. PNN often improves ENN [12] and CCW-kNN [13], two recent imbalanced classifiers based on kNN, while significantly reducing their computation cost. PNN also outperforms re-sampling and cost-sensitive learning strategies, namely SMOTE [5] and MetaCost [7], for imbalanced classification. Our work highlights that learning generalised concepts for disjuncts for the rare class is an effective approach to improving the performance of the nearest neighbour algorithm for imbalanced classification.

2 Related Work

The nearest neighbour algorithm has been advocated for imbalanced classification [9,19,20]. However the standard kNN algorithm experiences difficulty in the presence of imbalanced class distribution. Recently ENN [12] and CCW-kNN were proposed to improve kNN for imbalanced classification. However both ENN and CCW-kNN require a training stage either to find exemplar training samples to enlarge the decision boundaries for the positive class, or to learn the class weight for each training sample by mixture modelling and Bayesian network learning. The computation cost can be substantial with both approaches. In this paper we focus on improving the classification strategy of kNN. We apply a simple yet powerful strategy to estimate the positive posterior probability from the class distribution in the neighbourhood of query instances. Our experiments show that our new classification strategy, further improves the classification performance of ENN significantly, and show comparable results with CCW-kNN (better but not statistically significant).

There have been research efforts trying to improve the overall classification accuracy of kNN, such as intelligent instance selection and construction [8] and new classification scheme rather than the standard majority vote from k nearest neibhours [21]. There have also been research efforts trying to improve the classification efficiency [1,2,24] of kNN. Various strategies have been proposed to avoid exhaustive search of all training instances in the input space while ensuring accurate classification. But these approaches do not consider the classification strategy for class imbalance.

Several studies [9,19] in literature have tried to choose a more appropriate induction bias for learning algorithms to deal with the imbalanced class distribution problem, but these work are focused on how to make the generality-oriented induction bias of classification systems like the decision tree more specific so as to improve their performance for the rare class.

Re-sampling and cost-sensitive learning are commonly used strategies in literature to combat imbalanced class distribution for classification. No consistent conclusions have been drawn from existing studies [22] on the effectiveness of re-sampling techniques on imbalanced classification. Assigning higher cost to false negatives than to false positives can make a classification model more sensitive to the rare class. However the specific cost information is not always available in most applications. Our experiments show that such strategies can improve the performance of the C4.5 decision tree for imbalanced learning but do not work on kNN. This may be partly explained by the maximum-specificity induction bias of kNN – classification decision is made by examining the local neighbourhood of query instances, and therefore the global re-sampling and cost-adjustment strategies may not have pronounced effect in the local neighbourhood under examination.

3 Positive-biased Nearest Neighbour Classification

The ideal neighbourhood for classifying a query instance should be local to the query instance as well as general enough to form generalised concept for classification. A too large neighbourhood may over-generalise the subconcept for the positive class and introduce false positives whereas a too small neighbourhood may form a very restrictive subconcept and miss positive subconcepts. Our main idea to adjust the neighbourhood for making classification decisions is based on the concept of Positive Nearest Neighbour (PNN) region of query instances. An m-PNN region of a query instance t is such that it contains m positive nearest

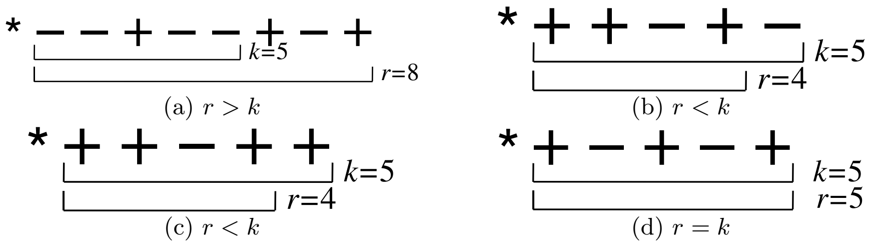


Fig. 1. For a query instance (denoted as * on the leftmost) and k=5, the 3-PNN region for the query instance. Instances are listed from left to right in its distance to the query instance.

neighbours of t. The m-PNN region for query instances means a varying number of nearest neighbours for making classification decisions — a small region for a subspace densely populated with positives, and a large region otherwise.

3.1 Positive Nearest Neighbours

With standard kNN classification algorithm, the k-nearest neighbour (k-NN) region for a query instance t may not contain any positive instances, especially when positive instances are scarce; the majority vote rule will compute $P(C_+|t) << 0.5$ and thus classify t to the negative class. Given query instance t and parameter k, to increase classification sensitivity to the positive class, we adjust the neighbourhood for classifying t so that it contains $\lceil k/2 \rceil$ positive nearest neighbours of t — namely the $\lceil k/2 \rceil$-PNN region of t. The total number of neighbours r in the $\lceil k/2 \rceil$-PNN region of t may be different from k: if the local region of t is densely populated with positive training instances, r is likely to be smaller than k. Otherwise r is likely to be larger than k.

Fig. 1 shows that for $k = 5$ (and therefore $\lceil k/2 \rceil = 3$) and a given query instance t (denoted as *), the different cases for the 3-positive nearest neighbour (3-PNN) region, in comparison to the corresponding 5-nearest neighbour (5-NN) region. The diagrams are in one dimension so that it is easier to explain the distance between instances and size of different regions. Depending on the distribution of positive instances in the neighbourhood of t, the total number of neighbours r in the 3-PNN region of the query instance may be larger or smaller than $k(= 5)$.

- Fig. 1(a) shows the most commonly occurring situation where positive instances are scarce. As positive instances are very rare in the neighbourhood of t, and very likely contain less than $\lceil k/2 \rceil$ positive instances, we have to expand to find a larger neighbourhood that contains $\lceil k/2 \rceil$ positive nearest neighbours. The radius of the region for t is 8. That is, $r = 8$ while $k = 5$.
- Fig. 1(b) and Fig. 1(c) show the case when the $\lceil k/2 \rceil$-PNN region of t is smaller than the k-NN region for t, even though it rarely happens in the presence of class imbalance. In this case, that $\lceil k/2 \rceil$ positive nearest

neighbours are observed in a smaller region centered at t suggests strongly that there is strong probability that t is positive.

- Fig. 1(d) shows the case when the k-NN region is the same as the $\lceil k/2 \rceil$-PNN region for a query instance t. In this case, distribution of classes in the $\lceil k/2 \rceil$-PNN region is the same as that in the k-NN region.

3.2 Estimating the Positive Posterior Probability

The region centered at a positive training instance likely forms a positive subconcept that is a component for the overall disjunctive positive concept. Ideally these positive subconcepts collectively should expand the decision boundary for the positive class. Roughly speaking the error rate for a positive subconcept region is the frequency of negative training instances in the region. However, the observed negative frequency is not accurate description of its error rate in independent tests. We estimate the false positive error rate by re-adjusting the observed negative frequency using pessimistic estimate [17,25]. The errors in a region follow the binomial distribution $B(N, q)$, where N is the total number of training instances in a region, and q is the true false positive error rate in the region. For a given confidence level c, the false positive error rate for a region can be estimated from the observed negative frequency f in the region as follows [25]:

$$q \approx \frac{f + z^2/2N + z\sqrt{f(1-f)/N + z^2/4N^2}}{1 + z^2/N} \tag{1}$$

where z is the *z-score* corresponding to a given confidence level, where for $c =$ 10% $z = 1.28$.

The false positive rate estimated from Equation 1 is always higher than the observed false positive frequency. A higher confidence level means the estimated false positive error rate closer to the observed negative frequency in the training instance space. *If the estimated false positive rate for a region is less than that estimated from the global negative class frequency in the training instance space, the region can form a positive subconcept.*

For a query instance t and a given value of k, when making classification decision for t, the $\lceil k/2 \rceil$-PNN and the region $S(t, r)$ for t are evaluated to decide if it can form a positive subconcept, where r is the total number of nearest neighbours for t in the $\lceil k/2 \rceil$-PNN region.[1] If the $\lceil k/2 \rceil$ region of t forms a positive subconcept, t is likely positive, that is $P(C_+|t) > P(C_-|t)$; otherwise t is likely negative, or $P(C_+|t) < P(C_-|t)$.

An important remaining question is how to estimate the positive posterior probability $P(C_+|t)$. We compute $P(C_+|t)$ based on the distribution of positives in the neighbourhood of t. Recall that we always have $\lceil k/2 \rceil$ positives in the $\lceil k/2 \rceil$-PNN region of t. Specifically we compute the positive posterior probability for t according to whether $r > k$ — that is whether $S(t, r)$ is a larger region than $S(t, k)$.

[1] For ease of discussion we overload r to represent the r^{th} nearest neighbour of t as well as its distance to t.

- If $r > k$, the neighbourhood of t lacks positive instances, and we need to adjust the computation of $P(C_+|t)$ so that classification is more sensitive to the positive. Especially if $S(t,r)$ is a positive subconcept, t is more likely to be positive and $P(C_+|t) = \lceil \frac{k}{2} \rceil / k$, which is greater than 0.5; otherwise $P(C_+|t)$ is estimated according to the positive frequency in the region $S(t,r)$, which is very likely less than 0.5.
- If $r \leq k$, $S(t,r)$ is a smaller neighbourhood of t and is densely populated with positives. In this case the positive frequency of $S(t,r)$ is used to estimate $P(C_+|t)$.

3.3 The Algorithm

We now present our PNN algorithm as shown in Algorithm 1, and for a given k it is denoted kPNN. In the algorithm first error rate threshold δ is computed using Equation 1 for confidence level c, number of training instances $|T|$ and the prior negative class frequency (line 1). Lines 3–11 compute $S(t,r)$, the $\lceil k/2 \rceil$-PNN neighbourhood of t, where $r = p' + n'$. Lines 14 and 15 describe that when $r > k$ and $S(t,r)$ is a positive subconcept, t is likely to be positive. So $P(C_+|t)=\lceil \frac{k}{2} \rceil / k$, the minimal probability > 0.5. Otherwise when $r > k$ but $S(t,r)$ is not a positive subconcept, or when $r < k$, $P(C_+|t)$ is computed based on the positive frequency in the region $S(t,r)$.

Line 4 involves a process sorting training instances by distance to the query instance. The sorting algorithm has a complexity of $O(n \log n)$, where n is the size of the training set. The loop from Line 4 to Line 11 repeats at most n times. Therefore, the time complexity of Algorithm 1 is $O(n \log n)$. Note the kPNN does not have a separate training stage searching the training space to compute exemplars as in ENN [12] or to learn the class weight for each training sample as in CCW-kNN [13]. So obviously kPNN can save the significant amount of training computation cost involved in both approaches.

Example 1. Given $k = 5$ (and so $\lceil k/2 \rceil = 3$), we use the examples in Fig. 1 to explain the classification process of PNN. In the figure, "*" denotes the query instance, and the size for the 3-PNN neighbourhood is represented as r =8, 4, 5 respectively. Let p and n denote the number of positive and negative instances in the k-NN neighbourhood of query instance "*" $S(*, 5)$. Let p' and n' denote the number of positive and negative instances in the $\lceil k/2 \rceil$-PNN neighbourhood of "*", $S(*, r)$ (r=8,4,5).

- $r > k$. Fig. 1(a) gives a scenario where $r = 8$. In the 3-PNN region of $S(*, 8)$, $p' = 3$, $n' = 5$. In the 5-NN region of $S(*, 5)$, $p = 1$, $n = 4$.

$$P(C_+|*) = 3/5, P(C_-|*) = 2/5, \text{ if S(*,8) is a positive subconcept;}$$
$$P(C_+|*) = 3/8, P(C_-|*) = 5/8, \text{ otherwise.}$$

- $r < k$. In Fig. 1(b) and Figure 1(c), $r = 4$ whereas $k = 5$. In Fig. 1(b), In the 3-PNN region of $S(*, 4)$, $p' = 3$, $n' = 1$, whereas in the 5-NN region of

Algorithm 1 The kPNN classification algorithm

```
Input:
    a) Training set $T$ ($|T|$ is the number of instances in $T$);
    b) Parameter $k$ and query instance $t$.
    c) Confidence level $c$
Output:
    ($P(C_+|t)$ and $P(C_-|t)$)
$\delta \leftarrow$ global error rate threshold by Equation 1 from $c$, $|T|$, and overall negative frequency
$G \leftarrow$ neighbours of $t$ in $T$ in increasing order of distance to $t$
$p' \leftarrow 0$; $n' \leftarrow 0$; $i \leftarrow 0$
while $i < |G|$ and $p' < \lceil k/2 \rceil$ do
    if $G[i]$ is a positive instance then
        $p' \leftarrow p' + 1$
    else
        $n' \leftarrow n' + 1$
    end if
    $i \leftarrow i + 1$
end while
$r \leftarrow n' + p'$
$e \leftarrow$ the FP rate by Equation 1 from $c$, $r$ and $\frac{n'}{r}$
if $r > k$ and $e \leq \delta$ then
    $P(C_+|t) \leftarrow \frac{\lceil k/2 \rceil}{k}$        {;; $P(C_-|t) = 1 - P(C_+|t)$. same for the other case}
else
    $P(C_+|t) \leftarrow$ positive frequency in the region $S(t, r)$
end if
```

$S(*, 5)$, $p = 3$, $n = 2$. In Fig. 1(c), in the 3-PNN region of $S(*, 4)$, $p' = 3$, $n' = 1$, whereas in the 5-NN region of $S(*, 5)$, $p = 4$, $n = 1$. In both cases

$$P(C_+|*) = 3/4; P(C_-|*) = 1/4.$$

- $r = k$. An example for the last case is shown in Fig. 1(d), where $r = 5$. The 3-PNN and 5-NN regions are the same, where $p' = p = 3$ and $n' = n = 3$.

$$P(C_+|*) = 3/5;\ P(C_-|*) = 2/5.$$

In this case the decision of kPNN is the same as that of kNN.

4 Experiments

We conducted 10-fold cross validation experiments to evaluate the performance of PNN in comparison to ENN, CCW-kNN. We also compare PNN against the SMOTE re-sampling [5] and MetaCost [7] cost-sensitive learning strategies for imbalanced classification. All classifiers were developed based on the WEKA data mining toolkit [25]. With all kNN-based classifiers k=3. The confidence levels for ENN and PNN are set to 10% and 20% respectively. Twelve real-world datasets

Table 1. The 12 real-world experimental datasets, ordered in decreasing level of imbalance

Dataset	size	#attr (num, symb)	classes (pos, neg)	minority (%)
Oil	937	47(47, 0)	(true, false)	4.38%
Hypo-thyroid	3163	25 (7, 18)	(true, false)	4.77%
PC1	1109	21 (21,0)	(true, false)	6.94%
Glass	214	9 (9,0)	(3, other)	7.94%
Satimage	6435	36 (36,0)	(4, other)	9.73%
CM1	498	21 (21,0)	(true, false)	9.84%
New-thyroid	215	5 (5,0)	(3, other)	13.95%
KC1	2109	21 (21,0)	(true, false)	15.46%
SPECT_F	267	44 (44,0)	(0, 1)	20.60%
Hepatitis	155	19 (6,13)	(1, 2)	20.65%
Vehicle	846	18 (18,0)	(van, other)	23.52%
German	1000	20 (7,13)	(2, 1)	30.00%

were used to evaluate the performance of classifiers in our experiments, from highly imbalanced (the minority frequency of 4.35%) to moderately imbalanced (the minority frequency of 30.00%). The datasets are summarised in Table 1, ordered in decreasing level of imbalance. The Oil dataset was provided by Robert Holte [11]. Datasets CM1, KC1 and PC1(`http://mdp.ivv.nasa.gov/index.html`) have been widely used in software engineering research to predict software defects [14]. The other datasets were compiled from the UCI Machine Learning Repository [3] by choosing one class as the positive and combining the remaining classes as the negative.

4.1 Performance Evaluation Using AUC and Convex Hull Analysis

We use both Receiver Operating Characteristic (ROC) curve [18] and Convex Hull analysis to evaluate the performance of classification algorithms. Area Under the ROC Curve (AUC) measures the overall classification performance [4], and a perfect classifier has an AUC of 1.0. All results reported next were obtained from 10-fold cross validation experiments and two-tailed t-tests at 95% confidence level were used to test the statistical difference between results. The ROC convex hull method provides visual performance analysis of classification algorithms at different levels of sensitivity [15,16].

Table 2 shows the AUC results for all models. CCW-3NN uses the multiplicative inverse strategy (additive inverse shows similar result). Compared with the remaining models, 3PNN has the highest average AUC of 0.841. 3PNN significantly outperforms 3ENN ($p < 0.05$) and shows statistically comparable ($p > 0.05$) result with CCW-3NN, despite a higher average AUC. 3PNN, 3ENN and CCW-3NN significantly outperform all of 3NN, 3NNSmt+, 3NN-Meta, C4.5Smt+ and C4.5Meta, This result confirms that our positive concept generalization strategy is very effective for improving the performance of kNN for imbalanced classification, and furthermore the strategy is more effective

Table 2. The AUC results for *k*PNN, in comparison with other approaches. Smt+ denotes the SMOTE [5] oversampling combined with under sampling strategy and Meta denotes the MetaCost [7] cost-sensitive learning strategy. The best result for each dataset is in bold. AUCs with difference <0.005 are considered equivalent.

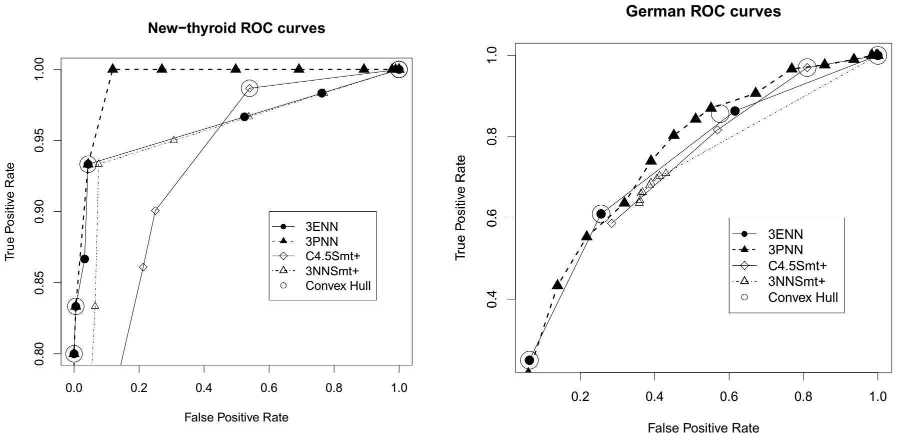
Dataset	3PNN	3ENN	CCW-3NN	3NN	3NNSmt+	3NNMeta	C4.5	C4.5Smt+	C4.5Meta
Oil	**0.847**	0.811	0.829	0.796	0.797	0.772	0.685	0.771	0.764
Hypo-thyroid	0.935	0.846	0.896	0.849	0.901	0.846	0.924	**0.948**	0.937
PC1	**0.846**	0.806	**0.845**	0.756	0.755	0.796	0.789	0.728	0.76
Glass	0.707	0.749	0.647	0.645	0.707	0.659	0.696	0.69	**0.754**
Satimage	**0.957**	0.934	0.839	0.918	0.902	0.928	0.767	0.796	0.765
CM1	0.726	0.681	**0.746**	0.637	0.666	0.625	0.607	0.666	0.668
New-thyroid	**0.988**	**0.99**	0.985	0.939	0.972	0.962	0.927	0.935	0.931
KC1	0.774	0.794	0.732	**0.815**	0.756	0.779	0.64	0.709	0.695
SPECT_F	0.749	0.767	**0.788**	0.72	0.725	0.735	0.626	0.724	0.643
Hepatitis	**0.841**	0.783	0.739	0.758	0.772	0.744	0.753	0.713	0.745
Vehicle	**0.983**	0.952	0.976	0.969	0.942	0.956	0.921	0.926	0.929
German	**0.737**	0.714	**0.739**	0.69	0.686	0.705	0.608	0.649	0.606
Average	**0.841**	0.818	0.828	0.786	0.798	0.792	0.745	0.771	0.766

than re-sampling and cost-sensitive learning strategies. Note also that SMOTE resampling and MetaCost demonstrate improvement on C4.5 (C4.5Smt+ and C4.5Meta vs. C4.5) but they do not demonstrate significant improvement on 3NN (3NNSmt+ and 3NNMeta vs. 3NN).

The New-thyroid dataset has a relatively high level of imbalance of 13.95%. From Table 2, 3PNN and 3ENN have an AUC result of 0.988 and 0.99, while 3NNSmt+ also has a competitive AUC result of 0.972. But as shown in Fig. 2(a), the ROC curves of the three models show very different trends. Notably more points of 3PNN and 3ENN (note their overlapping points on ROC curves) lie on the convex hull at low FP rates (<10%). On the other hand more points of 3NNSmt+ lie on the convex hull at high FP rates (>50%). It is desirable in many applications to achieve accurate prediction at low false positive rate and so 3PNN and 3ENN are obviously good choices for this purpose. German has a moderate imbalance level of 30%. ROC curves of the four models demonstrate similar trends on German, as shown in Fig. 2(b). Still at low FP rates, more points from 3PNN and 3ENN lie on the ROC convex hull, which again shows that 3PNN and 3ENN are strong models. The convex hull analysis has confirmed again that the positive concept generalisation strategy is very effective for imbalanced classification.

4.2 The Impact of Confidence Level

As discussed in Section 3.2, the confidence level affects the decision in PNN of whether to generalise to a positive subconcept. We applied 3PNN to two highly imbalanced datasets (Oil and Glass) and two moderately imbalanced datasets (KC1 and German) with confidence level from 1% to 50%. The AUC results are shown in Fig. 3. For the two datasets with high imbalance (Oil 4.38% and Glass 7.94%) AUC is positively correlated with confidence level. For example on Oil



(a)New-thyroid (b)German

Fig. 2. The convex hull analysis of 3PNN

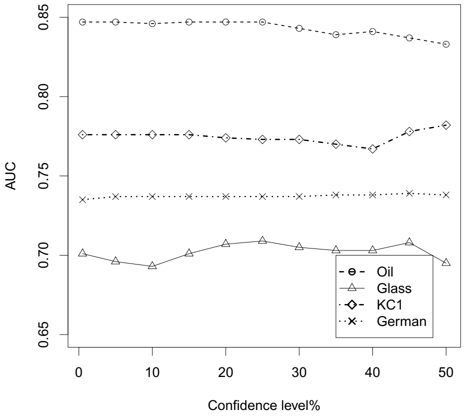


Fig. 3. The AUC results of 3PNN with varying confidence levels on four real-world datasets

when the confidence level increases from 1% to 50% the AUC decreases from 0.847 to 0.833. However for the two datasets with moderate imbalance (KC1 15.46% and German 30.00%) AUC is inversely correlated with confidence level. On German when confidence level increases from 1% to 50% AUC increases moderately.

The opposite behaviour of AUC in relation to confidence level may be explained by that on highly imbalanced data, to predict more positive instances, it is desirable to tolerate more negative samples in the training instance space in forming positive subconcepts, which is achieved by setting a low confidence level. Such an aggressive strategy increases the sensitivity of PNN to the positive class. On less imbalanced datasets where there are relatively sufficient positive instances, a high confidence level is desired to ensure a low error level in positive subconcepts.

5 Discussions and Conclusion

With the standard kNN classification strategy, the class of a query instance is decided by the majority class among its k nearest neighbours. In the presence of class imbalance, a query instance is often classified to the majority class and as a result many minority class instances are misclassified. Our experiments show that existing popular re-sampling and cost-sensitive learning strategies to combat imbalance can not produce significant improvement on the performance of the kNN algorithm.

We have proposed a Positive-biased Nearest Neighbour (PNN) algorithm to combat imbalanced class distribution at the classification stage. Generalised positive subconcepts are formulated to improve kNN induction for imbalanced classification, based on adjusting the positive posterior probability estimation via comparing the positive frequency in the local region of a query instance to the overall positive frequency in the training instance space. The size of local regions of query instances (the setting of k) and confidence level for forming generalised positive subconcepts are parameters for the PNN algorithm. Generally setting these parameters of PNN for optimal performance in different applications requires empirical experiments.

Extensive experiments on real-world imbalanced datasets have shown that PNN significantly improves the performance of kNN for imbalanced classification. PNN also outperforms popular re-sampling and cost-sensitive learning strategies for the class imbalance problem. Compared with recent improvements to the kNN algorithm for imbalanced classification, PNN significantly reduces computation while often improving classification accuracy.

Our study highlights that adjusting the induction bias of classification algorithms in general and maximum-specificity induction algorithms in particular is a cost-effective strategy to improve classification performance for the imbalanced class distribution problem.

References

1. Aha, D.W. (ed.): Lazy learning. Kluwer Academic Publishers, Norwell (1997)
2. Aha, D.W., Kibler, D.F., Albert, M.K.: Instance-based learning algorithms. Machine Learning 6, 37–66 (1991)
3. Asuncion, A., Newman, D.: UCI machine learning repository (2007), `http://www.ics.uci.edu/~mlearn/`
4. Bradley, A.P.: The use of the area under the roc curve in the evaluation of machine learning algorithms. Pattern Recognition 30, 1145–1159 (1997)
5. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: Synthetic minority over-sampling technique. Journal of Artificial Intelligence Research 16, 321–357 (2002)
6. Cover, T., Hart, P.: Nearest neighbor pattern classification. Institute of Electrical and Electronics Engineers Transactions on Information Theory 13, 21–27 (1967)
7. Domingos, P.: MetaCost: A general method for making classifiers cost-sensitive. In: Proceedings of the 5th International Conference on Knowledge Discovery and Data Mining (KDD 1999), pp. 155–164. ACM Press (1999)

8. Ferrandiz, S., Boullé, M.: Bayesian instance selection for the nearest neighbor rule. Machine Learning 81(3), 229–256 (2010)
9. Holte, R.C., Acker, L., Porter, B.W.: Concept learning and the problem of small disjuncts. In: Proceedings of the Eleventh International Joint Conference on Artificial Intelligence, pp. 813–818 (1989)
10. Jo, T., Japkowicz, N.: Class imbalances versus small disjuncts. SIGKDD Explorations 6(1), 40–49 (2004)
11. Kubat, M., Holte, R., Matwin, S.: Machine learning for the detection of oil spills in satellite radar images. In: Machine Learning, pp. 195–215 (1998)
12. Li, Y., Zhang, X.: Improving k nearest neighbor with exemplar generalization for imbalanced classification. In: Huang, J.Z., Cao, L., Srivastava, J. (eds.) PAKDD 2011, Part II. LNCS, vol. 6635, pp. 321–332. Springer, Heidelberg (2011)
13. Liu, W., Chawla, S.: Class confidence weighted knn algorithms for imbalanced data sets. In: Huang, J.Z., Cao, L., Srivastava, J. (eds.) PAKDD 2011, Part II. LNCS, vol. 6635, pp. 345–356. Springer, Heidelberg (2011)
14. Menzies, T., Greenwald, J., Frank, A.: Data mining static code attributes to learn defect predictors. IEEE Transactions on Software Engineering 33, 2–13 (2007)
15. Provost, F., Fawcett, T., Kohavi, R.: The case against accuracy estimation for comparing induction algorithms. In: Proceedings of the 15th International Conference on Machine Learning (ICML 1998), pp. 445–453. Morgan Kaufmann (1998)
16. Provost, F.J., Fawcett, T.: Robust classification for imprecise environments. Machine Learning 42(3), 203–231 (2001)
17. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann (1993)
18. Swets, J.: Measuring the accuracy of diagnostic systems. Science 240(4857), 1285–1293 (1988)
19. Ting, K.: The problem of small disjuncts: its remedy in decision trees. In: Proceedings of the 10th Canadian Conference on Artificial Intelligence, pp. 91–97 (1994)
20. Van Den Bosch, A., Weijters, A., Van Den Herik, H.J., Daelemans, W.: When small disjuncts abound, try lazy learning: A case study. In: Proceedings of the Seventh Belgian-Dutch Conference on Machine Learning, pp. 109–118 (1997)
21. Wang, J., Neskovic, P., Cooper, L.: Neighborhood size selection in the k-nearest-neighbour rule using statistical confidence. Pattern Recognition 39, 417–423 (2006)
22. Weiss, G.M.: Mining with rarity: a unifying framework. SIGKDD Explorations 6(1), 7–19 (2004)
23. Weiss, G.M., Hirsh, H.: A quantitative study of small disjuncts. In: Proceedings of the National Conference on Artificial Intelligence, pp. 665–670 (2000)
24. Wilson, D.R., Martinez, T.R.: Reduction techniques for instance-based learning algorithms. In: Machine Learning, pp. 257–286 (2000)
25. Witten, I.H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann (2005)

Class Based Weighted K-Nearest Neighbor over Imbalance Dataset

Harshit Dubey and Vikram Pudi

International Institute of Information Technology, Hyderabad,
Andhra Pradesh, India 500032
harshit.dubeyug08@students.iiit.ac.in, vikram@iiit.ac.in

Abstract. K-Nearest Neighbor based classifier classifies a query instance based on the class labels of its neighbor instances. Although kNN has proved to be a ubiquitous classification tool with good scalability, but it suffers from some drawbacks. The existing kNN algorithm is equivalent to using only local prior probabilities to predict instance labels, and hence it does not take into account the class distribution around neighborhood of the query instance, which results into undesirable performance on imbalanced data. In this paper, a modified version of kNN algorithm is proposed so that it takes into account the class distribution in a wider region around the query instance. Our empirical experiments with several real world datasets show that our algorithm outperforms current state-of-the-art approaches.

Keywords: Classification, K-Nearest Neighbor, Probability, Imbalance, Distribution.

1 Introduction

A data set is imbalanced, if its dependent variable is categorical and the number of instances in one class is different from those in the other class. In many real world applications such as Web page search, scam sites detection, fraudulent calls detection etc, there is a highly skewed distribution of classes. Various classification techniques such as kNN [6], SVM [5], and Neural Networks[10] etc have been designed and used, but it has been observed that the algorithms do not perform as good on imbalanced datasets as on balanced datasets. Learning from imbalanced data sets has been identified as one of the 10 most challenging problems in data mining research [17]. In the literature of solving class imbalance problems, various solutions have been proposed. Such techniques broadly include two different approaches: (1) modifying existing methods or (2) application of a pre-processing stage.

In the recent past, a lot of research centered at nearest neighbor methodology has been done. Although kNN is computationally expensive, it is very simple to understand, accurate, requires only a few parameters to be tuned and is robust with regard to the search space. Also kNN classifier can be updated at a very little cost as new training instances with known classes are presented. A

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 305–316, 2013.

strong point of kNN is that, for all data distributions, its probability of error is bounded above by twice the Bayes probability of error [16]. However one of the major drawbacks of kNN is that, it uses only local prior probabilities to predict instance labels, and hence does not take into account, class distribution around the neighborhood of query instance. This results in undesirable performance on imbalanced data sets. The performance of kNN algorithm over imbalanced datasets can be improved, if it uses information about local class distribution while classifying instances.

Fig. 1 shows an artificial two-class imbalance problem, where the majority class "A" is represented by circles and the minority class "B" by triangles. The query instance is represented by cross. As can be seen from the figure, the query instance would have been classified as the majority class "A" by a regular kNN algorithm with k value equal to 7. But if the algorithm had taken into account the imbalance class distribution around the neighborhood of the query instances (say in the region represented by dotted square), it would have classified the query instance as belonging to minority class "B", which is the desired class.

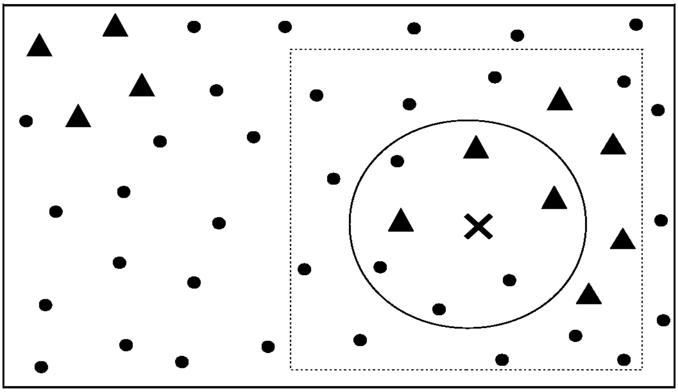


Fig. 1. A sample scenario where regular kNN algorithm will fail

In this paper, we propose a modified K-Nearest Neighbor algorithm to solve the imbalanced dataset classification problem. More specifically the contributions of this paper are as follows:

1. First we present a mathematical model of K-Nearest Neighbor algorithm and show that, it does not take into account nature of the data around the query instance.
2. To solve the above problem, we propose a Weighted K-Nearest Neighbor algorithm in which a weight is assigned to each of the class based on how its instances are classified in the neighborhood of query instance by the regular K-Nearest Neighbor classifier. The modified algorithm takes into account class distribution around the neighborhood of query instance. We ensure that the weights assigned do not give undue advantage to outliers.

3. A thorough experimental study of the proposed approach over several real world dataset was performed. The study confirms that our approach performs better than the current state-of-the-art approaches.

The organization of rest of the paper is as follows. In section 2, we throw light on related, and recent, work in the literature. Section 3 deals with problem formulation and mathematical model of kNN. We explain the modified algorithm in Section 4. In Section 5, experimental results are presented together with a thorough comparison with the state-of-the-art algorithms. Finally, in Section 6, conclusions are drawn.

2 Related Work

In the literature of solving class imbalance problems, various solutions have been proposed; such techniques broadly include two different approaches, modifying methods or the application of a pre-processing stage. The pre-processing approach focuses on balancing the data, which may be done either by reducing the set of examples (undersampling) or replicate minority class examples (oversampling) [8]. One of such earliest and classic work is the SMOTE method [3] which increases the number of minor class instances by creating synthetic samples. This work is also based on the nearest neighbor analogy. The minority class is over sampled by taking each minority class sample and introducing synthetic examples along the line segments joining any/all of the k minority class nearest neighbors. A recent modification to SMOTE proposes that, using different weight degrees on the synthetic samples (so-called safe-level-SMOTE [2]) produces better accuracy than SMOTE. Alternative approaches that modify existing methods focus on extending or modifying the existing classification algorithms so that they can be more effective in dealing with imbalanced data. HDDT [4] and CCPDT [15] are examples of such methods, which are modification of decision tree algorithms.

One of the oldest, accurate and simplest method for pattern classification and regression is K-Nearest-Neighbor (kNN) [6]. kNN algorithms have been identified as one of the top ten most influential data mining algorithms [19] for their ability of producing simple but powerful classifiers. It has been studied at length over the past few decades and is widely applied in many fields. The kNN rule classifies each unlabeled example by the majority label of its k-nearest neighbors in the training dataset. Despite its simplicity, the kNN rule often yields competitive results. A recent work on prototype reduction, called Weighted Distance Nearest Neighbor (WDNN) [11] is based on retaining the informative instances and learning their weights for classification. The algorithm assigns a non negative weight to each training instance tuple at the training phase. Only the training instances with positive weight are retained (as the prototypes) in the test phase. Although the WDNN algorithm is well formulated and shows encouraging performance, in practice it can only work with $K = 1$. A more recent approach WDkNN [20] tries to reduce the time complexity of WDNN and extend it to work for values of K greater than 1.

Chawla and Liu in one of their recent work [14] presented a novel K-Nearest Neighbors weighting strategy for handling the problem of class imbalance. They proposed CCW (class confidence weights) that uses the probability of attribute values given class labels to weight prototypes in kNN. While the regular kNN directly uses the probabilities of class labels in the neighborhood of the query instance, they used conditional probabilities of classes. They have also shown how to calculate CCW weights using mixture modeling and Bayesian networks. The method performed more accurately than the existing state-of-art algorithms.

KaiYan Feng and others [7] defined a new neighborhood relationship known as passive nearest neighbors. For two points A and B belonging to class L, point B is the local passive k^{th}-order nearest neighbor of A, only and only if A is the k^{th} nearest neighbor of B among all data of class L. For each query point, its k actual nearest neighbor and k passive nearest neighbors are first calculated and based on it, a overall score is calculated for each class. The class score determines the likelihood that the query points belong to that class.

In another recent work [12], Evan and others proposes to use geometric structure of data to mitigate the effects of class imbalance. The method even works, when the level of imbalance changes in the training data, such as online streaming data. For each query point, a k dimensional vector is calculated for each of the classes present in the data. The vector consist of distances of the query point to it's k nearest neighbors in that class. Based on this vector probability that the query point belongs to a particular class is calculated. However the approach is not studied in depth.

Yang Song and others proposes [18] two different versions of kNN based on the idea of informativeness. According to them, a point is treated to be informative, if it is close to the query point and far away from the points with different class labels. One of the proposed versions LI-KNN takes two parameters k and I, It first find the k nearest neighbor of the query point and then among them it find the I most informative points. Based on the class label of the informative points, class label is assigned to the query point. They also showed that the value of k and I have very less effect on the final result. The other version GI-KNN works on the assumption that some points are more informative then others. It tries to find global informative points and then assigns a weight to each of the points in training data based on their informativeness. It then uses weighted euclidean metric to calculate distances.

In another recent work [13], a k Exemplar-based Nearest Neighbor (kENN) classifier was proposed which is more sensitive to the minority class. The main idea is to first identify the exemplar minority class instances in the training data and then generalize them to Gaussian balls as concept for the minority class. The approach is based on extending the decision boundary for the minority class.

3 Problem Formulation and Mathematical Model of *k*NN

In this section, we present a mathematical model for kNN algorithm along with the notation used to model the dataset. We also show that kNN only makes use of local prior probabilities for classification.

The problem of classification is to estimate the value of the class variable based on the values of one or more independent variables (known as feature variables). We model the tuple as $\{x, y\}$ where x is an ordered set of attribute values and y is the class variable to be predicted. There are d attributes overall corresponding to a d-dimensional space.

Formally, the problem has the following inputs:

- A set of n tuples called the training dataset, $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$.
- A query tuple x_t.

The output is an estimated value of the class variable for the given query x_t, mathematically it can be expressed as:

$$y_t = f(x_t, D, parameters), \tag{1}$$

Where *parameters* are the arguments that the function $f()$ takes. These are generally set by the user or are learned by some method.

3.1 Mathematical Model of *k*NN

For a given query instance x_t, kNN algorithm works as follows:

$$y_t = \underset{c \in \{c_1, c_2, \ldots, c_m\}}{\arg\max} \sum_{x_i \in N(x_t, k)} E(y_i, c) \tag{2}$$

Where y_t is the predicted class for the query instance x_t and m is the number of classes present in the data. Also

$$E(a, b) = \begin{cases} 1 & if a = b \\ 0 & else \end{cases} \tag{3}$$

$$N(x, k) = Set\, of\, k\, nearest\, neighbor\, of\, x$$

Eq. (2) can also be written as

$$y_t = \arg\max \left\{ \sum_{x_i \in N(x_t, k)} E(y_i, c_1), \sum_{x_i \in N(x_t, k)} E(y_i, c_2), \quad \ldots \quad, \sum_{x_i \in N(x_t, k)} E(y_i, c_m) \right\} \tag{4}$$

$$y_t = \arg\max \left\{ \sum_{x_i \in N(x_t, k)} \frac{E(y_i, c_1)}{k}, \sum_{x_i \in N(x_t, k)} \frac{E(y_i, c_2)}{k}, \quad \ldots \quad, \sum_{x_i \in N(x_t, k)} \frac{E(y_i, c_m)}{k} \right\} \tag{5}$$

and we know that

$$p(c_j)_{(x_t,k)} = \sum_{x_i \in N(x_t,k)} \frac{E(y_i, c_j)}{k} \tag{6}$$

Where $p(c_j)_{(x_t,k)}$ is the probability of occurrence of j^{th} class in the neighborhood of x_t . Hence Eq. 5 turns out to be

$$y_t = \arg\max\{p(c_1)_{(x_t,k)}, p(c_2)_{(x_t,k)}, \ldots, p(c_m)_{(x_t,k)}\} \tag{7}$$

It is clear from Eq. 7, that kNN algorithm uses only prior probabilities to calculate the class of the query instance. It ignores the class distribution around the neighborhood of query point.

4 Algorithm

In this section, we will explain in detail our proposed algorithm. To tune the existing kNN algorithm , we introduce a weighting factor for each class. Our algorithm can be formally expressed as follows, for a given query instance x_t:

$$y_t = \underset{c \in \{c_1, c_2, \ldots, c_m\}}{\arg\max} \sum_{x_i \in N(x_t,k)} W[c, x_t] * E(y_i, c) \tag{8}$$

Where $W[c, x_t]$ denotes the weighting factor for the class c, while classifying query instance x_t. For Weighting factor equal to 1 for all the classes, our algorithm reduces to the existing kNN classifier. This weighting factor is introduced to take into account, class distribution around the query instance. The proposed algorithm is sensitive to the value of weighting factor.

Now we discuss on how to learn the value of weighting factor for each of the classes. Fig. 2 illustrates an imbalance dataset, in which data points are present in clusters with each cluster having exactly one major class. In this case, regular kNN algorithm would fail to classify the minority class instances present at the boundary of the cluster region (for example query instance 1).

To design the weights, we considered both query dependent and query independent weighting factor. If our learned weighting factors have a constant value for each of the class through out the dataset i.e. they do not depend on the query instance, and favors the minority class then, our algorithm would have classified the minority class instances present at the boundary of the clusters correctly, but have not classified points like instance 2 correctly. Having only class dependent weighting factor values would not capture the data distribution around the neighborhood of the query instance.

Our weighting factor value $W[c, x_t]$ can be denoted as :

$$W[c, x_t] = \frac{alpha(c, x_t)}{1 + alpha(c, x_t)} \tag{9}$$

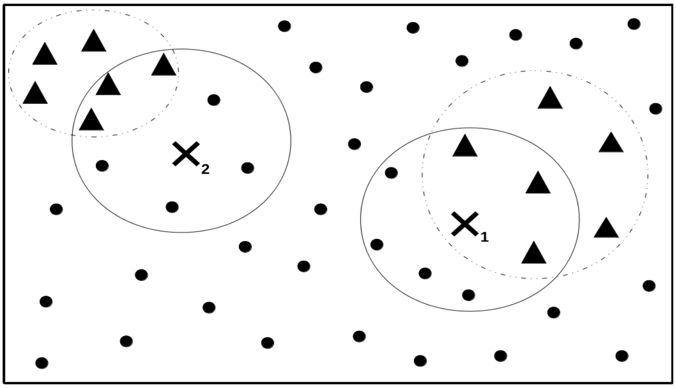


Fig. 2. A sample scenario where data points are present in clusters (dotted circle represent clusters containing minority class) with each cluster having one major class

where

$$alpha(c, x_t) = \sum_{x_i \in N(x_t, \frac{k}{m}, c)} \frac{m * getcoef(x_i)}{k} \tag{10}$$

$$N(x, k, c) = Set\ of\ k\ nearest\ neighbor\ of\ x\ belonging\ to\ class\ c$$

$$getcoef(x_i) = \frac{N(x_i, k, c')}{N(x_i, k, y_i)} \tag{11}$$

where c' is the class to which x_i is classified by existing kNN classifier. if c' equals to y_i then $getcoef(x_i)$ turns out to be 1.

Hence for a query point the weighting factor of a class is calculated based on how the k/m nearest neighbors of query point belonging to that class are classified by the existing kNN classifier. If a instance is classified correctly $getcoef$ will return 1 for it, else it will return the value by which the class prior probability should be multiplied, so that it is classified correctly. The basic intuition about the above formulae is simple: “If instances of a specific class are poorly classified in a particular region, then that class is likely to be a minority class in that region and should be given a higher weight.”

4.1 Properties of Weighting Factor

1. The weighting factor for each of the class is calculated based on how the k/m nearest neighbors of query point belonging to that class are classified by regular kNN classifier. If points belonging to a particular class in the neighborhood of the query points are classified incorrectly by regular kNN approach, then the weighting factor of that class will have a high value indicating that this class might be a minority class in that region. Hence the learned weighting factor takes into account the class distribution around neighborhood of the query point.

Algorithm 1. Pseudo code

```
Input: $Query\,instance\,Q, Training\,Data\,D, Parameter\,k,\ Set\,of\,all\,the\,class\,label\,C$
Output: $Class\,Label\,Cl\,for\,Query\,instance\,Q$
1: for $each\,class\,j\,in\,C$ do
2:     $Coefficient = (\frac{m}{k}) * \sum_{x_i \in N(Q,\frac{k}{m},j)} (getcoef(x_i))$
3:     $W[j,Q] = \frac{Coefficient}{1+Coefficient}$
4: end for
5: for $each\,neighbor\,n\,in\,N(Q,k)$ do
6:     $Probability[class(n)] = Probability[class(n)] + W[class(n),Q]$
7: end for
8: $Cl = \arg\max Probability$
```

2. Value of weighting factor is bounded between 0.5 to 1, proof of which is :

   ```
   if $x_i$ $is\,classified\,correctly\,by\,regular\,kNN$ then
       $\Rightarrow getcoef(x_i) \leftarrow 1$
   else $x_i$ $is\,classified\,as\,belonging\,to\,class\,c'$
       $\Rightarrow N(x_i,k,c') > N(x_i,k,y_i)$
       $\Rightarrow getcoef(x_i) > 1$ (from eq. 11)
   end if
   ```

   Hence $getcoef(x_i)$ value is always greater than or equal to 1, that implies $alpha(c,x_t)$ which is average of $getcoef$ values of k/m nearest neighbors of x_t is always greater than or equal to 1. Eq. 10 can also be written as

   $$W[c,x_t] = \frac{1}{1 + \frac{1}{alpha(c,x_t)}}$$

   Which impliles that

   $$0.5 \leq W[c,x_t] \leq 1$$

   If some outlier point is present in the neighborhood of the query point its $getcoef$ factor would have a high value, but as the weighting factor for a class is calculated by taking average of $getcoef$ values of k/m nearest neighbors of query point belonging to that class, its value would not be much affected by a outlier point. This makes our learned weighting factors resistant to outliers.

4.2 Complexity Analysis

The proposed algorithm needs to search for the k nearest neighbors of the query point (global nearest neighbors, line 5 of Algorithm 1.), same as the regular kNN algorithm. Apart from finding the global nearest neighbors, it also need to calculate the weighting factor for each of the class. Following calculations are involved in the calculation of weighting factors :

1. For each of the class, find k/m nearest neighbors of query point among that class (class neighbors). We can make use of the fact that the global k nearest

neighbors will be among the k nearest neighbors for each of the class, to optimize the search. Rather then finding k/m nearest neighbor for each class, we first get the k nearest neighbor for each class and then get the global k nearest neighbors, with some extra cost involved. For example, assuming that no index structure is present in the training data, while searching for global k nearest neighbors of query point a binary heap structure of size k is needed. For our proposed approach, we maintain such a heap structure one per class. While searching for neighbors, each of the points in the training data is inserted in the heap belonging to its class. When all the points are inserted in the heap, we have the k nearest neighbors of query points among each class in the respective class heap. The total *heapinsert* operations remains same as when finding global k nearest neighbors. Then we can find the global k nearest neighbors and class neighbors from this $k * (numberofclass)$ points.

2. For each of the points obtained above calculate $getcoef$ function, which needs the k nearest neighbors for each of the points (line 2). If $getcoef$ function is calculated for all the points present in the training data during the pre processing step then, this runtime overhead can be avoided. Else we need to find the k nearest neighbors of all the points obtained above i.e. k points.

5 Experimental Study

5.1 Performance Model

In this section, we demonstrate our experimental settings. The experiments were conducted on a wide variety of datasets obtained from UCI data repository [1] and Weka Datasets [9]. A short description of all the datasets is provided in Table 1. These datasets have been selected as they typically have a low minority class percentage and hence are imbalance. We have evaluated our algorithm against the existing state of art approaches. All the results have been obtained using 10-fold cross validation technique, except for SMOTE. For SMOTE each of the dataset is first randomized and then divided into training and testing data. SMOTE sampling is applied on training data to oversample the minority class and then regular kNN is used to classify instances present in testing data using the sampled training data.

As it is known that the use of overall accuracy is not an appropriate evaluation measure for imbalanced datasets, because of the dominating effect of the majority class, we have used F-Score as the evaluation metric. F-Score considers both the precision and the recall of the test to compute the score. We compared our performance against the following approaches: Regular K Nearest Neighbors (kNN) [1], Exemplar kNN (kENN) [2], SMOTE , HDDT [3] , C4.5 , CCPDT [4],

[1] For kNN , SMOTE, C4.5 (available as j48) and Naive, implementation available in Weka toolkit is used.

[2] The code is obtained from `http://goanna.cs.rmit.edu.au/$\sim$zhang/ENN/Weka-3-6_ENN.zip`

[3] The code is obtained from `http://nd.edu/$\sim$dial/software/hddt.tar.gz`

[4] The code is obtained from `http://www.cs.usyd.edu.au/$\sim$weiliu/CCPDT_src.zip`

Table 1. Dataset Description

Dataset	#Instances	#Attributes	#Class	Minority Class%
Balance	625	5	3	7.84
Cmc	1473	10	3	22.61
Diabetes	768	9	2	34.9
Glass	214	10	6	4.2
Heart	270	14	2	44.44
Hungarian	294	14	2	36.05
Ionosphere	351	34	2	35.9
Tranfusion	748	5	2	23.7
Wine	178	13	3	26.9

NaiveBayes (Naive). For all kNN based approaches (including SMOTE) F-Score obtained at maximum accuracy is mentioned.

5.2 Results and Discussion

Table 2. compares the result of our modified algorithm with existing state of the art algorithm. The number in the parenthesis indicates the rank of the respective algorithm. Also the top two algorithms are highlighted in bold. It can be seen that our approach produces consistently accurate classifier and outperforms other algorithms in most of the datasets. Also our proposed algorithm always outperform regular kNN on all the datasets, this confirms that the modified kNN algorithms takes into account the nature of the data to classify it. However for Ionosphere dataset decision tree based algorithms perform better than other state of the art algorithms.

Table 2. Experimental results over several real world dataset

Dataset	Our Algo	kNN	kENN	SMOTE	HDDT	C4.5	CCPDT	Naive
Balance	**0.361 (1)**	0.077 (6)	**0.167 (2)**	0.154 (3)	0.089 (5)	0.000 (7)	0.092 (4)	0.000 (7)
Cmc	0.419 (3)	0.418 (4)	**0.424 (2)**	0.364 (7)	0.380 (6)	0.409 (5)	0.356 (8)	**0.445 (1)**
Pima	**0.628 (2)**	0.601 (6)	0.610 (5)	0.593 (7)	0.613 (4)	0.614 (3)	0.587 (8)	**0.643 (1)**
Glass	**0.778 (1)**	**0.778 (1)**	0.560 (7)	0.750 (3)	0.571 (6)	0.636 (5)	0.235 (8)	0.696 (4)
Heart	**0.812 (2)**	0.805 (5)	**0.812 (2)**	0.765 (7)	0.784 (6)	0.736 (8)	**0.828 (1)**	**0.812 (2)**
Hungarian	0.779 (3)	0.747 (6)	0.747 (6)	**0.805 (2)**	0.767 (4)	0.656 (8)	**0.815 (1)**	0.762 (5)
Ionosphere	0.824 (5)	0.779 (8)	0.793 (6)	0.833 (4)	**0.891 (2)**	0.874 (3)	**0.894 (1)**	0.781 (7)
Tranfusion	**0.489 (2)**	0.442 (7)	0.486 (4)	**0.509 (1)**	**0.489 (2)**	0.481 (6)	0.486 (4)	0.281 (8)
Wine	**0.980 (1)**	**0.980 (1)**	0.950 (6)	**0.980 (1)**	0.958 (5)	0.923 (7)	0.871 (8)	0.970 (4)
Average Rank	**2.22**	4.88	4.44	**3.88**	4.44	5.77	4.77	4.33

Fig. 3 compares performance of our algorithm with kNN in terms of overall accuracy and accuracy to classify minority class, as the value of k varies for Hungarian dataset. It becomes clear from the figure that our algorithm based classifier are more sensitive to classify minority class and are still highly accurate. Also classifier learned from our approach are more accurate for larger values of k, this is evident from the fact that, for high value of k large region around the neighborhood of query point is considered to determine the local class distribution.

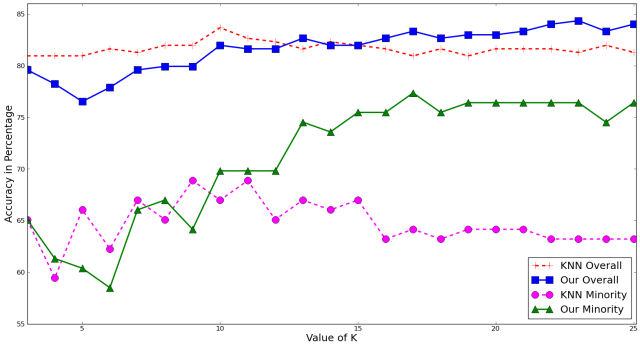


Fig. 3. Performance Comparison between Our Algorithm and kNN

6 Conclusion

In this paper, we have proposed a modified version of K-Nearest Neighbor algorithm so that it takes into account, class distribution around neighborhood of query instance during classification. In our modified algorithm, a weight is calculated for each class based on how its instances are classified by existing K-Nearest Neighbor classifier around the query instance and then a weighted kNN is applied. We have also evaluated our approach against the existing standard algorithms. Our work is focused on tuning the K-Nearest Neighbor for imbalance data, so that its performance on imbalance data is enhanced. As shown in the experimental section, our approaches more than often, outperforms existing state of the art approaches on a wide variety of datasets. Also our modified algorithm perform as good as the existing K-Nearest Neighbor classifier on balance data.

References

1. Asuncion, D.N.A.: UCI machine learning repository (2007)
2. Bunkhumpornpat, C., Sinapiromsaran, K., Lursinsap, C.: Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem. In: Theeramunkong, T., Kijsirikul, B., Cercone, N., Ho, T.-B. (eds.) PAKDD 2009. LNCS, vol. 5476, pp. 475–482. Springer, Heidelberg (2009)
3. Chawla, N., Bowyer, K., Hall, L., Kegelmeyer, W.: Smote. Journal of Artificial Intelligence Research 16(1), 321–357 (2002)

4. Cieslak, D.A., Chawla, N.V.: Learning decision trees for unbalanced data. In: Daelemans, W., Goethals, B., Morik, K. (eds.) ECML PKDD 2008, Part I. LNCS (LNAI), vol. 5211, pp. 241–256. Springer, Heidelberg (2008)
5. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning 20, 273–297 (1995), doi:10.1007/BF00994018
6. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification, 2nd edn. Wiley, New York (2001)
7. Feng, K., Gao, J., Feng, K., Liu, L., Li, Y.: Active and passive nearest neighbor algorithm: A newly-developed supervised classifier. In: Huang, D.-S., Gan, Y., Gupta, P., Gromiha, M.M. (eds.) ICIC 2011. LNCS, vol. 6839, pp. 189–196. Springer, Heidelberg (2012)
8. Garcia, S., Herrera, F.: Evolutionary undersampling for classification with imbalanced datasets: proposals and taxonomy. Evolutionary Computation 17(3), 275–306 (2009)
9. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: an update. SIGKDD Explor. Newsl. 11, 10–18 (2009)
10. Haykin, S.: Neural Networks: A Comprehensive Foundation. Prentice-Hall (1999)
11. Jahromi, M.Z., Parvinnia, E., John, R.: A method of learning weighted similarity function to improve the performance of nearest neighbor. Inf. Sci. 179, 2964–2973 (2009)
12. Kriminger, E., Principe, J., Lakshminarayan, C.: Nearest neighbor distributions for imbalanced classification. In: The 2012 International Joint Conference on Neural Networks (IJCNN), pp. 1–5 (June 2012)
13. Li, Y., Zhang, X.: Improving k nearest neighbor with exemplar generalization for imbalanced classification. In: Huang, J.Z., Cao, L., Srivastava, J. (eds.) PAKDD 2011, Part II. LNCS, vol. 6635, pp. 321–332. Springer, Heidelberg (2011)
14. Liu, W., Chawla, S.: Class confidence weighted knn algorithms for imbalanced data sets. In: Huang, J.Z., Cao, L., Srivastava, J. (eds.) PAKDD 2011, Part II. LNCS, vol. 6635, pp. 345–356. Springer, Heidelberg (2011)
15. Cieslak, D., Liu, W., Chawla, S., Chawla, N.: A robust decision tree algorithms for imbalanced data sets. In: Proceedings of the Tenth SIAM International Conference on Data Mining, pp. 766–777 (2010)
16. Loizou, G., Maybank, S.J.: The nearest neighbor and the bayes error rates. IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-9(2), 254–262 (1987)
17. Yang, Q., Wu, X.: 10 challenging problems in data mining research. International Journal of Information Technology and Decision Making 5(4), 597–604 (2006)
18. Song, Y., Huang, J., Zhou, D., Zha, H., Giles, C.L.: Iknn: Informative k-nearest neighbor pattern classification. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) PKDD 2007. LNCS (LNAI), vol. 4702, pp. 248–264. Springer, Heidelberg (2007)
19. Wu, X., Kumar, V., Ross Quinlan, J., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G.J., Ng, A., Liu, B., Yu, P.S., Zhou, Z.-H., Steinbach, M., Hand, D.J., Steinberg, D.: Top 10 algorithms in data mining. Knowl. Inf. Syst. 14, 1–37 (2007)
20. Yang, T., Cao, L., Zhang, C.: A novel prototype reduction method for the K-nearest neighbor algorithm with $K \geq 1$. In: Zaki, M.J., Yu, J.X., Ravindran, B., Pudi, V. (eds.) PAKDD 2010. LNCS, vol. 6119, pp. 89–100. Springer, Heidelberg (2010)

ProWSyn: Proximity Weighted Synthetic Oversampling Technique for Imbalanced Data Set Learning

Sukarna Barua[1], Md. Monirul Islam[1], and Kazuyuki Murase[2]

[1] Bangladesh University of Engineering and Technology (BUET), Dhaka, Bangladesh
[2] University of Fukui, Fukui, Japan

Abstract. An imbalanced data set creates severe problems for the classifier as number of samples of one class (majority) is much higher than the other class (minority). Synthetic oversampling methods address this problem by generating new synthetic minority class samples. To distribute the synthetic samples effectively, recent approaches create weight values for original minority samples based on their importance and distribute synthetic samples according to weight values. However, most of the existing algorithms create inappropriate weights and in many cases, they cannot generate the required weight values for the minority samples. This results in a poor distribution of generated synthetic samples. In this respect, this paper presents a new synthetic oversampling algorithm, Proximity Weighted Synthetic Oversampling Technique (ProWSyn). Our proposed algorithm generate effective weight values for the minority data samples based on sample's proximity information, i.e., distance from boundary which results in a proper distribution of generated synthetic samples across the minority data set. Simulation results on some real world datasets shows the effectiveness of the proposed method showing improvements in various assessment metrics such as AUC, F-measure, and G-mean.

Keywords: Imbalanced learning, clustering, synthetic oversampling.

1 Introduction

Imbalanced learning problem is to deal with imbalanced data sets where one or more classes have much higher number of data samples than other classes. Generally speaking, imbalanced learning occurs whenever some types of data distribution significantly dominate the sample space compared to other data distributions. By convention, in imbalanced data sets, we call the classes having more samples the majority classes and the ones having fewer samples the minority classes. Classifiers tend to produce greater classification errors over the minority class samples [1–3]. Many real world problems suffer from this phenomenon such as medical diagnosis [4], information retrieval [5], detection of fraudulent telephone calls [6] and oil spills in radar images [7], data mining from

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 317–328, 2013.

direct marketing [8], and helicopter fault monitoring [9]. Thus the identification of the minority class samples is of utmost importance.

There have been many attempts in solving imbalanced learning problems, such as various oversampling and undersampling methods [10]. Roughly speaking, an undersampling method remove some majority class samples from an imbalanced data set with an aim to balance the distribution between the majority class and minority class samples [11–14], while an oversampling method does the opposite, i.e, generates synthetic minority class samples and adds them to the data set [15–19]. Both undersampling and oversampling methods have been shown to improve classifiers' performance on imbalanced data sets. While comparing oversampling and undersampling one natural observation favoring oversampling is that undersampling may remove essential information from the original data, while oversampling does not suffer from this problem. It has been shown that oversampling is lot more useful than undersampling and the performance of oversampling was shown to improve dramatically even for complex data sets [20].

Most of the existing oversampling methods (e.g. [17–19]) first try to estimate the difficulty levels of the minority class samples in generating synthetic minority class samples. In doing so, they assign weights to the minority class samples based on a computed value, i.e., δ. The methods then use δ to decide how many synthetic samples are to be generated for a particular minority class sample. In Sect. 2, we illustrate that the way δ is computed is not reasonable in many scenarios. Consequently, the sample generation process is not able to generate synthetic samples appropriately, which affects classifiers performance.

In this paper, we present a new flexible oversampling method, named Proximity Weighted Synthetic Oversampling Technique (ProWSyn). Unlike previous work, ProWSyn uses the distance information of the minority class samples from the majority samples in assigning weights to the minority class samples. The effectiveness of the proposed technique has been evaluated on several benchmark classification problems with high imbalanced ratio. It has been found that ProWSyn performs better compared to some other existing methods in most of the cases.

The remainder of this paper is divided into five sections. In Sect. 2, we present the related works for solving imbalanced learning problems. Section 3 describes the details of the proposed algorithm. In Sect. 4, we present the experimental study and simulation results. Finally, in Sect. 5, we provide some future aspects of this research and conclude the paper.

2 Related Work

The main objective of oversampling methods is to oversample the minority class samples to shift the classifier learning bias toward the minority class. Synthetic Minority Over-sampling Technique (SMOTE) is one such method [15]. For every minority class sample, this technique first finds its k (which is set to 5 in SMOTE) nearest neighbors of the same class and then randomly selects some of them according to the over-sampling rate. Finally, SMOTE generates new synthetic samples along the line between the minority sample and its selected nearest neighbors. The problem of SMOTE is that it does not consider the importance of minority class samples, thus generates an arbitrary equal number of synthetic minority class samples. However, all the minority class samples are not equally important (hard). The minority class samples that are surrounded by many majority class samples or closer to the classifier's decision boundary are more important than the other ones.

Adaptive synthetic (ADASYN) oversampling [17], Ranked Minority Oversampling (RAMO) [18] and CBSO [19] try to address the aforementioned problem in dealing with imbalanced learning problems. These methods are based on the idea of assigning weights to minority class samples according to their importance. These weights are used for generating synthetic samples. More synthetic samples are generated for a large weight than for a small weight. CBSO, like earlier approaches [17, 18], uses a parameter δ, the number of majority samples among the k nearest neighbors of a minority sample x, for assigning weight to x. Let , $N_k(x)$ is the set of k nearest neighbors of x. Then, δ for x equals to the number of the majority class samples in $N_k(x)$. If this number is large, then δ is high, resulting a large weight assignment to the minority sample [17–19]. However, the use of δ for assigning weights to individual minority samples may not be appropriate in the situations described below.

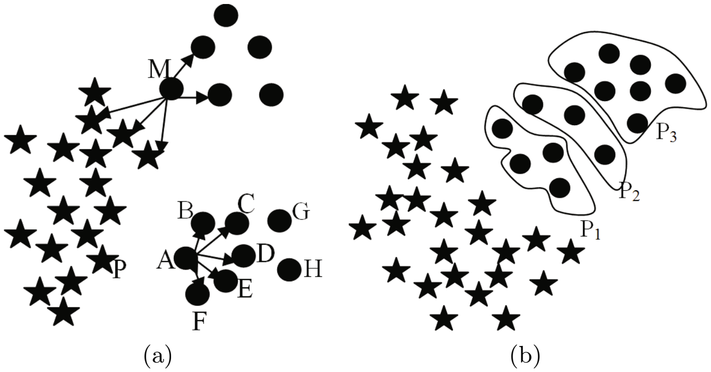


Fig. 1. (a) k nearest neighbors are shown by arrows for some minority samples. (b) Minority samples are partitioned according to their proximity, i.e., distance from boundary.

1. *δ may be inappropriate for assigning weights to the minority class samples located near the decision boundary.*
   To understand this fact, consider Fig. 1(a) where the minority class and majority class samples are shown by circles and stars respectively. It is clear from this figure that the minority class sample A has no majority class samples in their k-nearest neighborhood (assume $k = 5$) and $N_5(A)$ contains only the minority class samples B, C, D, E and F. We use arrow in Fig. 1(a) to show the neighbors of a minority sample. Since, $N_k(A)$ does not contain

any majority class sample, δ for A would be 0. For similar reasons, δ for B would also be 0. It means A and B will be given zero or the lowest weight, although seemingly they are the most important samples due to their position near the decision boundary.

2. *δ may be insufficient to discover the difference of minority samples w.r.t their importance in learning.* It can be seen from Fig. 1(a) that the minority samples A and B are closer to the decision boundary than those of G and H. It is, therefore, reasonable that A and B should be given higher weight than G and H. However, if we assume $k = 5$ and compute δ for A, B, G and H, then it is evident from the figure that the δ would is 0 for each of them, because no $N_5(x)$ $(x \in \{A, B, G, H\}$ contain any majority class sample. It is now understood that δ cannot differentiate the minority class samples according to their importance in learning. Another problem is that δ for some or all samples, i.e. A, B and so on, of one region, is 0, while for the sample M of another region, δ gets a good positive value (Fig. 1(a)). Under this condition, all synthetic samples will be generated in the M's region and no or very few will be generated in the A's region. This example again illustrate the same thing i.e. δ cannot discover the difference of the minority class samples w.r.t their importance in learning.

The above scenarios confirms that earlier approaches based on δ e.g. [17–19] cannot effectively assign weights to the minority class samples. In most scenarios, if the parameter k is not sufficiently large, then most of the minority samples will get zero weight [17, 19] or the lowest weight [18]. Minority class samples having zero or the lowest weight will get no or very few synthetic samples around them. It may seem the problem can be avoided by increasing the value of k to contain the majority class samples. However, the required value of k cannot be determined in advance. In some regions, a small k may suffice, while in other regions a large k is to be required. Even if k is increased, it cannot solve the skewed distribution of synthetic samples. For example, suppose we increase k to 6 to contain the majority class samples for the minority class sample A (Fig. 1(a)). For this case, $N_6(A) = \{B, C, D, E, F, P\}$, which contains one majority sample P. Hence, A's delta will be 1. However, M's delta will be 4, because $N_6(M)$ contains four majority class samples (Fig. 1(a)). Still, A's δ is smaller compared to that for M and more synthetic samples will be generated in the neighborhood of M. This justifies that increasing k cannot effectively solve the problem at all.

3 Proposed Algorithm

Motivated by problems stated in Sect. 2, we have extended the recently proposed CBSO [19] algorithm and proposed a new improved algorithm, named Proximity Weighted Synthetic Oversampling Technique (ProWSyn). The new algorithm uses a different weight generation technique to alleviate the problems described earlier. The complete algorithm is shown in [Algorithm ProWSyn]. Our ProWSyn differs from CBSO in Steps 2 to 7 in which each minority sample

x is now weighted based on the proximity level of x i.e.,PL_x. We measure PL_x using the Euclidean distance of x from the majority class samples. Steps 8 to 11 create the synthetic samples and produce an output oversampled minority data set, S_{omin}. The details of weight generation procedure are discussed below.

[**Algorithm ProWSyn**]
Input:
Training data samples D_{tr} with m samples $\{x_i, y_i\}$, $i = 1 \cdots m$, where x_i is an instance in n dimensional feature space X, and $y_i \in \{-1, 1\}$ is the class identity level associated with x_i. Define S_{maj} and S_{min} to be the majority and minority class set respectively and m_l and m_s as the number of majority class and minority class samples respectively. Therefore, $m_s \leq m_l$ and $m_s + m_l = m$.
Procedure:

1. Calculate the number of synthetic samples that need to be generated for the minority class:
$$G = (m_l - m_s) \times \beta$$
where $\beta \in [0, 1]$ is a parameter used to specify the desired balance level after generation of the synthetic samples. We can obtain a fully balanced dataset by assigning $\beta = 1$.
2. Initialize, $P = S_{min}$
3. For $i = 1$ to $L - 1$ do the following:
   (a) From each majority sample y, find the nearest K minority samples in P according to Euclidean distance. Let, the set of these K samples to be $N_K(y)$.
   (b) Form partition P_i as the union of all $N_K(y)$s:
$$P_i = \bigcup_{y \in S_{maj}} N_K(y) \tag{1}$$
   (c) Set proximity level of each minority sample x in partition P_i to be i:
$$PL_x = i, \forall x \in P_i \tag{2}$$
   (d) Remove selected minority samples from P, $P = P - P_i$.
4. Form partition P_L with the remaining unpartitioned samples in P.
5. Set proximity level of each x in P_L to be L:
$$PL_x = L, \forall x \in P_L$$
6. For each x, calculate a weight w_x from its proximity level PL_x defined as:
$$w_x = \exp\left(-\theta * (PL_x - 1)\right) \tag{3}$$
where θ is a smoothing factor and $w_x \in [0, 1]$.
7. Normalize w_x according to $\widehat{w_x} = w_x / \sum_{z \in S_{min}} w_z$, so that $\widehat{w_x}$ is a density distribution ($\sum \widehat{w_x} = 1$)

```
8. Calculate the number of synthetic samples $g_x$ that need to be generated for $x$:
       $g_x = \widehat{w_x} \times G$
9. Find the clusters of minority set, $S_{min}$
10. Initialize set, $S_{omin} = S_{min}$
11. For each $x$, generate $g_x$ synthetic minority class samples according to the following steps:
    Do the loop from 1 to $g_x$
      (a) Randomly select one minority sample $y$, from $x$'s cluster (as found in step 9).
      (b) Generate a synthetic sample, $s$, according to
          $s = x + \alpha \times (y - x)$, where $\alpha$ is a random number in the range $[0, 1]$.
      (c) Add $s$ to $S_{omin}$: $S_{omin} = S_{omin} \bigcup \{s\}$
    End Loop
```

Output: Oversampled minority data set, S_{omin}

3.1 Weight Generation Mechanism of ProWSyn

The goal of our weight generation mechanism is assign appropriate weights to the minority class samples according to their importance in learning. To do this, ProWSyn works in two phases. In first phase (Steps 2-5), it divides the minority data set in a number of partitions (say, L) based on their distance from the decision boundary. Each partition is assigned a proximity level where the level increases with increasing distance from the boundary. Minority class samples with lower proximity levels are the difficult samples and therefore are important for learning, while they with higher proximity levels are less important and may not have significant importance at all. In second phase, ProWSyn generates synthetic minority class samples using the proximity information so that it can generate more synthetic samples in lower proximity regions, i.e., regions that are very near to the decision boundary. The whole procedure is described below (Steps 2 to 7 of [Algorithm ProWSyn]).

From each majority sample ProWSyn finds the nearest K minority class samples according to the Euclidean distance. The set of all these minority class samples form the first partition, P_1 (of proximity level 1), the nearest level of samples from the boundary. Then, it finds the next K minority samples according to distance from each majority. These samples together form the second partition, P_2 (of proximity level 2). In this way, the procedure is repeated for $L-1$ such partitions of proximity levels 1 to $L-1$. The rest of the unpartitioned samples will form partition L, the farthest set of samples from the boundary. A simulated partitioning is shown in Fig. 1(b) for $K = 3$ and $L = 3$. It is seen that the minority class samples are properly identified and partitioned according to their distance from the boundary, which also signifies their importance in oversampling.

Table 1. Description of data set characteristics used in simulation

Dataset	Minority Class	Features	Instances	Minority	Majority	%Minority
Pageblocks	Class of ’Graphic’, ’Vert.line’, ’Picture’	10	5476	231	5245	4%
Abalone	Class of ’18’	7	731	42	689	6%
CTG	Class of ’3’, ’4	21	2126	134	1992	6.3%
Segment	Class of ’Grass’	19	2310	330	1980	14%
Libra	Class of ’1’, ’2’, ’3’	90	360	72	288	20%
Yeast	Class of ME3’, ’ME2’, ’EXC’, ’VAC’, ’POX’, ’ERL’	8	1484	304	1180	21%
Robot	Class of ’slight-left-turn’, ’slight-right-turn’	24	5456	1154	4302	22%
Vehicle	Class of ’1’	18	940	219	721	24%
Breast-tissue	Class of ’CAR’, ’FAD’	9	106	36	70	34%
Pima	Class of ’1’	8	768	268	500	35%

Minority class samples are given weight according to their proximity level (Step 6 of [Algorithm ProWSyn]). All minority samples in the same partition, i.e., having same proximity level, gets the same weight. As the level increases, weight of the minority samples also decreases exponentially (Eqn. 3). The parameter θ in (3) controls the rate with which weight decreases with respect to levels.

The above weight generation technique of ProWSyn has several advantages over earlier δ-based approaches. First, our ProWSyn can effectively find weight for a minority sample according to its position from the decision boundary, while earlier approaches fail to do so (Sect. 2). Secondly, the proposed method successfully partitions the minority class samples based on the distance from the decision boundary. So, samples closer to boundary get higher weight than samples that are further. This is not guaranteed in earlier approaches (Sect. 2). Thirdly, while earlier approaches may lead to generation of synthetic samples in a few small regions due to positive weights of a few minority class samples (Sect. 2), ProWSyn can avoid it by assigning proper weight values for all minority class samples. Fourthly, the procedure of ProWSyn is a more general one, we can easily control the size of each partition and number of partitions to be created based on the problem domain by varying the parameters K and L. However, there is no such scope in earlier approaches.

4 Experimental Study

In this section, we evaluate the effectiveness of our proposed ProWSyn algorithm and compare its performance with ADASYN [17], RAMO [18], and CBSO [19] methods. We use two different classifier models: backpropagation neural network and C4.5 decision tree [21]. We collect ten datasets from UCI machine learning repository [22]. The data sets were chosen in such a way that they contained a varied level of imbalanced distribution of samples. Some of these original data sets were multi-class data. Since, we are only interested in two-class classification problem, these data sets were transformed to form two-class data sets in a way which ensures a certain level of imbalance. Table 1 shows the minority class composition (these classes in the original dataset were combined to form the minority class and rest of the classes form the majority class) and other characteristics of the data sets such as the number of features, the number of total samples, and the number of majority and minority class samples. As evaluation metrics, we use the most popular measure for imbalanced problem domains i.e., the area under the Receiver Operating Characteristics (ROC) graph [23], usually known as AUC. Furthermore, we use two other popular performance metrics such as F-measure and G-mean [10].

We run single decision tree classifier and single neural network classifier on the selected datasets described in Table 1. For the neural network classifier, the number of hidden neurons is randomly set to 5, the number of input neurons is set to be equal to the number of features in the dataset and the number of output neurons is set to 2. We use Sigmoid function as an activation function for neural network. The number of training epochs is randomly set to 300 and learning rate is set to 0.1. For ADASYN and CBSO, the value of the nearest neighbors, K, is set to 5 [17, 19]. The values of the nearest neighbors, i.e., $k1$ and $k2$, of RAMO are chosen as 5 and 10 respectively [18]. The scaling coefficient, α, for RAMO is set to 0.3 [18]. For ProWSyn and CBSO, $C_p = 3$ [17, 19]. For ProWSyn, other parameter values are $\theta = 1$, $K = 5$, and $L = 5$. These values are chosen after some preliminary simulation runs and they are not meant to be optimal. Number of synthetic samples generated is set by $\beta = 1$ for ADASYN, ProWSyn, and CBSO. Same number of synthetic samples were generated for RAMO for fair comparison.

Simulation results of F-measure (F-meas), G-mean, averaged AUC, and standard deviation of AUC values (SDAUC) are presented in Table 2. Each result is found after a 10-fold cross-validation. The best result in each category is highlighted with a bold-face type. We obtain averaged AUC values by averaging the AUC values of multiple simulation runs [18]. We also provide the number of times an algorithm win (win time) against any other method we compare here.

It is observed from Table 2 that for both classifiers, ProWSyn outperforms CBSO, ADASYN, and RAMO algorithms in terms of F-measure, G-mean, and averaged AUC in most of the datasets. As described in Sect. 2, δ based weight generation technique cannot create appropriate distribution of synthetic minority class samples. Generation of appropriate weight values by the ProWSyn approach leads to better distribution of synthetic samples along the difficult minority class regions making its performance better than other approaches. We apply Wilcoxon signed-rank test [24] to statistically compare the performance (AUC metric) of the four methods. The test is applied to compare our proposed

Table 2. Performance of PROWSYN, ADASYN [17], RAMO [18], and CBSO [19] on ten Real World Datasets using single Decision Tree and single Neural Network classifiers

		Decision Tree Classifier				Neural Network Classifier			
Dataset	Method	F-meas	G-mean	AUC	SDAUC	F-meas	G-mean	AUC	SDAUC
Pageblocks	ADASYN	0.9848	0.991	0.9833	0.0118	0.8692	0.9567	0.9723	0.0251
	RAMO	0.9864	0.9928	0.9854	0.011	0.9536	0.9827	0.9848	0.008
	CBSO	0.9834	0.9908	0.984	0.009	0.8932	0.9731	0.9828	0.0077
	PROWSYN	**0.9878**	**0.9939**	**0.9857**	0.0075	**0.9779**	**0.9911**	**0.9867**	0.0027
Abalone	ADASYN	0.3055	**0.5372**	0.7495	0.1919	0.4646	**0.6321**	0.8875	0.0609
	RAMO	0.2401	0.4361	0.7301	0.2013	0.393	0.5229	0.8892	0.0741
	CBSO	**0.3101**	0.5155	**0.7615**	0.2087	0.4493	0.5415	0.8938	0.0736
	PROWSYN	0.294	0.4772	0.7528	0.1706	**0.4728**	0.5965	**0.8976**	0.0706
CTG	ADASYN	0.6415	0.8093	0.891	0.0324	0.4637	0.6866	0.9016	0.0248
	RAMO	0.6639	0.8128	**0.9206**	0.0259	0.5179	0.7268	0.9125	0.0386
	CBSO	**0.6758**	**0.8250**	0.9067	0.0359	0.5434	**0.7514**	**0.9192**	0.0232
	PROWSYN	0.5882	0.7852	0.9137	0.0365	**0.5527**	0.73	0.914	0.0403
Segment	ADASYN	0.7227	0.8702	0.9609	0.0153	0.5258	0.7443	0.9062	0.0892
	RAMO	**0.7761**	**0.9068**	0.9653	0.0139	0.5792	0.7488	0.9391	0.0252
	CBSO	0.7509	0.8877	0.9547	0.015	0.5237	0.716	0.9164	0.0696
	PROWSYN	0.7152	0.8853	**0.9665**	0.0177	**0.6000**	**0.8018**	**0.9410**	0.027
Libra	ADASYN	0.7367	0.8347	0.8546	0.0861	0.8792	0.9153	0.8919	0.1557
	RAMO	0.7285	0.812	0.8327	0.1156	**0.9072**	**0.9338**	**0.9596**	0.0507
	CBSO	0.7278	0.8162	0.8397	0.1218	0.8832	0.931	0.9112	0.1527
	PROWSYN	**0.7717**	**0.8557**	**0.9054**	0.0707	0.8928	0.9284	0.9467	0.0953
Yeast	ADASYN	0.6224	0.7617	0.8728	0.0185	0.6693	0.8311	0.8906	0.0208
	RAMO	0.6381	0.7739	0.8766	0.0203	0.6837	0.8206	0.8825	0.022
	CBSO	0.6412	0.7726	0.8712	0.0267	**0.6936**	**0.8447**	0.8937	0.0176
	PROWSYN	**0.6631**	**0.7986**	**0.8865**	0.028	0.6821	0.8289	**0.8991**	0.0188
Robot	ADASYN	0.9832	0.9922	0.9863	0.0029	0.5702	0.7628	0.8189	0.0571
	RAMO	0.9819	0.9903	0.984	0.0027	0.5899	0.7692	0.8403	0.0466
	CBSO	**0.9922**	**0.9963**	**0.9871**	0.0023	0.6071	0.79029	0.8499	0.0182
	PROWSYN	0.9667	0.9839	0.9837	0.0031	**0.6198**	**0.7931**	**0.8557**	0.0389
Vehicle	ADASYN	0.8844	0.9265	0.9628	0.0187	0.9072	0.9603	0.9764	0.012
	RAMO	**0.8884**	0.9277	0.9591	0.0228	0.9108	**0.9617**	**0.9801**	0.0062
	CBSO	0.8761	0.9251	0.961	0.017	0.8818	0.9496	0.9673	0.0205
	PROWSYN	0.8832	**0.9283**	**0.9634**	0.02	**0.9120**	0.9556	0.9726	0.0147
Btissue	ADASYN	0.6671	0.7286	0.8103	0.1408	0.6557	0.7055	0.7966	0.1369
	RAMO	0.7149	0.7754	0.8754	0.0903	0.639	0.6897	0.809	0.1081
	CBSO	0.5857	0.6693	0.8004	0.1327	0.6533	0.6732	0.7933	0.1027
	PROWSYN	**0.7805**	**0.8328**	**0.8985**	0.0876	**0.6844**	**0.7213**	**0.8482**	0.0759
Pima	ADASYN	0.5536	0.6471	0.7382	0.0452	0.6643	0.7181	**0.8165**	0.0459
	RAMO	0.5772	0.6669	0.75	0.0227	0.65	0.7034	0.7894	0.0762
	CBSO	0.5836	0.6707	0.7427	0.0437	0.656	0.7092	0.8144	0.04
	PROWSYN	**0.5962**	**0.6810**	**0.7611**	0.0541	**0.6883**	**0.7502**	0.8125	0.0361
Win Time	ADASYN	0	1	0		0	1	1	
	RAMO	2	1	1		1	2	2	
	CBSO	3	2	2		1	2	1	
	PROWSYN	5	6	7		8	5	6	

Table 3. Detailed computation of Wilcoxon test [24] statistic on AUC results of PROWSYN vs. ADASYN [17] for Decision Tree Classifier

Dataset	PROWSYN	ADASYN	Difference	Rank
PageBlocks	0.98573	0.98338	0.00235	+2
Abalone	0.75282	0.74956	0.00326	+4
CG	0.91376	0.89108	0.02268	+7
Segment	0.9665	0.96093	0.00557	+5
Libra	0.90546	0.85464	0.05082	+9
Yeast	0.88657	0.87287	0.0137	+6
Robot	0.98374	0.98636	-0.00262	-3
Vehicle	0.96346	0.96288	0.00058	+1
Btissue	0.89857	0.81033	0.08824	+10
Pima	0.76118	0.73829	0.02289	+8
	$R+ = 52$, $R- = 3$, $T = min\{52, 3\} = 3$			

Table 4. Significance tests of averaged AUC between PROWSYN vs. ADASYN [17], RAMO [18], and CBSO [19] for Decision Tree and Neural Network Classifiers. All results are significant (less than or equal to critical value 8) except ProWSyn vs. RAMO for Neural Network classifier ($T = 10$ is larger than critical value 8)

	Decision Tree Classifier			Neural Network Classifier		
	PROWSYN vs.			PROWSYN vs.		
	ADASYN	RAMO	CBSO	ADASYN	RAMO	CBSO
$R+$	52	48	47	52	45	50
$R-$	3	7	8	3	10	5
T	3	7	8	3	10	5

ProwSyn with each of the other methods in a pairwise manner. For 10 datasets, the test statistic, i.e. T should be less than or equal to critical value 8 [25] to reject the null hypothesis at the significance level of 0.05. Table 3 shows the detailed computation of the Wilcoxon statistic, i.e. T for ProsSyn vs. ADASYN results of Decision tree classifier. The obtained value of $T = 3$ is less than the critical value 8. This proves that ProwSyn is statistically better than the ADASYN. For space consideration, we avoid the detailed computation and show only the test statistic values for all other comparisons in Table 4. We see that ProwSyn statistically outperforms other methods for all comparisons except ProWSyn vs. RAMO for neural network classifier ($T = 10$ is larger than critical value 8). In this case, ProWSyn can not statistically outperform RAMO. However, in Table 2, AUC column under Neural Network Classifier shows that ProWSyn wins in 6 cases while RAMO wins 2 cases. This difference in winning time shows that ProWSyn performs better than RAMO for neural network classifier.

5 Conclusion

In this paper, we try to identify the problems related to the synthetic sample generation process of oversampling methods in dealing with imbalanced data sets. Existing algorithms [17–19] generate inaccurate weights for the minority samples that results in very poor and skewed distribution of synthetic samples (Sect. 2). We thus present a new synthetic oversampling algorithm, ProWSyn, for balancing the majority and minority class distributions in an imbalanced data set. Our ProWSyn avoids the aforementioned problem by assigning effective weights to the minority class samples using their Euclidean distance from the majority class samples in the data set. ProWSyn partitions the minority data set into several partitions based on samples' proximity from the decision boundary. and uses this proximity information for the minority samples in such a way that the closest sample get highest weight and the furthest ones get the lowest weight. By doing so, our method ensures a proper distribution of weights among the minority samples according to their position from the decision boundary. This results in a effective distribution of generated synthetic samples across the minority data set. The simulation results show that ProWSyn can statistically outperform ADASYN, CBSO, and RAMO algorithms in terms of a number of performance metrics such as AUC, F-measure, and G-mean. Several other research issues can be investigated using ProWSyn such as application of ProWSyn in multi-class problems, integration of ProWSyn with some other undersampling methods, and integration of ProWSyn with an ensemble technique such as Adaboost.M2 boosting ensemble.

Acknowledgments. The work has been done in the Computer Science & Engineering Department of Bangladesh University of Engineering and Technology (BUET). The authors would like to acknowledge BUET for its generous support.

References

1. Weiss, G.M.: Mining with Rarity: A Unifying Framework. ACM SIGKDD Explorations Newsletter 6(1), 7–19 (2004)
2. Holte, R.C., Acker, L., Porter, B.W.: Concept Learning and the Problem of Small Disjuncts. In: Proc. Int'l J. Conf. Artificial Intelligence, pp. 813–818 (1989)
3. Quinlan, J.R.: Induction of Decision Trees. Machine Learning 1(1), 81–106 (1986)
4. Murphy, P.M., Aha, D.W.: UCI repository of Machine learning databases. University of California Irvine, Department of Information and Computer Science
5. Lewis, D., Catlett, J.: Heterogeneous Uncertainty Sampling for Supervised Learning. In: Proc. of the Eleventh International Conference of Machine Learning, pp. 148–156 (1994)
6. Fawcett, T.E., Provost, F.: Adaptive Fraud Detection. Data Mining and Knowledge Discovery 3(1), 291–316 (1997)
7. Kubat, M., Holte, R.C., Matwin, S.: Machine Learning for the Detection of Oil Spills in Satellite Radar Images. Machine Learning 30(2/3), 195–215 (1998)
8. Ling, C.X., Li, C.: Data Mining for Direct Marketing: Problems and Solutions. In: Proc. Int'l Conf. on Knowledge Discovery & Data Mining (1998)

9. Japkowicz, N., Myers, C., Gluck, M.: A Novelty Detection Approach to Classification. In: Proc. of the Fourteenth International Joint Conference on Artificial Intelligence, pp. 518–523 (1995)
10. He, H., Garcia, E.A.: Learning from imbalanced data. IEEE Trans. Knowl. Data Eng. 21(10), 1263–1284 (2009)
11. Liu, X.Y., Wu, J., Zhou, Z.H.: Exploratory Under Sampling for Class Imbalance Learning. In: Proc. Int'l Conf. Data Mining, pp. 965–969 (2006)
12. Zhang, J., Mani, I.: KNN Approach to Unbalanced Data Distributions: A Case Study Involving Information Extraction. In: Proc. Int'l Conf. Machine Learning, ICML 2003, Workshop Learning from Imbalanced Data Sets (2003)
13. Kubat, M., Matwin, S.: Addressing the Curse of Imbalanced Training Sets: One-Sided Selection. In: Proc. Int'l Conf. Machine Learning, pp. 179–186 (1997)
14. Batista, G.E.A.P.A., Prati, R.C., Monard, M.C.: A Study of the Behavior of Several Methods for Balancing Machine Learning Training Data. ACM SIGKDD Explorations Newsletter 6(1), 20–29 (2004)
15. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: Synthetic Minority Over-Sampling Technique. J. Artificial Intelligence Research 16, 321–357 (2002)
16. Cieslak, D.A., Chawla, N.V.: Start Globally, Optimize Locally, Predict Globally: Improving Performance on Imbalanced Data. In: Proc. IEEE Int'l Conf. Data Mining, pp. 143–152 (2008)
17. He, H., Bai, Y., Garcia, E.A., Li, S.: ADASYN: Adaptive Synthetic Sampling Approach for Imbalanced Learning. In: Proc. Int'l J. Conf. Neural Networks, pp. 1322–1328 (2008)
18. Chen, S., He, H., Garcia, E.A.: RAMOBoost: Ranked Minority Oversampling in Boosting. IEEE Trans. Neural Networks 21(20), 1624–1642 (2010)
19. Barua, S., Islam, M. M., Murase, K.: A Novel Synthetic Minority Oversampling Technique for Imbalanced Data Set Learning. In: Lu, B.-L., Zhang, L., Kwok, J. (eds.) ICONIP 2011, Part II. LNCS, vol. 7063, pp. 735–744. Springer, Heidelberg (2011)
20. Japkowicz, N., Stephen, S.: The Class Imbalance Problem: A Systematic Study. Intelligent Data Analysis 6(5), 429–449 (2000)
21. Quinlan, J.R.: C4.5: programs for machine learning. Morgan Kaufmann, San Francisco (1993)
22. UCI Machine Learning Repository, `http://archive.ics.uci.edu/ml/`
23. Fawcett, T.: ROC Graphs: Notes and Practical Considerations for Data Mining Researchers. Technical Report HPL-2003-4, HP Labs (2003)
24. Corder, G.W., Foreman, D.I.: Nonparametric Statistics for Non-Statisticians: A step-by-Step Approach. Wiley, New York (2009)
25. Critical Value Table of Wilcoxon Signed-Ranks Test, `http://www.sussex.ac.uk/Users/grahamh/RM1web/WilcoxonTable2005.pdf`

Differential Privacy Preserving Spectral Graph Analysis

Yue Wang, Xintao Wu, and Leting Wu

University of North Carolina at Charlotte
{ywang91,xwu,lwu8}@uncc.edu

Abstract. In this paper, we focus on differential privacy preserving spectral graph analysis. Spectral graph analysis deals with the analysis of the spectra (eigenvalues and eigenvector components) of the graph's adjacency matrix or its variants. We develop two approaches to computing the ϵ-differential eigen decomposition of the graph's adjacency matrix. The first approach, denoted as *LNPP*, is based on the Laplace Mechanism that calibrates Laplace noise on the eigenvalues and every entry of the eigenvectors based on their sensitivities. We derive the global sensitivities of both eigenvalues and eigenvectors based on the matrix perturbation theory. Because the output eigenvectors after perturbation are no longer orthogonormal, we postprocess the output eigenvectors by using the state-of-the-art vector orthogonalization technique. The second approach, denoted as *SBMF*, is based on the exponential mechanism and the properties of the matrix Bingham-von Mises-Fisher density for network data spectral analysis. We prove that the sampling procedure achieves differential privacy. We conduct empirical evaluation on a real social network data and compare the two approaches in terms of utility preservation (the accuracy of spectra and the accuracy of low rank approximation) under the same differential privacy threshold. Our empirical evaluation results show that *LNPP* generally incurs smaller utility loss.

Keywords: differential privacy, spectral graph analysis, privacy preservation.

1 Introduction

There have been attempts [1–3] to formalize notions of differential privacy in releasing aggregate information about a statistical database and the mechanism to providing privacy protection to participants of the databases. Differential privacy [1] is a paradigm of post-processing the output of queries such that the inclusion or exclusion of a single individual from the data set make no statistical difference to the results found. Differential privacy is usually achieved by directly adding calibrated laplace noise on the output of the computation f. The calibrating process of this approach includes the calculation of the global sensitivity of the computation f that bounds the possible change in the computation output over any two neighboring databases. The added noise is generated

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 329–340, 2013.

from a Laplace distribution with the scale parameter determined by the global sensitivity of f and the user-specified privacy threshold ϵ. This approach works well for traditional aggregate functions (often with low sensitivity values) over tabular data. In [4], McSherry and Talwar introduced a general mechanism with differential privacy that comes with guarantees about the quality of the output, even for functions that are not robust to additive noise. The idea is to sample from the distribution specified by the exponential mechanism distribution. This mechanism skews a base measure to the largest degree possible while ensuring differential privacy, focusing probability on the outputs of highest value.

In this paper, we focus on differential privacy preserving spectral graph analysis. Spectral graph analysis deals with the analysis of the spectra (eigenvalues and eigenvector components) of the graph's adjacency matrix or its variants. We develop two approaches to computing the ϵ-differential private spectra, the first k eigenvalues and the corresponding eigenvectors, from the input graph G. The first approach, denoted as *LNPP*, is based on the Laplace Mechanism [1] that calibrates Laplace noise on the eigenvalues and every entry of the eigenvectors based on their sensitivities. We derive the global sensitivities of both eigenvalues and eigenvectors based on the matrix perturbation theory [5]. Because the output eigenvectors after perturbation are no longer orthogonormal, we postprocess the output eigenvectors by using the state-of-the-art vector orthogonalization technique [6]. The second approach, denoted as *SBMF*, is based on the exponential mechanism [4] and the properties of the matrix Bingham-von Mises-Fisher density for network data spectral analysis [7]. We prove that the Gibbs sampling procedure [7] achieves differential privacy. We conduct empirical evaluation on a real social network data and compare the two approaches in terms of utility preservation (the accuracy of spectra and the accuracy of low rank approximation) under the same differential privacy threshold. Our empirical evaluation results show that *L*NPP generally incurs smaller utility loss.

2 Preliminaries

2.1 Differential Privacy

We revisit the formal definition and the mechanism of differential privacy. For differential privacy, a database is treated as a collection of *rows*, with each row corresponding to an individual record. Here we focus on how to compute graph statistics (eigen-pairs) from private network topology described as its adjacency matrix. We aim to ensure that the inclusion or exclusion of a link between two individuals from the graph make no statistical difference to the results found.

Definition 1. *(Differential Privacy [1]) A graph analyzing algorithm Ψ that takes as input a graph G, and outputs $\Psi(G)$, preserves ϵ-differential edge privacy if for all closed subsets S of the output space, and all pairs of neighboring graphs G and G' from $\Gamma(G)$,*

$$Pr[\Psi(G) \in S] \leq e^{\varepsilon} \cdot Pr[\Psi(G') \in S], \tag{1}$$

where $\Gamma(G) = \{G'(V, E') | \exists!(u, v) \in G \text{ but } (u, v) \notin G'\}$.

A differentially private algorithm provides an assurance that the probability of a particular output is almost the same no matter whether any particular edge is included or not. A general method for computing an approximation to any function while preserving ϵ-differential privacy is given in [1]. This mechanism for achieving differential privacy computes the sum of the true answer and random noise generated from a Laplace distribution. The magnitude of the noise distribution is determined by the sensitivity of the computation and the privacy parameter specified by the data owner. The sensitivity of a computation bounds the possible change in the computation output over any two neighboring graphs (differing at most one link).

Definition 2. *(Global Sensitivity [1]) The global sensitivity of a function* $f : D \rightarrow \boldsymbol{R}^d$ *(*$G \in D$*),in the analysis of a graph* G*, is*

$$GS_f(G) := \max_{G, G' s.t. G' \in \Gamma(G)} \|f(G) - f(G')\|_1 \tag{2}$$

Theorem 1. *(The Laplace Mechanism [1]) An algorithm* A *takes as input a graph* G*, and some* $\varepsilon > 0$*, a query* Q *with computing function* $f : D^n \rightarrow \boldsymbol{R}^d$*, and outputs*

$$\boldsymbol{A}(G) = f(G) + (Y_1, ..., Y_d) \tag{3}$$

where the Y_i *are drawn i.i.d from* $Lap(GS_f(G)/\varepsilon)$*. The Algorithm satisfies* ϵ*-differential privacy.*

Another exponential mechanism was proposed to achieve differential privacy for diverse functions especially those with large sensitivities [4]. The exponential mechanism is driven by a score function q that maps a pair of input(G) and output(r) from $D^n \times \boldsymbol{R}^d$ to a real valued score($q(G, r)$) which indicates the probability associated with the output. Given an input graph G, an output $r \in \boldsymbol{R}^d$ is returned such that $q(G, r)$ is approximately maximized while guaranteeing differential privacy.

Theorem 2. *(The General Exponential Mechanism [4]) For any function* q*:* $(D^n \times \boldsymbol{R}^d) \rightarrow \mathbb{R}$*, based on a query* Q *with computing function* $f : D^n \rightarrow \boldsymbol{R}^d$*, and base measure* μ *over* $\boldsymbol{R}^d$*, the algorithm* Υ *which takes as input a graph* G *and some* $\alpha > 0$ *and outputs some* $r \in \boldsymbol{R}^d$ *is defined as*

$$\Upsilon_q^\alpha(G) := Choosing\ r\ with\ probability\ proportional\ to\ exp(\alpha q(G, r)) \times \mu(r).$$

$\Upsilon_q^\alpha(G)$ *gives* $(2\alpha\Delta q)$*-differential privacy, where* Δq *is the largest possible difference in* q *when applied to two input graphs that differ only one link, for all* r*.*

Theorem 3. *(Composition Theorem [2]) If we have* n *numbers of* ϵ*-differentially private mechanisms* $M_1, \cdots, M_n$*, computed using graph* G*, then any composition of these mechanisms that yields a new mechanism* M *is* $n\varepsilon$*-differentially private.*

Differential privacy can extend to group privacy as well: changing a group of k edges in the data set induces a change of at most a multiplicative $e^{k\epsilon}$ in the corresponding output distribution. In this paper, we focus on the edge privacy. We can extend the algorithm to achieve the node privacy by using the composition theorem [2].

2.2 Spectral Analysis of Network Topologies

A graph G can be represented as a symmetric adjacent matrix $A_{n \times n}$ with $A_{i,j} = 1$ if there is an edge between nodes i and j, and $A_{i,j} = 0$ otherwise. We denote the i-th largest eigenvalue of A by λ_i and the corresponding eigenvector by $\boldsymbol{u}_i$. The eigenvector $\boldsymbol{u}_i$ is a $n \times 1$ column vector of length 1. The matrix A can be decomposed as

$$A = \sum_{i=1}^{n} \lambda_i \boldsymbol{u}_i \boldsymbol{u}_i^T. \tag{4}$$

One major application of the spectral decomposition is to approximate the graph data A by a low dimension subspace A_k that captures the main information of the data, i.e., minimizes $\|A - A_k\|_F$. Given the top-k eigenvalues and corresponding eigenvectors, we have a rank-k approximation to A as

$$A_k = \sum_{i=1}^{k} \lambda_i \boldsymbol{u}_i \boldsymbol{u}_i^T = U_k \Lambda_k {U_k}^T, \tag{5}$$

where Λ_k is a diagonal matrix with $\Lambda_{ii} = \lambda_i$ and $U_k = (\boldsymbol{u}_1, .., \boldsymbol{u}_k)$.

U_k belongs to the Stiefel manifold. Denoted as $\nu_{k,n}$, the Stiefel manifold is defined as the set of rank-k $k \times n$ orthonormal matrices. One of the commonly used probability distributions on the Stiefel manifold $\nu_{k,n}$ is called the matrix Bingham-von Mises-Fisher density (Definition 3).

Definition 3. *(The matrix Bingham-von Mises-Fisher density [7]) The probability density of the matrix Bingham-von Mises-Fisher distribution is given by*

$$\mathbb{P}_{\mathrm{BMF}}(X|C_1, C_2, C_3) \propto etr\{C_3^T X + C_2 X^T C_1 X\}, \tag{6}$$

where C_1 and C_2 are assumed to be symmetric and diagonal matrices, repectively.

The matrix Bingham-von Mises-Fisher density arises as a posterior distribution in latent factor models for multivariate and relational data. Recently, a Gibbs sampling scheme was developed for sampling from the matrix Bingham-von Mises-Fisher density with application of network spectral analysis [7] based on the latent factor model(Definition 4).

Definition 4. *(The latent factor model for network data [7]) The network data is represented with a binary matrix A so that $A_{i,j}$ is the 0-1 indicator of a link between nodes i and j. The latent factor model with a probit link for such network data is defined as:*

$$\begin{aligned} A_{i,j} &= \delta_{(c,\infty)}\left(Z_{i,j}\right) \\ Z_{i,j} &= \boldsymbol{u}_i^T \Lambda \boldsymbol{u}_j + e_{i,j} \\ Z &= U \Lambda U^T + E \end{aligned}$$

where E is modeled as a symmetric matrix of independent normal noise, Λ is a diagonal matrix and U is an element of $\nu_{k,n}$, with k generally much smaller than n. Given a uniform prior distribution for U, we have

$$\mathbb{P}(U|Z, \Lambda) \propto etr(Z^T U \Lambda U^T/2) = etr(\Lambda U^T Z U/2),$$

which is a Bingham distribution with parameters $C_1 = Z/2$, $C_2 = \Lambda$ and $C_3 = 0$.

Lemma 1. *[7]A uniform prior distribution on eigenvectors U and independent normal(0, τ^2) prior distributions for the eigenvalues Λ give*

$$\mathbb{P}(\Lambda|Z, U) = \Pi_{i=1}^{k} normal(\tau^2 \boldsymbol{u}_i^T Z \boldsymbol{u}_i/(2+\tau^2), 2\tau^2/(2+\tau^2))$$
$$\mathbb{P}(U|Z, \Lambda) \propto etr(Z^T U \Lambda U^T/2) = etr(\Lambda U^T Z U/2),$$

where 'normal(u, σ^2)' denotes the normal density with mean u and variance σ^2.

The sampling scheme by Hoff [7] ensures Lemma 1 to approximate inferences for U and Λ for a given graph topology. As suggested in [7], the prior parameter τ^2 is usually chosen as the number of nodes n since this is roughly the variance of the eigenvalues of an $n \times n$ matrix of independent standard normal noise.

3 Mechanism for Spectral Differential Privacy

In this section, we present two approaches to computing the ϵ-differential private spectra: *LNPP*, which is based on the Laplace Mechanism (Theorem1), and *SBMF*, which is based on the exponential mechanism [4] and the properties of the matrix Bingham-von Mises-Fisher density for network data spectral analysis [7].

3.1 LNPP: Laplace Noise Perturbation with Postprocessing

In this approach, we output the first k eigenvalues, $\boldsymbol{\lambda}^{(k)} = (\lambda_1, \lambda_2, ..., \lambda_k)$, and the corresponding eigenvectors, $U_k = (\boldsymbol{u}_1, \boldsymbol{u}_2, ..., \boldsymbol{u}_k))$, under ϵ-differential privacy with the given graph G and parameters k, ϵ. We first derive the sensitivities for the eigenvalues and eigenvectors in Results 1, 2. We then follow Theorem 1 to calibrate Laplace noise to the eigenvalues and eigenvectors based on the derived sensitivities and privacy parameter. Because the perturbed eigenvectors will no longer be orthogonalized to each other, we finally do a postprocess to normalize and orthogonalize the perturbed eigenvectors following Theorem 4.

Result 1. *Given a graph G with its adjacent matrix A, the global sensitivity of each eigenvalue is $GS_{\lambda_i}(G) = 1$,($i \in [1, n]$); the global sensitivity of the first $k(k > 1)$ eigenvalues as a vector, $\boldsymbol{\lambda}^{(k)} = (\lambda_1, \lambda_2, ..., \lambda_k)$, is $GS_{\boldsymbol{\lambda}^{(k)}}(G) = \sqrt{2k}$.*

Proof. We denote adding/deleting an edge between nodes i and j on the original graph G as a perturbation matrix P added to the original adjacent matrix A. $P_{n\times n}$ is a symmetric matrix where only $P_{i,j}$ and $P_{j,i}$ have value $1/-1$ and all other entries are zeros. We denote λ_i as the eigenvalue of the matrix A and $\tilde{\lambda}_i$

as that of matrix $A + P$. We have the Euclidean norm and Frobenius norm of P respectively as $\|P\|_2 = 1$ and $\|P\|_F = \sqrt{2}$. Based on the matrix perturbation theory [5](Chapter IV, Theorem 4.11), we have

$$GS_{\lambda_i}(G) \leq \max |\tilde{\lambda}_i - \lambda_i| \leq \|P\|_2 = 1$$

and

$$GS_{\boldsymbol{\lambda}^{(k)}}(G) = \sum_{i=1}^{k} |\tilde{\lambda}_i - \lambda_i| \leq \sqrt{k}\sqrt{\sum_{i=1}^{k}(\tilde{\lambda}_i - \lambda_i)^2} \leq \sqrt{k}\|P\|_F = \sqrt{2k}.$$

Result 2. *Given a graph G with its adjacent matrix A, the sensitivity of each eigenvector, $\boldsymbol{u}_i (i > 1)$, is $GS_{\boldsymbol{u}_i}(G) = \frac{\sqrt{n}}{\min\{|\lambda_i - \lambda_{i-1}|, |\lambda_i - \lambda_{i+1}|\}}$, where the denominator is commonly referred as the eigen-gap of λ_i. Specifically, the sensitivities of the first and last eigenvector are respectively $GS_{\boldsymbol{u}_1}(G) = \frac{\sqrt{n}}{\lambda_1 - \lambda_2}$ and $GS_{\boldsymbol{u}_n}(G) = \frac{\sqrt{n}}{\lambda_{n-1} - \lambda_n}$.*

Proof. We define the perturbation matrix P and other terminologies the same as those in the proof of Result 1. We denote eigenvectors of matrix $A, A + P$ respectively as column vectors $\boldsymbol{u}_i$ and $\tilde{\boldsymbol{u}}_i$ $(i \in [1, k])$. Based on the matrix perturbation theory [5](Chapter V, Theorem 2.8), for each eigenvector $\boldsymbol{u}_i (i > 1)$, we have

$$\begin{aligned} GS_{\boldsymbol{u}_i}(G) \leq \sqrt{n}\|\tilde{\boldsymbol{u}}_i - \boldsymbol{u}_i\|_2 &\leq \frac{\sqrt{n}\|P\boldsymbol{u}_i\|_2}{\min\{|\lambda_i - \lambda_{i-1}|, |\lambda_i - \lambda_{i+1}|\}} \\ &\leq \frac{\sqrt{n}}{\min\{|\lambda_i - \lambda_{i-1}|, |\lambda_i - \lambda_{i+1}|\}}. \end{aligned}$$

Specifically for $i = 1$ (similarly for $i = n$),

$$GS_{\boldsymbol{u}_1}(G) \leq \sqrt{n}\|\tilde{\boldsymbol{u}}_1 - \boldsymbol{u}_1\|_2 \leq \frac{\sqrt{n}\|P\|_2}{\lambda_1 - \lambda_2} = \frac{\sqrt{n}}{\lambda_1 - \lambda_2}.$$

Theorem 4. *(Orthogonalization of vectors with minimal adjustment [6]) Given a set of non-orthogononal vectors $\boldsymbol{x}_1, ..., \boldsymbol{x}_k$, we could construct components $\boldsymbol{u}_1, ..., \boldsymbol{u}_k$ such that $\boldsymbol{x}_i$ is close to $\boldsymbol{u}_i$ for each i, and $U^T U$ is an identity matrix where $U = (\boldsymbol{u}_1, ..., \boldsymbol{u}_k)$ following*

$$U = XC,$$

where $X = (\boldsymbol{x}_1, ..., \boldsymbol{x}_k)$ is the set of $n \times 1$vectors and $X^T X$ is non-singular, C is the symmetric square-root of $(X^T X)^{-1}$ and is unique.

Algorithm 1. *LNPP: Laplace noise calibration approach*

```
Input: Graph adjacent matrix $A$, privacy parameter $\epsilon$ and dimension parameter $k$
Output: The first $k$ eigenvalues $\widetilde{\boldsymbol{\lambda}}^{(k)} = (\tilde{\lambda}_1, \tilde{\lambda}_2, ..., \tilde{\lambda}_k)$ and corresponding eigenvectors $\tilde{\boldsymbol{u}}_1, \tilde{\boldsymbol{u}}_2, ..., \tilde{\boldsymbol{u}}_k$, which satisfies $\epsilon$-differential privacy.
1: Decomposition $A$ to obtain the first $k$ eigenvalues $\boldsymbol{\lambda}^{(k)} = (\lambda_1, \lambda_2, ..., \lambda_k)$ and the corresponding eigenvectors $\boldsymbol{u}_1, \boldsymbol{u}_2, ..., \boldsymbol{u}_k$;
2: Distribute $\epsilon$ into $\epsilon_0, ..., \epsilon_k$, s.t.$\epsilon = \sum_{i=0}^{k} \epsilon_i$;
3: Follow Theorem 1 to add Laplace noise to $\boldsymbol{\lambda}^{(k)}$ with $\epsilon_0$ based on $GS_{\boldsymbol{\lambda}^{(k)}}(G)$ derived in Result 1 and obtain $\widetilde{\boldsymbol{\lambda}}^{(k)} = (\tilde{\lambda}_1, ..., \tilde{\lambda}_k)$;
4: For i:=1 to k do
       Follow Theorem 1 to add Laplace noise to $\boldsymbol{u}_i$ with $\epsilon_i$ based on $GS_{\boldsymbol{u}_i}(G)$ derived in Result 2 and obtain $\tilde{\boldsymbol{x}}_i$;
   Endfor
5: Normalize and orthogonalize $\tilde{\boldsymbol{x}}_1, ..., \tilde{\boldsymbol{x}}_k$ to obtain $\tilde{\boldsymbol{u}}_1, ..., \tilde{\boldsymbol{u}}_k$ following Theorem 4.
6: Output $\tilde{\lambda}_1, \tilde{\lambda}_2, ..., \tilde{\lambda}_k$ and $\tilde{\boldsymbol{u}}_1, \tilde{\boldsymbol{u}}_2, ..., \tilde{\boldsymbol{u}}_k$
```

Algorithm 1 illustrates our *LNPP* approach. We output the first k eigenvalues, $\widetilde{\boldsymbol{\lambda}}^{(k)} = (\tilde{\lambda}_1, \tilde{\lambda}_2, ..., \tilde{\lambda}_k)$, and the corresponding eigenvectors, $\tilde{\boldsymbol{u}}_1, \tilde{\boldsymbol{u}}_2, ..., \tilde{\boldsymbol{u}}_k$), under ϵ-differential privacy with the given graph topology A and parameters k, ϵ. We first compute the real values of eigenvalues $\boldsymbol{\lambda}^{(k)}$ and eigenvectors $\boldsymbol{u}_i (i \in [1, k])$ from the given graph adjacent matrix A (Line 1). Then we distribute the privacy parameter ϵ among $\boldsymbol{\lambda}^{(k)}$ and $\boldsymbol{u}_1, \boldsymbol{u}_2, ..., \boldsymbol{u}_k$ respectively as ϵ_0 and $\epsilon_1, \epsilon_2, ..., \epsilon_k$ where $\epsilon = \sum_{i=0}^{k} \epsilon_i$ (Line 2). With the derived the sensitivities for the eigenvalues $(GS_{\boldsymbol{\lambda}^{(k)}}(G))$ and each of the k eigenvectors $(GS_{\boldsymbol{u}_i}(G), i \in [1, k])$ from Results 1 and 2, next we follow Theorem 1 to calibrate Laplace noise and obtain the private answers $\widetilde{\boldsymbol{\lambda}}^{(k)}$ (Line 3) and $\tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}_2, ..., \tilde{\boldsymbol{x}}_k$ (Line 4). Finally we do a postprocess to normalize and orthogonalize $\tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}_2, ..., \tilde{\boldsymbol{x}}_k$ into $\tilde{\boldsymbol{u}}_1, \tilde{\boldsymbol{u}}_2, ..., \tilde{\boldsymbol{u}}_k$ following Theorem 4 (Line 5).

3.2 SBMF: Sampling from *BMF* Density

The *SBMF* approach to provide spectral analysis of network data is based on the sampling scheme proposed by Hoff [7] as an application of their recently-proposed technique of sampling from the matrix Bingham-von Mises-Fisher density (Definitions 3, 4). In [8], the authors investigated differentially private approximations to principle component analysis and also developed a method based on the general exponential mechanism [4]. In our work we focus on the eigen-decomposition of the 0-1 adjacency matrix (rather than the second moment matrix of the numerical data) and prove that the sampling scheme from the matrix Bingham-von Mises-Fisher density satisfies differential privacy through the general exponential mechanism (Theorem 2). The sampling scheme proposed by Hoff [7] ensures Lemma 1, with the purpose to build the latent factor model (Definition 4) for network data, i.e, to approximate inferences for U and Λ. We derive the privacy bounds of the output eigenvalues and eigenvectors following the sampling scheme respectively in Claims 1 and 2, based on Lemma 1. Then following the

Composition Theorem (Theorem 3), we come to the conclusion that the *SBMF* approach satisfies ϵ-differential privacy (Theorem 5).

Claim 1. *The sampling scheme which outputs $\boldsymbol{\lambda}^{(k)}$ satisfies ϵ_Λ-differential privacy where $\epsilon_\Lambda = k(\frac{2\tau^2}{2+\tau^2})^{3/2}$.*

Proof. We denote A and A' as the adjacent matrix of any neighboring graph G and G'. The calibrated noise to a function f from the Gaussian distribution $normal(0, \sigma^2)$, similar as that from the Laplace distribution, provides a $2\sigma GS_f$-differential privacy [1]. Based on Lemma 1, we have for each eigenvalue λ_i, the sampling scheme satisfies

$$\begin{aligned}\epsilon_{\lambda_i} &= 2\sigma GS_{\lambda_i} = 2(\frac{2\tau^2}{2+\tau^2})^{1/2}\{\tau^2 \boldsymbol{u}_i^T A \boldsymbol{u}_i/(2+\tau^2) - \tau^2 \boldsymbol{u}_i^T A' \boldsymbol{u}_i/(2+\tau^2)\} \\ &= 2(\frac{2\tau^2}{2+\tau^2})^{1/2} \frac{\tau^2}{2+\tau^2} \boldsymbol{u}_i^T (A - A')\boldsymbol{u}_i \leq (\frac{2\tau^2}{2+\tau^2})^{3/2}\end{aligned}$$

where the proof of $\boldsymbol{u}_i^T(A - A')\boldsymbol{u}_i \leq 1$ is straightforward. With the composition theorem (Theorem 3), $\epsilon_\Lambda = \sum_{i=1}^k \epsilon_{\lambda_i} = k(\frac{2\tau^2}{2+\tau^2})^{3/2}$.

Claim 2. *Given the graph G's adjacent matrix A, the sampling scheme which outputs U satisfies ϵ_U-differential privacy where $\epsilon_U = k^2\lambda_1$.*

Proof. The sampling scheme for U can be considered as an instance for the exponential mechanism(Theorem 2) with $\alpha = 1$ and $q(A, U) = tr(\Lambda U^T A U/2)$. We have

$$\begin{aligned}\Delta q(A, U) &= \left|tr(\Lambda U^T A U/2) - tr(\Lambda U^T A' U/2)\right| = \frac{1}{2}\left|tr(\Lambda U^T (A - A')U)\right| \\ &\leq \frac{1}{2}k\lambda_1 \left|tr(U^T(A - A')U)\right| \leq \frac{1}{2}k^2\lambda_1.\end{aligned}$$

Following Theorem 2, we have $\epsilon_U = 2\alpha\Delta q(A, U) = k^2\lambda_1$.

Theorem 5. *The* SBMF *approach to computing the spectra, the first k eigenvalues and the corresponding eigenvectors of a given graph topology A satisfies $\epsilon = (\epsilon_\Lambda + \epsilon_U)$-differential privacy, where $\epsilon_\Lambda = k(\frac{2\tau^2}{2+\tau^2})^{3/2}$ and $\epsilon_U = \alpha k^2\lambda_1$.*

In this work, we take the prior parameter τ^2 as n, which is suggested by Hoff [7] since this is roughly the variance of the eigenvalues of an $n \times n$ matrix of independent standard normal noise. We illustrate the *SBMF* approach in Algorithm 2. In the Algorithm, the parameter α is used to change the privacy magnitude by changing ϵ_U (Theorems 2, 5). Given the input graph topology A and dimension parameter k, we acquire the eigenvalues $\widetilde{\Lambda}_k$ and corresponding eigenvectors $\widetilde{U}_k$ from the sampler application provided by Hoff [7] with input matrix αA. The output satisfies $\epsilon = (\epsilon_\Lambda + \epsilon_U)$-differential privacy following Theorem 5.

Algorithm 2. *SBMF: Sampling from BMF density approach*

```
Input: Graph adjacent matrix $A_{n \times n}$, privacy magnitude $\alpha$ and dimension parameter $k$
Output: The first $k$ eigenvalues $\widetilde{\Lambda}_k$ and corresponding eigenvectors $\widetilde{U}_k$, which satisfies $\epsilon = (\epsilon_\Lambda + \epsilon_U)$-differential privacy.
Set the input matrix $Y = \alpha A$, the parameter $\tau^2 = n$ and the number of iterations $t$;
Acquire $\widetilde{\Lambda}_k$ and $\widetilde{U}_k$ from the sampler provided by Hoff [7] with the input matrix $Y$, the output satisfies $\epsilon = (\epsilon_\Lambda + \epsilon_U)$-differential privacy(Theorem 5);
Output $\widetilde{\Lambda}_k$ and $\widetilde{U}_k$
```

4 Empirical Evaluation

We conduct experiments to compare the performance of the two approaches, *LNPP* and *SBMF*, in producing the differentially private eigenvalues and eigenvectors. For the *LNPP*, we implement Algorithm 1. For the *SBMF*, we use the R-package provided by Hoff [7]. We use 'Enron' (147 nodes, 869 edges) data set that is derived from an email network [1] collected and prepared by the CALO Project. We take the dimension $k = 5$ since it has been suggested in previous literatures [9] that the first five eigenvalues and eigenvectors are sufficient to capture the main information of this graph. The first two rows in Table 1 show the eigenvalues and their corresponding eigen-gaps (Result 2).

4.1 Performance Comparison with $\alpha = 1$

In this section, we compare the performance of the *LNPP* approach with that of the *SBMF* approach in three aspects: the accuracy of eigenvalues, the accuracy of eigenvectors and the accuracy of graph reconstruction with the private eigenpairs. With $\tau^2 = n$ and $\alpha = 1$, we compute that $\epsilon_\lambda = 14$ and $\epsilon_U = 446$ following Claims 1 and 2. Therefore the *SBMF* approach satisfies $\epsilon = 460$ differential privacy following Theorem 5. On the other hand, the same ϵ is taken as the input for the *LNPP* approach. Different strategies have been proposed to address the ϵ distribution problem(Line 2 in Algorithm 1) in previous literatures [10, 11]. In our work, we just take one simple strategy, distributing ϵ as $\epsilon_0 = 10$ to the eigenvalues and $\epsilon_i = 90, (i \in [1, k])$ equally to each eigenvector. Therefore *LNPP* approach also satisfies $\epsilon = 460$ differential privacy.

For eigenvalues, we measure the output accuracy with the absolute error defined as $E_\Lambda = |\widetilde{\boldsymbol{\lambda}}^{(k)} - \boldsymbol{\lambda}^{(k)}|_1 = \sum_{i=1}^{k} |\widetilde{\lambda}_i - \lambda_i|$. The absolute errors E_Λ for *LNPP* and *SBMF* are respectively 0.9555 and 345.2301. One sample o eigenvalues In the third and fourth rows of Table 1, we show the output eigenvalues from the *LNPP* and the *SBMF* approaches. We can see that the *LNPP* outperforms the *SBMF* in more accurately capturing the original eigenvalues.

For eigenvectors, we define the absolute error as $E_U = |\widetilde{U}_k - U_k|_1$. E_U for *LNPP* and *SBMF* approaches are respectively 11.9989 and 13.4224. We also

[1] `http://www.cs.cmu.edu/~enron/`

Table 1. Eigenvalues Comparison

	λ_1	λ_2	λ_3	λ_4	λ_5
eigenvalue	17.8317	12.7264	10.6071	9.7359	9.5528
eigen-gap	5.1053	2.1193	0.8712	0.1832	0.1832
LNPP	18.1978	13.2191	10.6030	9.7311	9.4650
SBMF	107.8450	88.9362	76.1712	76.0596	56.6721

Table 2. Eigenvector Comparison

Approaches	E_U	$cos\langle \widetilde{\boldsymbol{u}}_i, \boldsymbol{u}_i \rangle = \widetilde{\boldsymbol{u}}_i' \cdot \boldsymbol{u}_i$				
		$\boldsymbol{u}_1$	$\boldsymbol{u}_2$	$\boldsymbol{u}_3$	$\boldsymbol{u}_4$	$\boldsymbol{u}_5$
LNPP	**11.9989**	**0.9591**	**0.7925**	0.4786	0.1217	0.1280
SBMF	13.4224	0.6605	0.6995	**0.7336**	**0.2921**	**0.4034**

define the cosine similarity to measure the accuracy of each private eigenvector as $cos\langle \widetilde{\boldsymbol{u}}_i, \boldsymbol{u}_i \rangle = \widetilde{\boldsymbol{u}}_i' \cdot \boldsymbol{u}_i (i \in [1, k])$. We show the detailed values of E_U and the cosine similarities in Table2. Note that the cosine value closer to 1 indicates better utility. We can see that *LNPP* generally outperforms *SBMF* in privately capturing eigenvectors that close to the original ones. Specifically, the *LNPP* approach is sensitive to eigen-gaps (second row in Table 1), i.e., it tends to show better utility when the eigen-gap is large such as for $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$. Thus a better strategy will be distributing privacy parameter ϵ according to magnitudes of eigen-gaps, instead of the equal distribution.

The *SBMF* approach outputs much larger eigenvalues than the original ones. It does not tend to accurately approximate anyone of the original eigenvectors either. The reason is that *SBMF* approach is designed to provide a low rank spectral model for the original graph rather than approximating of the original eigenvalues and eigenvectors.

We consider the application of graph reconstruction using the differentially private first k eigenvalues and the corresponding eigenvectors. $A_k = \sum_{i=1}^{k} \lambda_i \boldsymbol{u}_i \boldsymbol{u}_i^T = U_k \Lambda_k U_k^T$ is commonly used as a rank-k approximation to the original graph topology A when A is not available for privacy reasons or A's rank is too large for analysis. Since A_k is not an 0/1 matrix, We discretize A_k as $\widetilde{A}_k^1$ by choosing the largest $2m$ entries as 1 and all others as 0 (so keeping the number of edges m the same as that of the original graph). We then compare the performance of the two approaches by the absolute reconstruction error defined as $\gamma = \|A - \widetilde{A}_k^1\|_F$.The γ values for *LNPP* and *SBMF* approaches are 47.7912 and 34.1760 respectively. We can see that the result of the *SBMF* approach outperforms the *LNPP*.

4.2 Performance Comparison with Varying α

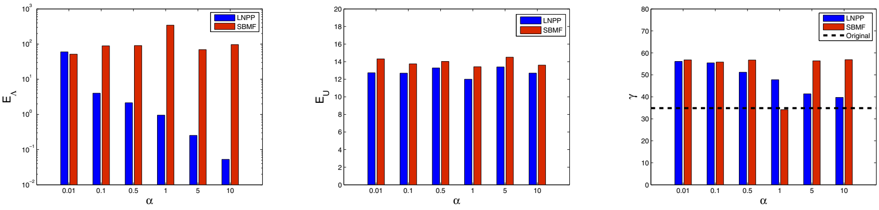
In this section, we change the privacy magnitude to additionally study the performance of the *LNPP* and *SBMF* approaches. α denotes the amplification factor of the privacy parameter ϵ used in section 4.1. We choose the value

Table 3. Comparison of two approaches for varying privacy magnitudes

	α	0.01	0.1	0.5	1	5	10
E_Λ	*LNPP*	60.1586	4.0160	2.1452	0.9555	0.2528	0.0527
	SBMF	51.6551	89.0678	90.9442	345.2301	69.6852	96.8904
E_U	*LNPP*	12.7419	13.2455	13.9874	11.9989	13.3967	12.7033
	SBMF	14.3155	13.7518	14.0238	13.4224	14.5114	13.6087
γ	*LNPP*	56.2139	55.4617	51.1859	47.4912	41.3763	39.7492
	SBMF	56.8155	55.8211	56.7450	34.1760	56.3715	56.9210

of α as $0.01, 0.1, 0.5, 1, 5, 10$ where the corresponding ϵ values are respectively $18.46, 58.6, 237, 460, 2244, 4474$ following Theorem 5.

We show the values of E_Λ, E_U and γ for the *LNPP* and the *SBMF* approaches in Table 3. The accuracy of the *LNPP* approach increases significantly with α for both the eigenvalues(E_Λ) and graph reconstruction (γ). Note that the greater the α, the weaker privacy protection, and hence the more utility preservation. However, the accuracy of eigenvectors measured by E_U is not changed much with α, as shown in Figure 1. This is because of the normalization of eigenvectors in the postprocess step. While the *SBMF* approach cannot accurately capture eigenvalues for any α value; as to graph reconstruction, the case of $\alpha = 1$ shows the best utility.



(a) E_Λ (b) E_U (c) γ

Fig. 1. Utility comparison for varying privacy magnitude

5 Conclusion

In this paper we have presented two approaches to enforcing differential privacy in spectral graph analysis. We apply and evaluate the Laplace Mechanism [1] and the exponential mechanism [4] on the differential privacy preserving eigen decomposition on the graph topology. In our future work, we will investigate how to enforce differential privacy for other spectral graph analysis tasks (e.g., spectral clustering based on graph's Laplacian and normal matrices). Nissim et al. [3] introduced a framework that calibrates the instance-specific noise with smaller magnitude than the worst-case noise based on the global sensitivity. We will study the use of smooth sensitivity and explore how to better distribute privacy budget in the proposed *LNPP* approach. We will also study how different sampling strategies in the proposed *SBMF* approach may affect the utility preservation.

Acknowledgments. This work was supported in part by U.S. National Science Foundation IIS-0546027, CNS-0831204, CCF-0915059, and CCF-1047621.

References

1. Dwork, C., McSherry, F., Nissim, K., Smith, A.: Calibrating noise to sensitivity in private data analysis. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 265–284. Springer, Heidelberg (2006)
2. Dwork, C., Lei, J.: Differential privacy and robust statistics. In: Proceedings of the 41st Annual ACM Symposium on Theory of Computing, pp. 371–380. ACM (2009)
3. Nissim, K., Raskhodnikova, S., Smith, A.: Smooth sensitivity and sampling in private data analysis. In: Proceedings of the Thirty-ninth Annual ACM Symposium on Theory of Computing, pp. 75–84. ACM (2007)
4. McSherry, F., Talwar, K.: Mechanism design via differential privacy. In: Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science, pp. 94–103. IEEE (2007)
5. Stewart, G., Sun, J.: Matrix perturbation theory. Academic Press, New York (1990)
6. Garthwaite, P., Critchley, F., Anaya-Izquierdo, K., Mubwandarikwa, E.: Orthogonalization of vectors with minimal adjustment. Biometrika (2012)
7. Hoff, P.: Simulation of the Matrix Bingham-von Mises-Fisher Distribution, With Applications to Multivariate and Relational Data. Journal of Computational and Graphical Statistics 18(2), 438–456 (2009)
8. Chaudhuri, K., Sarwate, A., Sinha, K.: Near-optimal algorithms for differentially-private principal components. In: Proceedings of the 26th Annual Conference on Neural Information Processing Systems (2012)
9. Wu, L., Ying, X., Wu, X., Zhou, Z.: Line orthogonality in adjacency eigenspace with application to community partition. In: Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence, pp. 2349–2354. AAAI Press (2011)
10. Xiao, X., Bender, G., Hay, M., Gehrke, J.: ireduct: Differential privacy with reduced relative errors. In: Proceedings of the ACM SIGMOD International Conference on Management of Data (2011)
11. Wang, Y., Wu, X., Zhu, J., Xiang, Y.: On learning cluster coefficient of private networks. In: Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (2012)

Sorted Nearest Neighborhood Clustering for Efficient Private Blocking

Dinusha Vatsalan and Peter Christen

Research School of Computer Science, College of Engineering and Computer Science
The Australian National University, Canberra ACT 0200, Australia
{dinusha.vatsalan,peter.christen}@anu.edu.au

Abstract. Record linkage is an emerging research area which is required by various real-world applications to identify which records in different data sources refer to the same real-world entities. Often privacy concerns and restrictions prevent the use of traditional record linkage applications across different organizations. Linking records in situations where no private or confidential information can be revealed is known as privacy-preserving record linkage (PPRL). As with traditional record linkage applications, scalability is a main challenge in PPRL. This challenge is generally addressed by employing a blocking technique that aims to reduce the number of candidate record pairs by removing record pairs that likely refer to non-matches without comparing them in detail. This paper presents an efficient private blocking technique based on a sorted neighborhood approach that combines k-anonymous clustering and the use of public reference values. An empirical study conducted on real-world databases shows that this approach is scalable to large databases, and that it can provide effective blocking while preserving k-anonymous characteristics. The proposed approach can be up-to two orders of magnitude faster than two state-of-the-art private blocking techniques, k-nearest neighbor clustering and Hamming based locality sensitive hashing.

Keywords: sorted neighborhood, nearest neighbor clustering, locality sensitive hashing, k-anonymity, reference values, scalability.

1 Introduction

Integrating large volumes of data from different sources is an important data pre-processing step in many data mining applications [1]. Since unique entity identifiers are not always available in all the databases to be linked, common identifying attributes, such as names and addresses, are often used to identify records that need to be reconciled to the same real-world entities. The degraded quality of data residing in databases (due to transcription errors, missing values, or inconsistent formats), makes the task of integrating databases challenging [2]. Integrating such data in the presence of data quality errors requires approximate comparison functions to be employed to identify if two entities are the same [1]. These comparison functions are often expensive in terms of computational complexity. A naive pair-wise comparison of two databases is of quadratic

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 341–352, 2013.

complexity in their sizes. This makes the record linkage process not scalable to large databases. The scalability problem has been studied by introducing two-step linkage algorithms that avoid all pair-wise comparisons by employing a blocking or indexing technique [3] in the first step, so that detailed comparisons using expensive similarity comparison functions are only made in the second step on a smaller number of candidate record pairs.

Linking data across databases from different organizations is more challenging due to privacy and confidentiality issues that often preclude exchanging sensitive information regarding the entities. Identifying which records in two databases have the same or approximately the same values for a set of attributes without revealing the actual values of these attributes is known as the problem of 'privacy-preserving record linkage' (PPRL) [4–6]. Blocking for PPRL needs to be conducted in such a way that no sensitive information that can be used to infer individual records and their attribute values is revealed to any party involved in the process, or to an external adversary. The scalability challenge of PPRL has been addressed by several recent approaches that adapt existing blocking techniques, such as standard blocking [7], mapping based blocking [8], clustering [9], and locality sensitive hashing [10], into a privacy-preserving context.

One popular blocking technique used in traditional record linkage is the sorted neighborhood approach [11, 12], where database tables are sorted according to a 'sorting key' over which a sliding window of fixed size is moved. Candidate record pairs are then generated from the records that are within the current window. This approach is very efficient compared to other blocking techniques in that its resulting number of candidate record pairs is $O((n_A + n_B)w)$, compared to $O((n_A \cdot n_B)/b)$ for other blocking techniques [3], where n_A and n_B are the number of records in the two databases to be linked, b is the number of blocks generated, and w is the size of the window. However, the use of sorted neighborhood methods for private blocking has so far not been studied.

We propose an efficient three-party blocking technique for PPRL based on the sorted neighborhood approach using a combination of two privacy techniques: k-anonymous clustering [13] and public reference values [14]. The aim of this approach is to efficiently create k-anonymous clusters represented by reference values from which candidate record pairs are generated, without revealing any information that can be used to infer individual records and their values.

The contributions of this paper are (1) an efficient blocking technique for PPRL based on the sorted neighborhood approach; (2) two variations to generate k-anonymous clusters; (3) an analysis of our solution regarding complexity, privacy, and quality; and (4) an empirical evaluation using real-world datasets. We compare our approach with two state-of-the-art three-party private blocking techniques, which are Karakasidis et al.'s [15] approach based on k-nearest neighbor clustering, and Durham's [16] approach based on Hamming based locality sensitive hashing.

The remainder of this paper is structured as follows. In the following section, we provide an overview of related work in private blocking. In Sect. 3 we describe our protocol using two small example datasets. In Sect. 4 we analyse the protocol and in Sect. 5 we validate these analyses through an empirical study. Finally we summarize our findings and provide directions for future research in Sect. 6.

2 Related Work

The use of a blocking technique is crucial in PPRL applications to make the linkage across large databases scalable [3]. Several approaches have been proposed for private blocking. Most work in PPRL that has investigated scalability has employed the basic standard blocking approach [17–20]. In traditional standard blocking, all records that have the same blocking key value (the value of a single attribute or a combination of attributes) are inserted into the same block, and only the records within the same block are compared in detail with each other in the comparison step. Each record is inserted into one block only [3]. Al-Lawati et al. [17] explored three methods of token blocking to group hash signatures of the TF-IDF distances of attribute values. Karakasidis et al. [20] proposed phonetic based private blocking where an encoding function, such as Soundex or NYSIIS [1], is used to group records that have similar (sounding) values into the same block. Generalization techniques, such as value generalization hierarchies [18], k-anonymity [13] and binning [21], have been used for private blocking to generalize records with similar characteristics into the same blocks.

Mapping based blocking [8] is another technique that has been employed in PPRL. Scannapieco et al. [22] and Yakout et al. [23] used a multi-dimensional embedded space into which attribute values are mapped while preserving the distances between these values. A clustering or nearest neighbor approach is then applied on these multi-dimensional objects to extract candidate record pairs.

Karakasidis et al. [15] used the k-nearest neighbor clustering algorithm to group records such that similar records are put into the same clusters, and each cluster consists of at least k elements to provide a k-anonymous privacy guarantee. Initially clusters are generated for a set of reference values that are shared by the database owners, such that each cluster consists of at least k reference values. Each database owner assigns their records into these clusters based on their similarity. These clusters are sent to a third party that merges the corresponding clusters to generate candidate record pairs.

Locality sensitive hashing (LSH) has recently been investigated as an efficient technique for scalable record linkage [10, 16]. LSH allows hashing of values in such ways that the likelihood that two similar values are hashed into the same block can be specified through the use of certain hashing functions. Durham [16] investigated how LSH can be applied in Bloom filter based PPRL to reduce the number of record pair comparisons. A Bloom filter is a bit array data structure where hash functions are used to map a set of elements (q-grams extracted from attribute values) into the bit array. For private blocking, an iterative pruning approach is employed, where random bits are sampled at each iteration from the Bloom filters and sent to a third party. The third party then uses Hamming based LSH functions to compute the Hamming distance that allows efficient generation of candidate record pairs.

D^A

RecID	Surname	GivenName	SKV
RA1	millar	robert	millarrobert
RA2	marten	amyas	martenamyas
RA3	melar	gail	melargail
RA4	miller	robart	millerrobart
RA5	morley	william	morleywilliam
RA6	philips	colin	philipscolin
RA7	smith	alisen	smithalisen
RA8	sampson	taylor	sampsontaylor

Reference values

R'

1	*millar*
2	*myler*
3	*robinson*
4	*smith*

D^B

RecID	Surname	GivenName	SKV
RB1	millar	robert	millarrobert
RB2	marris	roberto	marrisroberto
RB3	malar	gayle	malargayle
RB4	perris	charles	perrischarles
RB5	robbins	william	robbinswilliam
RB6	robertson	amy	robertsonamy
RB7	samuell	tailor	samuelltailor
RB8	smeeth	rupert	smeethrupert
RB9	smith	alison	smithalison

Fig. 1. Example databases held by Alice ($\mathbf{D^A}$) and Bob ($\mathbf{D^B}$) with surname and given name attributes and their sorting key values (SKVs), and a list of reference values ($\mathbf{R}'$) along with their position values, used to illustrate the protocol described in Sect. 3

3 Sorted Neighborhood Based Private Blocking

In this section we describe the steps of our sorted neighborhood clustering (SNC) based private blocking method, and illustrate it with an example consisting of two small databases with given names and surnames, as shown in Fig. 1. Assume *Alice* and *Bob* are the two owners of their respective databases $\mathbf{D^A}$ and $\mathbf{D^B}$, and *Carol* is the trusted third party. Alice and Bob share the sorted reference list $\mathbf{R}'$ containing n_R reference values selected from the publicly available reference dataset $\mathbf{R}$. Fig. 2 illustrates the three-party setting for private blocking.

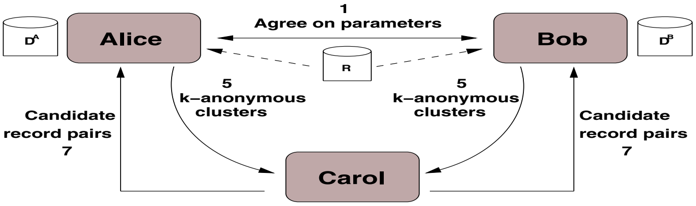


Fig. 2. Three-party private blocking. Numbers given correspond to the steps described in Sect. 3 that involve an exchange of data between parties

The two database owners Alice and Bob perform the following steps:

1. Agree upon the list of attributes to be used as the sorting keys, the reference dataset $\mathbf{R}$, the number of reference values to be used n_R, the minimum number of elements in a cluster k, a similarity function $sim(\cdot,\cdot)$ to compare reference values, and the minimum similarity threshold s_t used to decide if two clusters are to be merged in the SNC-Sim approach described in Step 4.
2. Alice and Bob each selects and sorts n_R ($n_R \leq |\mathbf{R}|$) reference values. The value for n_R can be chosen as $n_R = \min(|\mathbf{D^A}|, |\mathbf{D^B}|)/k$, so that each cluster will contain roughly around k database records. It is important to note that both Alice and Bob have the same list of sorted reference values $\mathbf{R}'$ at the end of this step. A secret random seed shared by Alice and Bob (but not known to Carol) can be used to select the same set of values from $\mathbf{R}$ into $\mathbf{R}'$ by both Alice and Bob.

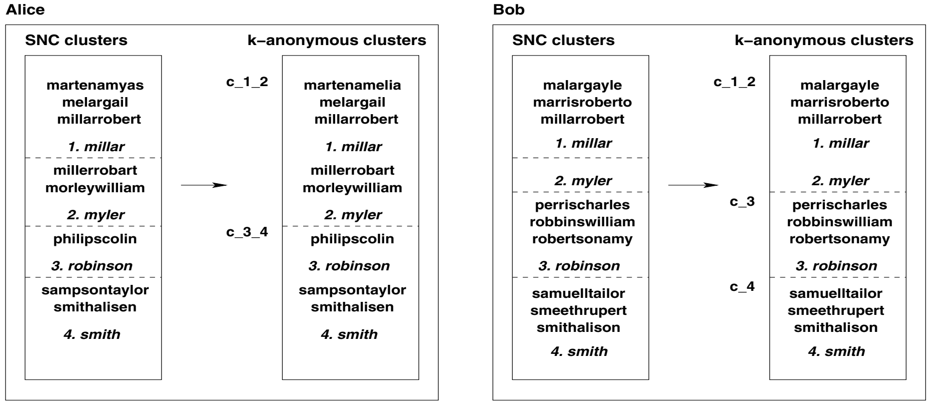


Fig. 3. Insertion of SKVs into the sorted list of reference values where each cluster is represented by one reference value (shown in italics font), and merging of clusters to create k-anonymous clusters (clusters that contain at least k SKVs) where each cluster is represented by one or more reference values (in this example, $k = 3$).

3. Alice and Bob individually insert their records based on the records' sorting key values (SKVs) into the sorted list of reference values to create SNC clusters, as shown in Fig. 3. An inverted index data structure can be used to efficiently insert records where the keys are the reference values and the corresponding values contain a list of SKVs which are lexicographically sorted before the reference value.
4. The next step is to create k-anonymous clusters. After inserting records into the sorted reference values there will be n_R clusters each represented by one reference value. However, to provide a k-anonymous privacy guarantee, each database owner has to merge their clusters in such a way that each cluster contains at least k database records. Cluster IDs are assigned to the merged clusters such that they consist of the position values of the reference values that reside in the merged clusters. We use cluster IDs of the form '$c\_x\_(x{+}1)\cdots\_(x{+}y)$', where x is the position value of the first reference value of the sorted reference values in the corresponding cluster, and $(y+1)$ is the number of reference values in that cluster. The merging of clusters to create k-anonymous clusters is shown in Fig. 3. This merging process can be done in two different ways.
   (a) SNC-Sim: Clusters are merged until the number of elements in them becomes greater than or equal to k and the similarity between reference values of adjacent clusters becomes less than the threshold s_t. This approach follows recent work on adaptive sorted neighborhood for duplicate detection [12]. Algo. 1 shows the main steps involved in this method. The clusters are merged until their size becomes greater than or equal to k (line 5 in Algo. 1). If the size of the (merged) cluster is greater than k, we compute the similarity of the next cluster's reference value r_{i+j+1} with the current cluster's reference value r_{i+j}, and if this similarity value $sim(r_{i+j+1}, r_{i+j})$

Algorithm 1. Merging Clusters using Sim

```
Input:
- $\mathbf{R}'$: List of sorted reference values $(r_1, \ldots, r_{n_R})$
- $\mathbf{S}$: Set of clusters $(c : [v_1, \ldots, v_l])$
- Minimum number of elements in a cluster $k$
- Minimum similarity threshold $s_t$
- Similarity comparison function $sim(\cdot, \cdot)$
Output:
- $\mathbf{O}$: Set of $k$-anonymous clusters
  $(`c\_x \ldots \_(x + y)' : [v_1, \ldots, v_k])$
1:  $i = 0$
2:  while $i < n_R$ do
3:      clus_vals = []; clus_id = ‘c_’
4:      num_vals = 0; sim_val = 0.0; $j = 0$
5:      while (num_vals $\leq k$ and $i + j < n_R$)
        or (sim_val $\geq s_t$ and $i + j < n_R$) do
6:          $r_i = \mathbf{R}'[i + j]$; $b = \mathbf{S}[r_{i+j}]$
7:          num_vals += len(b)
8:          clus_vals += b
9:          sim_val = $sim(r_{i+j}, r_{i+j+1})$
10:         clus_id += str(i+j)+‘_’
11:         $j$ += 1
12:     $\mathbf{O}$[clus_id] = clus_vals
13:     $i$ += $j$
```

Algorithm 2. Merging Clusters using Size

```
Input:
- $\mathbf{S}$: Set of clusters $(c : [v_1, \ldots, v_l])$
- Minimum number of elements in a cluster $k$
Output:
- $\mathbf{O}$: Set of $k$-anonymous clusters
  $(`c\_x \ldots \_(x + y)' : [v_1, \ldots, v_k])$
1:  ids = []; sizes = []
2:  for $(r, b) \in \mathbf{S}$ do
3:      ids += [r]; sizes += [len(b)]
4:  min_size = min(sizes)
5:  while min_size $< k$ and len(ids) $> 1$ do
6:      min_size_clus=$\mathbf{S}$.getID(len(b)=min_size)
7:      clus_vals=$\mathbf{S}$[min_size_clus]
8:      $i$ = ids.getindex(min_size_clus)
9:      prev_clus = ids[i-1]; next_clus = ids[i+1]
10:     if len($\mathbf{S}$[prev_clus]) < len($\mathbf{S}$[next_clus])
11:         clus_id = min_size_clus + prev_clus
12:         clus_vals += $\mathbf{S}$[prev_clus]
13:     else
14:         clus_id = min_size_clus + next_clus
15:         clus_vals += $\mathbf{S}$[next_clus]
16:     update(sizes); min_size = min(sizes)
17:     $\mathbf{O}$[clus_id] = clus_vals
```

is greater than or equal to s_t, then we continue to merge the next cluster with the current cluster. Lines 5-11 show this loop of merging. In line 12, the values in the merged cluster are stored in the output set of k-anonymous clusters with its cluster ID. This method inserts similar values into one cluster, at the cost that the resulting larger clusters will generate more candidate record pairs.

(b) SNC-Size: Clusters are merged until the minimum size of the clusters becomes greater than or equal to k by iteratively identifying the smallest cluster. Algo. 2 provides an overview of this method. In lines 2-4, we find the cluster with the smallest number of elements, and if this number is less than k (line 5) we merge it with the smaller of its two neighboring clusters. We repeat this merging (lines 5-17) until the minimum cluster size is at least k. The values in the merged cluster are stored in the output set of k-anonymous clusters in line 17 along with its cluster ID. Compared to SNC-Sim, this method results in a smaller number of records in most clusters. However, true matches might be missed depending on the value for k, because the similarity between values is not considered.

5. Once the k-anonymous clusters are created, they need to be sent to a third party, Carol, to generate candidate record pairs. The values in the clusters are replaced by their (encrypted) record IDs.

Carol receives the k-anonymous clusters from Alice and Bob and performs the following steps:

6. Find corresponding clusters from Alice and Bob based on the reference position values in the cluster IDs to generate candidate record pairs as is illustrated in Fig. 4.

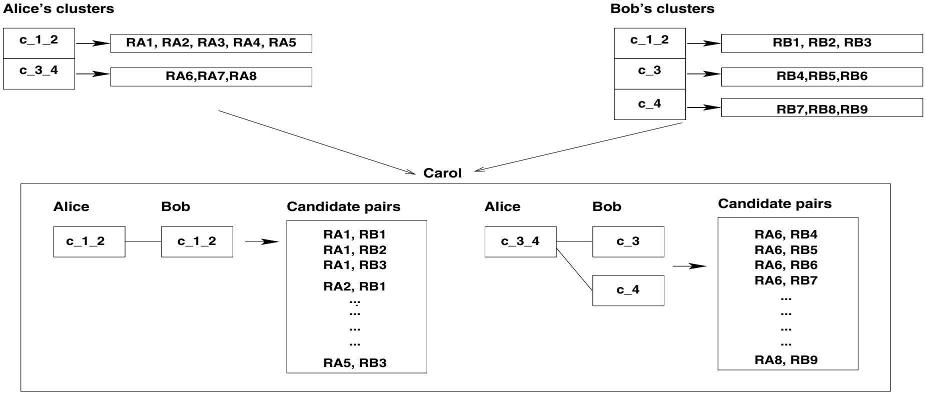


Fig. 4. The merging of corresponding clusters from Alice and Bob to generate candidate record pairs (made of record IDs) as conducted by Carol in Step 6 of the protocol

Algorithm 3. Generating Candidate Record Pairs

Input:
- $\mathbf{O^A}$: Alice's set of k-anonymous clusters
- $\mathbf{O^B}$: Bob's set of k-anonymous clusters

Output:
- **C**: Set of candidate record pairs

```
1:  for $(i^A, b^A) \in \mathbf{O^A}$ do
2:      ref_pos_vals_alice = get_ref_pos_vals($i^A$)
3:      bob_clusters = []; $b^B$ = []
4:      for ref_pos $\in$ ref_pos_vals_alice do
5:          bob_clusters += $\mathbf{O^B}$.getIDs(ref_pos $\in$ ID)
6:          $b^B$ += $\mathbf{O^B}$[bob_clusters]
7:      for alice_rec_ID $\in b^A$ do
8:          for bob_rec_ID $\in b^B$ do
9:              $\mathbf{C}$ += [alice_rec_ID,bob_rec_ID]
```

This process is explained in Algo. 3. From the cluster IDs, Carol extracts the position values of the reference values that reside in the corresponding clusters (line 2). In lines 3-6, Carol finds for each of Alice's clusters all of Bob's clusters that need to be merged by extracting the position values from Alice's cluster IDs. Carol then performs a nested loop (lines 7-9) over Alice's clusters and Bob's corresponding clusters, and stores the record pairs from Alice's and Bob's records in the output set of candidate record pairs.

7. Carol sends the record IDs of the candidate pairs **C** back to Alice and Bob which then employ a PPRL protocol on each block individually [14, 21, 24].

4 Analysis of the Protocol

In this section we analyse our SNC private blocking approach in terms of complexity, privacy, and quality.

1. **Complexity:** Assuming both databases contain n records ($n = n_A = n_B$) and n_R reference values are selected from the reference dataset, sorting

these n_R reference values is of $O(n_R \ log \ n_R)$ complexity, and inserting the n database records into the sorted list of reference values is of $O(n)$ complexity. At the end of the insertion of records into the sorted list, there will be n_R clusters each represented by one reference value. Merging clusters to create k-anonymous clusters requires a loop over n_R clusters, which is of $O(n_R)$ complexity, and results in less than or equal to n_R merged clusters.

Sending these clusters to Carol is of $O(n)$ communication complexity. Carol performs a loop over the clusters (a maximum of n_R clusters) to merge and generate candidate record pairs from Alice's and Bob's records. The computation complexity of this step is $O(n_R^2)$.

The overall complexity of our approach is linear in the size of the databases n and quadratic in the number of reference values n_R. The number of resulting clusters will be on average n/k. Assuming each cluster contains k records, the number of candidate record pairs generated by our approach is $\frac{n}{k} \times k^2 = n \ k$.

2. **Privacy:** We assume that all parties that participate in the protocol follow the 'honest but curious' (HBC) adversary model [4, 5], in that they try to find out as much as possible about the data from other parties while following the protocol. Since each cluster consists of at least k elements, it is difficult for Carol to perform an attack to infer individual records. The value for k has to be chosen carefully. A higher value for k provides stronger privacy guarantees but more candidate record pairs will be generated. The variance between the cluster sizes generated with our approach is very low compared to other private blocking techniques. Therefore, Carol cannot learn the frequency distribution of the clusters generated to conduct a frequency attack. We present the cluster sizes generated on some real databases in Sect. 5.

   Merging the blocks to create k-anonymous clusters makes the protocol more secure and harder for a frequency attack. Further, Carol does not know how the reference values $\mathbf{R}'$ were selected from the reference dataset $\mathbf{R}$ ($\mathbf{R}' \in \mathbf{R}$), and therefore she cannot learn the clustering details. However, as with other three-party protocols, collusion between the third party and one of the database owners with the aim to identify the other database owner's data, is a privacy risk in this approach as well [4, 5].
3. **Quality:** A good blocking technique should have two properties [3]: (1) effectiveness - all similar records should be grouped into the same cluster, and (2) efficiency - the number candidate record pairs should be as small as possible while including all true matching record pairs. SNC-Sim retrieves more similar records compared to SNC-Size as similarity is used in SNC-Sim to determine the maximum size of a cluster. However, SNC-Sim is more likely to group more records into one cluster. This results in higher effectiveness and lower efficiency for the SNC-Sim method comparatively.

   The value of k also determines the effectiveness and efficiency of blocking. A higher value for k results in a more effective but less efficient blocking. An optimal value for k needs to be set such that high values for both effectiveness and efficiency are achieved while k guarantees sufficient privacy as well.

Table 1. The number of records in the datasets used for experiments, and the number of records that occur in both datasets of a pair (i.e. the number of true matches)

Dataset sizes	25% overlap	50% overlap	75% overlap
1730 / 1730	446	897	1310
17,294 / 17,294	4365	8611	12,973
172,938 / 172,938	42,980	86,363	129,542
1,729,379 / 1,729,379	432,538	864,487	1,297,029

5 Experimental Evaluation

We conducted experiments using a real Australian telephone database containing 6,917,514 records. We extracted four attributes commonly used for record linkage: Given name (with 78,336 unique values), Surname (with 404,651 unique values), Suburb (town) name (13,109 unique values), and Postcode (2,632 unique values). To generate datasets of different sizes, we sampled 0.1%, 1%, 10% and 100% of records in the full database twice each, and stored them into pairs of files such that 25%, 50% or 75% of records appeared in both files of a pair. Table 1 provides an overview of the twelve pairs of datasets we generated.

The record pairs that occur in both datasets are exact matches. To investigate the performance of our protocol in the context of 'dirty data' (where attribute values contain errors and variations), we generated another series of datasets where we modified each attribute value by applying one randomly selected character edit operation (insert, delete, substitute, or transposition) [25]. As it turns out, there is no difference in the performance of our protocol with modified and not modified datasets, and we therefore only report averaged results.

We prototyped the protocol using the Python programming language (version 2.7.3). We also implemented prototypes of Karakasidis et al. [15]'s k-anonymous nearest neighbor clustering and Durham [16]'s Hamming based locality sensitive hashing for comparative analysis and evaluation. We name these two techniques as k-NN and HLSH in the results, respectively. We used parameter settings for the k-NN and HLSH methods in a similar range as used by the authors of these two state-of-the-art private blocking methods. For k-NN, k is set to 3 and the similarity threshold is set as $s_t = 0.6$. In the HLSH method, the number of iterations is set to $\mu = 20$, the number of hash functions is 30, and the number of bits to be sampled from the Bloom filters at each iteration is $\phi = 24$. The parameters for the SNC approaches were set as $k = 100$ and $s_t = 0.9$. All tests were run on a compute server with 64 bit Intel Xeon (2.4 GHz) CPUs, 128 GBytes of main memory and running Ubuntu 11.04. The prototype and test datasets are available from the authors.

Fig. 5 shows the total time required for private blocking of the four approaches. As can be seen from the figure, the SNC approach (both variations of SNC-Sim and SNC-Size) requires nearly two magnitudes less time by the database owners than the other two approaches (i.e. around 100 times faster) and is also linear in the size of the databases. We were unable to conduct experiments for the HLSH and k-NN approaches on the 1,729,379 datasets due to their memory requirements. The k-NN approach requires less time for the third party than

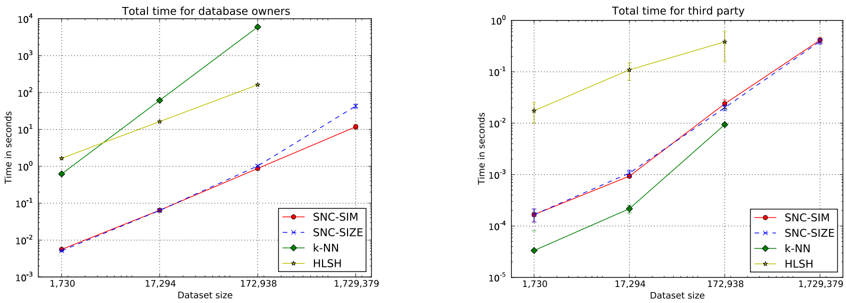


Fig. 5. Total time of the four blocking approaches required by database owners (left) and third party (right) averaged over the results of all variations of each dataset

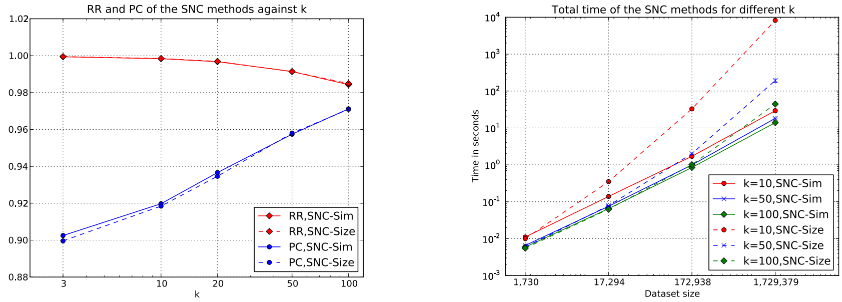


Fig. 6. RR and PC values for the k-anonymous SNC blocking approaches with the 17,294 datasets (left plot) and total time of the SNC approaches for different k against the dataset size (right plot) averaged over the results of all variations of each dataset

the SNC approaches, because a smaller number of candidate record pairs are generated with the k-NN approach at the cost of much reduced number of true matched record pairs (as is illustrated in the right plot in Fig. 7).

Reduction Ratio (RR) and Pairs Completeness (PC) can be used to assess the efficiency and effectiveness of blocking, respectively [3]. RR is the fraction of record pairs that are removed by a blocking technique and PC is the fraction of true matching record pairs that are included in the candidate record pairs generated by a blocking technique. The SNC approaches achieve high values for both PC and RR even when $k = 100$, which gives a strong privacy guarantee (Fig. 6 left plot). As discussed in Sect. 4, the effectiveness of blocking increases with k while efficiency decreases. As expected, the SNC-Sim approach provides a relatively higher PC than the SNC-Size. The right plot in Fig. 6 shows the scalability of our approach with different values for k.

The size of the clusters generated by the four approaches and the resulting PC and RR values are presented in Fig. 7. The SNC approaches have lower variances between the cluster sizes which makes a frequency attack by Carol harder. As illustrated in the left plot, the SNC approaches generate nearly uniform distributions of clusters. The k-NN approach has a higher RR but lower PC, while

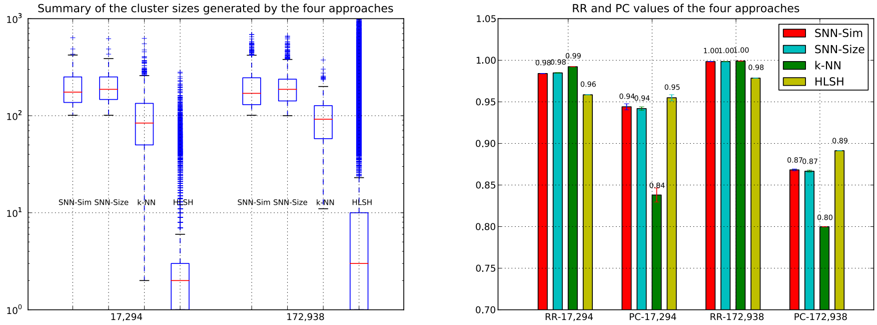


Fig. 7. Sizes of the clusters generated by the four approaches - with the bottom and top of the boxes representing the lower and upper quartiles, the band near the middle of the boxes the median, and the two ends of the whiskers the standard deviation above and below the mean of the distribution (left plot); and RR and PC values of the four approaches (right plot) using the 17,294 and 172,938 datasets

a lower RR and higher PC are achieved with the HLSH approach. The SNC approach performs superior compared to the other two approaches by achieving higher results for both RR and PC.

6 Conclusion

In this paper, we proposed an efficient private blocking technique that can be used to make privacy-preserving record linkage applications scalable to large databases. Our method is based on the sorted nearest neighborhood clustering approach, and uses a combination of the privacy techniques reference values and k-anonymous clustering. Experiments conducted on real-world large databases, each containing nearly 2 million records, validate that our approach is scalable and effective in generating candidate record pairs while preserving k-anonymity privacy characteristics. Our approach also outperforms two existing state-of-the-art private blocking techniques in terms of speed, efficiency, and effectiveness.

As discussed earlier, three-party solutions are often susceptible to collusion between parties. As future work, we aim to study how the sorted neighborhood clustering can be used for private blocking in a two-party context. Theoretically analyzing and modelling the privacy, complexity, and quality of our solution is another direction for future research.

References

1. Christen, P.: Data Matching. Data-Centric Systems and Appl. Springer (2012)
2. Batini, C., Scannapieca, M.: Data quality: Concepts, methodologies and techniques. In: Data-Centric Systems and Appl. Springer (2006)

3. Christen, P.: A survey of indexing techniques for scalable record linkage and deduplication. IEEE Transactions on Knowledge and Data Engineering 12(9) (2012)
4. Vatsalan, D., Christen, P., Verykios, V.: A taxonomy of privacy-preserving record linkage techniques. Information Systems (2013)
5. Hall, R., Fienberg, S.: Privacy-preserving record linkage. In: Domingo-Ferrer, J., Magkos, E. (eds.) PSD 2010. LNCS, vol. 6344, pp. 269–283. Springer, Heidelberg (2010)
6. Churches, T., Christen, P.: Blind data linkage using *n*-gram similarity comparisons. In: Dai, H., Srikant, R., Zhang, C. (eds.) PAKDD 2004. LNCS (LNAI), vol. 3056, pp. 121–126. Springer, Heidelberg (2004)
7. Fellegi, I.P., Sunter, A.B.: A theory for record linkage. Journal of the American Statistical Society 64(328), 1183–1210 (1969)
8. Jin, L., Li, C., Mehrotra, S.: Efficient record linkage in large data sets. In: DASFAA 2003, pp. 137–146 (2003)
9. Cohen, W.W., Richman, J.: Learning to match and cluster large high-dimensional data sets for data integration. In: ACM SIGKDD, pp. 475–480 (2002)
10. Kim, H., Lee, D.: Harra: fast iterative hashed record linkage for large-scale data collections. In: EDBT, Lausanne, Switzerland, pp. 525–536 (2010)
11. Hernandez, M.A., Stolfo, S.J.: Real-world data is dirty: Data cleansing and the merge/purge problem. Data Mining and Knowledge Discovery 2(1), 9–37 (1998)
12. Draisbach, U., Naumann, F., Szott, S., Wonneberg, O.: Adaptive windows for duplicate detection. In: ICDE, pp. 1073–1083 (2012)
13. Sweeney, L.: K-anonymity: A model for protecting privacy. International Journal of Uncertainty Fuzziness and Knowledge Based Systems 10(5), 557–570 (2002)
14. Pang, C., Gu, L., Hansen, D., Maeder, A.: Privacy-preserving fuzzy matching using a public reference table. In: McClean, S., Millard, P., El-Darzi, E., Nugent, C. (eds.) Intelligent Patient Management. Studies in Computational Intelligence, vol. 189, pp. 71–89. Springer, Heidelberg (2009)
15. Karakasidis, A., Verykios, V.: Reference table based k-anonymous private blocking. In: ACM Symposium on Applied Computing, Riva del Garda, Italy (2012)
16. Durham, E.: A framework for accurate, efficient private record linkage. PhD thesis, Vanderbilt University (2012)
17. Al-Lawati, A., Lee, D., McDaniel, P.: Blocking-aware private record linkage. In: IQIS, pp. 59–68 (2005)
18. Inan, A., Kantarcioglu, M., Bertino, E., Scannapieco, M.: A hybrid approach to private record linkage. In: IEEE ICDE, Cancun, Mexico, pp. 496–505 (2008)
19. Inan, A., Kantarcioglu, M., Ghinita, G., Bertino, E.: Private record matching using differential privacy. In: EDBT (2010)
20. Karakasidis, A., Verykios, V., Christen, P.: Fake injection strategies for private phonetic matching. In: DPM, Leuven, Belgium (2011)
21. Vatsalan, D., Christen, P., Verykios, V.: An efficient two-party protocol for approximate matching in private record linkage. In: AusDM, CRPIT 121 (2011)
22. Scannapieco, M., Figotin, I., Bertino, E., Elmagarmid, A.: Privacy preserving schema and data matching. In: ACM SIGMOD, pp. 653–664 (2007)
23. Yakout, M., Atallah, M., Elmagarmid, A.: Efficient private record linkage. In: IEEE ICDE, Shanghai, pp. 1283–1286 (2009)
24. Schnell, R., Bachteler, T., Reiher, J.: Privacy-preserving record linkage using Bloom filters. BMC Medical Informatics and Decision Making 9(1) (2009)
25. Christen, P., Pudjijono, A.: Accurate synthetic generation of realistic personal information. In: Theeramunkong, T., Kijsirikul, B., Cercone, N., Ho, T.-B. (eds.) PAKDD 2009. LNCS (LNAI), vol. 5476, pp. 507–514. Springer, Heidelberg (2009)

On Linear Refinement of Differential Privacy-Preserving Query Answering

Xiaowei Ying, Xintao Wu, and Yue Wang

University of North Carolina at Charlotte
{xying,xwu,ywang91}@uncc.edu

Abstract. Recent work showed the necessity of incorporating a user's background knowledge to improve the accuracy of estimates from noisy responses of histogram queries. Various types of constraints (e.g., linear constraints, ordering constraints, and range constraints) may hold on the true (non-randomized) answers of histogram queries. So the idea was to apply the constraints over the noisy responses and find a new set of answers (called refinements) that are closest to the noisy responses and also satisfy known constraints. As a result, the refinements expect to boost the accuracy of final histogram query results. However, there is one key question: is the ratio of the distributions of the results after refinements from any two neighbor databases still bounded? In this paper, we introduce a new definition, ρ-differential privacy on refinement, to quantify the change of distributions of refinements. We focus on one representative refinement, the linear refinement with linear constraints and study the relationship between the classic ϵ-differential privacy (on responses) and our ρ-differential privacy on refinement. We demonstrate the conditions when the ρ-differential privacy on refinement achieves the same ϵ-differential privacy. We argue privacy breaches could incur when the conditions do not meet.

Keywords: differential privacy, linear constraint, refinement, background knowledge.

1 Introduction

Research on differential privacy [1, 2] has shown that it is possible to carry out data analysis on sensitive data while ensuring strong privacy guarantees. Differential privacy is a paradigm of post-processing the output of queries. Differential privacy is defined as a property of a query answering mechanism, and a query answering mechanism satisfying differential privacy must meet the requirement that the distribution of its noisy query responses change very little with the addition or deletion of any record, so that the analyst can not infer the presence or absence of some record from the responses. Formally, differential privacy uses a user-specified privacy threshold ϵ to bound the ratio of the probabilities of the noisy responses from any two neighbor databases (differing one record).

Recent work [3–5] showed the necessity of incorporating a user's background knowledge to improve the accuracy of estimates from noisy responses of histogram queries. Various types of constraints (e.g., linear constraints, ordering

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 353–364, 2013.

constraints, and range constraints) may hold on the true (non-randomized) answers of histogram queries. So the idea was to apply the constraints over the noisy responses and find a new set of answers (called refinements) that are closest to the noisy responses and also satisfy known constraints. As a result, the refinements expect to boost the accuracy of final histogram query results.

However, there is one key question: is the ratio of the distributions of the results after refinements from any two neighbor databases still bounded? In this paper, we introduce a new definition, ρ-differential privacy on refinement, to quantify the change of distributions of refinements. We focus on one representative refinement, the linear refinement with linear constraints and study the relationship between the classic ϵ-differential privacy (on responses) and our ρ-differential privacy on refinement. We demonstrate the conditions when the ρ-differential privacy on refinement achieves the same ϵ-differential privacy. We argue privacy breaches could incur when the conditions do not meet.

2 Differential Privacy Revisited

We revisit the formal definition and the mechanism of differential privacy. We denote the original database as $\mathcal{D}$, and its neighboring database as $\mathcal{D}'$. We will concentrate on pairs of databases (D, D') differing only in one row, meaning one is a subset of the other and the larger database contains just one additional row.

Definition 1. *(ϵ-differential privacy) [1]. A mechanism $\mathcal{K}$ is ϵ-differentially private if for all databases D and D' differing on at most one element, and any subsets of outputs $S \subseteq Range(\mathcal{K})$,*

$$\Pr[\mathcal{K}(D) \in S] \le e^{\epsilon} \times \Pr[\mathcal{K}(D') \in S] \tag{1}$$

Theorem 1. *[1] For $f : D \to \mathbf{R}^d$, the mechanism $\mathcal{K}_f$ that adds independently generated noise with distribution $Lap(\Delta f/\epsilon)$ to each of the d output terms satisfies ϵ-differential privacy, where the sensitivity, Δf, is $\Delta f = max_{D,D'} \|f(D) - f(D')\|_1$ for all D, D' differing in at most one element.*

The mechanism for achieving differential privacy computes the sum of the true answer and random noise generated from a Laplace distribution. The magnitude of the noise distribution is determined by the sensitivity of the computation and the privacy parameter specified by the data owner. Differential privacy maintains composability, i.e., differential privacy guarantees can be provided even when multiple differentially-private releases are available to an adversary, and can extend to group privacy, i.e., changing a group of k records in the data set induces a change of at most a multiplicative $e^{k\epsilon}$ in the corresponding output distribution [6].

3 ρ-Differential Privacy on Refinement

In this section, we first describe the notations and then formally define refinement based on background knowledge. We present definitions of unbiased refinement and constrained refinement. We finally introduce our key concept, ρ-**differential privacy on refinement**, and use an illustrating example to show the difference between the proposed ρ-differential privacy on refinement and the classic ϵ-differential privacy.In a differentially private query answering mechanism, the analyst submits queries, the mechanism generates true values for the query, and perturbs them with calibrated noise to derive the responses, then returns the responses to the analyst. Usually, the analyst may possess some background knowledge about the database. With background knowledge, the analyst can refine the responses given by the mechanism, and may obtain more accurate values for his queries.

3.1 Definition

We denote the original database as $\mathcal{D}$, and its neighboring database as $\mathcal{D}'$ which differs from the original database by a single record. The vector-valued query is denoted as $\boldsymbol{Q}$, $\boldsymbol{Q} = (q_1, q_2, \cdots, q_n)^T$. We denote the true value from database D for the query as $\boldsymbol{\mu}$, $\boldsymbol{\mu} = (\mu_1, \mu_2, \cdots, \mu_n)^T$, and the response from database $\mathcal{D}$ for the query as X, $X = (X_1, X_2, \cdots, X_n)^T$. And we denote the true value from database $\mathcal{D}'$ for the query as $\boldsymbol{\mu}'$, $\boldsymbol{\mu}' = (\mu'_1, \mu'_2, \cdots, \mu'_n)^T$, and the response from database $\mathcal{D}'$ for the query as X', $X' = (X'_1, X'_2, \cdots, X'_n)^T$. The randomization mechanism satisfies ϵ-differential privacy, i.e., for an arbitrary set of integers $S = \{i, j, \ldots, k\} \subseteq \{1, \ldots, n\}$,

$$e^{-\epsilon} \le \frac{\Pr[X_S]}{\Pr[X'_S]} \le e^{\epsilon} \tag{2}$$

Assume that the user knows some background knowledge about $\mathcal{D}$ and $\mathcal{D}'$ denoted by $\mathcal{B}$ and $\mathcal{B}'$ respectively. For database $\mathcal{D}$, we denote the estimated value as $\widehat{X}$ derived by the analyst from the response using background knowledge, $\widehat{X} = (\widehat{X}_1, \widehat{X}_2, \cdots, \widehat{X}_n)^T$, for database $\mathcal{D}'$, we denote it as $\widehat{X}'$, $\widehat{X}' = (\widehat{X}'_1, \widehat{X}'_2, \cdots, \widehat{X}'_n)^T$.

Definition 2. ***(Refinement)*** *Given the background knowledge $\mathcal{B}$ on database $\mathcal{D}$, the refinement $\widehat{X} = (\widehat{X}_1, \ldots, \widehat{X}_n)^T$ is the user's estimation on the true value of query $\boldsymbol{Q}(\mathcal{D})$ based on the response X: $\widehat{X} = \mathrm{rf}(X|\mathcal{B}, \mathcal{D})$.*

Similarly, given response X' from $\mathcal{D}'$, the refinement to estimate $\boldsymbol{Q}(\mathcal{D}')$ is $\widehat{X}' = \mathrm{rf}(X'|\mathcal{B}', \mathcal{D}')$.

Definition 3. ***(UnbiasedRefinement)*** *TherefinementX̂ isunbiasedif$\mathbf{E}(\widehat{X}) = \boldsymbol{\mu}$ stands for any $\boldsymbol{\mu}$.*

Definition 4. ***(Constrained Refinement)*** *The refinement $\widehat{X}$ is a constrained refinement if $\widehat{X}$ always satisfies the background knowledge $\mathcal{B}$ for any response X.*

The two refinements, $\widehat{X} = (\widehat{X}_1, \ldots, \widehat{X}_n)$ from $\mathcal{D}$ and $\widehat{X}' = (\widehat{X}'_1, \ldots, \widehat{X}'_n)$ from $\mathcal{D}'$, may be mapped to two disjoint spaces by the refinement function rf(). In this case, either the numerator or the denominator of ratio $\frac{\Pr(\widehat{X}=\boldsymbol{x})}{\Pr(\widehat{X}'=\boldsymbol{x})}$ is 0. However, this difference is due to the refinement strategy and does not disclose any privacy information.

Definition 5. *(ρ-**differential privacy on refinement**) Given the refinements $\widehat{X}$ and $\widehat{X}'$ and an arbitrary set of integers $S = \{i, j, \ldots, k\} \subseteq \{1, \ldots, n\}$, define*

$$\widehat{X}_S = (\widehat{X}_i, \widehat{X}_j, \ldots, \widehat{X}_k) \text{ and } \widehat{X}'_S = (\widehat{X}'_i, \widehat{X}'_j, \ldots, \widehat{X}'_k).$$

Let $\mathcal{R}_S$ and $\mathcal{R}'_S$ be the sets of all possible values of $\widehat{X}_S$ and $\widehat{X}'_S$ respectively. The refinement satisfies differential privacy, if $\mathcal{R}_S \cap \mathcal{R}'_S \neq \varnothing$ and for any subset $\Omega \subseteq \mathcal{R} \cap \mathcal{R}'$ the following inequality stands

$$e^{-\rho} \leq \frac{\Pr[\widehat{X}_S \in \Omega]}{\Pr[\widehat{X}'_S \in \Omega]} \leq e^{\rho}. \tag{3}$$

3.2 An Illustrating Example

Example 1. The analyst submits a vector-valued query $\boldsymbol{Q}$, $\boldsymbol{Q} = (q_1, q_2)^T$, $\max |\boldsymbol{\mu} - \boldsymbol{\mu}'| = 1$, and $\sigma = \frac{1}{\epsilon}$. The analyst has the background knowledge that $\mu_1 + \mu_2 = c$ and $\mu'_1 + \mu'_2 = c'$. One method to refine the response is shown in (4):

$$\widehat{X}_1 = \frac{1}{2}(X_1 + c - X_2), \ \widehat{X}_2 = \frac{1}{2}(X_2 + c - X_1). \tag{4}$$

Equivalently expressed in matrix:

$$\begin{pmatrix} \widehat{X}_1 \\ \widehat{X}_2 \end{pmatrix} = \begin{pmatrix} \frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} + \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \end{pmatrix} c \tag{5}$$

The refinement in (5) belongs to constrained refinement. So we can calculate the ratio $\Pr(\widehat{X}_1 = x_1)/\Pr(\widehat{X}'_1 = x_1)$ to obtain the bound. First, from formula (5), we derive the probability density function of $\widehat{X}_1$ and $\widehat{X}'_1$, shown in (6) and (7).

$$f_{\widehat{X}_1}(x_1) = \frac{1}{2\sigma} \int_{\mathbb{R}} \exp\left\{-\frac{|2x_1 + x_2 - c - \mu_1| + |x_2 - \mu_2|}{\sigma}\right\} \mathrm{d}x_2 \tag{6}$$

$$f_{\widehat{X}'_1}(x_1) = \frac{1}{2\sigma} \int_{\mathbb{R}} \exp\left\{-\frac{|2x_1 + x_2 - c' - \mu'_1| + |x_2 - \mu'_2|}{\sigma}\right\} \mathrm{d}x_2 \tag{7}$$

Without loss of generality, we assume that $\mu_1 - \mu'_1 = 1$. When x_1 is sufficiently large, we can simplify formulas (6) and (7) to formulas (8) and (9) respectively.

$$f_{\widehat{X}_1}(x_1) = \frac{1}{2\sigma}(\sigma + 2x_1 - 2\mu_1) \exp\left\{-\frac{2(x_1 - \mu_1)}{\sigma}\right\} \tag{8}$$

$$f_{\widehat{X}'_1}(x_1) = \frac{1}{2\sigma}(\sigma + 2x_1 - 2\mu'_1) \exp\left\{-\frac{2(x_1 - \mu'_1)}{\sigma}\right\} \tag{9}$$

The ratio of the two PDFs can then be calculated as shown in (10), which tends to be $e^{2\epsilon}$ for large value of response X_1.

$$\begin{aligned}\frac{f_{\widehat{X}_1}(x_1)}{f_{\widehat{X}'_1}(x_1)} &= \frac{(\sigma + 2x_1 - 2\mu_1)}{(\sigma + 2x_1 - 2\mu'_1)} \exp\left\{\frac{2(\mu_1 - \mu'_1)}{\sigma}\right\} \\ &= \frac{(\sigma + 2x_1 - 2\mu_1)}{(\sigma + 2x_1 - 2\mu'_1)} e^{2\epsilon} \to e^{2\epsilon} \text{ (as } x_1 \to \infty) \end{aligned} \qquad (10)$$

So we can conclude that the ratio between the distributions of refinements for databases $\mathcal{D}$ and $\mathcal{D}'$ could be different from the ratio between the distributions of responses. In this example, the classic ϵ-differential privacy incurs 2ϵ-differential privacy on refinement. □

4 Background Knowledge and Refinement Analysis

In this section, we formally the linear constraint based background knowledge and conduct theoretical analysis on how refinement strategies affect differential privacy on refinement. We will use the following scenario as a running example throughout this section. Consider that a data publisher (such as a school) has collected grade information about a group of students and would like to allow the third party to query the data while preserving the privacy of the individuals involved. Assume the analyst submits a simple vector query: $\boldsymbol{Q}$ =(q_A, q_B, q_C, q_D, q_F, q_p, q_t). q_A, q_B, q_C, q_D, and q_F represent the numbers of students receiving grades A, B, C, D, and F respectively; q_p represents the number of passing students (grade D or higher) and q_t represents the query for the number of all the students.

$$\begin{cases} \mu_A + \mu_B + \mu_C + \mu_D - \mu_p = 0 \\ \mu_F + \mu_p - \mu_t = 0 \\ \mu_A + \mu_B = 80 \end{cases} \qquad (11)$$

The analyst may have the background knowledge in terms of the linear constraints shown in (11). The first two constraints are by the definition and independent of the underlying database whereas the third constraint holds specifically on the current database.

The analyst may have the background knowledge in terms of the ordering constraint, e.g., $\mu_A \leq \mu_p$. Ordering constraint can also be enforced when the user submits the vector query. For example, the analyst may submit a simple ascending ordering query that shows the number of students in each category. In other words, the analyst knows for sure that $\mu_1 \leq \mu_2 \leq ... \leq \mu_n$ although the responses may not hold the order constraints due to calibrated noises. Similarly the range constraint denotes the true answer of a particular query is within some finite range, e.g., $\mu_A \in [1, 10]$. Range constraints are often implicitly used in the post-process of noisy output. For example, users apply non-negative constraints when dealing with responses with negative values for attributes like age or salary. We will study these types of background knowledge in our future work.

4.1 Refinement with Linear Constraint

Assume that the user knows m linear combinations of the true answers:

$$b_{1i}\mu_1 + b_{2i}\mu_2 + \cdots + b_{ni}\mu_n = c_i, \quad i = 1, \ldots, m.$$

Equivalently, $\boldsymbol{b}_i^T \boldsymbol{\mu} = c_i$, where b_{ji} is the j-th entry of $\boldsymbol{b}_i$. Let $\boldsymbol{B} = [\boldsymbol{b}_1, \ldots, \boldsymbol{b}_m]$ and $\boldsymbol{c} = (c_1, \ldots, c_m)^T$.

Definition 6. ***(Linear Constraint)*** *The background knowledge with linear constraint can be expressed as*

$$\boldsymbol{B}^T \boldsymbol{\mu} = \boldsymbol{c},$$

where $\boldsymbol{B}$ *is an* $n \times m$ *matrix and* $\boldsymbol{c}$ *is an* m*-dimensional constant vectors.*

Under the linear constraint based background knowledge, a constrained refinement $\widehat{X}$ must satisfy $\boldsymbol{B}^T \widehat{X} = \boldsymbol{c}$ for any response X.

Definition 7. ***(Refinement with Linear Constraint)*** *The refinement* $\widehat{X}$ *is linear if it can be expressed as*

$$\widehat{X} = \boldsymbol{A}X + \boldsymbol{D}\boldsymbol{c} + \boldsymbol{h}, \tag{12}$$

where $\boldsymbol{A}$ *and* $\boldsymbol{D}$ *are* $n \times n$ *and* $n \times m$ *matrices respectively, and* $\boldsymbol{h}$ *is an* n*-dimensional constant vector.*

4.2 A General Result

Theorem 2. *Suppose that the user possesses the linear background knowledge* $\boldsymbol{B}^T \boldsymbol{\mu} = \boldsymbol{c}$ *and* $\boldsymbol{B}^T \boldsymbol{\mu}' = \boldsymbol{c}'$ *for database* $\mathcal{D}$ *and* $\mathcal{D}'$ *respectively, and he implements some constrained linear refinement as shown in* (12) *to estimate* $\boldsymbol{\mu}$*. Assume* $\mathrm{rank}(\boldsymbol{B}) = m$ *and* $\mathrm{rank}(\boldsymbol{A}) = r = n - m$*. Let* $\boldsymbol{U} = (\boldsymbol{U}_1, \boldsymbol{U}_2), \boldsymbol{V} = (\boldsymbol{V}_1, \boldsymbol{V}_2), \Sigma = \left(\begin{smallmatrix} \boldsymbol{\Sigma}_1 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{smallmatrix}\right)$*,be the SVD of* $\boldsymbol{A}$*:* $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$*, and* $\boldsymbol{A}^* = \boldsymbol{V}\begin{pmatrix} \boldsymbol{\Sigma}_1^{-1} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I}_m \end{pmatrix}\boldsymbol{U}^T$*. Adding noise from distribution Lap*(σ)*,* $\sigma = \Delta \boldsymbol{Q}/\epsilon$*, would result in*

$$\rho = \frac{\epsilon \|\boldsymbol{A}^* \boldsymbol{D}(\boldsymbol{c} - \boldsymbol{c}') - (\boldsymbol{\mu} - \boldsymbol{\mu}')\|_1}{\|\boldsymbol{\mu} - \boldsymbol{\mu}'\|_1},$$

where $\boldsymbol{\mu}$ *and* $\boldsymbol{\mu}'$ *are from the two databases that achieves* $\|\boldsymbol{\mu} - \boldsymbol{\mu}'\|_1 = \Delta \boldsymbol{Q}$*.*

Proof. Let $\Omega = \{\omega_1, \ldots, \omega_k\} \subseteq \{1, 2, \ldots, n\}$, and P_Ω be the $n \times k$ matrix with $P(\omega_i, i) = 1$ and 0 elsewhere. Similarly, $\bar{\Omega} = \{1, \ldots, n\} - \Omega$. with $n \times (n - k)$ matrix $P_{\bar{\Omega}}$ defined likewise. We can rewrite the refinement function to

$$\widehat{X} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T X + \boldsymbol{D}\boldsymbol{c} + \boldsymbol{h}.$$

Let $Z = \left(\begin{smallmatrix} Z_1 \\ Z_2 \end{smallmatrix}\right) = \left(\begin{smallmatrix} \boldsymbol{\Sigma}_1 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I} \end{smallmatrix}\right) \boldsymbol{V}^T X$. With $\boldsymbol{V}^T = \boldsymbol{V}^{-1}$ and $|\boldsymbol{V}| = 1$, we can have that the PDF of Z is

$$f_Z(\boldsymbol{z}) = \frac{1}{|\boldsymbol{\Sigma}_1|} f_X(\boldsymbol{V}\boldsymbol{\Sigma}^* \boldsymbol{z}), \text{ where } \boldsymbol{\Sigma}^* = \begin{pmatrix} \boldsymbol{\Sigma}_1^{-1} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I} \end{pmatrix}.$$

Let $W = UZ$, $S = \widehat{X} - \boldsymbol{D}\boldsymbol{c} - \boldsymbol{h} = \boldsymbol{U}_1 Z_1$, and $T = \boldsymbol{U}_2 Z_2$. Then, $W = S + T$, $S \in \mathcal{U}_\infty$, $S \in \mathcal{U}_\in$, and the PDF of W is given by

$$f_W(\boldsymbol{w}) = \frac{1}{|\boldsymbol{\Sigma}_1|} f_X(\boldsymbol{V}\boldsymbol{\Sigma}^* \boldsymbol{U}^T \boldsymbol{w}) = \frac{1}{|\boldsymbol{\Sigma}_1|} f_X(\boldsymbol{A}^* \boldsymbol{w}).$$

Notice that S and T are actually the projection of W onto the space spanned by $\boldsymbol{U}_1$ and $\boldsymbol{U}_2$ respectively, and the two spaces are orthogonal. For any $\boldsymbol{s} \in \mathcal{U}_\infty$ and $\boldsymbol{t} \in \mathcal{U}_\in$, $W = \boldsymbol{s} + \boldsymbol{t}$ would always give $S = \boldsymbol{s}$. Therefore, if $\boldsymbol{s} \in \mathcal{U}_\infty$, the PDF of S is given by

$$\begin{aligned} f_S(\boldsymbol{s}) &= \frac{1}{|\boldsymbol{\Sigma}_1|} \int\limits_{\mathcal{U}_\in} f_X[\boldsymbol{A}^*(\boldsymbol{s} + \boldsymbol{t})] \mathrm{d}\boldsymbol{t} = \frac{1}{|\boldsymbol{\Sigma}_1|} \int f_X \left[\boldsymbol{A}^* \left(\boldsymbol{s} + U \left(\begin{smallmatrix} \boldsymbol{0} \\ \boldsymbol{z}_2 \end{smallmatrix}\right)\right)\right] \mathrm{d}U \left(\begin{smallmatrix} \boldsymbol{0} \\ \boldsymbol{z}_2 \end{smallmatrix}\right) \\ &= \frac{1}{|\boldsymbol{\Sigma}_1|} \int f_X \left(\boldsymbol{A}^* \boldsymbol{s} + \boldsymbol{V}_2 \boldsymbol{z}_2\right) \mathrm{d}\boldsymbol{z}_2. \end{aligned}$$

Hence, the PDF of $\widehat{X}$ can be given by

$$f_{\widehat{X}}(\boldsymbol{x}) = \frac{1}{|\boldsymbol{\Sigma}_1|} \int f_X \left[\boldsymbol{A}^*(\boldsymbol{x} - \boldsymbol{D}\boldsymbol{c} - \boldsymbol{h}) + \boldsymbol{V}_2 \boldsymbol{z}_2\right] \mathrm{d}\boldsymbol{z}_2,$$

if $\boldsymbol{U}_2^T(\boldsymbol{x} - \boldsymbol{D}\boldsymbol{c} - \boldsymbol{h}) = 0$, and $f_{\widehat{X}}(\boldsymbol{x}) = 0$ otherwise.

Similarly, the PDF of $\widehat{X}'$ is given by

$$f_{\widehat{X}'}(\boldsymbol{x}) = \frac{1}{|\boldsymbol{\Sigma}_1|} \int f_{X'} \left[\boldsymbol{A}^*(\boldsymbol{x} - \boldsymbol{D}\boldsymbol{c}' - \boldsymbol{h}) + \boldsymbol{V}_2 \boldsymbol{z}_2\right] \mathrm{d}\boldsymbol{z}_2,$$

if $\boldsymbol{U}_2^T(\boldsymbol{x} - \boldsymbol{D}\boldsymbol{c}' - \boldsymbol{h}) = 0$, and $f_{\widehat{X}}(\boldsymbol{x}) = 0$ otherwise.

Notice that $\widehat{X}_\Omega = P_\Omega^T \widehat{X}$, $\widehat{X}_{\bar{\Omega}} = P_{\bar{\Omega}}^T \widehat{X}$ and $\widehat{X} = P_\Omega \widehat{X}_\Omega + P_{\bar{\Omega}} \widehat{X}_{\bar{\Omega}}$. The PDF of $\widehat{X}_\Omega$ can be expressed as

$$f_{\widehat{X}_\Omega}(\boldsymbol{x}_\Omega) = \frac{1}{|\boldsymbol{\Sigma}_1|} \iint\limits_{\mathcal{D}(\boldsymbol{x}_\Omega)} \mathrm{d}\boldsymbol{z}_2 \mathrm{d}\boldsymbol{x}_{\bar{\Omega}} f_X \left[\boldsymbol{A}^* \left(P_\Omega \boldsymbol{x}_\Omega + P_{\bar{\Omega}} \boldsymbol{x}_{\bar{\Omega}} - \boldsymbol{D}\boldsymbol{c} - \boldsymbol{h}\right) + \boldsymbol{V}_2 \boldsymbol{z}_2\right], \tag{13}$$

where $\mathcal{D}(\boldsymbol{x}_\Omega) = \{\boldsymbol{x}_{\bar{\Omega}} : \boldsymbol{U}_2^T(P_\Omega \boldsymbol{x}_\Omega + P_{\bar{\Omega}} \boldsymbol{x}_{\bar{\Omega}} - \boldsymbol{D}\boldsymbol{c} - \boldsymbol{h}) = 0\}$.

The PDF of $\widehat{X}'_\Omega$ can be derived in a similar manner. When the ratio of the integral kernels in (13) is bounded, i.e.,

$$e^{-\epsilon} \leq \frac{f_X \left[\boldsymbol{A}^*(\boldsymbol{x} - \boldsymbol{D}\boldsymbol{c} - \boldsymbol{h}) + \boldsymbol{V}_2 \boldsymbol{z}_2\right]}{f_{X'} \left[\boldsymbol{A}^*(\boldsymbol{x} - \boldsymbol{D}\boldsymbol{c}' - \boldsymbol{h}) + \boldsymbol{V}_2 \boldsymbol{z}_2\right]} \leq e^{\epsilon}, \tag{14}$$

the ratio of the integrals, $f_{\widehat{X}_\Omega}(\boldsymbol{x}_\Omega)/f_{\widehat{X}'_\Omega}(\boldsymbol{x}_\Omega)$, is also bounded by $[e^{-\epsilon}, e^{\epsilon}]$. Note that f_X and f_X are the Laplace distribution p.d.f., and hence

$$\begin{aligned}\frac{f_{\widehat{X}_\Omega}(\boldsymbol{x}_\Omega)}{f_{\widehat{X}'_\Omega}(\boldsymbol{x}_\Omega)} &\leq \frac{f_X\left[\boldsymbol{A}^*(\boldsymbol{x}-\boldsymbol{D}\boldsymbol{c}-\boldsymbol{h})+\boldsymbol{V}_2\boldsymbol{z}_2\right]}{f_{X'}\left[\boldsymbol{A}^*(\boldsymbol{x}-\boldsymbol{D}\boldsymbol{c}'-\boldsymbol{h})+\boldsymbol{V}_2\boldsymbol{z}_2\right]}\\ &\leq \frac{\exp\{\frac{1}{\sigma}\|\boldsymbol{A}^*(\boldsymbol{x}-\boldsymbol{D}\boldsymbol{c}-\boldsymbol{h})+\boldsymbol{V}_2\boldsymbol{z}_2-\boldsymbol{\mu}\|_1\}}{\exp\{\frac{1}{\sigma}\|\boldsymbol{A}^*(\boldsymbol{x}-\boldsymbol{D}\boldsymbol{c}'-\boldsymbol{h})+\boldsymbol{V}_2\boldsymbol{z}_2-\boldsymbol{\mu}'\|_1\}}\\ &\leq \exp\left\{\pm\frac{1}{\sigma}\|\boldsymbol{A}^*\boldsymbol{D}(\boldsymbol{c}-\boldsymbol{c}')-(\boldsymbol{\mu}-\boldsymbol{\mu}')\|_1\right\}.\end{aligned}$$

Therefore, when the noise is added according to the classical schema , i.e., take $\sigma = \frac{\Delta \boldsymbol{Q}}{\epsilon} = \frac{1}{\epsilon}\|\boldsymbol{\mu}-\boldsymbol{\mu}'\|_1$, we can have $f_{\widehat{X}_\Omega}(\boldsymbol{x}_\Omega)/f_{\widehat{X}'_\Omega}(\boldsymbol{x}_\Omega)$ is bounded by $e^{\pm\rho}$ where

$$\rho = \frac{\epsilon\|\boldsymbol{A}^*\boldsymbol{D}(\boldsymbol{c}-\boldsymbol{c}')-(\boldsymbol{\mu}-\boldsymbol{\mu}')\|_1}{\|\boldsymbol{\mu}-\boldsymbol{\mu}'\|_1}. \qquad \square$$

A special case of the above result is that $\rho = \epsilon$ when $\boldsymbol{c} = \boldsymbol{c}'$ (no difference on constants of linear background constraints over two neighbor databases). However, in practice, $\boldsymbol{c}$ could be different from $\boldsymbol{c}'$ (refer to the example shown in Appendix), the ρ-differential privacy on refinement is generally different from the ϵ-differential privacy. A direct result from the above theorem is that, in order to guarantee $e^{-\epsilon} \leq f_{\widehat{X}_\Omega}(\boldsymbol{x}_\Omega)/f_{\widehat{X}'_\Omega}(\boldsymbol{x}_\Omega) \leq e^{\epsilon}$, we can choose $\sigma = \frac{1}{\epsilon}\max_{\mathcal{D},\mathcal{D}'}$ $\|\boldsymbol{A}^*\boldsymbol{D}(\boldsymbol{c}-\boldsymbol{c}')-(\boldsymbol{\mu}-\boldsymbol{\mu}')\|_1$.

4.3 The Best Linear Refinement

Consider the following least square refinement based on the linear background knowledge:

$$\min\|\widehat{X}-X\|_2 \quad \text{s.t. } \boldsymbol{B}^T\widehat{X} = \boldsymbol{c}. \tag{15}$$

Theorem 3. *The least square refinement from the optimization problem in* (15) *is given by*

$$\widehat{X} = \left[\boldsymbol{I}-\boldsymbol{B}(\boldsymbol{B}^T\boldsymbol{B})^{-1}\boldsymbol{B}^T\right]X + \boldsymbol{B}(\boldsymbol{B}^T\boldsymbol{B})^{-1}\boldsymbol{c}. \tag{16}$$

The refinement shown in (16) *is a constrained unbiased refinement. It has the minimum variance of $\widehat{X}_i$, $i = 1,\ldots,n$, among all linear unbiased refinements.*

Proof. The Lagrange function of (16) is

$$\mathcal{L} = (\widehat{X}-X)^T(\widehat{X}-X) - 2\boldsymbol{\Lambda}(\boldsymbol{B}^T\widehat{X}-\boldsymbol{c}).$$

Taking $\frac{\partial\mathcal{L}}{\partial\widehat{X}} = \boldsymbol{0}$, we can have $\widehat{X} = X + \boldsymbol{B}^T\boldsymbol{\Lambda}$, and hence

$$\begin{aligned}\boldsymbol{B}^T\widehat{X} &= \boldsymbol{B}^T(X+\boldsymbol{B}\boldsymbol{\Lambda}) = \boldsymbol{c}\\ \boldsymbol{\Lambda} &= (\boldsymbol{B}^T\boldsymbol{B})^{-1}(\boldsymbol{c}-\boldsymbol{B}^TX)\\ \widehat{X} &= X+\boldsymbol{B}[(\boldsymbol{B}^T\boldsymbol{B})^{-1}(\boldsymbol{c}-\boldsymbol{B}^TX)]\end{aligned}$$

Equivalently, $\widehat{X}$ can be expressed as follows:

$$\widehat{X} = [\boldsymbol{I} - \boldsymbol{B}(\boldsymbol{B}^T\boldsymbol{B})^{-1}\boldsymbol{B}^T]X + \boldsymbol{B}(\boldsymbol{B}^T\boldsymbol{B})^{-1}\boldsymbol{c}.$$

Next, we show that $\widehat{X}$ is a unbiased constrained refinement:

$$\begin{aligned}\boldsymbol{B}^T\widehat{X} &= \boldsymbol{B}^T X - \boldsymbol{B}^T\boldsymbol{B}(\boldsymbol{B}^T\boldsymbol{B})^{-1}\boldsymbol{B}^T X + \boldsymbol{B}^T\boldsymbol{B}(\boldsymbol{B}^T\boldsymbol{B})^{-1}\boldsymbol{c} = \boldsymbol{c}.\\ \mathbf{E}(\widehat{X}) &= [\boldsymbol{I} - \boldsymbol{B}(\boldsymbol{B}^T\boldsymbol{B})^{-1}\boldsymbol{B}^T]\,\mathbf{E}(X) + \boldsymbol{B}(\boldsymbol{B}^T\boldsymbol{B})^{-1}\boldsymbol{c}\\ &= \boldsymbol{\mu} - \boldsymbol{B}(\boldsymbol{B}^T\boldsymbol{B})^{-1}\boldsymbol{B}^T\boldsymbol{\mu} + \boldsymbol{B}(\boldsymbol{B}^T\boldsymbol{B})^{-1}\boldsymbol{c} = \boldsymbol{\mu}.\end{aligned}$$

Next, we prove the minimal variance property. We use $\boldsymbol{M}$ to denote the matrix $\boldsymbol{I} - \boldsymbol{B}(\boldsymbol{B}^T\boldsymbol{B})^{-1}\boldsymbol{B}^T$. Then we have $\boldsymbol{M} = \boldsymbol{M}^T$, $\boldsymbol{M}\boldsymbol{M}^T = \boldsymbol{M}$. We can further show that $(\boldsymbol{A} - \boldsymbol{I})\boldsymbol{M} = \mathbf{0}$. Notice that the following equalities stand for any $\boldsymbol{\mu}$,

$$\mathbf{E}(\widehat{X}) = \boldsymbol{A}\,\mathbf{E}(X) + \boldsymbol{D}\boldsymbol{c} + \boldsymbol{h} \Rightarrow \boldsymbol{\mu} = \boldsymbol{A}\boldsymbol{\mu} + \boldsymbol{D}\boldsymbol{B}^T\boldsymbol{\mu} + \boldsymbol{h}.$$

We can thus have $\boldsymbol{I} - \boldsymbol{A} = \boldsymbol{D}\boldsymbol{B}^T$ and $\boldsymbol{h} = \mathbf{0}$. Therefore,

$$(\boldsymbol{A} - \boldsymbol{I})\boldsymbol{M} = -\boldsymbol{D}\boldsymbol{B}^T[\boldsymbol{B}(\boldsymbol{B}^T\boldsymbol{B})^{-1}\boldsymbol{B}^T - \boldsymbol{I}] = \mathbf{0}.$$

Since $\text{Cov}(\widehat{X}) = \boldsymbol{A}\,\text{Cov}(X)\boldsymbol{A'} = 2\sigma^2\boldsymbol{A}\boldsymbol{A}^T$, $\mathbf{V}(\widehat{X}_i)/2\sigma^2$ is the i-th diagonal entry of matrix $\boldsymbol{A}\boldsymbol{A}^T$. With $\boldsymbol{M}\boldsymbol{M}^T = \boldsymbol{M}$ and $(\boldsymbol{A} - \boldsymbol{I})\boldsymbol{M} = \mathbf{0}$, we can have

$$\boldsymbol{A}\boldsymbol{A}^T = [(\boldsymbol{A} - \boldsymbol{M}) + \boldsymbol{M}][(\boldsymbol{A} - \boldsymbol{M})^T + \boldsymbol{M}^T] = (\boldsymbol{A} - \boldsymbol{M})(\boldsymbol{A} - \boldsymbol{M})^T + \boldsymbol{M}.$$

Since $(\boldsymbol{A}-\boldsymbol{M})(\boldsymbol{A}-\boldsymbol{M})^T$ is the semi-positive definite matrix, and the the diagonal entries are non-negative, and hence $(\boldsymbol{A}\boldsymbol{A}^T)_{ii} \geq \boldsymbol{M}_{ii}$ with $\boldsymbol{A} = \boldsymbol{M}$ minimizes $(\boldsymbol{A}\boldsymbol{A}^T)_{ii}$, $i = 1, \ldots, n$.

5 Conclusion and Further Discussion

In this paper we have introduced a new definition, ρ-differential privacy on refinement, to quantify the change of distributions of results after refinements. We focus on one representative refinement, the linear refinement with background knowledge as linear constraints and investigate the relationship between the classic ϵ-differential privacy (on responses) and our ρ-differential privacy on refinement.

Three techniques were proposed to use constraints to boost accuracy of answering range queries over histograms [3–5]. The refinement approach (also called constrained inference) [3] focused on using consistency constraints, which should hold over the noisy output, to improve accuracy for a variety of correlated histogram queries. The idea was to find a new set of answers $\bar{q}$ that is the closet set to the set of noisy answers $\tilde{q}$ and that also satisfies the consistency constraints. The proposed approach, *the minimum least squares solution*, was a special case of our linear refinement with linear constraints presented in this paper. Hay et al. in [3] also showed that the inferred $\bar{q}$ based on the minimum L_2 solution satisfies

ϵ-differential privacy. In our work, we introduced the general linear refinement and showed the conditions on when the refinement based on the general linear constraints achieves the same ϵ-differential privacy as defined over distributions of responses. The authors extended to refine degree distribution of networks under the context of publishing private network data [7]. Xiao et al. in [4] proposed an approach based on the Haar wavelet. In [5], the authors unified the two approaches [3, 4] in one general framework based on the matrix mechanism that can answer a workload of predicate counting queries.

One key question is whether background knowledge can be exploited by adversaries to breach privacy. It is well known that for the pre-processing based privacy preserving data mining models, several works [8, 9] showed the risks of privacy disclosure by incorporating a user's background knowledge in the reasoning process. In contrast, in the context of differential privacy, the authors in [1, 2] stated that differential privacy provides formal privacy guarantees that do not depend on an adversary's background knowledge (including access to other databases) or computational power. In [10], the authors gave an explicit formulation of *resistance to background knowledge.* The formulation follows the implicit statement: *Regardless of external knowledge, an adversary with access to the sanitized database draws the same conclusions whether or not my data is included in the original data.* They presented a mathematical formulation of background knowledge and belief. The belief is modeled by the posteriori distribution: given a response, the adversary draws his belief about the database using Baye's rule to obtain a posterior refinement. In [3–5], the authors also stated that the refinement has *no impact* on the differential privacy guarantee. This is because the analyst performs the refinement without access to the private data, using only the constraints and the perturbed responses. The perturbed responses are simply the output of a differentially private mechanism and post-processing of responses cannot diminish the rigorous privacy guarantee.

In [11], the authors examined the assumptions of differential privacy from the data generation perspective and proposed a participation-based guideline - *does deleting an individual's tuple erase all evidence of the individual's participation in the data-generation process?* - for determining the applicability of differential privacy. They showed that the privacy guarantee from differential privacy can degrade when applied to social networks or when deterministic statistics (of a contingency table) have been previously released. The deterministic statistics can be modeled as linear constraints with fixed $\boldsymbol{c}$ values. In this case, $\boldsymbol{c}$ could be different from $\boldsymbol{c}'$. Based on our Theorem 2, the ρ-differential privacy on refinement is different from the ϵ-differential privacy and we have to add larger noise to prevent privacy breaches. In practice, the adversary may possess any kind of background knowledge, which may even include the a-priori knowledge of the exact values of all other $n-1$ individuals. We refer readers to the example shown in Appendix where the adversary can exploit the background knowledge of the other $n-1$ individuals in the database to infer the value of a specific individual. We argue that the privacy breach is caused by the combination of the randomization mechanism and the background knowledge. In our future work, we would explore whether refinements with some particular background knowledge (e.g., ordering or range constraints) can incur privacy breaches, i.e., enabling the adversary to draw *significantly* different beliefs about the databases.

Acknowledgments. This work was supported in part by U.S. National Science Foundation IIS-0546027 and CCF-0915059.

References

1. Dwork, C., McSherry, F., Nissim, K., Smith, A.: Calibrating Noise to Sensitivity in Private Data Analysis. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 265–284. Springer, Heidelberg (2006)
2. Dwork, C.: A Firm Foundation for Private Data Analysis. Communications of the ACM (January 2011)
3. Hay, M., Rastogi, V., Miklau, G., Suciu, D.: Boosting the Accuracy of Differentially Private Histograms Through Consistency. Proceedings of the VLDB Endowment 3(1) (2010)
4. Xiao, X., Wang, G., Gehrke, J.: Differential Privacy via Wavelet Transforms. In: Proceedings of the 26th IEEE International Conference on Data Enginering, pp. 225–236. IEEE (2010)
5. Li, C., Hay, M., Rastogi, V., Miklau, G., McGregor, A.: Optimizing Linear Counting Queries Under Differential Privacy. In: Proceedings of the 29th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems of Data, pp. 123–134. ACM (2010)
6. Dwork, C., Lei, J.: Differential Privacy and Robust Statistics. In: Proceedings of the 41st Annual ACM Symposium on Theory of Computing, pp. 371–380. ACM (2009)
7. Hay, M., Li, C., Miklau, G., Jensen, D.: Accurate Estimation of the Degree Distribution of Private Networks. In: Proceedings of the 9th IEEE International Conference on Data Mining, pp. 169–178. IEEE (2009)
8. Martin, D., Kifer, D., Machanavajjhala, A., Gehrke, J., Halpern, J.: Worst-Case Background Knowledge for Privacy-Preserving Data Publishing. In: Proceedings of the 26th IEEE International Conference on Data Enginering. IEEE (2007)
9. Du, W., Teng, Z., Zhu, Z.: Privacy-MaxEnt: Integrating Background Knowledge in Privacy Quantification. In: Proceedings of the ACM SIGMOD International Conference on Management of Data, ACM (2008)
10. Ganta, S., Kasiviswanathan, S., Smith, A.: Composition Attacks and Auxiliary Information in Data Privacy. In: Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 265–273. ACM (2008)
11. Kifer, D., Machanavajjhala, A.: No Free Lunch in Data Privacy. In: Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 193–204. ACM (2011)

A Appendix

A.1 Example When $c \neq c'$

Database $\mathcal{D}$ with n records is obtained by adding one record to database $\mathcal{D}_0$. Every record in $\mathcal{D}$ belongs to one of two categories. The attacker knows that in $\mathcal{D}_0$, $k\mu_1 = \mu_2$, where μ_i denotes the count of category i in $\mathcal{D}_0$, $i = 1, 2$. The added record belongs to either of the two categories, denoted by $\mathcal{D}'$ and $\mathcal{D}''$ respectively. Let $\boldsymbol{\mu}' = \binom{\mu_1'}{\mu_2'}$ and $\boldsymbol{\mu}'' = \binom{\mu_1''}{\mu_2''}$ be the counts of $\mathcal{D}'$ and $\mathcal{D}''$ respectively. The background knowledge can be expressed as:

$$\begin{aligned}&\text{if } \mathcal{D}' \text{ is true: } k\mu_1' - \mu_2' = \boldsymbol{B}^T\boldsymbol{\mu}' = \boldsymbol{c}' = k,\\ &\text{if } \mathcal{D}'' \text{ is true: } k\mu_1'' - \mu_2'' = \boldsymbol{B}^T\boldsymbol{\mu}'' = \boldsymbol{c}'' = -1,\end{aligned}$$

where $\boldsymbol{B} = \binom{k}{-1}$.

Response $X = (X_1, X_2)$ is obtained by adding noise $Lap(\frac{2}{\epsilon})$. Next, we show that, for $\mathcal{D}'$ and $\mathcal{D}''$, the refinements $\widehat{X}'$ and $\widehat{X}''$ do not satisfy differential privacy. Consider the following refinement:

$$\text{For } \mathcal{D}' : \widehat{X}_1 = \frac{X_1 + X_2 + k}{k+1}, \text{ and } \widehat{X}_2 = X_2; \tag{17}$$

$$\text{For } \mathcal{D}'' : \widehat{X}_1 = \frac{X_1 + X_2 - 1}{k+1}, \text{ and } \widehat{X}_2 = X_2. \tag{18}$$

Comparing (17) and (18) with the general linear refinement formula in (12), we can have

$$\boldsymbol{A} = \begin{pmatrix} \frac{1}{k+1} & \frac{1}{k+1} \\ 0 & 1 \end{pmatrix}, \ \boldsymbol{D} = \begin{pmatrix} \frac{1}{k+1} \\ 0 \end{pmatrix}, \quad \text{and} \ \boldsymbol{h} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

When X_1 satisfies $x_1 \geq \frac{\mu_1' + \mu_2' + k}{k+1}$, we can have

$$\begin{aligned} f_{\widehat{X}_1'}(x_1) &= \int_{\mathbb{R}} f_{X_1}(z) f_{X_2}[(k+1)x_1 - k - z]\mathrm{d}z \\ &\propto \int_{\mathbb{R}} \exp\left\{-\frac{|z - \mu_1'| + |z - (k+1)x_1 + k + \mu_2'|}{\sigma}\right\}\mathrm{d}z \\ &= \exp\left\{\frac{n + k - (k+1)x_1}{\sigma}\right\}[(k+1)x_1 - k - n] \\ &\quad (\text{note } n = \mu_1' + \mu_2'). \end{aligned}$$

For $\mathcal{D}''$, we can similarly have that when $x_1 \geq \frac{\mu_1'' + \mu_2'' - 1}{k+1}$,

$$f_{\widehat{X}_1''}(x_1) \propto \exp\left\{\frac{n - 1 - (k+1)x_1}{\sigma}\right\}[(k+1)x_1 + 1 - n].$$

With $\sigma = \frac{2}{\epsilon}$ (satisfying ϵ-differential privacy), we can have:

$$\lim_{x_1 \to \infty} \frac{f_{\widehat{X}_1'}(x_1)}{f_{\widehat{X}_1''}(x_1)} = \exp\left[\frac{(k+1)\epsilon}{2}\right].$$

Therefore, the ratio $f_{\widehat{X}_1'}/f_{\widehat{X}_1''}$ reaches $e^{\frac{(k+1)\epsilon}{2}}$ for sufficiently large $X_1 + X_2$, which indicates the adversary can tell which database of $\mathcal{D}'$ and $\mathcal{D}''$ the response is from. In other words, the adversary can derive the value of the added record by refinement.

A Coupled Clustering Approach for Items Recommendation

Yonghong Yu[1], Can Wang[2], Yang Gao[1], Longbing Cao[2], and Xixi Chen[3]

[1] State Key Lab for Novel Software Technology, Nanjing University, P.R. China
[2] Advanced Analytics Institute, University of Technology Sydney, Australia
[3] Shandong Branch, Bank of Communications, P.R. China
{yuyh.nju,canwang613}@gmail.com, gaoy@nju.edu.cn, LongBing.Cao@uts.edu.au

Abstract. Recommender systems are very useful due to the huge volume of information available on the Web. It helps users alleviate the information overload problem by recommending users with the personalized information, products or services (called items). Collaborative filtering and content-based recommendation algorithms have been widely deployed in e-commerce web sites. However, they both suffer from the scalability problem. In addition, there are few suitable similarity measures for the content-based recommendation methods to compute the similarity between items. In this paper, we propose a hybrid recommendation algorithm by combing the content-based and collaborative filtering techniques as well as incorporating the coupled similarity. Our method firstly partitions items into several item groups by using a coupled version of k-modes clustering algorithm, where the similarity between items is measured by the *Coupled Object Similarity* considering coupling between items. The collaborative filtering technique is then used to produce the recommendations for active users. Experimental results show that our proposed hybrid recommendation algorithm effectively solves the scalability issue of recommender systems and provides a comparable recommendation quality when lacking most of the item features.

Keywords: Recommender systems, collaborative filtering, coupled object similarity, clustering algorithm.

1 Introduction

A recommender system is an important intelligent tool due to the huge volume of information on the Web. A recommender system overcomes the information overload problem by providing users with the personalized information, products or services (in this paper, we call them 'items'), which satisfy the user taste and preferences. In our daily life, typical applications of recommender systems including Amazon, Last.fm and MovieLens, recommend products, radios and movies, respectively. In addition, more and more e-commerce sites deploy recommender systems to attract public users, and therefore prompt their sale revenues.

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 365–376, 2013.

Collaborative Filtering (CF) [1] is one of the most widely used techniques for building recommender systems with a great success in e-commerce for its high recommendation quality. CF algorithms recommend items (e.g., products and movies) based on the opinions of other users that have the similar taste or preferences, rather than the content of items. However, CF algorithms suffer from the *scalability* problem [1].

Alternatively, content-based filtering (CBF) [2,3] make recommendations by analyzing the content of users or items. Balabanović et al. [2] and Melville et al. [3] have empirically shown that CBF techniques produce significant improvement against CF techniques in terms of the prediction quality. However, for CBF, it is hard to extract the reasonable features associated with items [1]. Moreover, CBF requires appropriate metrics to compute the similarity between items. But the existing metrics are not well-defined and not effective [4]. In addition, CBF also faces the serious scalability problem. The computational complexity exponentially rises when the number of users and items increase dramatically.

In order to overcome those challenges and solve the above issues, in this paper, we propose a hybrid recommendation algorithm by combining collaborative filtering with content-based filtering techniques. Our method firstly partitions items into several item groups by using a coupled version of the k-modes clustering algorithm, where the similarity between items is measured by the *Coupled Object Similarity (COS)* [4], which considers the coupling relation between items [5]. Then, CF is used to provide recommendations for active users. The key contributions are as follows: (1) we capture the correlation among items based on *COS*, which has been evaluated to outperform other similarity measures (e.g., *SMS* [6],*ADD* [7]) for categorical data. By this means, we overcome the similarity measure problem in content-based filtering; (2) we apply an effective clustering algorithm to group items, and compute the prediction within a small range of the item neighborhood. By applying the clustering algorithm, we solve the scalability problem in recommender systems; (3) we evaluate our proposed method on MovieLens data set in terms of the scalability and recommendation quality.

This paper is organized as follows. Section 2 briefly reviews the related work. The notations used in this paper are presented in Section 3. Section 4 describes the framework of our proposed hybrid recommendation algorithm combining the content-based with collaborative filtering techniques. The coupled similarity based k-modes clustering algorithm is proposed in Section 5. Experiments are evaluated in Section 6. Finally, we conclude this paper in Section 7.

2 Related Work

Generally, recommender systems can be classified into three classes [1]: collaborative filtering (CF) approaches, content-based filtering (CBF) approaches and hybrid approaches.

CF [8,9] approaches produce recommendations or predictions based on the assumption that similar users have similar tastes. The similarity between users is measured according to their history rating behaviors. CBF [2,3] approaches recommend items for users by analyzing the content of items and the profiles of users. Typical CBF recommender system such as InforFinder [10]. However, pure content-based recommendation algorithms suffer from feature extraction problem [1].

Both CF and CBF approaches have limitations, since they make recommendations only relying on user-item matrix or features of users and items, respectively. Hybrid approaches by combining CF and CBF techniques help avoid certain limitations of CF with CBF approaches. For example, Balabanović et al.[2] proposed the recommender system Fab, which maintains user profiles based on content analysis, and then use CF techniques to find similar users for collaborative recommendation. Melville et al. [3] presented a content-boosted CF algorithm, which uses a content-based predictor to enhance existing user rating data and then makes prediction using a weighted person correlation-based CF algorithm. They are different from our proposed method, in which the similarity between items is measured through *COS* when a recommender system analyzes the content of items.

In addition, several clustering algorithms have been applied in recommender systems. Rashid et al. proposed *CLUSTKNN* [11], which uses a variant of basic k-means algorithm to partition users into clusters, and then leverages a CF algorithm to produce recommendations. Xue et al. proposed *CBSMOOTH* [12], which uses the clusters as the computed groups and smoothes the unrated data for individual users. Unlike our focus here, these algorithms group users or items over a user-item matrix, while ours groups items over the set of items. Furthermore, in order to group items, our hybrid recommendation algorithm adopts a coupled version of k-modes clustering algorithm, which outperforms other variants of k-modes clustering algorithms for categorical data sets.

3 Preliminaries

In a typical scenario, a recommender system consists of a set of n users $U = \{u_1, u_2, ..., u_n\}$, and a set of m items $O = \{o_1, o_2, ..., o_m\}$. Each user $u_i \in U$ expresses his/her preferences by rating a subset of items on a scale from one to five. This set of items rated by the user u_i is denoted as $O_{u_i}(O_{u_i} \subseteq O)$. Each item $o_j \in O$ is represented as a feature vector $o_j = \{a_{j_1}, a_{j_2}, ..., a_{j_l}\}$, and those features extracted from all the items are categorical. For example, if the item set O represents a collection of movies, then the features, i.e., *director, actor, genre* etc., are extracted to express a movie item. In addition, those features have categorical values, such as "*Koster*","*Grant*" and "*Comedy*" etc. for the feature genre.

Generally, user preferences on items are usually converted into a user-item matrix R, with n rows and m columns. Each element r_{ij} of R represents the rating given by user u_i on item o_j. The integer value of ratings falls into [0,5], in which 0 indicates that the user has not rated the item. The higher rating corresponds to the better satisfactory.

In essence, the objective of recommender systems is to predict the rating on the specified item o_j for an active user u_a, by leveraging all the various data mining and machine learning techniques.

4 Integrating Content-Based and CF Recommendation Algorithms

Our proposed approach is a hybrid recommendation algorithm by combining the content-based and collaborative filtering techniques. The framework of our proposed recommendation algorithm is presented in Fig.1. Our proposed recommendation algorithm consists of four major components: (1) Data Extraction: extract user-item matrix R and the set of items O from data source; (2) Item Neighborhood Formation: partition the set of items O into several clusters by applying a coupled version of k-modes clustering algorithm; (3) Model Building: compute the similarity between each pair of items in the same cluster and store these similarities in a model, namely a *HashMap*; (4) Prediction Computation: based on the trained model, use a nearest neighbor algorithm to produce recommendations for active users. We describe these four components in detail below.

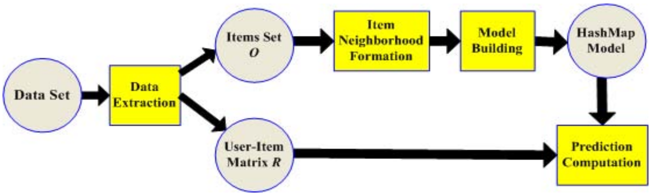


Fig. 1. The framework of our proposed hybrid recommendation algorithm

Data Extraction

In this component, our proposed hybrid recommendation algorithm extracts user-item matrix R and movie item set O from data source, which are used by movie item neighborhood formation and prediction computation component, respectively.

Item Neighborhood Formation and Model Building

1. Randomly select k distinct items from O.
2. Conduct the CK-modes algorithm on the set of items O, until the loss function converges or the number of iterations reaches the specified number of times. Once this CK-modes algorithm stops, the set of items O is divided into k disjoint clusters $\{O_1, O_2, ..., O_k\}(1 \leq i \leq k)$. Formally, $O = O_1 \cup O_2 \cup ... \cup O_k$, where $O_i \cap O_j = \emptyset(1 \leq i, j \leq k, i \neq j)$.

3. For each cluster, compute the similarity between each pair of items according to *COS*, and then store the pairs (e.g. $< itemid1, < itemid2, similarity >>$) in the *HashMap* model, where $itemid1$ is the key of the pair and $< itemid2, similarity >$ is the corresponding value. By using the map model, we quickly obtain the coupled similarity between $itemid1$ and $itemid2$.

Prediction Computation

Once the model with the coupled pairwise similarity has been built, the prediction method of the item-based collaborative filtering is adopted to generate the prediction on item o_i for an active user u. It takes the weighted sum of the ratings given by the active user u on the items similar to item o_i as the prediction. Weight measures the coupled similarity between the target item o_i and its similar item. Formally, the prediction P_{u,o_i} on item o_i for active user u is computed by the following formula.

$$P_{u,o_i} = \begin{cases} \frac{\sum_{\forall N_j \in N}(sim_{o_i,N_j} * R_{u,N_j})}{\sum_{\forall N_j \in N}(|sim_{o_i,N_j}|)} & \sum(|sim_{o_i,N_j}|) > 0 \\ \overline{r_u} & \sum(|sim_{o_i,N_j}|) = 0 \end{cases} \tag{1}$$

where N is the intersection of items rated by the active user u and items grouped by the CK-modes algorithm, R_{u,N_j} represents the rating on item N_j given by the user u. sim_{o_i,N_j} is the coupled similarity between item the o_i and the item N_j . $\overline{r_u}$ is the average of the active user's ratings.

5 Coupled Similarity Based K-Modes Algorithm

In this section, we present the coupled variant of k-modes clustering algorithm (CK-modes), which is used in our proposed recommendation algorithm to group items, by taking into account the coupling relationship among their features. In the recommender system, the features of items are categorical. For example, we use the categorical features (i.e. director, actor, genre and country) to represent a movie. K-modes [13], an extension of the basic k-means algorithm, is designed to deal with the categorical data sets. However, the similarity measure in k-modes is too rough to capture the closeness of two items. Hence, we decide to adapt the basic k-modes clustering algorithm to cluster items by incorporating the coupled similarity measure. The main difference between the basic k-modes algorithm and the adapted k-modes algorithm lies on the similarity measure and the method of updating modes. We further discuss these difference below.

5.1 Similarity Metric

For two items described by the categorical features, the k-modes clustering algorithm employs the Simple Matching Similarity [6] (*SMS*, which only uses 0 and 1 to distinguish similarities between distinct and identical categorical values) to compute the similarity feature values. However, *SMS* fails to capture the genuine relationship between categorical feature values. In contrast, we adopt the Coupled Object Similarity (COS)$\in$ [0,1] to measure the similarity between items, which is more accurate than *SMS*. *COS* [4] considers both the intra-coupled similarity within a feature and the inter-coupled similarity between features, which has been evaluated to outperform other similarities (e.g. *SMS* [6],*ADD* [7]) in term of clustering quality.

Formally, the Coupled Object Similarity (COS) between categorical items X and Y is defined as follows.

$$COS(X,Y) = \sum_{j=1}^{n} \delta_j^A(X_j, Y_j), \tag{2}$$

where X_j and Y_j are the values of feature j for X and Y, respectively; and δ_j^A is Coupled Attribute Value Similarity($CAVS$).

The $CAVS$ consists of the *Intra-coupled Attribute Value Similarity (IaAVS)* measure $\delta_j^{Ia}(X_j, Y_j)$ and the *Inter-coupled Attribute Value Similarity (IeAVS)* measure $\delta_j^{Ie}(X_j, Y_j)$ for feature j. The definition of $CAVS$ between attribute values X_j and Y_j of feature j is as follows.

$$\delta_j^A(X_j, Y_j) = \delta_j^{Ia}(X_j, Y_j) \cdot \delta_j^{Ie}(X_j, Y_j) \tag{3}$$

IaAVS measures the feature value similarity by considering the feature value occurrence frequencies within a feature, while *IeAVS* measures the feature value similarity by taking the feature dependency aggregation into account [4].

5.2 Updating Modes

Let S be a cluster generated by the previous partition of k-modes algorithm. There are mm items described by categorical features $\{a_{j_1}, a_{j_2}, ..., a_{j_l}\}$ belonging to the cluster S. A mode of the cluster S is a item vector $Q = [q_1, q_2, ..., q_l]$ to maximize the sum of the similarity between each element of S and Q. The classic k-modes clustering algorithm updates the mode Q of cluster S by reassigning each component of Q with the corresponding feature value that occurs the most among all those values of the items in S.

In our proposed adapted k-modes algorithm, we update the mode of each cluster according to the following definition.

Definition 1. *The mode of item set S with mm items is a vector $Q = [q_1, q_2, ..., q_l]$ that maximizes:*

$$Sim(Q,S) = \sum_{i=1}^{mm} COS(S_i, Q) \tag{4}$$

Since the latter method needs to compute the similarity between each pair of items, it causes a high computation cost. The former method is more efficient than the latter one. However, in order to group items for recommender system, we have to select the latter way. In recommender system, a movie item always has more than one genres. We extend the features of the movie item with t additional genre features, if the total number of genres of all the movies is t. In other words, a movie item is described by features $\{a_1, a_2, ..., a_l, g_1, g_2, ...g_t\}$. When a movie item has the genre $g_i(1 \leq i \leq t)$, we assign 1 to the corresponding value of the movie feature g_i, and otherwise 0. After the extension of features, the former mode updating method does not work well. The reason is that sparsity of genres: one movie has several genres, but the number of genres of all the movies is rather large, which leads to the phenomena that the occurrence frequency of 0 in feature g_i is much higher than that of 1. Hence, most of the corresponding mode values of $g_i(1 \leq i \leq l)$ are 0, resulting in the inaccurate description of the modes.

Therefore, we take advantage of the latter method formalized in Definition 1 in our proposed recommendation algorithm.

5.3 Coupled Similarity Based K-Modes

By using the coupled similarity measure and the method of modes updating described in the above sections 5.1 and 5.2, we design the Coupled similarity based k-modes clustering algorithm (CK-modes) described in Algorithm 1.

Algorithm 1. The Coupled Similarity Based K-modes Algorithm (CK-modes)

```
Input:
    $K$ : the number of clusters.
    $O$ : the set of $m$ items.
Output:
    a set of $K$ clusters are generated when the loss function value converges.
1: Randomly select K initial modes from $O$
2: Allocate each item to the nearest mode, and the similarity between each item and
   the mode is measured by COS. At the same time, accumulate the loss function
   value when each item is assigned to a cluster.
3: Once each item is allocated to the corresponding cluster, update the modes of
   clusters according to the principle described in Section 5.2.
4: Reallocate every item against the current modes, and accumulate a new loss function.
5: Repeat (2) and (3) until the loss function value converges.
```

6 Experiments and Evaluation

In this section, we conduct several experiments to show the accuracy and the scalability of our proposed hybrid recommendation algorithm.

6.1 Data Set and Evaluation Metric

We use MovieLens data set in our evaluation, which has been widely used in collaborative filtering research in the last decade. MovieLens data set contains

100,000 ratings from 943 users and 1,682 movies, and users with less than 20 ratings have been removed. MovieLens data set was converted into a user-item matrix R with 943 rows (users) and 1682 columns (movies).

Since our proposed approach is a hybrid of collaborative filtering and content based recommendation algorithm, we extract features of movie from MovieLens data set, and present each movie item as a feature vector $o = (mid, director, actor, country, genre)$. In addition, MovieLens data set lacks of director, actor and country etc. features, we only retain genre in feature vector o. Particularly, a movie item has several genres, such as movie titled 'Toy Story' is both an animation and a comedy. In order to make use of these genres to group movie, we extend the features of movie item with t additional genre features, if the total number of genres of all the movies in data set is t. In other words, a movie item is described by features $\{g_1, g_2, ...g_t\}$. When a movie item has genre $g_i(1 \leq i \leq t)$, we assign 1 to the corresponding feature value g_i, and otherwise 0.

We choose MAE to evaluate experimental results, as MAE metric is simple to calculate and intuitive to interpret. Formally,

$$MAE = \frac{\sum_{i=1}^{N} |p_i - q_i|}{N} \tag{5}$$

where p_i and q_i are the real rating and the corresponding prediction, respectively, and N denotes the total number of predictions generated for all active users. The lower the *MAE*, the better the recommendation algorithm generates the predictions for users.

6.2 Experimental Settings

Benchmark Recommendation Algorithms. We evaluate several widely discussed algorithms in the recommender system research, including user-based collaborative filtering algorithm [8], item-based collaborative filtering algorithm [9] and *CLUSTKNN* [11]. Each benchmark recommendation algorithm has been tuned to produce the best prediction quality according to the principles described in the corresponding literature.

We conduct a five-fold cross validation over the MovieLens data set by randomly extracting different training and test sets each time, which accounts for 80% and 20%, respectively. Finally, we use the average of MAE and the run time costs over the five folds to present the experimental results.

6.3 Experimental Results

In this section, we firstly determine the sensitivity of some parameters of our proposed hybrid recommendation algorithm and then compare with other benchmark recommendation algorithms. The parameters in our proposed algorithm include the number of clusters K and the number of neighbors that are selected to compute predictions for the target movie items.

Sensitivity of the Number of Clusters K. We perform a group of experiments to evaluate the prediction quality on the number of clusters K, ranging from 5 to 150 with a step of 10. Fig.2 reports the results. We observe that the number of clusters K does have an impact on the prediction quality. As K increases from 5 to 20, the prediction quality downgrades. After that, the MAE fluctuates; and then the curve tends to be flat. The best prediction quality is $MAE = 0.73$, when the number of clusters K equals to 5. Thus, we select $K = 5$ as the optimal choice in our following experiments.

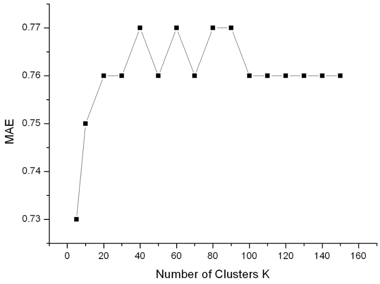


Fig. 2. Impact of K on our proposed recommendation algorithm

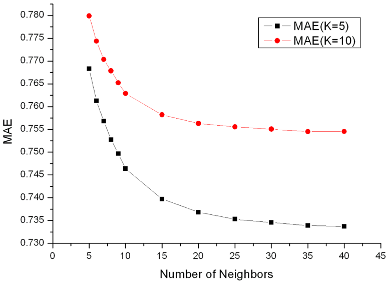


Fig. 3. Impact of the number of neighbors on our proposed recommendation algorithm

Sensitivity of the Neighborhood Size. We conduct another group of experiments to assess the prediction quality on the size of neighborhood used to produce the recommendation, ranging from 5 to 40 with a step of 5. Fig.3 describes that $MAEs$ decrease as the number of neighbors increases from 5 to 30, and then $MAEs$ remain stable. That is to say, our proposed method achieves a better prediction quality when more similar items are taken into account. Therefore, we select 30 as the optimum item neighborhood size.

From Fig.3, we observe that the more neighbors involved in making recommendation, the better the recommendation quality. There are two reasons: (1) the average size of neighborhoods decreases as the number of clusters K increases; (2) the intersection between the item neighborhood generated by the CK-modes algorithm and the set of items rated by the active user becomes smaller as the value of K increases. As a result, our proposed recommendation algorithm takes average of the active user's ratings as the prediction value when there are no neighbors for the target item.

Prediction Quality Comparisons. Once we determine the optimal parameters, we compare our proposed algorithm with those selected benchmark recommendation algorithms in terms of the prediction quality.

Table 1 presents the results of the best prediction quality by using different recommendation algorithms. It can be observed from Table 1 that the item-based algorithm outperforms other recommendation algorithms, and followed by *CLUSTKNN* recommendation algorithm. The prediction qualities of the user-based and our proposed hybrid recommendation algorithm are comparable to the item-based recommendation algorithms (i.e., MAE =0.73). However, The difference between our proposed algorithm and the item-based algorithm is small and is not statistically significant. In fact, our proposed hybrid recommendation algorithm depends on the features of items, and does not work not well when most of the important features are missing. Only the genres of movie has been extracted from the MovieLens data set, and other features, such as director, actor and country, are missing. Thus, the lacking of information limits of our proposed hybrid recommendation algorithm to some extent.

Table 1. Comparison of prediction quality of recommendation algorithms

Recommendation algorithm	**MAE**
User-based CF	0.730
Item-based CF	0.72
CLUSTKNN	0.725
Our proposed algorithm	0.730

Performance. Here, we focus on the overall performance of the system. We denote the throughput as the number of recommendations generated per second. Fig.4 shows the throughputs of both our proposed recommendation algorithm and other benchmark recommendation algorithms. Note that the user-based recommendation algorithm scans the whole user-item matrix R, its throughput do not change with the number of clusters. However, the throughput of the item-based recommendation algorithm varies with the number of neighbors selected to produce predictions. We plot the throughput of the item-based recommendation algorithm when the number of neighbors is 30, where it generates the best prediction quality.

As we can clearly see from Fig.4, clustering based recommendation algorithms (i.e., *CLUSTKNN* and our proposed hybrid recommendation algorithm) outperform both the user-bused and the item-based recommendation algorithms. The throughputs of both the *CLUSTKNN* and our proposed hybrid algorithm are substantially higher than other approaches at all values of the number of clusters. We can observe that for the number of clusters $K = 20$, our proposed recommendation algorithm produces a throughput rate of 12492 while the user-based and the item-base recommendation algorithms produce only 1333 and 2564, respectively. In addition, increasing the number of clusters corresponds to scanning the decreasing movie item neighborhood. Fig.4 also shows that the throughput of our proposed method increases rapidly as the number of clusters goes up. By contrast, the throughput of *CLUSTKNN* drops down as the number of clusters grows. The reason is that the *CLUSTKNN* takes centroids of clusters as the neighbors of the active user. Then, it takes more neighbors into account as the number of clusters increases.

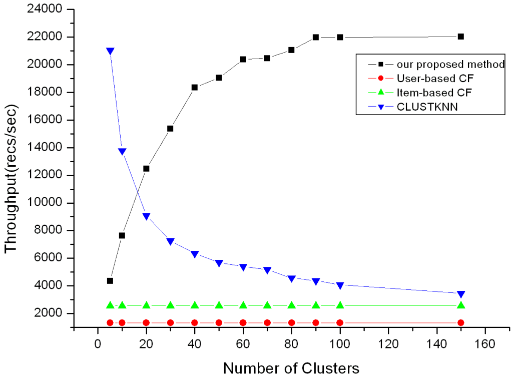


Fig. 4. Throughput of the selected recommendation algorithms

We make the following conclusions from the above experimental evaluation. First, the CK-modes algorithm is applied to divide the set of items O into several clusters, and then the rating predictions are computed within a small neighborhood. Hence, our proposed recommendation algorithm is highly scalable, which gives a quick response for active users. Second, the prediction quality of the hybrid recommendation algorithm is comparable with other classic recommendation algorithms. Only a minor difference on the prediction quality is observed. Finally, only the genres of movie are available in the MovieLens data set. Lacking of features information limits the accuracy of our hybrid recommendation algorithm. We regard that this hybrid recommendation algorithm will produce a better prediction quality when more features of movie item are available. As a matter of fact, finding similar movie items by using the movies' content is more reasonable and intuitive than by using the users' rating behaviors.

7 Conclusion and Future Work

Recommender systems play an important role in e-commerce for both users and businesses. It provides personalized recommendations for better business revenue. In this paper, we propose a hybrid recommendation algorithm by combining the content-based and the collaborative filtering techniques. This hybrid recommendation algorithm firstly partitions the items into several groups by using the coupled version of k-modes clustering algorithm. Then it uses the collaborative filtering technique to produce the recommendations for active users. Experimental results show that our proposed hybrid recommendation algorithm effectively solves the scalability issue of recommender systems with a comparable recommendation quality under the condition of lacking of most of the features.

We plan to extract more features of items to improve our proposed hybrid recommendation algorithm. We will also extend other recent clustering algorithms, such as spectral clustering algorithm, to speed up the process of model building and improve the prediction quality.

Acknowledgments. We would like to acknowledge the support for this work from the National Science Foundation of China (61035003, 61175042,61021062), the National 973 Program of China (2009CB320702), the 973 Program of Jiangsu, China (BK2011005), the Program for New Century Excellent Talents in University (NCET-10-0476), the Australian Research Council Discovery Grant (DP1096218, DP130102691) and ARC Linkage Grant (LP100200774).

References

1. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge and Data Engineering 17(6), 734–749 (2005)
2. Balabanović, M., Shoham, Y.: Fab: content-based, collaborative recommendation. Communications of the ACM 40(3), 66–72 (1997)
3. Melville, P., Mooney, R., Nagarajan, R.: Content-boosted collaborative filtering for improved recommendations. In: Proceedings of the National Conference on Artificial Intelligence, pp. 187–192. AAAI Press, MIT Press (1999, 2002)
4. Wang, C., Cao, L., Wang, M., Li, J., Wei, W., Ou, Y.: Coupled nominal similarity in unsupervised learning. In: CIKM, pp. 973–978. ACM (2011)
5. Cao, L., Ou, Y., Yu, P.: Coupled behavior analysis with applications. IEEE Transactions on Knowledge and Data Engineering 24(8), 1378–1392 (2012)
6. Gan, G., Ma, C., Wu, J.: Data clustering: theory, algorithms & applications (asasiam series on statistics & applied probability, n 20). Recherche 67, 02 (2007)
7. Ahmad, A., Dey, L.: A k-mean clustering algorithm for mixed numeric and categorical data. Data & Knowledge Engineering 63(2), 503–527 (2007)
8. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.: Grouplens: an open architecture for collaborative filtering of netnews. In: CSCW, pp. 175–186. ACM (1994)
9. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Item-based collaborative filtering recommendation algorithms. In: WWW, pp. 285–295. ACM (2001)
10. Krulwich, B., Burkey, C.: Learning user information interests through extraction of semantically significant phrases. In: Proceedings of the AAAI Spring Symposium on Machine Learning in Information Access, pp. 100–112 (1996)
11. Al Mamunur Rashid, S., Karypis, G., Riedl, J.: Clustknn: a highly scalable hybrid model-& memory-based cf algorithm. In: Proc. of WebKDD 2006, Citeseer (2006)
12. Xue, G., Lin, C., Yang, Q., Xi, W., Zeng, H., Yu, Y., Chen, Z.: Scalable collaborative filtering using cluster-based smoothing. In: SIGIR, pp. 114–121. ACM (2005)
13. Huang, Z.: Extensions to the k-means algorithm for clustering large data sets with categorical values. Data Mining and Knowledge Discovery 2(3), 283–304 (1998)

Location Recommendation Based on Periodicity of Human Activities and Location Categories

Seyyed Mohammadreza Rahimi and Xin Wang

Department of Geomatics Engineering, Schulich School of Engineeting,
University of Calgary, 2500 University Dr. N.W. Calgary, Alberta, Canada, T2N 1N4
{smrahimi,xcwang}@ucalgary.ca

Abstract. Location recommendation is a popular service for location-based social networks. This service suggests unvisited sites to the users based on their visiting history and site information. In this paper, we first present how to build the temporal and spatial probability distribution functions (PDF) to model the temporal and spatial checkin behavior of the users. Then we propose two recommender algorithms, Probabilistic Category Recommender (PCR) and Probabilistic Category-based Location Recommender (PCLR), based on the periodicity of user checkin behavior. PCR uses the temporal PDF to model the periodicity of users' checkin behavior. PCLR combines the temporal category model used in PCR with a geographical influence model built on the spatial PDF. The experimental results show that the proposed methods achieve better precision and recall than two well-known location recommendation methods.

Keywords: Recommender system, Location-based Social Networks, Location-Category, probability model.

1 Introduction

In location-based social networks (LBSN), people share location-related information with each other, and also leverage collaborative knowledge learned from user-generated and location-related content. Among various LBSN services, the location recommendation service suggests unvisited sites to the users based on the information collected on LBSNs, such as checkins, social ties, user profiles and location profiles.

The location recommendation has been an active research area. The existing methods focus on the "geographical influence" and the "social influence" on users checkin behavior [1, 2, 4]. Modeling the geographical influence, the recommender finds the probability of a user visiting locations based on the distance of the locations to the user's home [2] or to the previously visited locations by that user [1]. The challenge of the existing geographical influence-based methods is that they do not consider temporal effect on human checkin behaviors. Given a user's checkin history, the methods would recommend the same set of locations regardless of noon or midnight. The other attempt in the literature is to make use of the social ties among users. The assumption is that people have the similar checkin patterns with their friends. However, only 10~30% of the checkins are influenced by social links [2].

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 377–389, 2013.

In this paper, we approach the problem from an activity-based perspective. We believe that people behave on a periodic pattern. In the other words, people are more likely to conduct the same activity around the similar time of the day. Such temporal pattern exists but it is different from one person to another. Since the category of a location reflects the activities happening in that location, we believe a similar temporal pattern exists for the location categories. After analyzing checkin data of a real LBSN, we find such a pattern for the categories of locations. We also find that the further a location is away from the home location of the user, the lower chance he or she will visit that location. The probability decreases exponentially as the distance increases. But the degree of the decrease varies for different reaching distances. Thus, we could make better location recommendation if we divide the reaching distance of the user into home and away zones and find the spatial probability distribution functions for each zone separately.

In this paper, we combine the periodicity of user's behavior and the geographical influence into location recommendations. Specifically, our contributions in this study include:

- We present how to build the temporal and spatial probability distribution functions (PDF) to model the temporal and spatial checkin behavior of the users. The temporal PDF models the periodic pattern of user checkins. It is discovered in temporal analysis of the checkins. The spatial PDF is discovered in the spatial analysis of the checkins and models the probability of checking in to a location as a function of the distance of that location to user's home.
- We propose a category recommender algorithm called Probabilistic Category Recommender (PCR). It recommends category of locations to the users at a given time using the temporal PDF and the checkin history of those users.
- Based on PCR, we propose a location recommender algorithm called Probabilistic Category-based Location Recommender (PCLR) which uses the PCR's category model along with a spatial model. The spatial model is built upon the spatial PDF and measures the probability of checking in to a location based on the distance of that location to user's home. PCLR combines the probabilities of PCR's category model and the spatial model to find the probability of checking in to a location.
- We conduct the experiments on a real LBSN checkin dataset to evaluate the performance of the PCR and PCLR algorithms. We discover that the category recommendation (PCR) is more effective compared to the exact location recommendation (PCLR). In addition, PCLR outperforms two existing well-known location recommenders, PMM and USG.

The paper is organized as follows: in Section 2, we give a literature review on location recommendation algorithms for LBSNs. Section 3 introduces the dataset used in this paper and confirms the assumptions used in the algorithms through the data analysis. In Sections 4 and 5, we propose PCR and PCLR algorithms, respectively. The experimental results are presented in Section 6. Finally, we conclude the study in Section 7.

2 Related Works

The location recommendation is an active research area. Zheng et al. [5] recommend locations to the users based on the real-world location history of the users collected in GPS trajectories. Park et al. [6] proposed a method of providing personalized location recommendation to the users based on the location history of the users. Simon et al. [7] and Beeharee et al. [8] used the mobile tour guide systems which collect real-time location of the users for location recommendations. Zhou et al. [4] proposed a method called probabilistic latent semantic analysis (PLSA). It first trains a latent semantic model, and then it uses that model to find the probability of a user checking in to a given location. However, none of these methods consider the fact that human checkin behavior is influenced by the distance of user to the location of checkin. To add on, they do not use the temporal patterns of human checkin behavior for location recommendation.

Ye et al. [1] proposed a fusion framework USG consisting of three different models 1) a user-based collaborative filtering (CF) model, 2) a social influence model and 3) a geographical influence model. The user-based CF model estimates the implicit preference of a user for a location combining the behavior of similar users. The social influence model, which also is a CF model, estimates the implicit preference of a user by aggregating the behavior of his or her friends. Finally, the geographical influence model uses a power law distribution to find the probability of checkin at given distances from the users previously visited locations. The method builds different models for different aspects of location recommendation and provides a fusion framework for combining these models. However, applying the CF method for location recommendation may not be suitable since the similar users in a location recommendation approach might be in different locations. Therefore, recommending one's behavior to the other is not appropriate. Another trend in the recent research is making preference-aware recommendations using the location categories. Bao et al. [10] make recommendations using the checkin history of the local experts. Local experts are users with high expertise in user's preferred categories and the venues in the geospatial range of the user. However, their method lacks the temporal feature for recommendation which can be modeled using a periodic movement model for the users.

In the more recent researches the periodicity of the human behavior has gained the attention of researchers [2, 9, 11, 12, 13]. Eagle et al, [13] model the behavior of an individual using the weighted sum of a set of characteristic vectors called "eigenbehaviors". Li et al. [10, 11] define the periodic behavior of a moving object as "the repeating activities at certain locations with regular time intervals." and then mine the periodic movements of moving objects. Cho et al. [2] propose a location recommendation method based on the periodicity of the human movement. They propose two methods PMM (Periodic Mobility Model) and PSMM (Periodic Social Mobility Model). In PMM the user can be in home or work states and being in different states is defined using a temporal probability distribution function. The PSMM is based on PMM and adds the effect of social ties to it. However, the two methods assign the training checkins to different states randomly and iterate until an optimal classification is reached. Therefore, the checkins classified as work or home state might be belonging to the other group.

3 Dataset Description and Data Analysis

In this section, we will first describe the Gowalla data used in the paper. Then we will confirm two assumptions used by our location recommendation algorithms with the Gowalla data and illustrate how to build the probability distribution functions based on the data analysis.

The dataset used in this paper is collected from Gowalla, which was one of the popular online LBSN services until it was closed in 2012 (for details of the data crawler and data collection see [4].) The dataset contains 5462 users, 5999 locations and 104851 checkins. A checkin indicates a user has visited a location at certain time. In our dataset, a checkin includes user-id, spot-id, spot-latitude, spot-longitude, spot-category and timestamp. Spot-latitude and spot-longitude are the latitude and longitude of the checked-in location. Spot-category is the category of the checkin location, for example, “Coffee shop” or “Office”. Finally, the timestamp of the checkin shows the date and time the user visited the spot.

Before introducing our algorithms, we will confirm two underlying assumptions of our algorithms on Gowalla dataset. These two assumptions are: 1) People have temporal patterns for their daily activities and checkins (confirmed in temporal analysis); 2) People visit locations closer to their home with a higher probability compared to the further locations (confirmed in spatial analysis).

3.1 Temporal Analysis

We believe that people have a periodic behavior for visiting similar type of locations. For example, a person might go to coffee shops everyday at 8am but she might go to different coffee shops on different days. To test this, we first find the pairs of checkins to the same category of location from the same user, and then we plot the frequency of checkin pairs based on the time interval of those checkins. Fig. 1 shows the plot of frequency of checkin to the same category at given time differences using 1 hour time window.

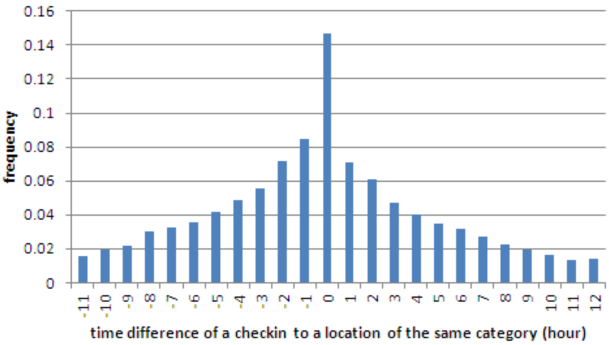


Fig. 1. Frequency of checkins to locations of the same category to the time difference of those checkins using 1 hour time window

As shown in Fig. 1, using the one-hour time window the probability of checkin is the highest (about 15%) for 0 hour time difference and it declines as the absolute value of the time difference increases. Adding up the frequency of checkins from -2 to 2 hour time difference, we find out that the probability of checking in to the locations from the same category is about 45% for the mentioned 5-hour time window. As an example, if a user checks in to a coffee shop at 8 am, the chance that she will checkin to a coffee shop (the same coffee shop or a different one) between 7:30am to 8:30am in the coming days is 15%, and the probability that she will checkin to a coffee shop between 5:30am and 10:30am in the coming days is about 45%.

After plotting the frequency of checkin to the same category given the time difference, we can make a temporal Probability Density Function (PDF) based on the plot. This PDF helps us quantify the probability of checkin to different categories at different times of the day. To do so, we first define the function *F*, consisting of 24 different constant outputs and the values of the outputs based on the checkin plot. For example the *F(Δt)* for Fig. 1 is defined as:

$$F(\Delta t) = \begin{cases} .15, & \lfloor \Delta t \rfloor = 0 \\ .07, & \lfloor \Delta t \rfloor = 1 \\ .085, & \lfloor \Delta t \rfloor = -1, \\ & \dots \\ .0018, & \lfloor \Delta t \rfloor = 12 \end{cases} \tag{1}$$

where Δt is the time difference and $\lfloor \Delta t \rfloor$ indicates the floor of t. Based on this, we can define the temporal probability distribution function TP for a given set of checkins as:

$$TP(t; \mu) = F(\lfloor t - \mu \rfloor), \tag{2}$$

where t is the time we compute the probability for and μ is the average time of the checkins in the subset.

3.2 Spatial Analysis

We believe the probability of a user visiting a location closer to their home location is higher than the probability of visiting locations farther from their homes. However, the home locations of the users are usually not given in the dataset. To find the home location, we assume that user checkins are centered at his home location. We are finding the home location of a user by averaging the locations visited by the user. But this estimation could be affected by some checkin locations when the user was on a trip, especially for the users with small number of checkins. To solve this problem, we first divide the surface of the earth into small non-overlapping regions and then find the region with the most number of checkins [2]. We consider the average point of the locations in that region as the candidate home location of the user. However, because we use fixed regions and average the locations in each of those regions, there's a chance that we are missing the actual home location. To solve this, we select all the checkins by the user in 100km radius of the candidate home location. 100 km is used as the human reach distance based on [1]. Finally, we average all the locations in this selected set of checkins to find the home location of the user.

Algorithm 1 shows the steps of finding the home location of the user using the user's checkin history. It first groups the checkins based on their regions (lines 1 to 3). In line 4 we select the region with maximum number of checkins, and find the average location of the checkins in the region as the candidate home location (line 5). Then we find all of the checkins in the reaching distance of the candidate home location (lines 6 to 10) and return the average of those locations as the home location of the user (line 11).

After having the home location of users, we will test whether the frequency of checking in to locations decreases as the distance to a user's home location increases. To do so, we first calculate the distance between locations checked in by the users and their home location and then compute the frequency of checking in at any distance of their home location. Fig. 2 shows the logarithmic scale plot of the probability of checkin over the distance to the home location of the user.

```
Algorithm 1 findHomeLocation (checkins)
// checkins is a set of checkins of a user
Begin
01-for each (c in checkins)
02-    region[c].add(c);
03-end for
04-selectedRegion <- maximumSized(region);
05-candidateHomeLocation <- average(checkins In selectedRegion);
06-for each (c in checkins)
07-    if (c is in reachingDistance of candidateHomeLocation)
08-      selectedCheckins.add(c);
09-    end if
10-end for
11-return average(selectedCheckins);
End.
```

Fig. 2 shows that: first, the checkin frequency values for distances greater than 50km vary randomly (shown as triangular points), which means that 50km is the range of human checkin behavior for our dataset and checkins happen on distances greater than 50km when the user is on a trip. Second, based on the slope of the linear relationship, we can separate the less than 50km part into two different parts, less than 16km (shown as diamond shaped points) and greater than 16km (shown as rectangular points). We can tell that the probability decreases more slowly in less than 16km part. In this case, we assume that the area within 16km radius of the user's home location is his home zone and the outside area is the away zone. Based on these findings we should use different PDFs for each of these zones in order to have a better fitting model.

To find the spatial PDF, we use exponential estimation to find the relationship of frequency and distance in each of the mentioned zones. Based on Fig. 2 we define the spatial probability distribution (SP) for Gowalla dataset as:

$$SP(l;h) = \begin{cases} 0.0886e^{-0.166*distance(l,h)}, & distance(l,h) \leq 16Km \\ 0.3122e^{-0.204*distance(l,h)}, & 16Km < distance(l,h) \leq 50Km, \\ 0, & 50Km < distance(l,h) \end{cases} \quad (3)$$

where l is the location for which we want to find the probability of checkin and h is the home location of the user.

4 Probabilistic Category Recommender

As discussed in Section 3.1, users are more likely to checkin to the locations of the same category around the same time of the day. Based on this, the user's checkin behavior and their checkin location categories can be used to predict the category of the location the user is going to visit. Additionally, this can be used to assign categories to uncategorized locations in dataset based on the behavior of the user explored those locations.

In this section, we describe how we build a temporal model based on the user behavior and how it is used for recommending the categories of locations. The temporal model is a user-specific model. Each user has a different model that is trained based on their checkins. For a given user and a given time of the day, this model will return a list of categories and the probability values that user will visit a location belonging to each of those categories.

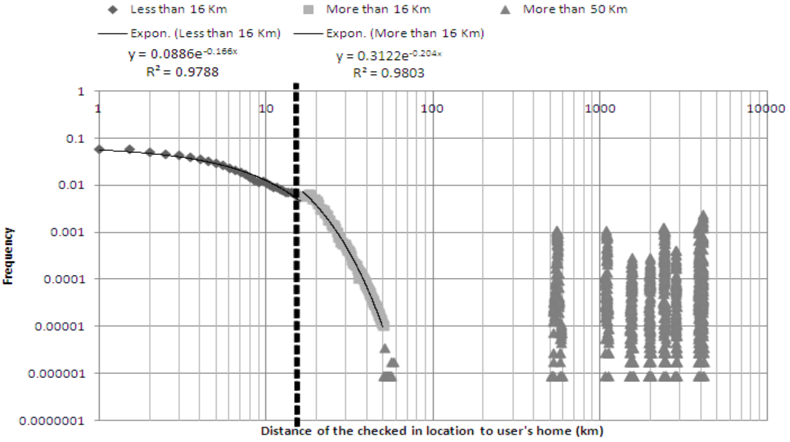


Fig. 2. Logarithmic scale of the frequency of checkins to the distance to user's home

To make this model, we start with the checkin history of the users. Then in order to find the similar checkins we separate the checkins into subsets based on their category and time. Doing this, we find checkin subsets where each subset contains checkins which have happened to locations of the same category and the timestamps showing the same time window. For example, all checkins to the coffee shops that happened between 4:00pm-4:59pm are put in one subset and checkins to coffee shops happened between 8:00am-8:59am are put in another subset. In order to build the temporal PDF introduced in the previous section, we need to find the average time of the checkins in each subset of checkins. Because we have one temporal PDF for each of the checkin subsets, we need a weighting value to normalize these temporal PDFs so that the whole model satisfies the second axiom of probability.

The average time of each subset helps us find the central point of the checkins of that specific type. The average time of the subset s_i is calculated as $\mu_{s_i} = \frac{\sum_{c_j \in s_i} t_{c_j}}{|s_i|}$, where s_i is a subset of checkins by a user. c_j is a checkin selected from s_i. μ_{s_i} is the average time of the checkins in s_i and t_{c_j} is the time of checkin c_j. The weight of each subset is the number of checkins in that subset divided by the whole number of checkins by that user. The weight of the subset s_i is defined as $w_{s_i} = \frac{|s_i|}{|checkins\ by\ u|}$, where $|s_i|$ the number of checkins in s_i. A larger value of weight shows that more checkins are assigned to that subset and hence that subset is of more importance.

Next, we calculate the probability of checkin to the category of that subset based on the temporal PDF.

The probability of user *u* checking in a location of category *c* at time *t* is:

$$T(u,c\,|t) = \sum_{\{s_{i\in}\in subsets[u] | category[s_i] = c\}} w_{s_i} * TP(t;\mu_{s_i}). \qquad (4)$$

This equation shows how to compute the temporal probability of a given category. The probability of checking in to a category is the summation of the weight of all subsets by that user matching the given category multiplied by the temporal probability of that subset at the given time. In this equation s_i is *i*-th subset of checkins with category *c* and w_{s_i} is the weight of subset s_i. *TP* is the temporal probability distribution function introduced in Eq. 2. Finally, $\mu_{\mathrm{s_i}}$ is the average time of the checkins in subset s_i.

```
Algorithm 2 buildCategoryModel (checkins, u)
// checkins is a set of checkins of an user u
Begin
For each (c in checkins)
  category <- getLocationCategory(c);
  h <- getHourOfDay(c.time);
  addToSubset(u.subsets[category,h],c);
endfor;
For each (s in u.subsets)
  Weight[s] <- size(s)/size(checkins);
  Average[s] <- averageTimeOfCheckins(s);
End for;
End.
```

```
Algorithm 3 calculateCategoryProbability (u, category, time)
Begin
P <- 0;
For each (s in u.subsets)
  If (category (s) = category )
     P <- P + weight[s] * F ([time - average[s]]);
  End if
End For
Return P;
End.
```

```
Algorithm 4 PCRrecommendCategories (u, time, k)
Begin
 for each (c in categories)
   probability[c] <- calculateCategoryProbability(u , c , time);
 end for
 sortedCategories <- sort(categories based on probability);
 return sortedCategories[1] ... [k];
End.
```

The algorithms for building the model and finding the probability of checkins are given in Algorithm 2, 3 and 4. The `buildCategoryModel` algorithm (Algorithm 2) is responsible for making the temporal probability model and it is the starting point of the PCR method. As stated earlier, the first step in building the PCR model is to group the checkins into subsets of checkins with the same category and the same time window. So the first loop (lines 1 to 5) separates the checkins into subsets based on the category and the time. The next step is to find the weight of the subsets and average time of the checkins in each subset. The second loop (lines 6 to 9) is responsible for this task.

In order to find the probability of each category at given times, we need to calculate the temporal probability introduced in Eq. 4. The Algorithm 3 `calculateCategoryProbability` finds the probability of checkins to a specific category at a given time for an individual user. The algorithm needs to first find the subsets belonging to the given category for the user. The loop starting on line 2 loops over all subsets, and using an If-statement finds the satisfying subsets. In line 4, the probability is computed using the Eq. 4.

Now we have the methods for building the model and calculating the probability of each category we can make the recommender algorithm (Algorithm 4). To make the category recommendation, we loop over all categories and calculate the probability of checkin to that category at the given time (lines 1 to 3). Then we sort the categories based on the probability (line 4) and return the top-k categories to the users (line 5).

Example: Consider a user with checkins to 8 locations. 3 of them to coffee shops around 8:15, 2 of them to coffee shops around 17:20 and the remaining three to fast food restaurants around 17:20. This user requests for recommendation at 18:01 on Sept 21st 2012. To recommend a category based on his checkin history, the first step is to build the PCR model using the Algorithm 2. So first we separate his checkin history into three different subsets (s1:"Coffee shop, around 8:15",s2:Fast food around 12:30. and s3:Coffee shop, around 17:20). Then we find the weight and average time of each subset: w_1 = 3/8, w_2=3/8, w_3 = 2/8, μ_1 = 8:15, μ_2 = 12:30 and μ_3 = 17:20. Now we have the PCR temporal model. The next step is finding the probability of different categories for the given time. Following the steps of Algorithm 3 the probability of checkin to a coffee shop and a fast food can be calculated as follows:

$$\begin{aligned} P(John, coffee\ shop\,|t) &= w_1 * TP(time; \mu_1) + w_3 * TP(time; \mu_3) \\ &= .375 * F\,(\lfloor 18{:}01 - 8{:}1\rfloor + .25 * F(\lfloor 18{:}01 - 17{:}20\rfloor) = 0.0425 \end{aligned}$$

$$P(John, fast\ food\,|t) = w_2 * TP(time; \mu_2) = .375 * F\,(\lfloor 18{:}01 - 12{:}30\rfloor) = 0.013$$

5 Probabilistic Category-Based Location Recommender

In this section, we propose a new location recommendation algorithm called Probabilistic Category-based Location Recommender (PCLR). PCLR extends from PCR and it combines the geographical influence and the recurring pattern of the user activities to improve the location recommendation. The idea of PCLR is to first find the locations the user is most likely to visit based on the geographical influence and then, in order include the effect of time, weight those locations using the probability values of the category of those locations which is done using the PCR algorithm.

The spatial component of the PCLR algorithm which is responsible for the geographical influence is based on the probability distribution suggested in Eq. 3; it also uses the PCR algorithm to weight the recommended locations found using their geographical influence. Combining these two, the probability of user (u) checkin to the location (l) at the given time (t) is defined as:

$$P(u, l\,|t) = SP(l; home_u) * \mathrm{T}(u, c_l|t), \tag{5}$$

where $home_u$ is the home location of user u and c_l is the category of location l. SP is the spatial probability of visiting location l given the home location of the user (Eq. 3) and T is the temporal probability of checking in the category of location l at given time t (Eq. 4).

For the previous example, if a coffee shop is in the distance of 5km from user's home location. The spatial probability that the user checks in that location is 0.024. The temporal probability of checkin to a coffee shop as we computed in the previous section is 0.0425. Then the probability of the sample user checking into that specific coffee shop is: $sp * tp = 0.024 * 0.0425 = 0.00102$

The PCLR algorithms are shown in Algorithms 5, 6 and 7. The `buildPCLRModel` algorithm (Algorithm 5) is responsible for making the PCLR model. It first makes the PCR model in line 1, and then finds the home location of the user using Algorithm 1 (line 2).

Algorithm 6 `calculateLocationProbability` computes the probability of a checkin for an individual user to a certain location at a given time using Eq. 5. The first line of this algorithm finds the spatial probability of location using the Eq. 3. The second line calls the `calculateCategoryProbability` to find the probability the user will checkin to a location of the same category as location l. Line 3 finds the probability of checkin to the location by multiplying these two probabilities as is suggested earlier in Eq. 5.

The PCLR location recommender (Algorithm 7). first finds the probability of the user checking in to candidate locations (lines 1 to 3) and recommends the top-k locations to the user (lines 4 and 5).

```
Algorithm 5 buildPCLRModel ( checkins , u)
// checkins is a set of checkins of user u
Begin
1- buildCategoryModel( chechins, u)
2- home[u] <- findHomeLocation( checkins )
End.
```

```
Algorithm 6 calculateLocationProbability (u, l, t)
// Returns the probability user u check in to location l at time t
Begin
// SP is the PDF introduced in the section 3.2.
1- spatialProbability <- sp(l;h);
2- tp <- calculateCategoryProbability (u, category[l], t);
3- return spatialProbability * tp;
End.
```

```
Algorithm 7 PCLRrecommendLocations (u, time, k)
Begin
1- for each (l in locations)
2-   probability[c] <- calculateLocationProbability (u , l , time);
3- end for
4- sortedLocations <- sort(locations based on probability);
5- return sortedLocations[1] ... [k];
End.
```

6 Experiments

We implement the algorithms in Java and use a Mac with 8GB of ram and a 2.3GHz, Intel Core i5 CPU for the experiments. We divide the Gowalla data described in Section 3 into the training and testing datasets. To do so, we randomly move one of the checkins of every user to the testing dataset and leave the rest in the training dataset. As the result, the training dataset contains 99389 checkins and the testing dataset contains 5462 checkins. We randomly generated 5 groups of different training and testing datasets and report the average performance from five runs.

To evaluate the performance of the algorithm, we use Precision and Recall. Precision is the ratio of the number of relevant instances to the number of retrieved instances, while recall is the ratio of the number of relevant instances retrieved to the number of relevant instances. They are defined as:

$$Precision = \frac{Number\ of\ correct\ recommendations}{Number\ of\ recommendation}$$

$$Recall = \frac{Number\ of\ correct\ recommendations}{Number\ of\ correct\ answers}$$

We compare the performance of our proposed algorithms with two existing methods, Periodic Mobility Model (PMM) proposed by Cho et al. [2] and the USG model proposed by Ye et al [1]. However, because these two methods do not provide a category recommender, we modify them in order to have a fair comparison with the PCR algorithm. We change the two methods to return the categories of the top locations instead of the exact location itself. We call the new PMM and USG methods PMM+c and USG+c, respectively. Fig. 3 shows the precision and recall values of different recommender algorithms.

From Fig. 3, we can tell that both PCR and PCLR perform better than competing methods. Considering the exact location recommender algorithms, we find out that PCLR outperforms both PMM and USG regardless precision and recall (Fig.3. (a) and (b)). This proves that using the location categories and periodic user behavior help improve the location recommendation. We also discover that PMM outperforms USG. This could be the result of PMM benefited from a periodic model of human movements which USG did not.

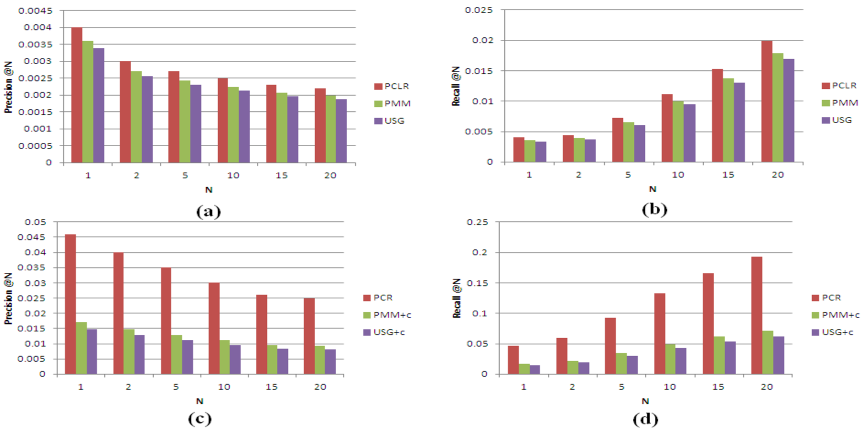


Fig. 3. Comparison of different recommender algorithms on Gowalla checkins. (a) Precision of PCLR, PMM and USG location Recommenders. (b) Recall of PCLR, PMM and USG location Recommenders. (c) Precision of category recommender algorithms. (d) Recall of category recommender algorithms.

As for the category recommender algorithms, PCR performs better than PMM+c and USG+c with a big margin considering both precision and recall (Fig. 3 (c) and (d)). The reason is that PCR is specifically built for category recommendation. It uses the temporal PDFs derived from the dataset for the categories. We also observe that PMM+c performs better than USG+c. Again we think this is because PMM+c uses a periodic model.

7 Conclusions and Future Work

In this paper, using the data collected from Gowalla, we discover that users have a recurring behavior of visiting locations over time. We also find that users are more likely to visit locations near their home. Based on these findings, two recommenders, a category recommender (PCR) and a location recommender (PCLR), are proposed. PCR provides suggestions with the location category for the next user visit. PCLR provides recommended locations to the users at a given time of the day. Experimental results show that the recommending location category to the user is more effective than the exact location. Our proposed algorithms perform much better than the existing well-known methods.

In the future, we will build more complicated models for both spatial and temporal components. We will evaluate our algorithms on larger datasets and compare it with other existing models as well. To add on, we are also planning to study the relationship of the social ties and the user checkin behaviors in order to improve our current models.

Acknowledgement. The research is supported by Canada NSERC Discovery Grant.

References

1. Ye, M., Ying, P., Lee, W., Lee, D.: Exploiting Geographical Influence for Collaborative Point-of-Interest Recommendation. In: 34th ACM International Conference on Research and Development on Information Retrieval, Beijing, China, pp. 325–344 (2011)
2. Cho, E., Myers, S., Leskovec, J.: Friendship and Mobility: User Movement In Location-Based Social Networks. In: 17th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, San Diego, California, USA, pp. 1082–1090 (2011)
3. Cheng, Z., Caverlee, J., Lee, K., Sui, D.: Exploring millions of footprints in location sharing services. In: 5th International Conference on Weblogs and Social Media, Barcelona, Spain, pp. 81–88 (2011)
4. Zhou, D., Wang, B., Rahimi, S.M., Wang, X.: A Study of Recommending Locations on Location-Based Social Network by Collaborative Filtering. In: Kosseim, L., Inkpen, D. (eds.) Canadian AI 2012. LNCS, vol. 7310, pp. 255–266. Springer, Heidelberg (2012)
5. Zheng, V.W., Zheng, Y., Xie, X., Yang, Q.: Collaborative Location and Activity Recommendations with GPS History Data. In: 19th International Conference on World Wide Web, Raleigh, North Carolina, USA, pp. 1029–1038 (2010)
6. Park, M.-H., Hong, J.-H., Cho, S.-B.: Location-Based Recommendation System Using Bayesian User's Preference Model in Mobile Devices. In: Indulska, J., Ma, J., Yang, L.T., Ungerer, T., Cao, J. (eds.) UIC 2007. LNCS, vol. 4611, pp. 1130–1139. Springer, Heidelberg (2007)
7. Simon, R., Frőhlich, P.: A Mobile Application Framework for the Geospatial Web. In: 16th International Conference on World Wide Web, Banff, Alberta, Canada, pp. 381–390 (2007)
8. Beeharee, A., Steed, A.: Exploiting Real World Knowledge in Ubiquitous Applications. Personal and Ubiquitous Computing Archive 11(6), 429–437 (2007)
9. Wang, J., Prabhala, B.: Periodicity Based Next Place Prediction. In: Workshop on Mobile Data Challenge by Nokia, Newcastle, UK (2012)
10. Bao, J., Zheng, Y., Mokbel, M.: Location-based and Preference-Aware Recommendation Using Sparse Geo-Social Networking Data. In: 20th ACM SIGSPATIAL International Conference on Advances in GIS. Redondo Beach, California (2012)
11. Li, Z., Ding, B., Han, J., Kays, R., Nye, P.: Mining periodic behaviors for moving objects. In: 16th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, pp. 1099–1108 (2010)
12. Li, Z., Wang, J., Han, J.: Mining event periodicity from incomplete observations. In: 18th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Beijing, China, pp. 444–452 (2012)
13. Eagle, N., Pentland, A.: Eigenbehaviors: identifying structure in routine. Behavioral Ecology and Sociobiology 63, 1057–1066 (2009)

Top-N Recommendations by Learning User Preference Dynamics

Yongli Ren, Tianqing Zhu, Gang Li*, and Wanlei Zhou

School of Information Technology, Deakin University,
221 Burwood Highway, Vic 3125, Australia
{yongli,ztianqin,gang.li,wanlei}@deakin.edu.au

Abstract. In a recommendation system, user *preference patterns* and the *preference dynamic effect* are observed in the *user* × *item* rating matrix. However, their value has barely been exploited in previous research. In this paper, we formalize the preference pattern as a sparse matrix and propose a *Preference Pattern Subspace* to iteratively model the *personal* and the *global* preference patterns with an EM-like algorithm. Furthermore, we propose a *PrepSVD-I* algorithm by transforming the Top-N recommendation as a pairwise preference learning process. Experiment results show that the proposed *PrepSVD-I* algorithm significantly outperforms the state-of-the-art Top-N recommendation algorithms.

1 Introduction

Although recommendation system research has seen the development of techniques about *rating prediction*, the majority of commercial recommender systems aims to generate a list of recommended items, which is the task of *Top-N recommendation* [9].

Various techniques have been proposed for Top-N recommendations. Most of them are based on the modelling of user rating patterns by analysing the *user* × *item* rating matrix. These methods show improved performance, but their abilities in Top-N recommendations are still limited by the availability of user ratings. Specifically, most of the available ratings are given to a small fraction of items, and this is known as the *long tail effect* [2]. As shown in Fig. 1c, 33% of ratings are observed from only around 5.5% of items in the *MovieLens* data set. These items are referred as *popular* items, while the other 94.5% are referred as *unpopular* or *long tail* items [4]. Thus, the *long tail effect* indicates that ratings on *long tail* items are much fewer. Consequently, these analysing methods for rating patterns are naturally limited by the *long tail effect.*

In this work, we observe that each user has a *preference pattern* that is different from his/her rating pattern, and the *preference pattern* tends to change over time. For example, Fig. 1a and 1b show the user *preference patterns* and the temporal dynamics observed on a real movie recommender system, *MovieLens.* Specifically, Fig. 1a shows that fresh users tend to rate movies from a larger

[*] Corresponding author.

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 390–401, 2013.

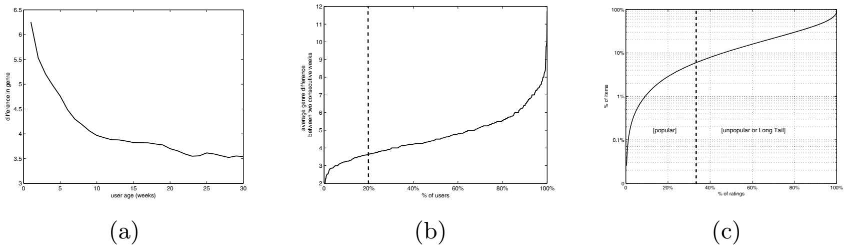


Fig. 1. Preference dynamics in *MovieLens*. a) genre difference by user age; b) user distribution over genre difference; c) rating distribution over item popularity.

range of genres than experienced users. Fig. 1b shows that, about 80% of users rated movies that spread over at least 3.5 genres on average during every two consecutive weeks. These observations indicate the existence of patterns on user preference styles, as well as their dynamics. In this paper, we name this new effect as the *preference dynamic effect.* Please note that the temporal characteristic in user *preference patterns* is different from the one observed by Koren [12], which is the temporal dynamics in user *rating patterns.*

In this paper, we focus on the modelling of the *preference dynamic effect* with the *Preference Pattern Subspace.* The basic idea is to model the *user preference styles* and their temporal dynamics by constructing a low-rank subspace. Firstly, a low-rank subspace is built to capture the *global preference patterns* for all users; then, the projection for each *personal preference pattern* on the subspace is individually refined based on his/her own preference styles. After that, the refined user projections on the subspace are used to improve the modelling of the *global preference patterns.* Iteratively, we can obtain a well-trained low-rank subspace to model both the *user preference styles* and their temporal dynamics. Based on the model, we formulate Top-N recommendation as a pairwise preference learning process, and propose a *PrepSVD-I* algorithm. Experimental results show that *PrepSVD-I* significantly outperforms the state-of-the-art Top-N recommendation techniques, especially when recommending *long tail* items. The contributions of this paper are as follows:

- For the first time, the *preference dynamic effect* is proposed to capture the personal and temporal characteristics of user preferences.
- We propose a novel *Preference Pattern* model and a subspace approach, *Preference Pattern Subspace*, to model the *preference dynamic effect.*
- Based on the *Preference Pattern Subspace*, we formulate Top-N recommendation as a pairwise preference learning process.

The rest of this paper is organized as follows. In Section 2, the *Preference Pattern* is proposed. In Section 3, we present *Preference Pattern Subspace.* We present the results of the experiment in Section 4, and the conclusion in Section 5.

2 The Preference Pattern

In this section, we propose a novel *Preference Pattern* model to capture the *preference dynamic effect*. Notations used in this paper are summarized in Table 1.

Table 1. Symbols

Symbol	Description
u_a	the active user for whom the recommendations are generated
r_{xl}	the rating on an item t_l by user u_x
$\mathcal{R}$	the *user* × *item* rating matrix
$T_N(u_x)$	the list of Top-N items recommended to user u_x
$\mathcal{U} = \{u_1, \cdots, u_m\}$	a set of m users
$\mathcal{T} = \{t_1, \cdots, t_n\}$	a set of n items
$\mathcal{C} = \{c_1, \cdots, c_q\}$	a set of q categories
$\mathbf{p}_x$	the *preference pattern* for user u_x
$\mathcal{P} = \{\mathbf{p}_1, \cdots, \mathbf{p}_m\}$	the *preference patterns* for m users
$\mathbf{v}_x$	the projection of $\mathbf{p}_x$
$\mathcal{V} = \{\mathbf{v}_1, \cdots, \mathbf{v}_m\}$	the projection of $\mathcal{P}$ on a low-rank subspace

Definition 1. *A* preference pattern *is a sequence of personal preference styles aligned in a time order. Precisely, for user u_x, the* preference pattern *is denoted as $\mathbf{p}_x = [p_{x1}, \cdots, p_{xi}]^T$, where $p_{xi} = [p^1_{xi}, \cdots, p^q_{xi}]$ denotes the preferences of u_x at time i over category $\mathcal{C} = [c_1, \cdots, c_q]$, and p^j_{xi} denotes the preference of u_x at time i on c_j.*

The *preference pattern* has two key characteristics, *personalization* and *time.* All preference styles within a *preference pattern* come from the same user, and are sorted in a time order. For a particular user u_x, a *preference style* refers to his/her preferences over a range of categories (e.g. *genre* in movies/songs) of items at a particular time. The preference style at time i can be represented as a q-D vector p_{xi}, which is defined as a *preference pattern vector.* Its value at position j, p^j_{xi}, indicates the preference of user u_x over category j at time i. For the value of p^j_{xi}, we approximate it as a function of the implicit rating history of user u_x. Formally, p^j_{xi} is defined as follows:

$$p^j_{xi} = \sum_{l=1}^{n} b^j_{xl}, \tag{1}$$

where n is the number of items, and

$$b^j_{xl} = \begin{cases} \frac{1}{|\mathcal{C}_l|} & r_{xl} \neq \oslash, t_l \in c_j, r_{xl} \text{ is given at time i} \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

where $r_{xl} \neq \oslash$ denotes that r_{xl} is available, $\mathcal{C}_l$ denotes a set of categories to which t_l belongs, and $t_l \in c_j$ denotes that item t_l belongs to category c_j. Please note that there is an inverse relationship between b^j_{xl} and $|\mathcal{C}_l|$ in Eq. 2.

The *preference pattern* is defined for each user in a way to naturally integrate a user's various needs with the corresponding dynamics. The *preference pattern* *vectors* within a preference pattern captures the user's corresponding *preference styles*, and the differences in two consecutive preference pattern vectors imply the dynamics of a user's *preference styles.* If all values in each *preference pattern vector* are available, the preference pattern is *complete*, otherwise it is *incomplete.* In a real recommender system, as many missing values exist within the preference patterns, the modelling of preference patterns is a challenge.

3 The Preference Pattern Subspace

In this section, we build a *Preference Pattern Subspace* to model the preference patterns for each user, then propose the *PrepSVD-I* algorithm by formulating the task of Top-N recommendation as a pairwise preference learning process.

3.1 Learning the Preference Pattern Subspace

To model the user preference patterns, we propose a *Preference Pattern Subspace* by applying the *Singular Value Decomposition* (SVD). Conventionally, the SVD of the preference patterns $\mathcal{P}$ is the factorization of the form: $\mathcal{P} = \mathcal{U} \cdot \Sigma \cdot \mathcal{V}^T$, where $\mathcal{U}$ is an $m \times m$ orthogonal matrix, Σ is an $m \times n$ diagonal matrix containing the singular values of $\mathcal{P}$ on the diagonal, $\mathcal{V}$ is an $n \times n$ orthogonal matrix.

According to the *Eckart-Young theorem* [7], it is well-known that the best rank-k approximation of matrix $\mathcal{P}$ can be achieved by SVD. However, conventional SVD is defined without considering the existence of missing values. Therefore, as $\mathcal{P}$ is highly incomplete, SVD can not be directly applied to analyse preference patterns $\mathcal{P}$. To overcome this problem, we propose an EM-like learning algorithm to capture as much as possible the main variance of the highly incomplete preference patterns $\mathcal{P}$. Specifically, $\mathbf{p}_x$ can be divided into two parts, $\mathbf{p}_x^a$ and $\mathbf{p}_x^m$, representing the *available* part and the *missing* part of $\mathbf{p}_x$, respectively. We estimate $\mathcal{P}$ using SVD to construct a low rank k subspace:

$$\hat{\mathcal{P}} = \mathcal{U}_k \cdot \Sigma_k \cdot \mathcal{V}_k^T, \tag{3}$$

where $\mathcal{U}_k$ contains the first k columns of $\mathcal{U}$, Σ_k is a $k \times k$ diagonal matrix that contains the first k singular values of $\mathcal{P}$, and $\mathcal{V}_k$ contains the first k columns of $\mathcal{V}$. Consequently, the reconstruction $\hat{\mathbf{p}}_x$ of $\mathbf{p}_x$ is defined as:

$$\hat{\mathbf{p}}_x = \mathcal{U}_k \cdot \Sigma_k \cdot \mathbf{v}_x^T, \tag{4}$$

where $\mathbf{v}_x$ is the xth row of $\mathcal{V}_k$, and denotes the projection of $\mathbf{p}_x$ for user u_x on the low rank k subspace. Similarly, the reconstruction $\hat{\mathbf{p}}_x$ can also be divided into two parts $\hat{\mathbf{p}}_x^a$ and $\hat{\mathbf{p}}_x^m$, representing the reconstruction of $\mathbf{p}_x^a$ and $\mathbf{p}_x^m$, respectively. The SVD guarantees to produce the best k-rank approximation of $\mathcal{P}$ with minimal reconstruction errors. However, as $\mathcal{P}$ is highly incomplete, we change the modelling objective to minimize the reconstruction error on the available preferences in $\mathcal{P}$. This is defined as the squared distance between the original available part of $\mathcal{P}$ and their reconstructions:

$$\varepsilon^a = \frac{1}{m} \sum_{x=1}^{m} (\mathbf{p}_x^a - \hat{\mathbf{p}}_x^a)^T \cdot (\mathbf{p}_x^a - \hat{\mathbf{p}}_x^a), \tag{5}$$

where m is the number of users.

To build the representative subspace, an EM-like algorithm is introduced as follows: first, the missing values of $\mathcal{P}$ are replaced with their corresponding values in $\mu = \frac{1}{m}\sum_{x=1}^{m}\mathbf{p}_x$. Then, in the j-th iteration, the standard SVD algorithm is applied to calculate a low-rank subspace defined by $\mathcal{U}_k$ and Σ_k. After that, the reconstruction $\hat{\mathbf{p}}_x$ of $\mathbf{p}_x$ can be calculated with Eq. 4. However, as only a small fraction of preferences are available in $\mathbf{p}_x \in \mathcal{P}$, its projection $\mathbf{v}_x$ can not be directly estimated from Eq. 3. Please note that we can estimate $\mathbf{v}_x$ from part of $\mathbf{p}_x$, e.g, the *available* part $\mathbf{p}_x^a$, and this estimation method has been widely used in the field of multimedia research [6]. Thus, we estimate $\mathbf{v}_x$ as the least squares solution for the following equation:

$$(\mathcal{U}_k \cdot \Sigma_k) \cdot \mathbf{v}_x^T = [\mathbf{p}_x]^a, \tag{6}$$

where $[\mathbf{p}_x]^a$ denotes $\mathbf{p}_x$ in the current iteration step but only has values on the positions corresponding to $\mathbf{p}_x^a$. After $\mathbf{v}_x$ is estimated, the reconstruction $\hat{\mathbf{p}}_x$ of $\mathbf{p}_x$ can be calculated using Eq. 4. $\mathbf{p}_x^m$ will then be updated with $\hat{\mathbf{p}}_x^m$, and the new $\mu^{(j+1)}$ in the next $(j+1)$-th iteration will be calculated with the updated $\mathbf{p}_x$ accordingly. With the updated $\mathcal{P}$ and the new mean vector $\mu^{(j+1)}$, the SVD algorithm is once again applied to calculate $\mathcal{U}_k$ and Σ_k. This iterative process will continue until the reconstruction error ε^a is below a pre-defined threshold. The proof for the convergence of this training algorithm is provided as follows.

Proof 1. In the j-th iteration, the reconstruction $\hat{\mathbf{p}}_x$ of $\mathbf{p}_x$ is defined as: $\hat{\mathbf{p}}_x^j = (\mathcal{U}_k \cdot \Sigma_k)^j \cdot \mathbf{v}_x^T$, where $(\mathcal{U}_k \cdot \Sigma_k)^j$ denote $\mathcal{U}_k$ and Σ_k in the j-th iteration. We denote $\hat{\mathbf{p}}_x^j$ obtained with $(\mathcal{U}_k \cdot \Sigma_k)^j$ as $f_{\hat{\mathbf{p}}_x}^j(\mathcal{U}_k \cdot \Sigma_k)^j$. Please note that $f_{\hat{\mathbf{p}}_x}^j(\mathcal{U}_k \cdot \Sigma_k)^j$ and the data in the next iteration, $\mathbf{p}_x^{j+1}$, share values on missing part of $\mathbf{p}_x$, thus the reconstruction error on $\mathbf{p}_x^j$ is represented as:

$$(\varepsilon_x^a)^j = d\left(f_{\hat{\mathbf{p}}_x}^j(\mathcal{U}_k \cdot \Sigma_k)^j, \mathbf{p}_x^{j+1}\right), \tag{7}$$

where $d(\cdot,\cdot)$ is the *Euclidean* distance between two vectors.

If we use $(\mathcal{U}_k \cdot \Sigma_k)^j$ to calculate the reconstruction of $\mathbf{p}_x^{j+1}$, we obtain

$$d\left(f_{\hat{\mathbf{p}}_x}^{j+1}(\mathcal{U}_k \cdot \Sigma_k)^j, \mathbf{p}_x^{j+1}\right) \leq d\left(f_{\hat{\mathbf{p}}_x}^j(\mathcal{U}_k \cdot \Sigma_k)^j, \mathbf{p}_x^{j+1}\right), \tag{8}$$

because the orthogonal property of $(\mathcal{U}_k \cdot \Sigma_k)^j$ makes it sure that $f_{\hat{\mathbf{p}}_x}^{j+1}(\mathcal{U}_k \cdot \Sigma_k)^j$ and $\mathbf{p}_x^{j+1}$ have the minimum *Euclidean* distance.

In the $(j+1)$-th iteration, after applying SVD to the updated training data $\mathbf{p}_x^{j+1}$, we observe the minimum reconstruction error by obtaining $(\mathcal{U}_k \cdot \Sigma_k)^{j+1}$:

$$d\left(f_{\hat{\mathbf{p}}_x}^{j+1}(\mathcal{U}_k \cdot \Sigma_k)^{j+1}, \mathbf{p}_x^{j+1}\right) \leq d\left(f_{\hat{\mathbf{p}}_x}^{j+1}(\mathcal{U}_k \cdot \Sigma_k)^j, \mathbf{p}_x^{j+1}\right). \tag{9}$$

For the reconstruction error in the $(j+1)$-th iteration, we obtain

$$(\varepsilon_x^a)^{j+1} = d\left(f_{\hat{\mathbf{p}}_x}^{j+1}(\mathcal{U}_k \cdot \Sigma_k)^{j+1}, \mathbf{p}_x^{j+1}\right) \leq \left(f_{\hat{\mathbf{p}}_x}^{j+1}(\mathcal{U}_k \cdot \Sigma_k)^j, \mathbf{p}_x^{j+1}\right) \leq (\varepsilon_x^a)^j. \tag{10}$$

$$(\varepsilon^a)^{j+1} = \frac{1}{m}\sum_{x=1}^{m}(\varepsilon_x^a)^{j+1} \leq \frac{1}{m}\sum_{x=1}^{m}(\varepsilon_x^a)^j = (\varepsilon^a)^j. \tag{11}$$

Thus, the algorithm will converge to minimize ε^a. □

The modelling process is an iterative refinement of the global and the personal preference patterns. One advantage of this EM-like learning algorithm is that the well-trained *Preference Pattern Subspace* can model both the personal preference patterns and the global preference patterns simultaneously.

3.2 Recommendation Generation

After learning the well-trained *Preference Pattern Subspace*, we propose a *PrepSVD-I* algorithm to generate Top-N recommendations. In this paper, we apply the latent factor model to estimate the ratings $\hat{r}_{xl}$ for user u_x on item t_l:

$$\hat{r}_{xl} = \rho_x^T \rho_l, \tag{12}$$

where ρ_x and ρ_l are the user factors and the item factors, and can be learnt with stochastic gradient descent method by looping through available ratings.

Given the user factors and the item factors, $T_N(u_x)$ can be generated by estimating ratings for un-known items with Eq. 12, then formed with the Top-N ranked ones. For $t_l \in T_N(u_x)$, t_l can be temporarily absorbed into u_x's preference pattern vector p_{xi}, and a tentatively changed preference pattern vector $\tilde{p}_{xi}$ is available. Please note that because we only want to measure the degree to which the recommendations match u_x's preference styles captured by the *Preference Pattern Subspace*, we initialize $\tilde{p}_{xi}$ as empty while keeping the other part the same as $\mathbf{p}_x$. The value at the jth position of $\tilde{p}_{xi}$ is then defined as:

$$\tilde{p}_{xi}^j = \begin{cases} \frac{1}{|\mathcal{C}_l|} & t_l \in c_j, c_j \subseteq \mathcal{C}_l \\ 0 & \text{otherwise,} \end{cases} \tag{13}$$

where $\mathcal{C}_l$ denotes a set of categories that t_l belongs to, $t_l \in c_j$ denotes that t_l belongs to category c_j. The reconstruction error for t_l to u_x at time i is defined as the squared distance between the changed preference pattern and its reconstruction:

$$\varepsilon_{t_l}^a = (\tilde{\mathbf{p}}_x^a - \hat{\mathbf{p}}_x^a)^T \cdot (\tilde{\mathbf{p}}_x^a - \hat{\mathbf{p}}_x^a), \tag{14}$$

where $\tilde{\mathbf{p}}_x^a$ is the available part of $\tilde{\mathbf{p}}_x$, $\hat{\mathbf{p}}_x$ is the reconstruction of $\tilde{\mathbf{p}}_x$ and is calculated with Eq. 6 and Eq. 4, while $\hat{\mathbf{p}}_x^a$ is the available part of $\hat{\mathbf{p}}_x$.

Moreover, the observed *preference dynamic effect* implies one constraint to the objective function of recommendation generations. It can be formulated as:

$$\forall t_l \in T_N(u_x), \varepsilon_{t_l}^a \leq \bar{\varepsilon}_{u_x}^a, \tag{15}$$

where $\bar{\varepsilon}_{u_x}^a = \frac{1}{|S(u_x)|} \sum_{t_{l'} \in S(u_x)} \varepsilon_{t_{l'}}^a$ and $S(u_x)$ is the set of items liked by u_x. Following this, we formulate the Top-N recommendation generation as a pairwise preference learning problem [8], and utilize the user average reconstruction $\bar{\varepsilon}_{u_x}^a$ as the negative preference:

$$\min_{\theta,\xi} \sum_x \xi_x + \lambda\Big(\sum_{u_x \in \mathcal{U}} \|\rho_x\|^2 + \sum_{t_l \in \mathcal{T}} \|\rho_l\|^2\Big) \tag{16}$$
$$\text{s.t.:} \varepsilon_{t_l}^a - \bar{\varepsilon}_{u_x}^a \geq 1 - \xi_x \text{ and } \xi_x \geq 0$$

where ξ_x is a non-negative value measuring the degree of violating the constraint in Eq. 15, λ is the regularization weight determined by cross validation.

We apply a simple gradient descent algorithm to optimize the objective function defined in Eq. 16. Moreover, as there is an inverse relationship between b_{xl}^{j} and $|\mathcal{C}_l|$ in the approximation of preference values in Eq. 2, we name this proposed algorithm as the *PrepSVD-I* algorithm. It loops on all $T_N(u_x), \forall u_x \in \mathcal{U}$, and updates the user factors and the item factors by following the negative gradient:

$$\rho_l = \rho_l - \gamma(h'(\rho_x^T \rho_l)\rho_x + \lambda\rho_l) \tag{17}$$

$$\rho_x = \rho_x - \gamma(\sum_{t_l \in T_N(u_x)} h'(\rho_x^T \rho_l)\rho_l + \lambda\rho_x), \tag{18}$$

where γ is the learning rate, $h' = -\frac{|T_N(u_x)|\Delta - 1}{|T_N(u_x)| - 1} H(1 - \varepsilon_{t_l}^a + \bar{\varepsilon}_{u_x}^a)$, $\Delta = |\varepsilon_{t_l}^a - \bar{\varepsilon}_{u_x}^a|$, and $H(z) = 1$ if $z > 0$ and 0 otherwise, denoting the *Heaviside* function [1]. When the training process is completed, we can calculate the predicted rating $\hat{r}_{xl}$ for each unknown item t_l to user u_x, then recommend the top ranked N items to u_x with Eq. 12. Here the proposed *PrepSVD-I* algorithm takes both the personal preference patterns and the global preference patterns into consideration.

4 Experiment and Analysis

The datasets we experimented with were the popular *MovieLens* dataset and *Netflix* dataset. *MovieLens* includes around 1 million ratings collected from 6,040 users on 3,900 movies. Following literature [13], the *Netflix* dataset is a subset extracted from the Netflix Prize dataset, in which each user rated at least 20 movies, and each movie was rated by 20 − 250 users. For each dataset, we split it into two subsets, the *training set* and the *test set*. Following the work of [3, 4, 11], we reasonably assume that 5-star rated items are relevant to the active user, and adopt a similar strategy to conduct experiments. Specifically, we randomly select 2% of ratings and use all 5-star selected ratings to form the *test set*, and make sure that at least one 5-star rating exists for each individual user. The remaining ratings in the data set form the *training set*. After training the model on the *training set*, we randomly select 1000 additional items that are not rated by the active user, then predict ratings on the test item and additional 1000 selected items. These items are then ranked and the top ranked N items are selected as Top-N recommendations for the active user. This testing strategy is common for Top-N recommendations research and has been adopted by [3,4,11]. To examine the algorithm performance thoroughly, we set up a series of configurations with different data sparsity levels. Specifically, on *MovieLens* data set, we keep the *test set* the same, but vary the percentage of observed ratings for each user in the *training set*, from 10% to 100% with a 10% step. These configurations are called as *Given*10% to *Given*100% accordingly. Moreover, as recommending popular items is trivial [4], we will focus on recommending *long tail* items and *all items* that include both popular and unpopular items.

4.1 Comparison and Evaluation

We examine the performance of the proposed *PrepSVD-I* algorithm by comparing it with 7 other Top-N recommendation algorithms, including *PureSVD* [4], *SLIM* [13], BPTF [14], *itemKNN* [5], *NNcosNgbr* [4], *Top Popular* (*TopPop*) [3, 4] and *Movie Average* (*MovieAvg*) [11]. Please note BPTF considers the *time* information [14]. In *PureSVD*, the number of factors is set to 50. In *SLIM*, we set $\beta = 0.1$ and $\lambda = 0.1$. The number of the nearest neighbors in *NNcosNgbr* is set to 200, and the number of neighbors in *itemKNN* is set to 20. For *BPTF*, we set *lrate* $= 0.001$, $D = 200$ and the number of samples to 50. For our method, to train the *Preference Pattern Subspace*, we set $k = 50$, $\beta = -1$, and set the max iteration of training to 50, the error threshold to 10^{-6}. For *PrepSVD-I*, $\gamma = 0.0001$, $\lambda = 0.03$, and the factors for both users and items are set to 50.

The quality of Top-N recommendations is measured by the *recall* (or Hit Rate), the *precision* and the *fall-out* [4,5,10]. For the active user, if the Top-N recommendation list contains the test item, we call this a *hit*. Therefore, *recall*, *precision* and *fall-out* are defined as follows:

$$recall = \frac{\#hits}{|X|}, \quad precision = \frac{\#hits}{N \cdot |X|}, \quad fall\text{-}out = \frac{|X| \cdot N - \#hits}{|irrelevant|},$$

where X is the *test set* and $|irrelevant|$ is the number of all non-relevant items. A higher *recall* or *precision* value indicates better Top-N recommendations, while a lower *fall-out* value means better recommendations.

4.2 Performance on Different Datasets

To fully examine the performance of the proposed model, we conduct experiments on two well-known data sets, *MovieLens* and *Netflix*. Table 2 shows the results on these datasets when $N = 20$. It is observed that *PrepSVD-I* outperforms all the compared algorithms on both data sets for both *long tail* and *all items* recommendations in all the measurement metrics. Specifically, on *MovieLens* for *long tail* item recommendations, when measuring in *recall*, *PrepSVD-I* obtains a recall at 0.5389, which outperforms the best result 0.4987 (from *PureSVD*) by 8.06%; when measuring in *precision*, *PrepSVD-I* achieves a precision at 0.0269 that also outperforms all the other compared algorithms; for *all items* recommendations, *PrepSVD-I* also achieves better performance in *recall* and *precision*. On *Netflix*, when measuring in *recall*, *PrepSVD-I* achieves a better recall at 0.6361 and 0.7526 for *long tail* and *all items* recommendations, respectively. When measuring in *fall-out*, it seems that, *PrepSVD-I*, *PureSVD* and *SLIM* show similar performance. The main reason behind this is that, according to the definition of *fall-out*, when the number of irrelevant items is large, the *fall-out* value tends to be small, and this diminishes the difference between the performance of compared algorithms. Nevertheless, *PrepSVD-I* still achieves comparable performance to the compared algorithms. This indicates that the proposed *Preference Pattern* model can benefit Top-N recommendations for both *long tail* and *all items* recommendations. This is mainly because

Table 2. Performance on *MovieLens* and *Netflix* when $N = 20$

		MovieLens			*Netflix*		
items	Algorithm	*recall*	*precision*	*fall-out*	*recall*	*precision*	*fall-out*
long tail	PrepSVD-I	**0.5389**	**0.0269**	**0.0195**	**0.6361**	**0.0318**	**0.0194**
	PureSVD	0.4987	0.0249	**0.0195**	0.6165	0.0308	**0.0194**
	SLIM	0.4527	0.0226	**0.0195**	0.5987	0.0299	**0.0194**
	NNcosNgbr	0.4518	0.0226	**0.0195**	0.4988	0.0249	0.0195
	itemKNN	0.3273	0.0164	0.0197	0.4393	0.0220	0.0196
	BPTF	0.1992	0.0100	0.0198	0.2960	0.0148	0.0197
	TopPop	0.0096	0.0005	0.0200	0.2041	0.0102	0.0198
	MovieAvg	0.0818	0.0041	0.0199	0.0312	0.0016	0.0200
		MovieLens			*Netflix*		
items	Algorithm	*recall*	*precision*	*fall-out*	*recall*	*precision*	*fall-out*
all items	PrepSVD-I	**0.6928**	**0.0346**	**0.0193**	**0.7526**	**0.0376**	**0.0192**
	PureSVD	0.6709	0.0335	**0.0193**	0.7193	0.0360	0.0193
	SLIM	0.6794	0.0340	**0.0193**	0.7295	0.0365	0.0193
	NNcosNgbr	0.4962	0.0248	0.0195	0.5258	0.0263	0.0195
	itemKNN	0.5625	0.0281	0.0194	0.6436	0.0322	0.0194
	BPTF	0.3389	0.0169	0.0197	0.3837	0.0192	0.0196
	TopPop	0.3857	0.0193	0.0196	0.5000	0.0250	0.0195
	MovieAvg	0.1734	0.0087	0.0198	0.0589	0.0029	0.0199

the *Preference Pattern* is based on users' personal preference styles, and also because it takes the global preference patterns into consideration. Therefore, it can lead to better recommendations regardless of whether the target item is popular or not.

4.3 Performance on *long tail* Item Recommendations

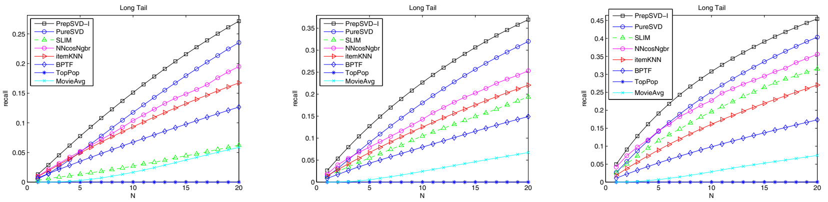
As recommending popular items is trivial [4], here we examine the proposed *PrepSVD-I* by comparing it with 7 other state-of-the-art recommendation algorithms under various data sparsity levels on *long tail* recommendations.

Table 3 shows the *recall* performance of the examined algorithms when N equals 10 and 20. It is clear that the proposed *PrepSVD-I* algorithm significantly outperforms all of the compared algorithms under all sparsity conditions except on *Given*10% when $N = 10$ with *BPTF* obtaining a slightly higher *recall*. However, *BPTF* performs badly on all other sparsity levels. For example, on *Given*90% when $N = 20$, *BPTF* only obtains a *recall* at 0.1824, which is much worse than all the other personalized algorithms, e.g. *PrepSVD-I*, *itemKNN*, *NNcosNgbr* and *SLIM*. Moreover, *PrepSVD-I* performs steadily through various sparsity levels, and always achieves better performance. Specifically, on *Given*50%, when $N = 20$, *PrepSVD-I* achieves a *recall* of 0.3147, which outperforms the best compared result of 0.2751 (from *PureSVD*) by 14.39%. This is, as expected, because the proposed *Preference Pattern Subspace* is capable of capturing user preference styles and the corresponding dynamics, and is not affected by whether the item is popular or not. Moreover, it is also observed that the sparser the training data set, the larger the improvements. For example, when $N = 10$, the improvement on *Given*100% is 12.66%, and increases to 29.76% on *Given*40% data set. The reason behind this is that when the training set is sparser, the *long tail* effect indicates that available ratings on *long tail*

Table 3. *recall* when $N = 10$ and $N = 20$ on *long tail* Items

	*Given*10%		*Given*20%		*Given*30%		*Given*40%		*Given*50%	
Algorithm	$N = 10$	$N = 20$	$N = 10$	$N = 20$	$N = 10$	$N = 20$	$N = 10$	$N = 20$	$N = 10$	$N = 20$
PrepSVD-I	0.0388	**0.0903**	**0.0677**	**0.1490**	**0.1092**	**0.2112**	**0.1526**	**0.2728**	**0.1851**	**0.3147**
PureSVD	0.0299	0.0818	0.0525	0.1311	0.0848	0.1815	0.1176	0.2353	0.1474	0.2751
SLIM	0.0000	0.0000	0.0000	0.0004	0.0058	0.0167	0.0270	0.0623	0.0587	0.1188
NNcosNgbr	0.0230	0.0598	0.0371	0.0785	0.0552	0.1141	0.1037	0.1949	0.1156	0.2257
itemKNN	0.0267	0.0501	0.0517	0.0901	0.0726	0.1321	0.0935	0.1673	0.1112	0.1970
BPTF	**0.0491**	0.0873	0.0484	0.0910	0.0549	0.1064	0.0673	0.1265	0.0738	0.1359
TopPop	0.0000	0.0000	0.0000	0.0001	0.0000	0.0001	0.0000	0.0001	0.0000	0.0001
MovieAvg	0.0036	0.0232	0.0073	0.0367	0.0095	0.0431	0.0165	0.0586	0.0169	0.0560

	*Given*60%		*Given*70%		*Given*80%		*Given*90%		*Given*100%	
Algorithm	$N = 10$	$N = 20$	$N = 10$	$N = 20$	$N = 10$	$N = 20$	$N = 10$	$N = 20$	$N = 10$	$N = 20$
PrepSVD-I	**0.2275**	**0.3701**	**0.2614**	**0.4088**	**0.3100**	**0.4559**	**0.3479**	**0.4957**	**0.3995**	**0.5389**
PureSVD	0.1801	0.3198	0.2091	0.3568	0.2515	0.4035	0.2988	0.4502	0.3546	0.4987
SLIM	0.1048	0.1938	0.1532	0.2679	0.1959	0.3151	0.2451	0.3741	0.3229	0.4527
NNcosNgbr	0.1429	0.2531	0.1872	0.3034	0.2272	0.3562	0.2666	0.3889	0.3293	0.4518
itemKNN	0.1256	0.2204	0.1414	0.2417	0.1616	0.2702	0.1816	0.2954	0.2086	0.3273
BPTF	0.0796	0.1490	0.0873	0.1600	0.0984	0.1733	0.1032	0.1824	0.1196	0.1992
TopPop	0.0000	0.0002	0.0000	0.0002	0.0001	0.0003	0.0001	0.0029	0.0013	0.0096
MovieAvg	0.0245	0.0673	0.0281	0.0716	0.0289	0.0745	0.0303	0.0755	0.0318	0.0818



(a) *Given*40% (b) *Given*60% (c) *Given*80%

Fig. 2. *recall* at *N* on *long tail*

items will be much more limited. Consequently, the user rating patterns will be extremely incomplete, and solely modelling them does not lead to good recommendations. In this case, the *preference dynamic effect* becomes more valuable to predict users' preferences on items. Therefore, the proposed *Preference Pattern Subspace* will show high effectiveness for recommendation purposes.

To thoroughly examine the performance of *PrepSVD-I*, we vary the *N* value from 1 to 20, and report the results on *Given*40%, *Given*60% and *Given*80% as shown in Fig. 2. We observe that *PrepSVD-I* outperforms all compared algorithms at all *N* values on all the data sets. This indicates that *PrepSVD-I* is effective in recommending the desired items at the top of the recommendation list. The experiment results show that, when recommending *long tail* items, *PrepSVD-I* is robust to the data sparsity issue, and can significantly outperforms state-of-the-art Top-*N* recommendation algorithms in terms of accuracy.

4.4 Performance on *all items* Recommendations

we also conduct experiments on recommending *all items*, including both popular and *long tail* items. Table 4 shows the *recall* performance of all compared algo-

Table 4. *recall* when $N = 10$ and $N = 20$ on *All Items*

	*Given*10%		*Given*20%		*Given*30%		*Given*40%		*Given*50%	
Algorithm	$N = 10$	$N = 20$	$N = 10$	$N = 20$	$N = 10$	$N = 20$	$N = 10$	$N = 20$	$N = 10$	$N = 20$
PrepSVD-I	0.1421	0.2300	0.1926	0.3007	0.2363	**0.3585**	**0.2826**	**0.4165**	**0.3267**	**0.4637**
PureSVD	0.1310	0.2102	0.1716	0.2701	0.2115	0.3205	0.2471	0.3698	0.2840	0.4131
SLIM	0.2032	0.3059	0.2221	**0.3238**	**0.2392**	0.3472	0.2811	0.3960	0.3211	0.4462
NNcosNgbr	0.2027	0.2990	0.2052	0.3095	0.2096	0.3248	0.2532	0.3709	0.2617	0.3818
itemKNN	0.0290	0.0548	0.1383	0.2032	0.2252	0.3223	0.2748	0.3855	0.3042	0.4265
BPTF	0.1014	0.1756	0.1285	0.2065	0.1475	0.2332	0.1653	0.2549	0.1803	0.2736
TopPop	**0.2088**	**0.3080**	**0.2247**	0.3222	0.2256	0.3251	0.2332	0.3320	0.2413	0.3410
MovieAvg	0.0036	0.0443	0.0152	0.0812	0.0235	0.1003	0.0353	0.1198	0.0451	0.1253

	*Given*60%		*Given*70%		*Given*80%		*Given*90%		*Given*100%	
Algorithm	$N = 10$	$N = 20$	$N = 10$	$N = 20$	$N = 10$	$N = 20$	$N = 10$	$N = 20$	$N = 10$	$N = 20$
PrepSVD-I	**0.3821**	**0.5242**	**0.4322**	**0.5746**	**0.4819**	**0.6213**	**0.5229**	**0.6588**	**0.5678**	**0.6928**
PureSVD	0.3267	0.4592	0.3721	0.5090	0.4275	0.5640	0.4939	0.6226	0.5534	0.6709
SLIM	0.3785	0.5090	0.4277	0.5640	0.4588	0.5897	0.5017	0.6273	0.5668	0.6794
NNcosNgbr	0.2796	0.3905	0.3112	0.4156	0.3374	0.4426	0.3558	0.4553	0.4036	0.4962
itemKNN	0.3360	0.4594	0.3595	0.4832	0.3836	0.5094	0.4128	0.5360	0.4423	0.5625
BPTF	0.1930	0.2884	0.2026	0.3003	0.2127	0.3118	0.2196	0.3221	0.2381	0.3389
TopPop	0.2484	0.3480	0.2559	0.3564	0.2661	0.3658	0.2717	0.3724	0.2860	0.3857
MovieAvg	0.0585	0.1456	0.0710	0.1631	0.0691	0.1629	0.0691	0.1613	0.0742	0.1734

rithms across various sparsity levels on *all item* recommendations. We can observe that *PrepSVD-I* achieves the best *recall* on all data sets, except *Given*10%, *Given*20% and *Given*30% (on $N = 10$). It is unexpected that *TopPop* achieves the highest *recall* values on *Given*10% and *Given*20% (on $N = 10$). This is mainly because when the training set is very sparse, the majority of available ratings are given for popular items. Therefore, the popularity of items will bias the performance of algorithms. This is consistent with the findings in [4]. *SLIM* achieves the best *recall* on *Given*20% (on $N = 20$) and *Given*30% (on $N = 10$). However, both *TopPop* and *SLIM* become almost useless in recommending *long tail* items on the same data sets, as shown in Table 3, which confirms the popularity-related bias.

On the other hand, it is also clear that *PrepSVD-I* outperforms all compared algorithms on 7 out of 10 data sets, including *TopPop* and *SLIM*. Specifically, when $N = 20$ on *Given*40% data set, *PrepSVD-I* achieves a *recall* at 0.4165 which outperforms the best compared results of 0.3960 (from *SLIM*); on *Given*100% data set, *PrepSVD-I* obtains a *recall* at 0.6928 that outperforms the best compared results of 0.6794 (from *SLIM*). This indicates that the *Preference Pattern Subspace* can also benefit *all items* recommendations, including popular items.

In terms of *accuracy* on recommending both *all items* and *long tail* items, it is clear that *PrepSVD-I* performs better than all compared state-of-the-art Top-N recommendation algorithms. Although *TopPop* and *SLIM* achieve a slightly better *recall* values on *Given*10%, *Given*20% and *Given*30% when recommending *all items*, they perform badly on the same data set when recommending *long tail* items, as shown in Table 3. This will limit their recommendation abilities. However, *PrepSVD-I* can perform very well on both *all items* and *long tail* item recommendations. This means *PrepSVD-I* possesses a better recommendation ability than other rating-pattern-based techniques.

5 Conclusion

This paper introduces a novel *preference dynamic effect* in the context of recommender systems, and proposes a *Preference Pattern Subspace* approach to model this effect. The basic idea is to build a low-rank subspace to capture the *personal preference patterns* together with their temporal dynamics by refining the *global* and the *personal* preference patterns iteratively with an EM-like algorithm. Experiment results show that the proposed *PrepSVD-I* significantly outperforms the other state-of-the-art Top-N recommendation techniques in terms of *accuracy*, and is robust to challenge the data sparsity issue.

References

1. Abramowitz, M., Stegun, I.A.: Handbook of Mathematical Functions: with Formulas, Graphs, and Mathematical Tables. Dover, New York (1972)
2. Anderson, C.: The Long Tail: Why the Future of Business Is Selling Less of More, vol. 33. Hyperion (2006)
3. Cremonesi, P., Garzotto, F., Negro, S., Papadopoulos, A.: Comparative Evaluation of Recommender System Quality. In: CHI Extended Abstracts, pp. 1927–1932 (2011)
4. Cremonesi, P., Koren, Y., Turrin, R.: Performance of Recommender Algorithms on Top-N Recommendation Tasks. In: Recsys, pp. 39–46. ACM (2010)
5. Deshpande, M., Karypis, G.: Item-based top- N recommendation algorithms. ACM TOIS 22(1), 143–177 (2004)
6. Geng, X., Zhou, Z.-H., Smith-Miles, K.: Automatic age estimation based on facial aging patterns. IEEE TPAMI 29(12), 2234–2240 (2007)
7. Golub, G.H., Van Loan, C.F.: Introduction to Matrix, 3rd edn., vol. 1. The Johns Hopkins University Press, Baltimore (1996)
8. Herbrich, R., Graepel, T., Obermayer, K.: Support Vector Learning for Ordinal Regression. In: ICANN 1999, vol. 470, pp. 97–102 (1999)
9. Herlocker, J.L., Konstan, J.A., Terveen, L.G., Riedl, J.T.: Evaluating Collaborative Filtering Recommender Systems. ACM Transactions on Information Systems 22(1), 5–53 (2004)
10. Karypis, G.: Evaluation of item-based top-n recommendation algorithms. In: CIKM 2001, pp. 247–254. ACM (2001)
11. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: SIGKDD 2008, pp. 426–434. ACM (2008)
12. Koren, Y.: Collaborative filtering with temporal dynamics. In: SIGKDD 2009, pp. 447–456. ACM (2009)
13. Ning, X., Karypis, G.: SLIM: Sparse Linear Methods for Top-N Recommender Systems. In: ICDM 2011, pp. 497 – 506 (2011)
14. Xiong, L., Chen, X., Huang, T.K., Schneider, J., Carbonell, J.G.: Temporal collaborative filtering with bayesian probabilistic tensor factorization. In: Proceedings of SIAM Data Mining (2010)

Semantic Title Evaluation and Recommendation Based on Topic Models

Huidong Jin[1,2], Lijiu Zhang[2], and Lan Du[3]

[1] CSIRO Mathematics, Informatics and Statistics, Acton ACT 2601, Australia
[2] Research School of Computer Science, CECS, the Australian National University, Acton ACT 2601, Australia
[3] Department of Computing, Macquarie University, NSW 2109, Australia
Warren.Jin@csiro.au, Lijiu.Zhang@anu.edu.au, Lan.Du@mq.edu.au

Abstract. To digest tremendous documents efficiently, people often resort to their titles, which normally provide a concise and semantic representation of main text. Some titles however are misleading due to lexical ambiguity or eye-catching intention. The requirement of reference summaries hampers using traditional lexical summarisation evaluation techniques for title evaluation. In this paper we develop semantic title evaluation techniques by comparing a title with other sentences in terms of topic-based similarity with regard to the whole document. We further give a statistical hypothesis test to check whether a title is favourable without any reference summary. As a byproduct, the top similar sentence can be recommended as a candidate for title. Experiments on patents, scientific papers and DUC'04 benchmarks show our Semantic Title Evaluation and Recommendation technique based on a recent Segmented Topic Model (STERSTM), performs substantially better than that based on the canonical model Latent Dirichlet Allocation (STERLDA). It can also recommend titles with quality comparable with the winners of DUC'04 in terms of summarising documents into very short summaries.

Keywords: Topic models, semantic, evaluation, hypothesis test.

1 Introduction

Text mining techniques have been sought after in order to make informed decisions efficiently based on tremendous textural information [1]. For a lot of documents, a good title, which gives a concise and semantic representation of contents in main text, often provides a shortcut for readers to digest documents. However, due to various reasons like lexical ambiguity (say, polysemy), eye-catching intention or being prepared by an inexperienced writer, a lot of documents come with titles whose semantics are away from their main texts. For example, "Learning to fly" can be a title of a book for flight training or an autobiography for Victoria Beckham. These motivate us to develop automatic techniques to evaluate to what degree a title captures the main contents of its associated document. As a byproduct, our title evaluation techniques can be used to recommend a title-worthy sentence from which a quality title could be generated.

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 402–413, 2013.

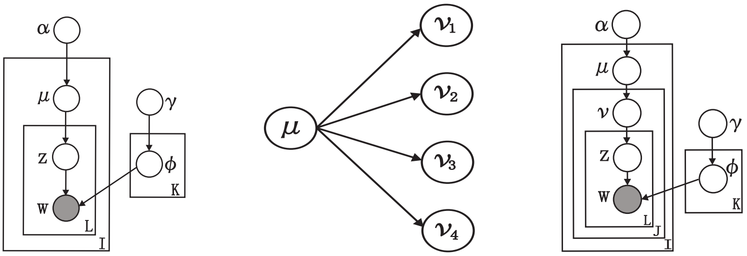
Two issues hamper adjusting traditional document summarisation evaluation techniques including ROUGE for title evaluation. To evaluate a title, these techniques require a, normally human generated, reference summary [12]. In addition, the evaluation is mainly based on whether words in a title appear or not in the reference summary. It is often not enough, especially considering polysemy and synonymy. To overcome these two issues, we will propose to compute semantic similarity in a space spanned by latent topics learnt by topic models, and then to use a statistical hypothesis test to check or classify whether a title is poor.

Our title evaluation and recommendation techniques are relevant to extractive text summarisation. Extractive summarisation only chooses information (words/sentences) from documents to compose concise representation for them [13]. There are mainly two types of approaches [13]. One type of approaches first derive an intermediate representation like topic words, TF*IDF, Latent Semantic Analysis (LSA), and Bayesian topic models, for documents that captures the contents in main text. Sentences are then scored for importance. Because of their modelling generalisability to unseen documents and short textual units [2, 8, 9], we choose topic models to learn a topic representation of a document and its sentences/title. In this way, not only can one handle polysemy and synonymy [5], but also make a title, sentences, a document directly comparable. We will develop and compare two semantic title evaluation techniques based on either a recent Segmented Topic Model (STM) [9] or the standard Latent Dirichlet Allocation (LDA) [2].

In the second type of summarisation approaches, indicator representation approaches, the text is represented by a diverse set of possible importance indicators that do not aim at discovering topicality [13]. These indicators are combined, using graph-based ranking methods, say PageRank [10] or machine learning techniques, say classification [6], to score the importance of each sentence. Different from Bayesian topic models, these approaches normally require extra information [10, 13, 14, 15], such as costly training data [6, 14], WordNet [6], Wikipedia [14, 15], or search query logs [14]. Our experiments show our title recommendation techniques can give very short summaries with quality comparable with the winners of DUC'04, including [6, 10].

The basic procedure of our Semantic Title Evaluation and Recommendation (STER) techniques is as follows. (1) We use topic models to generate latent topics, each of which is a probability distribution over words, from documents as well as sentences/titles in them. Based on two topic models STM and LDA, we have STERSTM and STERLDA techniques respectively. (2) Each document/sentence/title is represented as a mixture of latent topics. (3) Semantic similarity values are calculated between documents and its sentences/title based on their topic distributions. (4) Being compared with other sentences in a document, a title with a statistically significantly low similarity is regarded as unfavourable. The top similar sentence is recommended as a title worthy sentence.

In Section 2, we first brief Bayesian topic modelling techniques. Section 3 presents STERSTM and STERLDA. Experimental results of STERSTM, and



(a) LDA (b) Topic hierarchy (c) STM

Fig. 1. LDA [2], hierarchical structure within a document, and STM [9]

comparison with STERLDA and the methods participated in DUC'04 are reported in Section 4, followed by concluding comments in Section 5.

2 Background of Bayesian Topic Modelling Techniques

In order to estimate semantic coverage of a title, we need to compute semantic similarity between a title and its whole document in the same latent topic space. Because of better generalisation capability to unseen documents and short textual units than LSA and its variants [9], we learn this topic space using Bayesian topic models that specify a probabilistic process by which text documents can be generated.

The canonical topic model, LDA [2], is a latent variable model of documents, where a document is regarded as a mixture of K latent topics, each of which is a probability distribution over words. Following [9], documents are indexed by i $(i = 1, \cdots, I)$, and words $\boldsymbol{w}$ are observed data, each is indexed by l $((l = 1, \cdots, L))$. The latent variables are $\boldsymbol{\mu}_i$ (*the topic distribution or topic proportion* for a document) and $\boldsymbol{z}$ (the *topic assignments* for observed words), and the model parameter of $\boldsymbol{\phi}_k$'s (*per-topic word distributions*). This generative model, as illustrated in Fig. 1(a), is as follows:

$$\begin{array}{llll} \boldsymbol{\phi}_k \sim \text{Dirichlet}_W(\boldsymbol{\gamma}) & \forall\, k; & \boldsymbol{\mu}_i \sim \text{Dirichlet}_K(\boldsymbol{\alpha}) & \forall\, i; \\ z_{i,l} \sim \text{Multinomial}_K(\boldsymbol{\mu}_i) & \forall\, i,l; & w_{i,l} \sim \text{Multinomial}_W\left(\boldsymbol{\phi}_{z_{i,l}}\right) & \forall\, i,l. \end{array}$$

$\text{Dirichlet}_K(\cdot)$ is a K-dimensional Dirichlet distribution, and W is the number of different words. The hyper-parameters $\boldsymbol{\gamma}$ and $\boldsymbol{\alpha}$ are Dirichlet priors for word and topic distributions respectively.

Since LDA was introduced, topic models have been widely extended in the text mining community (see [5, 8] and references therein). Topic models have been successfully used in document summarisation [13], opinion mining [16], sequential topic evolution [8, 7], etc. Via leveraging hierarchical structure within a document, such as a document consisting of sentences (Fig. 1(b)), STM can generate much more accurate topics than LDA and its variants [9]. In addition, it models a document and its sentences in the same topic space, which is required by our semantic title evaluation. In fact, in STM, topic proportions of sentences distribute around the topic proportion of the whole document, as it is described by a Poisson-Dirichlet Process (PDP). Conditioned on the model parameters $\boldsymbol{\alpha}, \boldsymbol{\gamma}, \boldsymbol{\Phi}$ and PDP parameters a, b (called *discount* and *strength* respectively, $0 \leq a < 1$, $b > -a$), STM that we used in this paper assumes the following generative process (graphical view see Fig. 1(c)):

1. For each document documents D_i $(i \in \{1, \cdots, I\})$, draw a document topic proportion or distribution $\boldsymbol{\mu}_i \sim \text{Dirichlet}_K(\boldsymbol{\alpha})$
2. For each sentence $S_{i,j}$ $(j \in \{1, \cdots, J_i\})$
   (a) Draw sentence topic proportion $\boldsymbol{\nu}_{i,j}$ around $\boldsymbol{\mu}_i$, i.e., $\boldsymbol{\nu}_{i,j} \sim \text{PDP}(a, b, \boldsymbol{\mu}_i)$
   (b) For each word $w_{i,j,l}$, where $l \in \{1, \ldots, L_{i,j}\}$
      i. Select a topic $z_{i,j,l} \sim \text{Multinomial}_K(\boldsymbol{\nu}_{i,j})$
      ii. Generate a word $w_{i,j,l} \sim \text{Multinomial}_W(\boldsymbol{\phi}_{z_{i,j,l}})$

3 Semantic Title Evaluation and Recommendation

The procedure of our semantic title evaluation methods is given as follows. It first represents a document, its sentences and title using the same set of latent topics learned by a topic model. The semantic similarity between a title/sentence and the document is computed based on their topic proportion (i.e., distributions) vectors. Via comparing the title's similarity value with those of sentences in main text, we use a hypothesis test to compute p-Value to check how semantically good a title is. As a byproduct, p-Values for those sentences can also be used to recommend a top one for a title candidate, from which a title can be generated quickly.

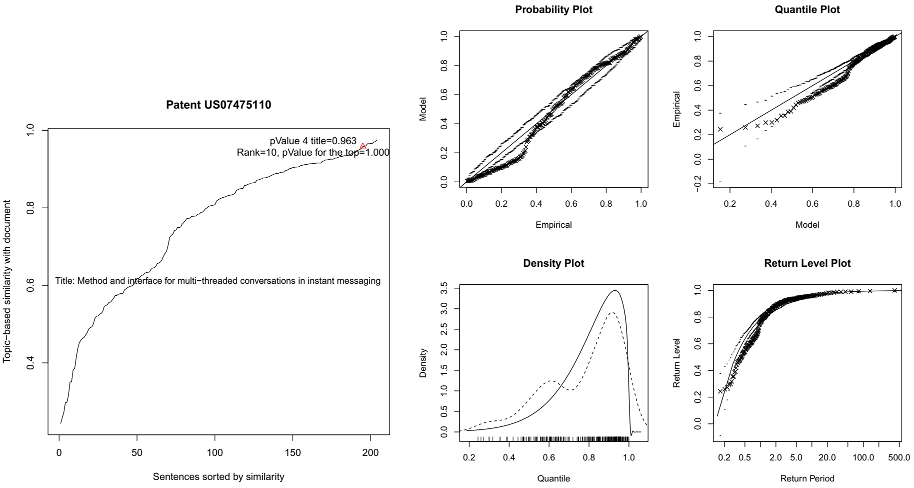
Algorithm 1 outlines our Semantic Title Evaluation and Recommendation method based on STM (STERSTM). In the preprocessing step (Step 1), a document is first split into its constituent sentences by a Perl programme (Lingua:en:sentence package) [3] based on a regular expression and a list of abbreviations. Hereinafter, a title is treated as a separate sentence for the sake of

Algorithm 1 STERSTM

Input: One corpus $\mathcal{D}$ with one or multiple documents, and the number of topics K.

```
1. Document preprocessing: Split documents $D_i$ ($\in \mathcal{D}$) into sentences $S_{i,j}$, and
   then split sentences $S_{i,j}$ into words $w_{i,j,l}$; remove most and least frequent words
2. Build a STM, and estimate its parameters using the collapsed Gibbs sampler in [9]
3. Infer topic proportions $\boldsymbol{\mu}_i$ for documents and $\boldsymbol{\nu}_{i,j}$ for sentences based on STM
4. FOR each document $D_i$ in $\mathcal{D}$ DO
5.     Compute similarity $s_{i,j}$ between $D_i$ and sentence $S_{i,j}$ using $\boldsymbol{\mu}_i$ and $\boldsymbol{\nu}_{i,j}$
6.     Fit a GEV distribution $G(s; \boldsymbol{\theta}_i)$ over $s_{i,j}$ via maximising likelihood
7.     Compute p-Value $G(s_{i,j}; \boldsymbol{\theta}_i)$ for each sentence $S_{i,j}$        /*Hypothesis test*/
8.     Categorise and rank sentences based on their p-Values
```

Output: p-Value for titles, and the sentence with largest p-Value for each document



(a) Similarity, rank and p-Value (b) Diagnosis plots for a GEV distribution

Fig. 2. Semantic title evaluation result of STERSTM for Patent US07475110

simplicity. Sentences are then split into words. After that, all stop-words, extremely frequent (*e.g.*, top 30 in our experiments) words, and least frequent (*e.g.*, less than 5 times) words are removed. We do not stem words in order to keep post-processed sentences with an acceptable length.

After having the word list $w_{i,j,l}$ for each sentence $S_{i,j}$ in document D_i, we run the efficient collapsed Gibbs sampling algorithm [9] to estimate parameters in STM (Step 2). In Step 3, with a sufficient number of samples being drawn from the converged Markov chain for STM, topic distributions of documents and sentences can be estimated by a fixed point estimation with inverting the generative process in Section 2.

Step 5 calculates the semantic similarity between a document and its sentences using their topic proportion vectors $\boldsymbol{\mu}_i$ and $\boldsymbol{\nu}_{i,j}$. The widely used cosine similarity measures similarity between two vectors by calculating the cosine of the angle between them:

$$s_{i,j} = cosine\_similarity\,(\boldsymbol{\mu}_i, \boldsymbol{\nu}_{i,j}) = \frac{\sum_{k=1}^{K}(\mu_{i,k} \times \nu_{i,j,k})}{\sqrt{\sum_{k=1}^{K}\mu_{i,k}^2} \times \sqrt{\sum_{k=1}^{K}\nu_{i,j,k}^2}} \qquad (1)$$

Because a topic proportion vector also indicates a multinomial distribution, we can also use the Hellinger distance or Kullback-Leibler divergence, which quantify the similarity between two probability distributions [13, 7]. As our preliminary title evaluation experimental results show there is little difference among these similarity metrics, we will only present results for the cosine similarity. Examples of cosine similarities of sentences within two patents and one conference paper could be found in Figs. 2(a) and 3.

Before introducing Steps 6-8, we show that it is not easy to specify a constant threshold for semantic similarities for determining a favourable title through examples in Fig. 3. Similarities for different documents have different value ranges. Comparing with other sentences in a document, 0.95 is reasonably good for the patent's title while just average for the paper's title in Fig. 3(a). Similarly, because the numbers of sentences in a document can range from a few dozens to several thousands, it is difficult to specify a threshold for rank based on similarity or relative rank (e.g., rank of title divided by the total number of sentences within a document). Rank 34th is possibly favourable for a title within a very long document, but doubtable for a short one in Fig. 3(a). A lot of sentences arguably have high semantic similarity with a document. A small change on the title's similarity value may lead to a big change on its rank as well as its relative rank.

We give a statistical mechanism to specify document-specific 'thresholds', as Generalised Extreme Value (GEV) distribution is able to fit well these similarity values in Figs. 2(a) and 3. In the extreme value theorem, the GEV distribution is a limited distribution of properly normalized minima of a sequence of independent and identically distributed random variables [4]. It is a family of continuous probability distributions, and it is a general distribution family, including Weibull and Gumbel distribution families. The GEV distribution we used has a cumulative distribution function:

$$G(x;\boldsymbol{\theta}) = exp\left\{-\left[1-\theta_3\left(\frac{x+\theta_1}{\theta_2}\right)\right]^{-1/\theta_3}\right\} \tag{2}$$

for $1-\theta_3(x+\theta_1)/\theta_2 > 0$, where $\theta_1 \in R$ is the location parameter, $\theta_2 > 0$ the scale parameter and $\theta_3 \in R$ the shape parameter.

In Step 6, parameter $\boldsymbol{\theta}$ of the GEV distribution are estimated via maximising likelihood of all the similarity values within the same document. The parameter estimation can be visually validated via such as probability plot, quantile (Q-Q) plot, density plot or return level plot [4]. Diagnosis plots for a fitted GEV distribution for similarity values in Fig. 2(a) are exemplified in Fig. 2(b).

Step 7 in Algorithm 1 computes p-Values of all the sentences within a document. The p-Values for a sentence/title here can be used for fulfilling a statistical hypothesis test. The p-Value is the probability of the similarity observation under the null hypothesis (H0) which hypothesises that its similarity value based on topics is not extreme in comparison with counterpart sentences. We can reject the null hypothesis if and only if the p-Value is less than the significance level threshold. We will use a conservative threshold, say, 10% in this work. Therefore, if the p-Value for a title is less than the threshold, we reject the null hypothesis and draw a statistically sound conclusion that the title is not semantically good enough (in comparison with other sentences in the associated document). In other words, we can categorise such a title as 'Unfavourable'. Our experiment results, some presented in Section 4.2, show that the sentences with large semantic similarity values can summarise the whole document excellently. As a matter of factor, STERSTM is very close to the runner-up of Task 1 (summarising an English document into a very short summary) in the Document Understanding

Conference (DUC) in 2004 [6]. Thus, we may categorise a title as 'Excellent' if its p-Value is larger than 90%. Other titles, with moderate p-Value ranging from 0.10 to 0.90, will be categorised as 'Average.' Step 8 conducts this categorisation. It also sorts sentences of a document based on their p-Values (equivalently, their semantic similarities). Finally, the p-Values of titles generated by STERSTM can evaluate titles in a statistically sound way without a reference summary. The top sentences with highest p-Value from STERSTM can be recommended as the title-worthy sentence or a title candidate for the document.

Steps of Algorithm 1 could be independently replaced with other techniques to develop new methods. For example, Step 5 can be replaced with other sentence scoring techniques [10, 13]. Steps 2 and 3 can be replaced with a modelling technique as soon as it can represent documents and sentences in the same semantic space. When steps 2 and 3 are replaced with LDA, we call the new method STERLDA. LDA does not consider document structure as STM does. In order to derive topic distributions for both documents and their sentences, we need to run LDA twice, one on the document level, another on the sentence level. However, these two LDAs will come up with two different sets of latent topics due to unsupervised learning. To tackle this problem, the topics generated on the document level are used and fixed in training LDA on the sentence level.

4 Experimental Setting and Results

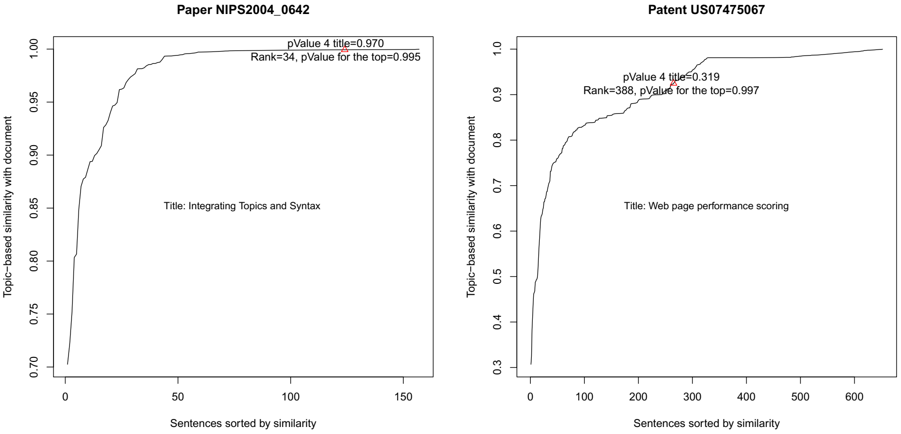
STERSTM can run on a single document, while STERLDA cannot. To facilitate a fair comparison, we ran both of them on a set of documents. We set the number of topics $K = 50$, and priors $\alpha = 0.05$ and $\gamma = 0.01$ for both STM and LDA, and $a = 0.02$ and $b = 10$ for STM in our experiments in this paper.

4.1 Semantic Title Evaluation Experiments

We used two sets of documents for title evaluation experiments. One is Patents-99, where 99 U.S. patents were randomly selected from 5000 U.S. patents [1] granted between Jan. and Mar. 2009 under the class "computing; calculating; counting" with international patent classification (IPC) code G06. After preprocessing, the numbers of post-processed sentences in these patents range from 60 to 2163. The second data set is NIPS-100, in which 100 papers were randomly selected from NIPS conference papers in 2004. These papers contain a lot of equations, which make the preprocessing step harder. The numbers of sentences range from 68 to 207.

As we discussed in Section 3, p-Value from a GEV distribution can give us more informative evaluation than ranks etc. When the similarity value of a title has a high rank, it often has a low p-Value. Though the rank and the p-Value are negatively correlated, p-Value takes into account of similarity values of other sentences within the same document, and becomes more informative. For example,

[1] All patents are from Cambia, `http://www.cambia.org/daisy/cambia/home.html`



(a) For Paper 642 in NIPS'04 (b) For Patent US07475067

Fig. 3. Semantic similarity, rank, and p-Value got by STERSTM for two documents

for NIPS paper 579, its title "*Validity estimates for loopy Belief Propagation on binary real-world networks*" has the semantic similarity of 0.921, and is ranked only 116th in comparison with the 131 sentences from the paper, and looks really unfavourable. However, p-Value of 0.417 does not provide evidence statistically significantly to claim this title is unfavourable. Another similar example could be found in Fig. 3(b).

Fig. 3 illustrates semantic similarities between sentences/title and a whole document based on topics learned by STM. For Paper 642 from NIPS'04, the title "*Integrating Topics and Syntax*" has the similarity value of 0.9993, and it is ranked 34th in comparison with 155 sentences from the paper. Its p-Value from the GEV distribution is 0.970. That means this is an excellent title. From Fig. 3(b), we can see the title "*Web page performance scoring*" has the semantic similarity of 0.9247. It is ranked as 388th in comparison with other 650 sentences from the patent. Its p-Value is 0.319, which says the title is not excellent from the viewpoint of covering the whole patent semantically. From its abstract[2], we can see it could be improved if some word related with '*tool*' or '*browser-based tool*' is appended to the title. As another evidence, the top semantically similar sentence chosen by STERSTM is "*More particularly, the invention relates to*

[2] ***Abstract*** *A browser-based tool is provided that loads a Webpage, accesses the document object model (DOM) of the page, collects information about the page structure and parses the page, determines through the use of heuristics such factors as how much text is found on the page and the like, produces statistical breakdown of the page, and calculates a score based on performance of the page. Key to the operation of the invention is the ability to observe operation of the Webpage as it actually loads in real time, scoring the page for several of various performance factors, and producing a combined score for the various factors.*

Table 1. Categorisation of titles for two sets of documents based on p-Values

Title Categorisation		Unfavourable	Average	Excellent
p-Value range		[0,0.1]	(0.1,0.9]	(0.9,1.0]
Patents-99	STERSTM	0	49	50
	STERLDA	6	57	36
NIPS-100	STERSTM	0	55	45
	STERLDA	11	66	23

a tool which analyses the content and structure of Web pages in real time and produces statistics and a performance score."

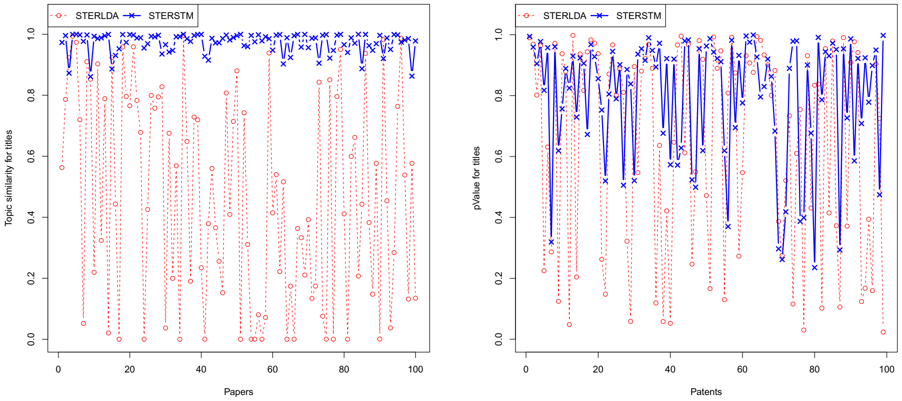
Fig. 4(a) illustrates the semantic similarity values of titles from NIPS-100. The 100 similarity values of these titles generated by STERSTM range from 0.86 to almost 1. They are normally quite high. The similarities by STERLDA range from almost 0 to 0.996 and have a broader value range. It seems that STERLDA generates less reliable evaluation than STERSTM in terms of similarity values. For this document set, according to STERSTM, 45 out of 100 papers have excellent titles, including the one in Fig. 3(a). STERSTM doesn't find any unfavourable title, which is not surprised as all the papers were prepared by experienced researchers. STERLDA surprisingly finds 11 unfavourable titles, and only 23 excellent ones as summarised in Table 1. For example, the title "*Methods Towards Invasive Human Brain Computer Interfaces*" of Paper 443 in NIPS'04 has the p-Value of 0.058 and is inappropriately regarded as unfavourable by STERLDA.

Fig. 4(b) gives the p-Values of these titles within the document set Patents-99. The p-Values of the 99 titles based on STERSTM range from 0.22 to very close to 1. STERSTM finds 45 excellent titles, and it does not find any unfavourable patent titles, as we would expect. In comparison, p-Values from STERLDA range from 0 to close to 1. It finds only 23 excellent titles and 11 unfavourable titles. Thus, STERSTM can evaluate titles more reliably than STERLDA based on the two document sets.

4.2 Semantic Title Recommendation Experimental Results

In this section, we empirically check whether our proposed techniques can recommend a title worthy sentence from the viewpoint of capturing the main idea of a document [11]. Due to limitation of space, we only report results on one set of documents, DUC-2004. DUC-2004 is the benchmarks used for Task 1 (generating a very short summary from a document) in NIST's DUC'04[3]. The corpus consists of 50 sets of documents each contains 10 same topic documents on average. The documents came from the AP newspapers and New York Times newspapers. The short summary generated is peer summary and it is automatically evacuated by one of widely used document summarisation metrics, Recall-Oriented

[3] http://www-nlpir.nist.gov/projects/duc/guidelines/2004.html



(a) Similarity values for NIPS-100

(b) p-Values for Patents-99

Fig. 4. Topic similarities values or p-Values from STERSTM and STERLDA

Understudy for Gisting Evaluation (ROUGE) [12]. ROUGE essentially calculates n-gram overlaps between given summaries and previously-written human summaries. A high level of overlap should indicate a high level of shared concepts between the two summaries. There are four reference summary (or model summary) per document in DUC-2004. ROUGE can evaluate a short given summary by comparing it with up to four reference summaries.

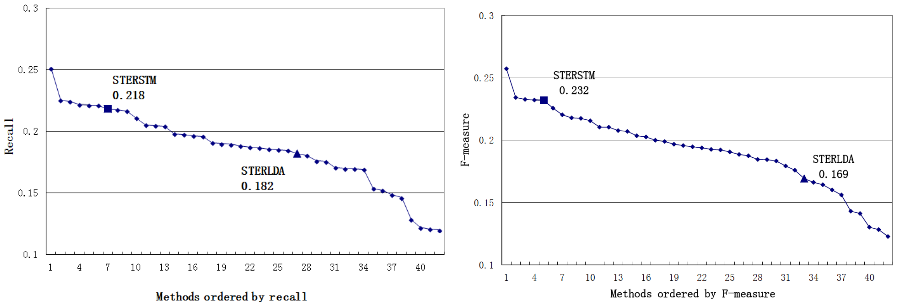
We report evaluation results based on ROUGE-1, i.e., checking unigram overlap between a given summary and a reference summary, partially because both STM and LDA are trained with unigrams. In particular, we use F-measure, which is a weighted harmonic mean of recall and precision.

$$\text{F-measure} = \frac{2 \cdot precision \cdot recall}{precision + recall}, \tag{3}$$

where the recall is the proportion of words in the reference summary appearing in the given sentence, and precision is the proportion of words in the given sentence appearing in the reference summary. Both precision and recall are based on an understanding and measure of relevance. An F-measure score reaches its best value at 1 and worst score at 0.

As we mentioned in Section 3, to facilitate fair comparison, a sentence was trimmed (removing duplicate words, frequent words, and semantically less important words which are not in top 100 word lists of in topic-word distributions) as ROUGE truncates summaries longer than the target length of 75 bytes (alphanumerics, whitespace, and punctuation included) before evaluation for DUC-2004.

For this corpus, the average recall, precision and F-measure of STERSTM are 0.218, 0.250, and 0.232, respectively. For STERLDA, they are 0.182, 0.160, and 0.169, respectively. STERSTM obviously outperforms STERLDA. In comparison with 40 participation methods in the DUC'04 conference, STERSTM did quite well in terms of all the three measures. It is ranked as 7th, 9th and



(a) Average recall (b) Average F-measure

Fig. 5. Title recommendation results of 42 methods for DUC-2004

5th in terms of average recall (Fig. 5(a)), precision, and F-measure (Fig. 5(b)). One DUC'04 participation method [6] that requires training data and WordNet, has F-measure of 0.234, which is the runner-up in Fig. 5(b). Its average recall is 0.217, quite close to that of STERSTM. The another graph-based document summarisation technique, the winner of several tasks in DUC'04, LexRank [10] has F-measure of 0.208 for this task and is 13th in Fig. 5(b). Thus, in terms of quality of very short summaries generated for DUC-2004, STERSTM is comparable with the top methods participated in the DUC'04 conference.

5 Conclusion and Discussion

Based on a recent topic modelling technique, Segmented Topic Model (STM), this work has presented one Semantic Title Evaluation and Recommendation (STER) technique, STERSTM. Through comparing title/sentences with the whole document in the topic space created by STM, STERSTM computes the semantic similarity of title/sentences, which can estimate the semantic coverage of a title/sentence. Via fitting a Generalised Extreme Value (GEV) distribution over the similarity values of sentences and a title within a document and calculating p-Value under the distribution, STERSTM is able to identify excellent and unfavourable titles without extra information like a human generated reference summary. The sentence with top p-Value is recommended as a title candidate. Experimental results on several different document sets have shown STERSTM can pick up some improvable titles, statistically significantly outperform STERLDA, a counterpart based on the canonical topic model LDA, and generate very short summaries with quality comparable with various document summarisation techniques.

There are several possible extensions of this work. Better trimming techniques to shorten a sentence to a concise and readable title could improve title recommendation [11]. It is appealing to explore more reliable statistical distributions for semantic similarity values, especially for those for small documents. We are also going to extend the proposed techniques for multiple relevant documents, embedding key words or other meta data.

Acknowledgements. Huidong Jin was financially supported by the Environmental and Agricultural Informatics program, CSIRO Mathematics, Informatics and Statistics. Lan Du was supported under the Australian Research Council's Discovery Projects funding scheme (project numbers DP110102506 and DP110102593). The authors are grateful to the anonymous reviewers and Dr Peter Caley for their constructive comments.

References

[1] Aggarwal, C., Zhai, C.: Mining Text Data. Springer-Verlag New York Inc. (2012)

[2] Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet allocation. J. Mach. Learn. Res. 3, 993–1022 (2003)

[3] Clough, P.: A perl program for sentence splitting using rules. University of Sheffield (2001)

[4] Coles, S.: An introduction to statistical modeling of extreme values. Springer (2001)

[5] Crain, S., Zhou, K., Yang, S., Zha, H.: Dimensionality Reduction and Topic Modeling: From Latent Semantic Indexing to Latent Dirichlet Allocation and Beyond. In: [1], ch. 5, pp. 129–161 (2012)

[6] Doran, W., Stokes, N., Newman, E., Dunnion, J., Carthy, J., Toolan, F.: News story gisting at university college dublin. In: The Proceedings of the Document Understanding Conference, DUC (2004)

[7] Du, L., Buntine, W., Jin, H.: Modelling sequential text with an adaptive topic model. In: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pp. 535–545. Association for Computational Linguistics (2012)

[8] Du, L., Buntine, W., Jin, H., Chen, C.: Sequential latent Dirichlet allocation. Knowledge and Information Systems 31(3), 475–503 (2012)

[9] Du, L., Buntine, W., Jin, H.: A segmented topic model based on the two-parameter Poisson-Dirichlet process. Machine Learning 81, 5–19 (2010)

[10] Erkan, G., Radev, D.: LexRank: Graph-based lexical centrality as salience in text summarization. J. Artif. Intell. Res. (JAIR) 22, 457–479 (2004)

[11] Jin, R., Hauptmann, A.G.: A new probabilistic model for title generation. In: COLING 2002, pp. 1–7 (2002)

[12] Lin, C., Och, F.: Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In: ACL 2004, p. 605. Association for Computational Linguistics (2004)

[13] Nenkova, A., McKeown, K.: A Survey of Text Summarization Techniques. In: [1], ch. 3, pp. 43–76 (2012)

[14] Svore, K., Vanderwende, L., Burges, C.: Enhancing single-document summarization by combining RankNet and third-party sources. In: EMNLP-CoNLL 2007, pp. 448–457 (2007)

[15] Xu, S., Yang, S., Lau, F.: Keyword extraction and headline generation using novel word features. In: AAAI 2010, pp. 1461–1466 (2010)

[16] Zhai, Z., Liu, B., Xu, H., Jia, P.: Constrained LDA for grouping product features in opinion mining. In: Huang, J.Z., Cao, L., Srivastava, J. (eds.) PAKDD 2011, Part I. LNCS, vol. 6634, pp. 448–459. Springer, Heidelberg (2011)

Video Quality Prediction over Wireless 4G

Chun Pong Lau, Xiangliang Zhang, and Basem Shihada

CEMSE, King Abdullah University of Science and Technology, Saudi Arabia
{lau.pong,xiangliang.zhang,basem.shihada}@kaust.edu.sa

Abstract. In this paper, we study the problem of video quality prediction over the wireless 4G network. Video transmission data is collected from a real 4G SCM testbed for investigating factors that affect video quality. After feature transformation and selection on video and network parameters, video quality is predicted by solving as regression problem. Experimental results show that the dominated factor on video quality is the channel attenuation and video quality can be well estimated by our models with small errors.

Keywords: Video Quality Prediction, Wireless 4G, Superposition Coded Multicasting.

1 Introduction

Fourth generation (4G) of mobile communication standards, such as Long Term Evolution Advanced (LTE-Advanced) and Worldwide Interoperability for Microwave Access Release 2 (WiMAX2), provide high speed and large range of wireless connectivity. Because of the large coverage of a base station (BS), users within the coverage area result in different channel quality causing the multi-users diversity problem. In order to provide a reliable video multicast/broadcast service by high speed wireless channels, She *et al.* proposed Superposition Coded Multicasting (SCM) method in [1]. In this scheme, a scalable video bitstream consists of two different quality layers that are modulated into two signals by two different modulations. These two signals are then superimposed into one broadcasting signal for all receivers under the BS coverage. Since the transmission of superimposed signal can be affected by wireless channel conditions, the quality of received video is key concern for both the providers and receivers.

From the video providers' point of view, the Quality of Service (QoS) is determined by the quality of video that customers will receive. If the quality of video can be predicted according to network conditions and video parameters, providers can guarantee certain level of service by taking effective actions. On the one hand, before broadcasting a video, providers can set the input features/parameters to achieve the targeted video quality. On the other hand, during video transmission, operators can adjust the video and network parameters according to this prediction model for various conditions.

There are three main challenges facing the accurate prediction of video quality over wireless channels. First, the behavior of wireless signal is difficult to predict and measure, since it fluctuates and can be significantly affected by complex

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 414–425, 2013.

and mixed factors, such as attenuation, fading, and interference [2]. Attenuation happens due to the distance between transmitter and receiver. Fading may vary over time, position or radio frequency due to multi-path propagation. Second, multi-user channel diversity challenge where users at different geographic locations within the same BS coverage area have different channel quality [3]. Third, collecting data for analysis and prediction is expensive. In order to accurately predict the video quality, all possible conditions that may affect video quality must be considered, and all corresponding data should be collected. Constructing an emulation testbed with implementation of SCM is often expensive. To the best of our knowledge, the testbed used in this paper is the first built emulation testbed on this research area. This work assists research work without real testbed by building a practical prediction model. It helps to reduce the dependence on the expensive testbed system through estimating transmission results, instead of conducting real experiments.

Video quality prediction can be formalized as a regression problem. Video transmission is described by a set of features, including video and network parameters. Video quality, measured by a common quality metric called Peak Signal-to-Noise Ratio (PSNR), is the target for prediction [4]. Given a raw data, feature selection is first used for selecting the useful features from the data set. Three popular regression approaches, k-nearest neighbor algorithm (k-NN), Support Vector Machines (SVM) and Neuron Network (NN) are employed for prediction. Nominal type feature (*modulation*) and a ratio type feature (*power boosting*) are transformed together into one ratio type feature. The improved results from advance approaches including the transformed feature demonstrate the main factors affecting video quality and should gain more attention in video transmission.

The rest of this paper is organized as follows. Section 2 introduces the background and related work of video quality prediction problem. Section 3 describes the methodology for prediction, which consists of data collection on real testbed, feature transformation and selection, and employed prediction techniques. Section 4 reports experimental results. Section 5 finally concludes and gives perspectives.

2 Background

Researchers face many challenges on video quality prediction. One of the difficulties is the videos are affected by numerous parameters. Through collecting data from simulations of 3G and WLAN network, Khan *et al.* predicted the video quality with both application and network level parameters [5]. However, this was limited to four parameters, which are frame rate, send bitrate, packet error rate, and link bandwidth. These limited number of features cannot represent the network condition well especially when network condition becomes complex in the wireless 4G.

Garcia *et al.* proposed a framework to find a parametric content description of videos [6]. Assuming that video content influences the video quality, they proposed to decompose the input video to spatial and temporal features which highlight a strong adequation with the perceived video quality. However, prediction of video quality was not quantified in their approach.

Dalal *et al.* investigated the feature selection for predicting video quality after network transmission [7]. Network statistic such as packet lost rate and error rate were considered as features. Two methods correlation ranking and principal component analysis (PCA) were investigated. However, video they studied were collected from a wired network.

Liu *et al.* proposed a framework for video prediction after encoding process, which can reduce time cost for choosing correct parameters for video encoding process [8]. Their approach was simple and efficient. However, they did not consider the parameters of network transmission for the prediction.

All of the above reported studies omitted the wireless network characteristics and assumed a simulation based data. Therefore, we attempt to collect real-life data and perform more accurate prediction considering the wireless network characteristics. In the next section, we introduce our methodology.

3 Methodology

This section introduces our proposed approach of video quality prediction. We first introduce the testbed for data collection. Then, as pre-processing is a crucial part of prediction, we introduce feature transformation, normalization and ranking mechanisms. Finally, k-NN, SVM and NN are introduced to build the regression models for video quality prediction.

3.1 Testbed Description and Data Collection

Video data in this paper is collected by a real-time SCM testbed, which consists of two network emulators, two personal computers (PC) and one variable attenuator. In the transmitter side, one PC is connected to one network emulator acted as the video server and BS. The other PC is connected to the second emulator acted as the mobile receiver and mobile device. Videos are transmitted over the emulators connected by coaxial radio frequency cables through the variable attenuator acting as the wireless channel with different values of channel attenuation. The emulators are implemented by National Instruments NI PXIe-1062Q chassis with embedded controller PXIe-8130, IF transceiver PXIe-5641R, RF Up converter PXIe-5610 and RF Down converter PXIe-5600. Figure 1 shows the system architecture of SCM with transmitter on the top. The video data passes through all layers and become superposition coded signal broadcast over the emulated wireless channel. Receiver receives the broadcasting signals that go through all the layers to reconstruct the video data.

Video streams are encoded as H.264-SVC [9] bitstream, and are stored in the video server. Each video sequence consists of two layers, base layer and enhancement layer. Base layer contains most of the important information of the video and should be correctly received and decoded in order to reconstruct the video after transmission. Enhancement layer is additional information of the base layer to improve the video quality when successfully decoded. It is expected that enhancement layer bitrate is higher than base layer bitrate since

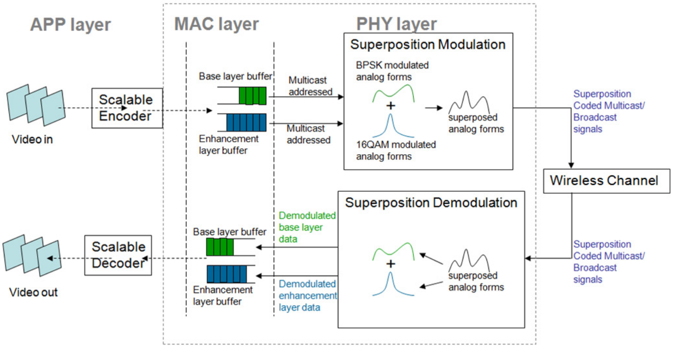


Fig. 1. SCM System Architecture

the encoding quantization parameters (QP) of enhancement layer is lower than the base layer QP. Different number of group of pictures (GOP) and intra period are considered as the video features/parameters in the system.

From the network prospective, transmission power, and reception power sensitivity are controlled by the transmitter and receiver emulators in physical layer. Channel attenuation is controlled manually by the variable attenuator. Assume that during each of the transmission, the attenuation value is constant to remain a static transmission environment to reduce complexity. Carrier frequency, modulation, power boosting, cyclic prefix and bandwidth are selected from the MAC layer control software.

A video transmission, when completed, is described by a set of video and network features. The first three columns of Table 1 show the features considered in this work. The last row is the prediction target PSNR, which is commonly used to measure the quality of the reconstructed video after network transmission. It is calculated by comparing the video frame by frame and pixel by pixel between the received video and the original video before encoding and transmission. A higher PSNR value introduces a higher video quality received. Given a video with its known feature values, our target is to predict the PSNR after it is received.

In our experiments throughout the paper, the video transmitted is a 40 seconds length movie trailer, which is a standard video provided by Durian Open Movie project named as “Sintel” [10]. Each test of experimental operation takes around 2 minutes. In addition, calculating the PSNR value between received video with original video requires 1 to 2 minutes. 985 testes were conducted in total. Ten random chosen examples are shown in Table 2.

3.2 Feature Transformation

Two network features called *Base Layer Modulation* and *Enhancement Layer Modulation* are of our interest while predicting video quality. The possible values of these two features are BPSK, QPSK, 16QAM and 64QAM, which are difficult to be directly used for prediction. Therefore, *Base Layer Modulation* and *Base Layer Power Boosting* are transformed to *Base Layer Probability of transmitted*

Table 1. Video and network features, with statistic, Pearson correlation coefficient and P-value w.r.t. target PSNR

	Features	Type	Min	Max	Range	Co-unt	Pearson Corr. Coef.	P-value
Video	Base bitrate	Ratio	399.5	739	339.5	7	0.116650	2.4360e-4
	Enh. bitrate	Ratio	777.88	3074.25	2296.4	7	0.101631	1.4037e-3
	Overall bitrate	Ratio	1189.3	3473.7	2284.3	6	0.109720	5.6144e-4
	Base Encoding QP	Interval	20	28	8	3	-0.107587	7.1926e-4
	Enh. Encoding QP	Interval	35	40	5	2	0.028953	3.6404e-1
	Group of Pictures	Interval	2	8	6	3	-0.074944	1.8652e-2
	Intra Period	Interval	2	32	30	5	-0.114500	3.1721e-4
Network	Transmission Power	Ratio	0	0	0	1	NaN	NaN
	Reception Power Sen.	Ratio	-10	0	10	2	-0.028953	3.6404e-01
	Attenuation	Ratio	2	21	19	18	-0.502685	3.3726e-64
	Carrier Frequency	Ratio	2.51	2.51	0	1	0.000000	1
	Base Modulation	Nominal	BPSK	BPSK	N/A	1	Transformed	Transformed
	Base Power Boosting	Ratio	-3	2	5	4	Transformed	Transformed
	Base Prob. of Tx Sym. Err	Ratio	2.328e-4	0.8858	0.0884	9	-0.190526	1.6672e-9
	Enh. Modulation	Nominal	QPSK	16QAM	N/A	2	Transformed	Transformed
	Enh. Power Boosting	Ratio	-11	0	11	7	Transformed	Transformed
	Enh. Prob. of Tx Sym. Err	Ratio	0.0789	0.5253	0.4465	9	0.151654	1.7421e-6
	Cyclic Prefix	Interval	0.25	0.25	0	1	NaN	NaN
	Bandwidth	Interval	5	10	5	2	0.145872	4.2871e-6
Target	PSNR	Ratio	25.26	66.91	41.64	461		

Table 2. Examples of video records

Base Bitrate	Enh. Bitrate	Overall Bitrate	B. QP	E. QP	GOP	I. P.	Tx P.	Rx P.	At.	Freq.	B. Mod	B. PB	E. Mod	E. PB	CP	BW	PSNR
520.63	1686.87	2207	25	35	2	4	0	0	2	2.51	BPSK	-1	QPSK	-7	1/4	10	58.5195
520.63	1686.87	2207	25	35	2	4	0	0	9	2.51	BPSK	-1	QPSK	-11	1/4	10	55.2172
739.	1634.75	2373	25	35	2	2	0	0	7	2.51	BPSK	2	16QAM	0	1/4	10	58.9206
442.	1201.25	1643	25	35	4	8	0	0	7	2.51	BPSK	2	16QAM	0	1/4	10	58.9995
415.13	991.62	1406.75	25	35	8	32	0	0	4	2.51	BPSK	2	16QAM	0	1/4	10	56.7126
411.5	777.88	1189	28	35	4	16	0	0	2	2.51	BPSK	-1	QPSK	-7	1/4	5	56.3021
410.63	1221.5	1189	25	35	4	16	0	0	2	2.51	BPSK	-1	QPSK	-7	1/4	10	55.9479
411.5	777.88	1189	28	35	4	16	0	0	11	2.51	BPSK	-1	QPSK	-11	1/4	5	47.0903
411.5	777.88	1189	28	35	4	16	0	0	5	2.51	BPSK	-1	QPSK	-5	1/4	5	55.5979
399.5	3074.25	3473	20	40	4	4	0	-10	10	2.51	BPSK	-1	16QAM	-5	1/4	10	52.9213

symbol error. Similarly for Enhancement layer, *Enhancement Layer Modulation* and *Enhancement Layer Power Boosting* are transformed to *Enhancement Layer Probability of transmitted symbol error*. The idea of the transformation is considering the transmitted symbol energy and the receiver symbol decision area. *Probability of symbol error* means a transmitted symbol falls into other symbols decision area with the amount of Gaussian noise added onto it. The probability of symbol error is calculated as follows according to [11]. Assuming that base layer modulation is always BPSK, the probability of correctly detecting the abscissa of each SPC symbol in the i^{th} region is:

$$P_{i,1} = Q\left(\frac{-\mu_{y_i}}{\sigma_{y_i}}\right) \tag{1}$$

where μ_{y_i} is the mean for received symbol abscissa coordination and σ_{y_i} is the variance. *Base layer probability of transmitted symbol error* is:

$$P_b = 1 - \frac{2\sqrt{m_2}}{M} \sum_{i=0}^{\sqrt{m_2}-1} P_{i,1} \tag{2}$$

where $M = m_1 \times m_2$, m_1 is the base layer modulation number and m_2 is the enhancement layer modulation number.

After the symbol is decoded on base layer by successive interference cancellation (SIC), the remained symbol will be decoded by enhancement layer demodulator. The standard symbol error equation for m_2-QAM demodulation is expressed as follow:

$$\begin{aligned} P_{m_2QAM} = 2 &\left[2\left(1 - \frac{1}{\sqrt{m_2}}\right) Q\left(\sqrt{\frac{3}{m_2 - 1}\frac{E_2}{N_0}}\right)\right] \\ &- \left[2\left(1 - \frac{1}{\sqrt{m_2}}\right) Q\left(\sqrt{\frac{3}{m_2 - 1}\frac{E_2}{N_0}}\right)\right]^2 \end{aligned} \tag{3}$$

where $Q(.)$ is known as Q-function, E_2 is the remained energy after SIC of symbol for enhancement layer decoding process, N_0 is the Gaussian noise power.

Finally the *probability of enhancement symbol error* is calculated as follow:

$$P_e = 1 - (1 - P_{m_2QAM})(1 - P_b) \tag{4}$$

Transformed feature P_b and P_e range from 0.00024 to 0.08858 for *base layer* and 0.0789 to 0.5253 for *enhancement layer.*

3.3 Feature Normalization

Since values of raw video features are on different scale, normalization is performed to linearly scale them to unit range. An original value x of a feature is normalized to be x^*:

$$x_i^* = \frac{x_i - x_{min}}{x_{max} - x_{min}} \tag{5}$$

where x_{min} is the smallest value and x_{max} is the largest value of the feature. To further investigate the improvement that can be made by normalization, original and normalized data are compared in the experimental results.

3.4 Feature Ranking

In order to eliminate irrelevant and redundant features in the raw data, supervised feature selection is performed in data preprocessing phase. Features are ranked by their Pearson correlation coefficients R_f with respect to the target PSNR in training data. As a measure of the linear dependence of a feature and the target,

$R_f = +1$ indicates a strong linear relationship between them, $R_f = -1$ represents a negative linear relationship and $R_f = 0$ shows the independence of them. For a feature X_f, its coefficient w.r.t. target Y is defined as:

$$R_f = \frac{cov(X_f, Y)}{\sqrt{var(X_f)var(Y)}} \tag{6}$$

Features with R_f close to 0 will be eliminated when learning a prediction model.

3.5 Regression Models

Three regression algorithms are employed in this paper for predicting video quality, k-Nearest Neighbor, Support Vector Machine and Neural Network.

k-Nearest neighbor algorithm (k-NN) is a lazy learning method. The quality of a testing video is estimated based on its k nearest neighbors in training data, which have similar feature values to the testing video. Assuming that its k most similar videos have PSNR value $T_j, j = 1...k$, the quality of the video i is

$$T_i = \frac{\sum_{j=1}^{k} w_{i,j} \times T_j}{\sum_{j=1}^{k} w_{i,j}} \tag{7}$$

where $w_{i,j}$ is the weight function inversely proportional to the distance of the training object j. It is defined as: $w_{i,j} = \frac{1}{c+(d_{i,j})^2}$ where c is the kernel width parameter and $d_{i,j}$ is the distance between video i and j. Considering the feature correlation coefficients, we define distance $d_{i,j}$ by $d_{i,j} = \sqrt{\sum_f R_f \times (x_{i,f} - x_{j,f})^2}$ where $x_{i,f}$ and $x_{j,f}$ is the value of feature f in video i and j.

To acquire the best results, the number of nearest neighbors k is set from 1 to 10, and the parameter c is tested by values in $[10^{-4} - 10^{-3}]$ stepped by 5×10^{-5}.

Support Vector Machine (SVM) is considered as one of the most robust and powerful algorithms for classification and regression. In this work, Epsilon-SVR, the regression model of LIBSVM 3.12 [12], is employed for video quality prediction. After testing of various kernel functions, Radial Basis Function (RBF) kernel is selected for its best performance: $\phi(x, x_i) = e^{-\gamma\|x-x_i\|^2}$. The setting of SVM parameters C ($cost$) and kernel parameter γ ($gamma$) are explored by searching in $[1 - 100]$ for C and in $[10 - 400]$ for γ. Experimental results on different settings of C and γ will be demonstrated in Figure 3 of the next section.

Neural Network (NN) has been widely used in predicting video quality based on application and network parameters [5,13,14]. Given enough hidden units, any continuous function can be uniformly approximated to arbitrary accuracy [15]. We applied two to three hidden layers NN with back-propagation learning algorithm. Different numbers of hidden neurons are evaluated ranged from $1 \times F$ to $3 \times F$, where F is the number of features. Results of NN with different topology settings will be reported in next section.

4 Experimental Results

This section presents the experimental results on 985 video testings. 5-fold cross validation is applied for reliable evaluation reason. In order to investigate the impact of feature normalization and transformation introduced in last section, four different cases summarized in Table 3 are evaluated and compared.

Table 3. Experimental Evaluation Cases

		Modulation and boosting	
		Excluded	Transformed
Normalization	Without	Case 1	Case 3
	With	Case 2	Case 4

Root Mean Squared Error (RMSE) and Coefficient of Determination (R^2) are considered as the criteria to evaluate the performance of the prediction methods. RMSE is used to quantify the difference between the predicted and actual PSNR value. A smaller value indicates a better result with higher accuracy. RMSE is calculated as $\sqrt{\sum_{i=1}^{N}(T_i - T_{i-act})^2/N}$, where T_i and T_{i-act} are the predicted and actual PSNR respectively, and N is the number of testing videos.

R^2 is another performance measurement of prediction. It is calculated by $R^2 = 1 - \frac{SSE}{SST}$, where the total sum of square $SST = \sum_{i=1}^{N}(T_{i-act} - \frac{1}{N}\sum_{i=1}^{N} T_{i-act})^2$ and sum of squares error $SSE = \sum_{i=1}^{N}(T_{i-act} - T_i)^2$. R^2 is in [0 1], where the maximum value 1 indicates the best prediction model.

4.1 Feature Ranking

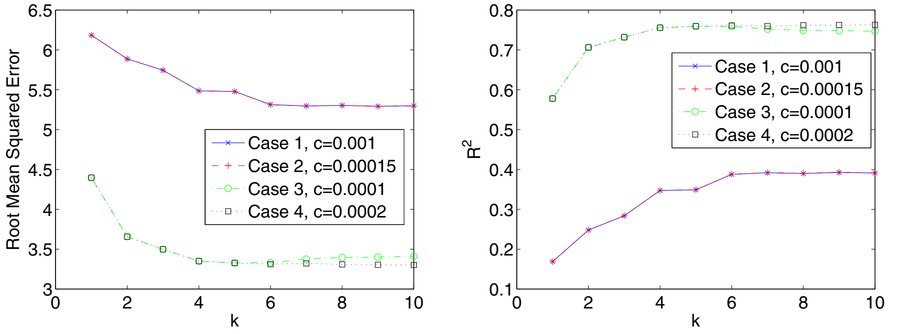
As introduced in section 3.4, each feature is ranked by its Pearson correlation coefficient R_f. The last two columns of Table 1 show the R_f and corresponding P-values of all features. Those features with $-0.03 < R_f < 0.03$ or NaN[1] are eliminated, such as Enhancement Encoding QP (0.028953), Transmission Power (NaN), Reception Power Sensitivity (-0.028953), Carrier Frequency (0.000000), and Cyclic Prefix (NaN). Their large P-values (> 0.05) also confirm that these feature and the target value have no significant correlation.

4.2 Importance of Transforming Modulation and Power Boosting

To investigate the importance of transforming modulation and power boosting features presented in Section 3.2, we compare results of Case 1 and 2 with that of Case 3 and 4.

Figure 2a and 2b show the RMSE and R^2 of k-NN on each case. The best setting of kernel parameter c when calculating $w_{i,j}$ was exhaustively explored. Figure 2 presents the results with the best setting. As we can see for all values

[1] The 0 or NaN of R_f for some features is caused by their single distinct value, as given in the 7th column (Count) of Table 1.

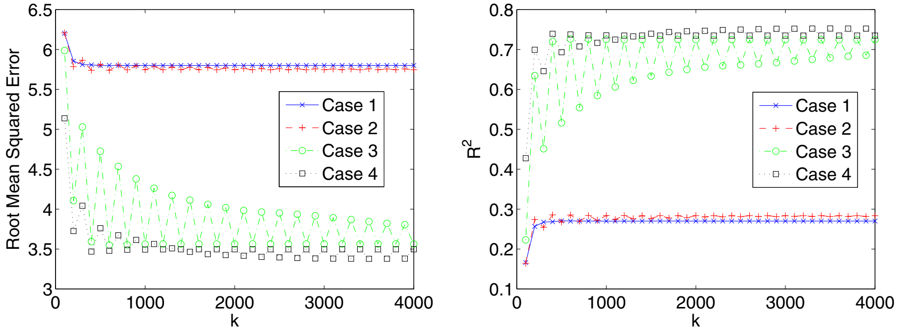


(a) k-NN - Root Mean Square Error (b) k-NN - Coefficient of determination

Fig. 2. Prediction performance of k-NN on each case with the best setting of c

of k, Case 3 and 4 are significantly better than Case 1 and 2 with lower RMSE values and higher R^2. We can then conclude that feature transformation plays a very important role on correctly predicting video quality.

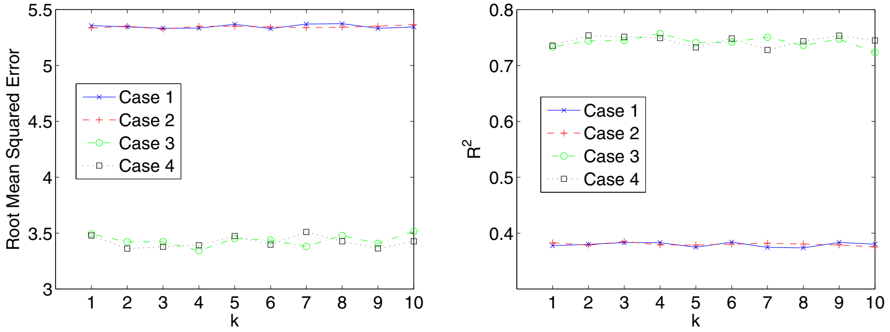
Figure 3 presents the prediction performance of SVM. As observed in k-NN, better results are obtained from Case 3 and 4 rather than Case 1 and 2. The setting of SVM parameter C and γ is tested by combinations of values indexed by k on x-axis. The best prediction is on Case 4 when $C = 9$ and $\gamma = 120$ ($k = 332$) resulting RMSE $= 3.218$ and $R^2 = 0.7669$.



(a) SVM - Root Mean Square Error (b) SVM Coefficient of determination

Fig. 3. Prediction performance of SVM on each case (sampled C and γ setting combinations)

Figure 4 shows the results of NN with different network topology setting indexed by k, e.g., $k = 1$ when NN has $F \times F$ neurons (2 hidden layers, each of which has F hidden neurons), $k = 5$ when NN has $F \times F \times F$ neurons (3 hidden layers), where F is the number of features after selection. The same observation can be found that Case 3 and 4 perform better than Case 1 and 2. The best prediction results with lowest RMSE and highest R^2 happened on Case 3 when NN has $3F \times 3F$ neurons ($k = 4$).



(a) NN - Root Mean Square Error (b) NN Coefficient of determination

Fig. 4. Prediction performance of NN with different combination of hidden layers and neurons

Results of k-NN, SVM and NN consistently show that prediction excluding modulation and power boosting features (Case 1 and 2) gives worse results than transforming them into probability of symbol errors (Case 3 and 4). In other words, transformation of modulation and power boosting is essential.

4.3 Effects of Normalization of Data

Since Case 3 and 4 achieve better prediction results, we study the normalization effect on these two cases in this subsection (Case 1 and 2 are exempted). In k-NN, normalization does not make significant effect. In Figure 2, normalized data (Case 4) performs exactly the same as original data (Case 3) when k is less than 4. Normalized data (Case 4) leads to better results than original data (Case 3) when k increases.

Normalization stabilized the performance of SVM when varying parameter setting. Fluctuating curves of Case 3 and smoothing curves of Case 4 in Figure 3 indicate that normalized data can produce better (lower RMSE and higher R^2) and more stable results. However, NN does not significantly benefit from normalization.

In summary, normalization can generally further improve the prediction performance.

4.4 Comparison of Regression Models

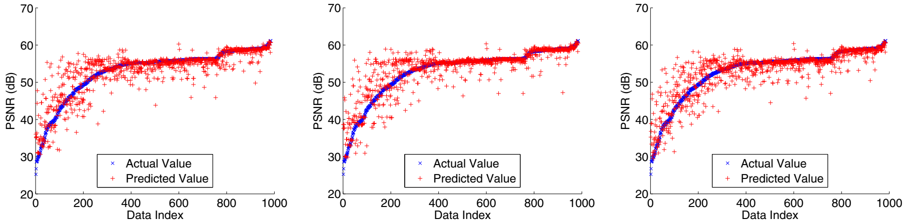
Table 4 compares the prediction performance of k-NN, SVM, and NN on Case 3 and Case 4. Case 1 and 2 are excluded because their prediction results are much worse than that of Case 3 and 4. The best RMSE and R^2 value of each model were presented when its model parameters are set appropriately. In general, three models give similar results, and they all produce small RMSE and high R^2. SVM on Case 4 can lead to the smallest RMSE and the highest R^2.

Figure 5 shows the predicted and actual PSNR values for the best prediction of all models, k-NN on Case 4 when k=10,c=0.0002, SVM on Case 4 when C=9,

Table 4. The best predictions of k-NN, SVM, and NN on Case 3 and 4

		k-NN	SVM	NN
RMSE	Case 3	3.32864	3.27009	3.34251
	Case 4	3.30163	**3.21764**	3.36173
R^2	Case 3	0.75914	0.75922	0.75707
	Case 4	0.76283	**0.76689**	0.75371

γ=120, and NN on Case 3 when topology is $3F \times 3F = 30 \times 30$, where F is the number of features after feature transformation and selection. Videos on x-axis are ordered by their actual PSNR values. We can observe that all prediction methods can give good prediction when actual PSNR is high. Especially, the prediction of SVM is more accurate when actual PSNR is larger than 54dB. The prediction deviates more from the true target when actual PSNR is relatively low. However, the deviation is only around 5% of the target on average. These less accurate predictions are due to the lack of effective learning examples. Videos with low PSNR have worse quality. In a well-developed transmission system, the number of damaged videos is smaller than the number of undamaged videos.



(a) k-NN at k=10,c=0.0002 (b) SVM at C=9, γ=120 (c) NN with 30 × 30

Fig. 5. Comparison of predicted PSNR and actual PSNR

5 Conclusion

This paper studies the problem of video quality prediction in wireless 4G network transmission. A real testbed is built for collecting videos transmitted under various network conditions. The raw transmission data is pre-processed by feature transformation, normalization and ranking. Regression models are learned by three different algorithms and used for predicting the quality of video.

The first motivation of this work is to investigate which features are strongly correlated to video quality. From the feature ranking results, we find that attenuation and base layer probability of symbol error are the two important features that highly correlated to the PSNR (target of prediction) with the correlation coefficient R_f= -0.5027 and -0.1905, P-value = 3.372e-64 and 1.667e-9, respectively. In addition, according to the evaluation results, we observe that feature transformation of modulation and power boosting for both base layer and enhancement layer significantly affected the prediction accuracy of video quality.

Through pre-processing raw data and exploring the most suitable parameter setting, we built prediction models based on three different regression algorithms.

The experimental results demonstrate that these models can accurately predict video quality with small errors and high relevance with target values.

This first attempt of video quality prediction in wireless 4G opens several perspectives for further research. First, the models are useful to reduce the time spent on running real experiments. As we mentioned in the introduction, imitating all of the different channel conditions and conducting video transmitting experiments are very time consuming. Our discovery of key features and prediction model can be used to guide the design of experiments. Second, we will enhance the prediction models to make them perform well on more complex channel conditions.

References

1. She, J., Yu, X., Hou, F., Ho, P.H., Yang, E.H.: A Framework of Cross-Layer Superposition Coded Multicast for Robust IPTV Services over WiMAX. In: IEEE Wireless Communications and Networking Conference WCNC, pp. 3139–3144. IEEE (2008)
2. Judd, G., Steenkiste, P.: Characterizing 802.11 wireless link behavior. Wireless Networks 16(1), 167–182 (2008)
3. Knopp, R., Humblet, P.: Information capacity and power control in single-cell multiuser communications. In: IEEE International Conference on Communications ICC 1995 Seattle, Gateway to Globalization, vol. 1, pp. 331–335 (1995)
4. Huynh-Thu, Q., Ghanbari, M.: Scope of validity of PSNR in image/video quality assessment. Electronics Letters 44, 9–10 (2008)
5. Khan, A., Sun, L., Ifeachor, E.: Content-based video quality prediction for MPEG4 video streaming over wireless networks. Journal of Multimedia 4(4), 228–239 (2009)
6. Garcia, M.N., Raake, A.: Towards Content-related Features for Parametric Video Quality Prediction of IPTV Services. In: IEEE International Conference on Acoustics Speech and Signal Processing, ICASSP 2008, pp. 757–760 (2008)
7. Dalal, A., Olson, J.: Feature Selection for Prediction of User-Perceived Streaming Media Quality. In: Preoceedings of International Symposium on Performance Evaluation of Computer and Telecommunication Systems SPECTS, pp. 285–294 (2007)
8. Liu, Y.X., Kurceren, R., Budhia, U.: Video classification for video quality prediction. Journal of Zhejiang University SCIENCE A 7(5), 919–926 (2006)
9. Schwarz, H., Marpe, D., Wiegand, T.: Overview of the Scalable Video Coding Extension of the H.264/AVC Standard. IEEE Transactions on Circuits and Systems for Video Technology 17(9), 1103–1120 (2007)
10. Roosendaal, T.: Sintel. In: ACM SIGGRAPH 2011 Computer Animation Festival, SIGGRAPH 2011, p. 71. ACM, New York (2011)
11. Ho, J.C.C.: Logical superposition coded modulation for wireless video multicasting. Master's thesis, Univ. of Waterloo, Waterloo, Canada (2009)
12. Chang, C., Lin, C.: LIBSVM: a library for support vector machines. ACM Transactions on Intelligent Systems and Technology (TIST) 2(3), 27 (2011)
13. Mohamed, S., Rubino, G.: A study of real-time packet video quality using random neural networks. IEEE Transactions on Circuits and Systems for Video Technology 12(12), 1071–1083 (2002)
14. Callet, P.L.: A convolutional neural network approach for objective video quality assessment. IEEE Transactions on Neural Networks 17(5), 1316–1327 (2006)
15. Bishop, C.M.: Pattern recognition and machine learning. Springer, New York (2006)

A Self-immunizing Manifold Ranking for Image Retrieval

Jun Wu[1], Yidong Li[1], Songhe Feng[1], and Hong Shen[1, 2]

[1] School of Computer and Information Technology, Beijing Jiaotong University,
Beijing 100044, China
[2] School of Computer Science, University of Adelaide, SA 5005, Australia
{wuj,ydli,shfeng,hshen}@bjtu.edu.cn

Abstract. Manifold ranking (MR), as a powerful semi-supervised learning algorithm, plays an important role to deal with the relevance feedback problem in content-based image retrieval (CBIR). However, conventional MR has two main drawbacks: 1) in many cases, it is prone to exploit "unreliable" unlabeled images when deployed in CBIR due to the semantic gap; 2) the performance of MR is quite sensitive to the scale parameter used for calculating the Laplacian matrix. In this work, a self-immunizing MR approach is presented to address the drawbacks. Concretely, we first propose an elastic *k*NN graph as well as its constructing algorithm to exploit unlabeled images "safely", and then develop a local scaling solution to calculate the Laplacian matrix adaptively. Extensive experiments on 10,000 Corel images show that the proposed algorithm is more effective than the state-of-the-art approaches.

Keywords: content-based image retrieval, relevance feedback, self-immunizing manifold ranking, elastic *k*NN graph, local scaling.

1 Introduction

With the ubiquitous use of digital images in a large number of practical applications, Content-Based Image Retrieval (CBIR) has drawn substantial research attention in many computer communities during the past two decades [2]. A main challenge in CBIR is the so-called semantic gap, i.e. the low-level visual features are not sufficient to characterize the high-level semantics of images. Relevant feedback has been shown as a powerful tool for bridging the semantic gap by exploiting the user's interaction with CBIR system. During the past years, a wide variety of relevant feedback techniques have been proposed, most of which belong to the family of supervised learning [14, 2].

One critical research topic related to relevance feedback is to learn with few labeled training examples, as few users are patient to label a lot of images during the interaction. To this end, semi-supervised learning [1] has been applied to relevance feedback [3, 4, 8, 9, and 13]. A popular semi-supervised learning method used in CBIR is the manifold ranking (MR) that aims to learn a ranking function by making use of the underlying geometrical structure of the given image database. Previous studies have shown that MR is one of the most promising and successful semi-supervised learning techniques for relevance feedback [3, 7, 10 and 11].

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 426–436, 2013.

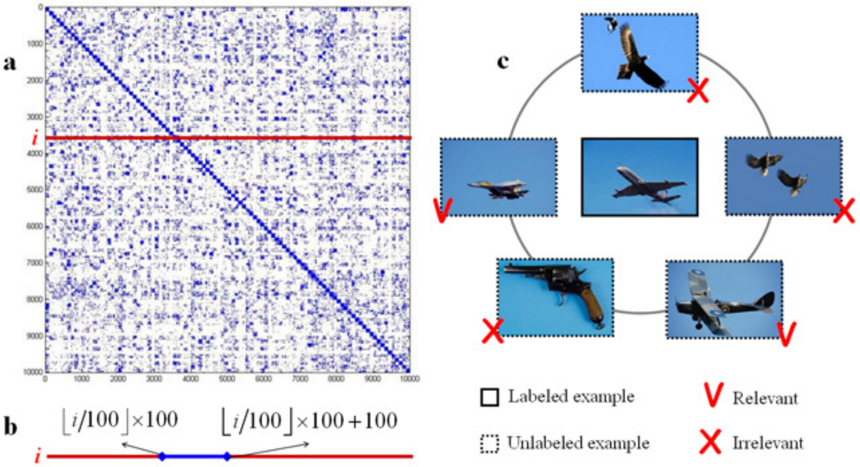


Fig. 1. An **illustration of our motivation**: (a) The adjacency matrix of a *k*NN graph built on a set of 10,000 images, each consecutively-numbered 100 images in which belong to the same semantic category; (b) The "trusted" interval of a labeled image; and (c) An example of a labeled image with "unreliable" unlabeled nearby neighbors.

However, it has been found that the performance of semi-supervised learning may be even worse than the supervised learning when "unreliable" unlabeled data is exploited [5]. Taking MR as an example, it assumes that a labeled example and its (unlabeled) nearby neighbors trend to have similar properties, and thus their ranking scores should be approximate, but this assumption may not be true in CBIR due to the semantic gap. To verify the efficacy of this assumption, we conducted an empirical study on a set of 10,000 images, each consecutively-numbered 100 images in which belong to the same semantic category. Given the image set, a *k*NN graph is constructed and corresponding adjacency matrix is shown by Figure 1a. Ideally, for each image i, we expect its k nearest neighbors appear within a "trusted" interval $\left[\text{floor}(i/100)\times 100,\ \text{floor}(i/100)\times 100+100\right]$ (e.g., if an image id is 1588, its "trusted" interval should be [1501, 1600]), since the images within this interval belong to the same class as illustrated in Figure 1b, where $\text{floor}(\bullet)$ denotes the integer operation. Figure 1a shows that most nonzero elements distribute around the principal diagonal of the adjacency matrix, which means most neighbor points are inside their trusted interval. But there are still many nonzero elements far away from the principal diagonal, i.e. the neighbor points are outside the trusted interval (an example is illustrated by Figure 1c), and, in this case, the performance of MR may degenerate. Moreover, the performance of MR is sensitive to the scale parameter used for calculating the Laplacian matrix. Such a parameter is usually hard to tune with very few labeled examples [11], which is a common issue in graph-based semi-supervised learning.

To address the above problems, this paper presents a **S**elf-**i**mmunizing **ma**nifold **r**anking (Simar) approach for relevance feedback in CBIR, which is able to exploit unlabeled images "safely" and tune the scale parameter adaptively. Concretely, we first propose a new graph structure named elastic *k*NN graph and corresponding constructing algorithm. In this structure, the creditable relationship between each labeled image and its nearby neighbors can be dynamically adjusted by monitoring the change of retrieval performance. Also, a local scaling solution is employed by Simar to tune the scale parameter for Laplacian matrix calculation, which is beneficial to the data distribution with multi-scales, e.g. image database. Our empirical study shows encouraging results in comparison to some existing semi-supervised learning algorithms widely used in CBIR.

The remainder of this paper is organized as follows. Section 2 elaborates the proposed Simar approach. Section 3 shows experimental evaluations. Finally, section 4 concludes this paper.

2 The Proposed Simar Approach

In this section, we first formulate relevance feedback in CBIR as a semi-supervised graph-based ranking problem, and then present an elastic *k*NN graph structure and a local scaling method to facilitate the setting of parameters.

2.1 Preliminaries

Let $\mathbf{X}=\{\mathbf{x}_1,\cdots,\mathbf{x}_n\}$ denote an image database, where each $\mathbf{x}_i \in \mathbb{R}^d$ represents an image by a *d*-dimensional feature vector. To discover the geometrical structure of the given image database, a graph (such as *k*NN graph) is usually built on $\mathbf{X}$ and we define $\mathbf{W} \in \mathbb{R}^{n\times n}$ as corresponding adjacency matrix with element w_{ij} saving the weight of the edge between point i and j. Normally the weight can be calculated using a Gaussian kernel

$$w_{ij} = \exp\left(-d^2\left(\mathbf{x}_i,\mathbf{x}_j\right)/\sigma^2\right) \tag{1}$$

if $\mathbf{x}_j \in N_k\left(\mathbf{x}_i\right)$ or $\mathbf{x}_i \in N_k\left(\mathbf{x}_j\right)$, otherwise $w_{ij}=0$, where $N_k\left(\mathbf{x}\right)$ denotes the set of the *k* nearest neighbors of $\mathbf{x}$, and $d\left(\mathbf{x}_i,\mathbf{x}_j\right)$ is a distance metric (such as L1 distance) between $\mathbf{x}_i$ and $\mathbf{x}_j$. Finally, we define a label vector as $\mathbf{y}=\left[y_1,\cdots,y_n\right]^T$ to record the user's judgment in relevance feedback loops, in which an element $y_i=1$ if $\mathbf{x}_i$ is the query or labeled as positive, $y_i=-1$ if $\mathbf{x}_i$ is labeled as negative, and $y_i=0$ otherwise.

Given $\mathbf{W}$ and $\mathbf{y}$, the goal of our Simar approach is to learn a ranking function $\mathrm{f}:\mathbf{X}\rightarrow\mathbb{R}$ that assigns each image $\mathbf{x}_i$ a ranking score f_i according to its relevance to user's query. Similar to other MR methods, Simar aims to find an optimal f^* by solving the following optimization problem:

$$\mathrm{O}(\mathrm{f})=\frac{1}{2}\left(\sum_{i,j=1}^{n} w_{ij}\left\|\frac{1}{\sqrt{D_{ii}}}f_i-\frac{1}{\sqrt{D_{jj}}}f_j\right\|^2+\mu\sum_{i=1}^{n}\left\|f_i-y_i\right\|^2\right) \quad (2)$$

where $\mu > 0$ is the regularization parameter and **D** is a diagonal matrix with $D_{ii}=\sum_{j=1}^{n} w_{ij}$. The first term is a smoothness constraint that makes the nearby images share close ranking scores. The second term is a fitting constraint which means the ranking result should fit to the label assignment. By minimizing $\mathrm{O}(\mathrm{f})$, we get the optimal f by the following closed form

$$\mathrm{f}^{*}=\left(\mathbf{I}_n-\alpha\mathbf{S}\right)^{-1}\mathbf{y} \quad (3)$$

where $\alpha=1/(1+\mu)$, $\mathbf{I}_n$ is an identity matrix with $n\times n$, and $\mathbf{S}=\mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$ is the symmetrical normalization of **W**. In large scale problems, we prefer to use the iteration scheme:

$$\mathrm{f}(t+1)=\alpha\mathbf{S}\,\mathrm{f}(t)+(1-\alpha)\mathbf{y}\,. \quad (4)$$

During each round of iteration, each data point receives information from its neighbors (first term). And retains its initial information (second term). The iteration process is repeated until convergence.

As illustrated by Eq. (3) and (4), one of the key issues is to design an appropriate $\mathbf{S}$, and more precisely to design $\mathbf{W}$, which depends on two key parameters: the number of the nearest neighbors *k* used for constructing *k*NN graph and the scale parameter σ used by Gaussian kernel. We will discuss how to tune the parameters in the following subsections.

2.2 Constructing an Elastic *k*NN Graph

Constructing an appropriate graph is one of the keys to develop a high-performance MR scheme. As mentioned, the *k*NN graph is a popularly used structure, but it is prone to exploit "unreliable" unlabeled images, as illustrated by Figure 1. To "safely" exploit unlabeled images, we expect that the constructed graph could dynamically update the *k* value in a query session, in order to maintain a relatively confidential connecting relationship between each labeled image and its nearby unlabeled neighbors. To this purpose, for each image $\mathbf{x}_i$, we suggest using a large *k* in our approach when its most neighbors are inside its "trusted" interval because corresponding (unlabeled) nearby neighbors are "reliable" in this case. Conversely, a small *k* is preferable in order to reduce the likelihood of exploiting the "unreliable" neighbors. At worst, no unlabeled images are considered and our Simar approach will degenerate to a supervised ranking method. In this way, we can guarantee that our semi-supervised ranking method will never worse than a supervised one.

Given the labeled image set, a challenge is to probe whether most of their (unlabeled) nearby neighbors are inside the corresponding "trusted" intervals, since the images are not indexed by semantic in real-world applications. Considering this, Simar adopts an indirect strategy, that is, in each round of feedback the "reliability" of the unlabeled images used by current ranker is evaluated by monitoring the changes in its retrieval performance. Concretely, the retrieval performance is measured by using the precision rate defined as $Precision = \frac{\text{Number of postive retrievals}}{\text{Number of total retrievals}} \times 100\%$ via the user's feedback on image retrievals. If the current precision $Precision_{cur}$ is greater than the previous precision $Precision_{pre}$, then the "reliability" of the unlabeled images exploited currently is enhanced, and the value of k should be enlarged. On the other hand, if $Precision_{cur} < Precision_{pre}$, then it means that the "reliability" of the unlabeled images exploited currently is receded, and the value of k should be decreased. With these considerations, we adaptively tune the parameter k according to

$$k_{cur} = \text{floor}\left(k_{pre}\left(1 + Precision_{cur} - Precision_{pre}\right)\right). \tag{5}$$

to "safely" exploit the unlabeled images.

Note that the Precision mentioned here is calculated with the number of relevant images that appear in a fixed number of retrievals. Suggested by Luxberg [6], the initial value of k, used at the first round of feedback, is set to $\text{floor}(\log n)$ for the asymptotic connectivity purpose.

2.3 Local Scaling

As mentioned before, the performance of MR is sensitive to the scale parameter σ. Some previous works [3, 7 and 10] suggested running their MR algorithms repeatedly for a number of σ values and selecting the one leading to the highest average precision. However, the performance of this approach is heavily depended on the testing data and the range of values to be tested still has to be set manually. What is worse, there may not be a single value of σ that works well for all data points when the input data with different local statistics, which is the common case in the image database. Therefore, we try to address this shortcoming from a local scaling view, i.e. calculating a local scale parameter for each image, instead of selecting a single scale parameter for all images.

Inspired by the self-tuning spectrum clustering technique [12], the scale parameter can be regarded as some measure when two data points are considered similar. This provides an intuitive way for selecting possible σ. Let σ_i and σ_j denote the local scaling parameters of image $\mathbf{x}_i$ and $\mathbf{x}_j$ respectively. The distance from $\mathbf{x}_i$ to $\mathbf{x}_j$ as 'seen' by $\mathbf{x}_i$ can be defined as $d\left(\mathbf{x}_i,\mathbf{x}_j\right)/\sigma_i$ while the converse is $d\left(\mathbf{x}_j,\mathbf{x}_i\right)/\sigma_j$. Hence, the square distance d^2 between two images can be generalized as:

$$d\left(\mathbf{x}_i,\mathbf{x}_j\right)d\left(\mathbf{x}_j,\mathbf{x}_i\right)/\sigma_i\sigma_j = d^2\left(\mathbf{x}_i,\mathbf{x}_j\right)/\sigma_i\sigma_j , \tag{6}$$

and the weight of the edge between a pair of images, i.e. Eq. (1), can be rewritten as:

$$w_{ij} = \exp\left(-d^2\left(\mathbf{x}_i,\mathbf{x}_j\right)/\left(\sigma_i\sigma_j\right)\right). \tag{7}$$

Intuitively, a small σ_i is preferable when $\mathbf{x}_i$ is residing in a tight local region, while a large σ_i is preferable when $\mathbf{x}_i$ is residing in a sparse local region. To this purpose, the selection of the local scale σ_i can be done by studying the local statistics of the neighborhood of $\mathbf{x}_i$. Considering the efficiency, we use the distance from $\mathbf{x}_i$ to its *k*-th nearest neighbor $\mathbf{x}_{ik}$ to represent the local statistic, i.e. $\sigma_i = d(\mathbf{x}_i, \mathbf{x}_{ik})$ where $k = \text{floor}(\log n)$ that gave good result in our experiment.

2.4 Implementation Issues

For the real-time response purpose, previous work suggested using a sparse representation for the affine matrix $\mathbf{W}$ and calculating it off-line [3]. However, different from conventional MR, Simar requires updating matrix $\mathbf{W}$ on-line because elastic *k*NN graph is considered. Our idea is to calculate an initial affine matrix with a large *k* value off-line, and then add or remove elements into/from the matrix according to the changes of *k* values on-line. In the way, we can update matrix $\mathbf{W}$ with low computational cost. The key steps are summarized as follows.

Step 1 (off-line): Starting with a large k_0 (=100) value, for each image, we search its k_0 nearest neighbors from database and store their identities in a matrix $\mathbf{G} \in \mathbb{R}^{n \times k_0}$, where each element g_{ij} denotes the identity of the *j*-th nearest neighbor of image $\mathbf{x}_i$. Based on $\mathbf{G}$, the initial affinity matrix $\mathbf{W}_0$ is calculated by Eq. 7.

Step 2 (off-line): In the first round of feedback, given $k_1 = \text{floor}(\log n)$, the affinity matrix $\mathbf{W}_1$ (initialized by $\mathbf{0}^{n \times n}$) is generated based on $\mathbf{W}_0$ and $\mathbf{G}$ by copying elements from $\mathbf{W}_0$ to $\mathbf{W}_1$. For example, given image $\mathbf{x}_i$, the identity of its j-th nearest neighbor is g_{ij}, and corresponding affinity between $\mathbf{x}_i$ and its *j*-th nearest neighbor can be gained by: $\mathbf{W}_1(i, g_{ij}) \leftarrow \mathbf{W}_0(i, g_{ij})$.

Step 3 (on-line): After the second round of feedback, the affinity matrix $\mathbf{W}_{cur}$ used currently is updated by monitoring the changes of retrieval performance. In details, k_{cur} is first calculated by Eq. 5; then, based on $\mathbf{W}_0$ and $\mathbf{G}$, we add elements into $\mathbf{W}_{cur}$ when $k_{cur} > k_{pre}$, while remove elements from $\mathbf{W}_{cur}$ when $k_{cur} < k_{pre}$. The detailed updating rules can be described as:

```
if $k_{cur} > k_{pre}$ then  /* adding elements into $\mathbf{W}_{cur}$ */
  for $i = 1$ to $n$
     for $j = k_{pre} + 1$ to $k_{cur}$
        $\mathbf{W}_{cur}(i, g_{ij}) \leftarrow \mathbf{W}_0(i, g_{ij})$;
     end for
  end for
```

```
else  /* removing elements from $\mathbf{W}_{cur}$ */
  for $i=1$ to $n$
    for $j=k_{pre}$ to $k_{cur}+1$ with step=-1
      $\mathbf{W}_{cur}(i,g_{ij}) \leftarrow 0$ ;
    end for
  end for
end if
```

Another issue is with respect to the out-of-sample search. If the query image is not in the database, we first connect the query with its k_0 nearest neighbors from database images, meanwhile, add a new row to $\mathbf{G}$, with each element store the identify of the corresponding neighbor. Then, we calculate the edge weights by Eq. 7 and add one row and one column to $\mathbf{W}_0$, with each element equal to the corresponding edge weight. All the other operations will be performed similarly using the enlarged matrix $\mathbf{W}_0$ and $\mathbf{G}$.

3 Experimental Study

In this section, we show several experimental results and comparisons to evaluate the effectiveness of Simar scheme on a real world image database. All algorithms in experiments are implemented in MATLAB 2008 and run on a PC with Intel Core (TM) Duo 2.93 GHZ processor and 2GB RAM.

3.1 Experimental Setup

Experiments are performed on a set of 10,000 images picked from the Corel database. These images belong to 100 semantic classes, each of which has 100 images.

Three different features are used to represent the images, including a 64-dimensional color histogram, an 18-dimensional wavelet-based texture and a 5-dimensional edge direction histogram. At last, each image is represented as an 87-dimensional feature vector.

We use PR-graph and P@TopN to evaluate the effectiveness of image retrieval methods. PR-graph depicts the relationship between precision and recall of a specific retrieval method. In general, a PR-graph can also be summarized into one statistic value, i.e. MAP (mean average precision). However, PR-graph can hardly reflect the changes of retrieval performance caused by feedbacks directly. P@TopN emphasizes the retrieval performance at a particular scope N, which describes the relationship between precision and round of feedback at top N retrieval results. Thus it can compensate for the deficiency of PR-graph.

3.2 Comparison Methods

To examine the efficacy of the proposed Simar approach, several existing semi-supervised learning solutions for relevance feedback in CBIR are compared in our empirical study. (1) Conventional **MR** [3] serves as a baseline method that applies regular MR algorithm to learn a ranking function. The setting of parameters is consistent with [3], i.e. α=0.99, σ=0.05 and k=200. (2) **Co-training** [15] first trains two independent rankers using different distance metrics, and then each ranker labels for the other ranker its two most confident images from unlabeled data for the purpose of enriching the training set. (3) **SemiBoost** [8] iteratively learns an ensemble of SVMs using a similar procedure of boosting algorithm. In particular, both labeled and

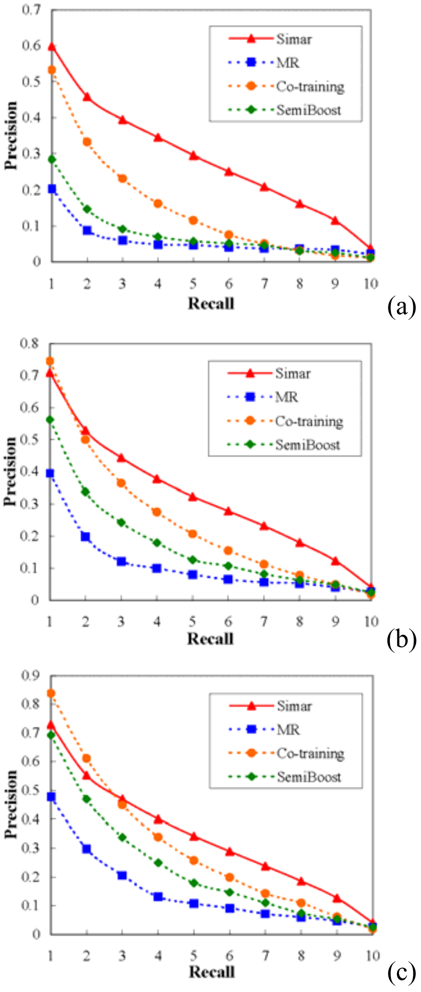


Fig. 2. PR-graphs of the proposed method compared with some existing methods at the (a) 1st, (b) 2nd and (c) 3rd round of feedback.

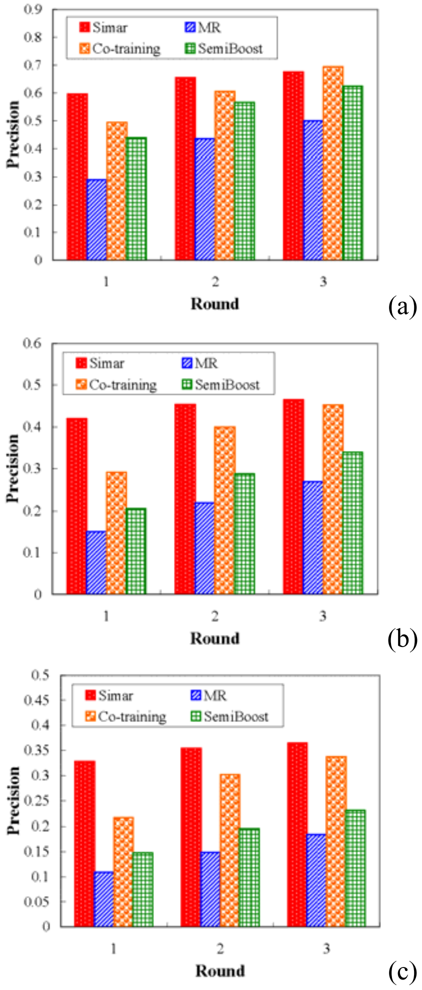


Fig. 3. Precisions of the proposed method compared with some existing methods at the Top (a) 20, (b) 60 and (c) 100 retrievals.

unlabeled images are exploited in the boosting procedure. The SVM is implemented using LIBSVM toolbox. Furthermore, in order to study whether the elastic *k*NN graph is useful, a degenerated variant of Simar, termed SimarDeg, is evaluated in the comparison. (4) **SimarDeg** is almost the same as Simar except that the former use the fixed *k*NN graph ($k = \text{floor}\left(\log n\right)$), instead of the elastic *k*NN graph, to calculate the Laplacian matrix.

3.3 Performance Evaluation

To evaluate the average performance, we conducted every experiment on a set of 200 random queries sampled from our image dataset. At the beginning of retrieval, the database images are ranked according to their Euclidean distances to the query image and top ten images are labeled as the initially labeled training data. Then, various methods are then applied to rerank the database images. For each compared method, after obtaining a query, several rounds of feedback were performed, and in each round the user labeled ten images as the feedback.

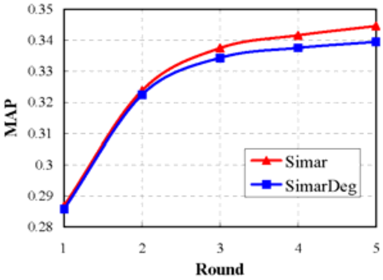


Fig. 4. MAP of the proposed method compared with its degenerated variant.

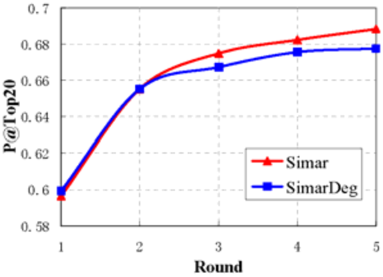


Fig. 5. P@Top 20 of the proposed method compared with its degenerated variant.

Table 1. MAPs of the four compared methods

	Simar	MR	Co-training	SemiBoost
Round 1	**0.287**	0.062	0.156	0.081
Round 2	**0.324**	0.113	0.25	0.177
Round 3	**0.338**	0.152	0.303	0.234

At first, the performance of Simar, MR, Co-training and SemiBoost are compared. The PR-graph at the 1st, 2nd, and 3rd round of feedback are shown in Figure 2, and the corresponding MAP statistic is tabulated in Table 1, where the best performance has been boldfaced. The precision curve at top 20, top 60, and top 100 retrieval results are presented in Figure 3. Several observations can be drawn from the experimental results. First, by comparing the two MR approaches, the performance of Simar is much better than conventional MR. Note that the main difference between them is that Simair calculates the Laplacian matrix using an adaptive scale parameter while conventional MR does this using a fixed scale parameter, which verifies the usefulness of our local scaling solution. Furthermore, in most cases, Simar outperforms Co-training and SemiBoost, especially at the first round of feedback, which is meaningful to the real world applications because it is not practical to require the user to provide many rounds of feedback and therefore the retrieval performance at the 1st round of feedback is the most important. Finally, it is impressive that at all rounds of feedback, the MAP of Simar is always the best. That means the MR approach is more effective than other semi-supervised ranking methods when the parameters are tuned appropriately.

In order to study whether the elastic *k*NN graph employed in our approach is beneficial or not, Simar is compared with its degenerated variant SimarDeg. Figure 4 and Figure 5 print the MAP and the P@Top20 of the two algorithms at 1st to 5th round of feedback, respectively. As can been seen, the performance of Simar and SimarDeg are close to each other at the first two rounds of feedback, and then Simar growingly outperforms SimarDeg with the increase of the rounds of feedback. It is conjectured that the number of labeled images is small at the first two rounds of feedback, the probability of exploiting the "unreliable" unlabeled images (the nearby neighbors of the labeled images) would be low, and thus the impact of the elastic *k*NN graph is trivial. By gradually adding the user's feedbacks, the elastic *k*NN graph is increasingly helpful to Simar.

4 Conclusions

In this paper, we presented a novel MR approach for relevance feedback in CBIR, which addressed the two main drawbacks of regular MR algorithm. In particular, we employed an elastic *k*NN graph in MR to reduce the risk of exploiting "unreliable" unlabeled data, and developed a local scaling solution to facilitate the setting of the scale parameter used for calculating Laplacian matrix. We conducted extensive experiments to evaluate the performance of our techniques for relevance feedback in CBIR, from which the promising results showed the advantages of the proposed approach in comparison to several existing methods. In the future work, we will take more visual features into consideration and evaluate our method on other databases.

Acknowledgements. This work was supported in part by the Fundamental Research Funds for the Central Universities (#2012JBM038 and #2012JBM035), the Natural Science Foundation of China (#61170232, #60973067 and #61175053), and the State Key Laboratory Research Funds #RS2012K011.

References

1. Chapelle, O., Scholkope, B., Zien, A.: Semisupervised Learning. MIT Press, Cambridge (2006)
2. Datta, R., Joshi, D., Li, J., Wang, J.Z.: Image Retrieval: Ideas, Influences, and Trends of The New Age. ACM Comput. Surv. 40(2), 5:1-5:60 (2008)

3. He, J., Li, M., Zhang, H., Tong, H., Zhang, C.: Manifold-Ranking Based Image Retrieval. In: Proc. ACM Int. Conf. Multimedia, MM (2004)
4. Hoi, S.C.H., Jin, R., Zhu, J., Lyu, M.R.: Semi-Supervised SVM Batch Mode Active Learning for Image Retrieval. In: Proc. IEEE Conf. Computer Vision and Pattern Recognition, CVPR (2008)
5. Li, Y.F., Zhou, Z.H.: Towards Making Unlabeled Data Never Hurt. In: Proc. Int. Conf. Machine Learning, ICML (2011)
6. Luxberg, U.: A Tutorial on Spectral Clustering. Statistics and Computing 17(4), 395–416 (2007)
7. Wang, B., Pan, F., Hu, K.M., Paul, J.C.: Manifold-Ranking Based Retrieval using k-Regular Nearest Neighbor Graph. Pattern Recognition 45(4), 1569–1577 (2012)
8. Wu, J., Lin, Z., Lu, M.: Asymmetric Semi-Supervised Boosting for SVM Active Learning in CBIR. In: Proc. ACM Int. Conf. Image and Video Retrieval, CIVR (2010)
9. Wu, J., Lu, M., Wang, C.: Collaborative Learning between Visual Content and Hidden Semantic for Image Retrieval. In: Proc. IEEE Int. Conf. Data Mining, ICDM (2010)
10. Xu, B., Bu, J., Chen, C., Cai, D., He, X., Liu, W., Luo, J.: Efficient Manifold Ranking for Image Retrieval. In: Proc. ACM Int. Conf. Research and Development in Information Retrieval, SIGIR (2011)
11. Yang, Y., Nie, F., Xu, D., Luo, J., Zhuang, Y., Pan, Y.: A Multimedia Retrieval Framework Based on Semi-Supervised Ranking and Relevance Feedback. IEEE Trans. Pattern Analysis and Machine Intelligence 34(4), 723–742 (2012)
12. Zelnik-Manor, L., Perona, P.: Self-Tuning Spectral Clustering. Adv. Neural. Inf. Process. Syst. (NIPS) 2 (2004)
13. Zhang, L., Wang, L., Lin, W.: Semisupervised Biased Maximum Margin Analysis for Interactive Image Retrieval. IEEE Trans. Image Processing 21(4), 2294–2308 (2012)
14. Zhou, X.S., Huang, T.S.: Relevance Feedback in Image Retrieval: A Comprehensive Review. Multimedia Syst. 8(6), 536–544 (2003)
15. Zhou, Z.H., Chen, K.J., Dai, H.B.: Enhancing Relevance Feedback in Image Retrieval using Unlabeled Data. ACM Transactions on Information Systems 24(2), 219–244 (2006)

Low-Rank Matrix Recovery with Discriminant Regularization

Zhonglong Zheng, Haixin Zhang, Jiong Jia, Jianmin Zhao,
Li Guo, Fangmei Fu, and Mudan Yu

Department of Computer Science, Zhejiang Normal University, Jinhua Zhejiang 321004, China

Abstract. Recently, image classification has been an active research topic due to the urgent need to retrieve and browse digital images via semantic keywords. Based on the success of low-rank matrix recovery which has been applied to statistical learning, computer vision and signal processing, this paper presents a novel low-rank matrix recovery algorithm with discriminant regularization. Standard low-rank matrix recovery algorithm decomposes the original dataset into a set of representative basis with a corresponding sparse error for modeling the raw data. Motivated by the Fisher criterion, the proposed method executes low-rank matrix recovery in a supervised manner, i.e., taking the with-class scatter and between-class scatter into account when the whole label information is available. The paper shows that the formulated model can be solved by the augmented Lagrange multipliers, and provide additional discriminating ability to the standard low-rank models for improved performance. The representative bases learned by the proposed method are encouraged to be structural coherence within the same class, and as independent as possible between classes. Numerical simulations on face recognition tasks demonstrate that the proposed algorithm is competitive with the state-of-the-art alternatives.

1 Introduction

With the ever-growing amount of digital image data in multimedia databases, there is a great requirement for algorithms that can provide effective semantic indexing. Categorizing digital images only using keywords is the quintessential, but not always executable example in image classification tasks. Face recognition (FR) is one typical image classification problem. Several aspects contribute to the difficulty of FR problem including the large variability in variance, illumination, pose, occlusion and even disguise of different subjects.

To design realistic FR systems, researchers usually focus on feature extraction of facial images and the generalization of classifiers. The testing sample from the same subjects will be used to evaluate the associated identification or verification performance. Although the testing sample might be corrupted, the training data sets are commonly assumed to be well taken in some desired conditions including reasonable illumination, pose, variations and without occlusion or disguise. When applying existing face recognition methods for practical scenarios, we will need to throw away the corrupted training images, and we might thus encounter small sample size and overfitting problems. Moreover, the disregard of corrupted training face images might give

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 437–448, 2013.

up some valuable information for recognition. Inspired by the sparse coding mechanism of human vision system [1][2], and with the rapid development of ℓ_1-norm minimization techniques in recent years, the sparse representation classification (SRC) ideas have been successfully used in various machine vision and pattern recognition applications [3][4][5][6]. Though interesting classification results have been reported in documentations, more investigations need to be made in order for a clearer understanding about the relationship between object representation and classification. Since SRC requires the training images to be well aligned for reconstruction purposes, [7] and [8] further extend it to deal with face misalignment and illumination variations. [5] also proposes modified SRC-based framework to handle outliers such as occlusions in face images. However, the above methods might not generalize well if both training and testing images are corrupted.

To address this issue, we propose formulating the face recognition problem under a matrix completion framework fueled by the recent advances in low-rank (LR) matrix recovery [9][10][11], together with the discriminant regularization denoted by within-class scatter and between-class scatter [12]. In this paradigm, low-rank matrix approximation is solved in a supervised manner as the whole label information of the training database is accessible. That is, we regularize the representative basis derived from standard LR matrix recovery using class-specific discriminant criterion which is motivated by Fisher criterion, and plays an important role in face recognition tasks [12][13][14]. By introducing this type of regularization, our matrix completion algorithm is able to capture discriminative portions extracted from different classes.

2 Related Works

2.1 Discrimination in Face Recognition

The face recognition literature is fairly dense and diverse and thus cannot be surveyed in its entirety in this limited space. In this paper, we focus on the class of face recognition approaches called subspace methods that are more closely related to our method. A prime instance of such methods is Eigenfaces [15], which attempts to group images by minimizing data variance. Fisherfaces [12], due to finding a subspace that minimizes the within-class distances while maximizing the between-class distances at the same time, achieves much better classification performance than Eigenfaces in face recognition problem. Some other subspace methods are geometrically inspired where the emphasis is on identifying a low dimensional sub-manifold on which the face images lie. The most successful of these methods include those which seek to project images to a lower dimensional subspace such that the local neighborhood structure present in the training set is maintained. These include Laplacianfaces [16], Locality Preserving Projections (LPP) [17], Orthogonal Laplacianfaces [18], Marginal Fisher Analysis (MFA)[19] etc.. Over time, improvements on discrimination of these methods have appeared in [20][21][22][23][24]. These generalizations seriously make the discriminant regularization as an indispensable part of their models, and therefore great improvements can be witnessed.

2.2 Sparse Representation-Based Classification

Recently, Wright et al [4] proposed a sparse representation-based classification algorithm for face recognition. In SRC-based algorithms, each testing image is regarded as a sparse linear combination of the whole training data by solving an ℓ_1 minimization problem, and very impressive results were reported in [4]. Several works have been proposed to further extend SRC-based algorithms for improved performance. For example, [25] utilizes a LASSO type regularization for computing the joint sparse representation of different features for visual signals. Jenatton et al. [26] utilizes a tree-structured sparse regularization for hierarchical sparse coding. Although promising face recognition results were reported by SRC-based algorithm, it still requires clean face images for training and thus might not be preferable for real-world scenarios. If corrupted training data is presented, SRC-based algorithms tend to recognize testing images with the same type of corruption and thus lead to poor performance. In the following section, we will introduce our proposed method for robust face recognition, in which both training and testing data can be corrupted.

2.3 Matrix Recovery via Rank Minimization

Low-rank matrix recovery is a procedure for reconstructing an unknown matrix with low-rank or approximately low-rank constraints from a sampling of its entries. This problem is motivated by the requirement of inferring global structure from a small number of local observations. [10], a breakthrough in matrix completion algorithms, states that the minimization of the rank function under broad conditions can be achieved using the minimizer obtained with the nuclear norm (sum of singular values). Since the natural reformulation of the nuclear norm gives rise to a semi-definite program, existing interior point methods can only handle problems with a number of variables in the order of the hundreds. Recently, Robust PCA method [9] has been proved to achieve the state-of-the-art performance using Augmented Lagrange Multipliers (ALM) method [11]. The proposed algorithm is also solved within the framework of ALM due to its fast efficiency. In the context of computer vision and pattern recognition, minimization of the nuclear norm in matrix completion has been applied to several problems: structure from motion [27], RPCA [9][28], subspace alignment [29], subspace segmentation [30] and signal denoising [31] etc..

3 Proposed Algorithm

3.1 Problem Setting

Given the original dataset $X = [x_1, x_2, \ldots, x_n] \in \mathbb{R}^{D \times n}$ consists of n columns, each column denotes a sample. Low-rank matrix recovery decomposes X into the following form

$$X = A + E, \tag{1}$$

where A is a low-rank matrix, and E is a sparse matrix. The dimension of matrices A and E is the same as X. According to [10], the solution of eq(1) can be solved by ALM

[11] method by optimizing the following model

$$\arg\min_{A,E} ||A||_* + \lambda||E||_1, \quad s.t.\ X = A + E, \tag{2}$$

where $||||_*$ denotes nuclear norm, and $||||_1$ denotes ℓ_1 norm.

3.2 Within-Class and Between-Class Scatters

Assume that all the labels of data X are available. Specifically, let x_i^s denote the i-th sample of the s-th class. We derived with-in class scatter and between class scatter matrices in the following manner which is different from Fisherfaces [12].

Let w_s denote the within-class scatter of class s. Define it as

$$w_s = \sum_{i=1}^{c_s} ||x_i^s - \bar{x^s}||_2^2, \quad s = 1, \ldots, c. \tag{3}$$

Let $X_s = [x_1^s, x_2^s, \ldots, x_{c_s}^s]$ denote the s-th class data matrix, c_s is the number of samples in class s, and e_{c_s} denote all-one column vector of length c_s. Then we have $\bar{x^s} = \frac{1}{c_s} X_s e_{c_s}$. Rewriting eq(3) shows

$$\begin{aligned} w_s &= \sum_{i=1}^{c_s} (x_i^s - \bar{x^s})(x_i^s - \bar{x^s})^T \\ &= Tr\{\sum_{i=1}^{c_s} x_i^s (x_i^s)^T\} - 2Tr\{\sum_{i=1}^{c_s} x_i^s (\frac{1}{c_s} X^s e_{c_s})^T\} + Tr\{(\frac{1}{c_s} X^s e_{c_s})(\frac{1}{c_s} X^s e_{c_s})^T\} \\ &= Tr(X_s X_s^T) - \frac{2}{c_s} Tr\{X_s e_{c_s} (e_{c_s})^T X_s^T\} + \frac{(e_{c_s})^T e_{c_s}}{c_s^2} Tr\{X_s e_{c_s} (e_{c_s})^T X_s^T\}, \end{aligned} \tag{4}$$

where Tr denotes trace operator of matrix. Thus we have

$$w_s = Tr\{X_s D_s X_s^T\}, \tag{5}$$

where $D_s = I_s - \frac{2}{c_s} e_{c_s} (e_{c_s})^T + \frac{(e_{c_s})^T e_{c_s}}{c_s^2} e_{c_s} (e_{c_s})^T$.

Next, we can define the between-class scatter of s-th class with the other classes

$$\beta_s = \sum_{j=1, j\neq s}^{c} ||\bar{x^s} - \bar{x^j}||_2^2, \tag{6}$$

where c is the number of classes. Following similar formulations from eq(3) to (5), we can rewrite eq(6) as

$$\begin{aligned} \beta_s &= \sum_{j=1, j\neq s}^{c} (\bar{x^s} - \bar{x^j})(\bar{x^s} - \bar{x^j})^T \\ &= \sum_{j=1, j\neq s}^{c} Tr\{\bar{x^s}(\bar{x^s})^T - 2\bar{x^s}(\bar{x^j})^T + \bar{x^j}(\bar{x^j})^T\} \\ &= \frac{c-1}{c_s^2} Tr\{X_s e_{c_s} (e_{c_s})^T X_s^T\} - 2Tr\{\bar{x^s} \sum_{j=1, j\neq s}^{c} \bar{x^j}\} + Tr\{\sum_{j=1, j\neq s}^{c} \bar{x^j}(\bar{x^j})^T\} \\ &= Tr\{X_s B_1 X_s^T\} - Tr\{X_s B_2\} + B_3, \end{aligned} \tag{7}$$

where $B_1 = \frac{c-1}{c_s^2} e_{c_s}(e_{c_s})^T$, $B_2 = \frac{2}{c_s} e_{c_s} \sum_{j=1,j\neq s}^{c} \bar{x^j}$ and $B_3 = Tr\{\sum_{j=1,j\neq s}^{c} x^j (x^j)^T\}$.

3.3 Low-Rank Matrix Recovery Discrimination

Although low-rank matrix recovery decomposes the original data X and produces a low-rank matrix A together with a sparse error matrix E for better representation purpose, as shown in eq(1), the derived low-rank matrix A might not contain sufficient discriminating information. Assume that the original X represents face image data, we can rewrite it into class-wise form $X = [X_1, X_2, \ldots, X_c]$.

Based on the within-class scatter and between-class scatter matrices shown in eq(5) and (7), it is a natural idea of adding a discriminant regularization to the low-rank matrix recovery problem shown in eq(1)

$$\begin{array}{c} \arg\min_{A,E} \sum_{s=1}^{c} \{||A_s||_* + \lambda ||E_s||_1 + \gamma(w_s(A_s) - \beta_s(A_s))\} \\ s.t.\ X_s = A_s + E_s, \end{array} \tag{8}$$

which is a class-wise optimization problem. In eq(8), $w_s(A_s)$ and $\beta_s(A_s)$ are the within-class scatter and between-class scatter of s-th class, respectively. Like LDA or Fisherfaces [12], to make projected samples favor of classification in feature space, we expect that the samples within the same class cluster as close as possible and samples between classes separate as far as possible in the learned low-rank matrix A. The term $||A_s||_* + \lambda ||E_s||_1$ shown in eq(8) performs the standard low-rank decomposition of the data matrix X. The term $\gamma(w_s(A_s) - \beta_s(A_s))$ is our discriminant regularizer based on within-class and between-class scatters, which is penalized by the parameter γ balancing the low-rank matrix approximation and discrimination. We refer to eq(8) as low-rank matrix recovery with discriminant regularization.

Meanwhile, we can rewrite $(w_s(A_s) - \beta_s(A_s))$ into the following form

$$\begin{aligned} w_s(A_s) - \beta_s(A_s) &= Tr\{A_s D_s A_s^T\} - Tr\{A_s B_1 A_s^T\} + Tr\{A_s B_2\} - B_3 \\ &= Tr\{A_s(D_s - B_1)A_s^T\} + Tr\{A_s B_2\} - B_3 \\ &\leq ||A_s||_F ||(D_s - B_1)||_F ||A_s||_F + ||A_s||_* ||B_2||_2 - B_3 \\ &= b_1 < A_s, A_s > + b_2 ||A_s||_* - b_3, \end{aligned} \tag{9}$$

where

$$b_1 = ||(D_s - B_1)||_F,\ \ b_2 = ||B_2||_2\ \ and\ \ b_3 = B_3. \tag{10}$$

As b_3 is irrelevant to A_s, the optimization of eq(8) can be rewritten as

$$\begin{array}{c} \arg\min_{A_s,E_s} ||A_s||_* + \lambda ||E_s||_1 + \gamma(b_1 < A_s, A_s > + b_2 ||A_s||_*) \\ s.t.\ X_s = A_s + E_s. \end{array} \tag{11}$$

The optimization of eq(11) can be solved by ALM [11]. The general method of ALM is introduced for solving the following constrained optimization problem

$$\min f(X) \quad s.t. \quad h(X) = 0. \tag{12}$$

The corresponding ALM function of eq(12) is defined as

$$L(X, Y, \mu) = f(X) + < Y, h(X) > + \frac{\mu}{2} ||h(X)||_F^2, \tag{13}$$

Algorithm 1 General Method of ALM

```
$\rho \geq 1$.
while not converged do
    solve $X_{k+1} = \arg\min_X L(X_k, Y_k, \mu_k)$
    $Y_{k+1} = Y_k + \mu_k h(X_k)$
update $\mu_k$ to $\mu_{k+1}$
end while
Output: $X_k$
```

where Y is a Lagrange multiplier matrix and μ is a positive scalar. The solution to eq(13) is outlined as Algorithm1.

In the proposed eq(11), let $X = (A_s, E_s)$, then

$$\begin{aligned} f(X) &= \|A_s\|_* + \lambda\|E_s\|_1 + \gamma(b_1 < A_s, A_s > +b_2\|A_s\|_*), \\ h(X) &= X_s - A_s - E_s \end{aligned} \tag{14}$$

respectively. The ALM function of our eq(11) is

$$\begin{aligned} L(A_s, E_s, Y_s, \mu, \gamma) = \|A_s\|_* + \lambda\|E_s\|_1 + \gamma(b_1 < A_s, A_s > +b_2\|A_s\|_*) \\ + < Y_s, X_s - A_s - E_s > + \frac{\mu}{2}\|X_s - A_s - E_s\|_F^2. \end{aligned} \tag{15}$$

To solve eq(15), we can optimize A_s, E_s and Y_s iteratively.

- Updating A_s:
  When updating A_s, we have to fix E_s and Y_s to solve the following problem based on eq(15), and the 3rd iteration of Algorithm(1) evolves

$$\begin{aligned} A_s^{k+1} &= \arg\min_{A_s^k} L(A_s^k, E_s^k, Y_s^k, \mu^k, \gamma) \\ &= \arg\min_{A_s^k} (1 + b_2)\|A_s^k\|_* + (\gamma b_1 + \frac{\mu^k}{2}) < A_s, A_s > +\mu^k < X_s^k - E_s^k + \frac{1}{\mu^k} Y_s^k, A_s^k > \\ &= \arg\min_{A_s^k} \epsilon\|A_s^k\|_* + \frac{1}{2}\|X_a - A_s^k\|_F^2, \end{aligned} \tag{16}$$

  where $\epsilon = \frac{1+b_2}{2\gamma b_1 + \mu^k}$ and $X_a = \frac{\mu^k}{2\gamma b_1 + \mu^k}(X_s^k - E_s^k + \frac{1}{\mu^k} Y_s^k)$. Introducing the following soft-thresholding operator

$$S_\epsilon[x] \doteq \begin{cases} x - \epsilon, if \quad x > \epsilon \\ x + \epsilon, if \quad x < -\epsilon \\ 0, \qquad otherwise \end{cases}, \tag{17}$$

  then we have the solution of eq(16) [11]

$$A_s^{k+1} = US_s[S]V^T, \tag{18}$$

  where USV^T is the SVD of X_a.

- Updating E_s:
  When updating E_s, we have to fix A_s and Y_s. The eq(15) can be derived as

$$E_s^{k+1} = \arg\min_{E_s^k} \eta \|E_s^k\|_1 + \frac{1}{2}\|X_e - E_s^k\|_F^2, \tag{19}$$

  where $\eta = \lambda \frac{1}{\mu^k}$ and $X_e = \frac{\mu^k}{2\gamma b_1 + \mu^k}(X_s^k - A_s^{k+1} + \frac{1}{\mu^k} Y_s^k)$.

Once we obtain A_s and E_s, Y_s can be updated using the 4th iteration of Algorithm1. The whole method we proposed is described in Algorithm2.

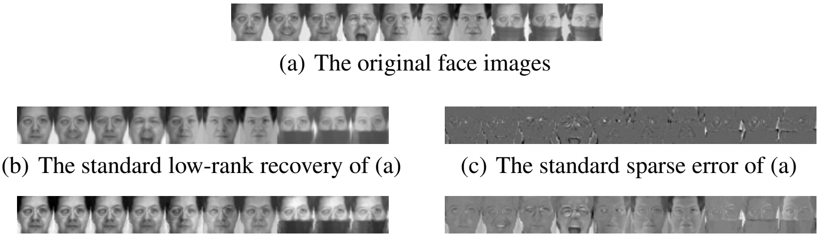
Algorithm 2 Low-rank Matrix Recovery with Discrimination

```
Input observation matrix $X$, $\lambda$.
Input $\mu_0 > 0$, $\rho > 1$ and $\eta$.
Compute $Y_0^* = sgn(X)/J(sgn(D))$.
while not converged do
    $A_0^{k+1} = A_*^k$, $E_0^{k+1} = E_*^k$, $j = 0$ and $b_{1,2}$ shown in eq(10);
    while not converged do
        $(U, S, V) = svd(X_s^k - E_s^k + \frac{1}{\mu^k} Y_s^k)$;
        $A_{j+1}^{k+1} = U S_\epsilon[S] V^T$;
        $E_{j+1}^{k+1} = S_\eta(X_s^k - A_s^{k+1} + \frac{1}{\mu^k} Y_s^k)$;
        $j \leftarrow j + 1$
    end while
    $Y_{k+1}^* = Y_k^* + \mu_k(X_s - A_*^{k+1} - E_*^{k+1})$
    update $\mu_k$ to $\mu_{k+1}$
end while
Output: $(A_*^k, E_*^k)$.
```

3.4 LR with Discrimination for Face Recognition

Occlusion is a common challenging encountered in face recognition tasks, such as eyeglasses, sunglasses, scarves and some objects placed in front of the faces. Moreover, even in the absence of an occluding object, violations of an assumed model for face appearance may act like occlusions: e.g., shadows due to extreme illumination. Robustness to occlusion is therefore essential to practical face recognition system. If the face images are partially occluded, popular recognition methods based on holistic features such Eigenfaces [15], Fisherfaces [12] and Laplacianfaces [16] would lead to unacceptable performance due to the corruption of the extracted features. Although SRC-based algorithm [28] achieves better results in recognizing occluded testing images, it still requires unoccluded face images for training and thus might not be preferable for real application scenarios.

Low-rank matrix recovery has been applied to alleviate the aforementioned problems by decomposing the collected data matrix into two different parts, one is a representation basis matrix of low rank and the other is the corresponding sparse error, as shown in Fig.1.



(a) The original face images

(b) The standard low-rank recovery of (a)

(c) The standard sparse error of (a)

(d) The low-rank recovery of our method

(e) The sparse error of our method

Fig. 1. The results of low-rank matrix recovery with and without discrimination

We can find out from Fig.1 that when the standard low-rank matrix recovery is combined with discrimination, the face images within the recovered representation basis matrix tend to be more similar to each other for the same subject, which means more compactness exists within the same classes and dissimilarity between different classes. In addition, we also can conclude from Fig.1 that the sparse error with discrimination can remove more sparse noise. As a result, the representation basis matrix of low-rank recovery with discrimination has a better representative ability than the original version. Since the face images usually lie in high dimensional spaces, traditional dimensionality reduction techniques, like PCA or LDA, can be performed on the recovered representation basis matrix. As a result, the derived subspace can be applied as the dictionary for training and the testing purposes. In the recognition stage, one can also use SRC-based classification strategy to identify the input image. Our scheme for face recognition is described as Algorithm3.

Algorithm 3 LR with Discrimination for Face Recognition

```
Input training data $X = [X_1, X_2, \ldots, X_c]$ and a testing image $y$.
Use Algorithm2 on $X$ to compute the representation basis matrix $A$.
Calculate the projection matrix of $P$ of $A$.
Compute the projection of $X$ and $y$:
$X_p = P^T X$, and $y_p = P^T y$.
Perform SRC-based classification on $y_p$:
$\arg\min_\alpha \|y_p - X_p\alpha\|_2^2 + \lambda\|\alpha\|_1$,
for $i = 1 : c$
    $err(i) = \|y_p - X_p^i \alpha_i\|_2^2$
end for
Output: $label(y) = \min_i err(i)$.
```

4 Experiments

In this section, we perform the proposed method shown in Algorithm3 on publicly available databases for face recognition to demonstrate the efficacy of the proposed classification algorithm. We will first examine the role of feature extraction within our framework, comparing performance across various feature spaces and feature dimensions, and comparing to several popular methods. Meanwhile, We will then demonstrate the robustness of the proposed algorithm to corruption and occlusion. Finally, the experimental results demonstrate the effectiveness of sparsity as a means of validating testing images.

Besides the standard low-rank matrix recovery without discrimination and our proposed method, we also consider Nearest Neighbor (NN), SRC [4], and LLC [32] for comparisons. Note that LLC can be regarded as an extended version of SRC exploiting data locality for improved sparse coding, and the classification rule is the same as that of SRC. To evaluate our recognition performance using data with different dimensions, we project the data onto the eigenspace derived by PCA using our LR with discrimination models. For the standard LR approach, the eigenspace spanned by LR matrices without discrimination is considered, while those of other SRC based methods are derived by the data matrix X directly. We vary the dimension of the eigenspace and compare the results in this section.

4.1 Two Databases

- The Extended Yale B database consists of $2,414$ frontal face images of 38 individuals around $59-64$ images for each person [33]. The cropped and normalized 192×168 face images were captured under various laboratory-controlled lighting conditions. Some sample face images of the Extended Yale B are shown in Fig.2(a).
- The AR database consists of over $4,000$ frontal images for 126 individuals [34]. For each individual, 26 pictures were taken in two separate sessions. In Fig.2(b), the left and right ones are some images collected in two sessions.

(a) (b)

Fig. 2. Some sample images of Extended Yale B and AR

4.2 Results

On the Extended Yale B Database. For each subject, we randomly select 10, 20 and 30 images of each subject for training respectively, and the left images for testing. Randomly choosing the training set ensures that our results and conclusions will not depend on any special choice of the training data. We vary the dimension of the eigenspace as 25, 50, 75, 100, 150, 200, 300 and 400 to compare the recognition performance between

Table 1. 10 training samples

METHOD	25	50	75	100	150	200	300
LR+SRC	67.3	75.7	79.1	82.2	83.4	84.6	86.8
SRC	61.8	69.6	76.9	82.1	83.3	85.8	86.2
LLC+SRC	44.6	62.5	68.7	73.4	75.7	77.3	78.6
NN	30.7	42.8	45.6	49.7	53.4	56.1	58.3
OURS	72.4	77.2	81.4	83.8	86.5	86.9	86.8

Table 2. 20 training samples

METHOD	25	50	75	100	150	200	300
LR+SRC	75.4	82.3	85.7	87.2	89.6	90.5	91.9
SRC	67.3	74.9	80.6	85.8	87.2	90.5	91.1
LLC+SRC	51.7	66.8	72.4	78.3	81.6	85.8	87.2
NN	38.1	45.7	52.2	58.4	62.4	66.3	69.8
OURS	82.4	84.6	86.3	89.1	92.9	93.1	93.5

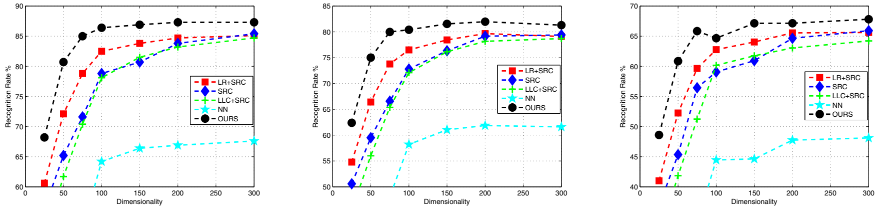
Table 3. 30 training samples

METHOD	25	50	75	100	150	200	300
LR+SRC	86.9	93.3	94.7	95.4	96.1	96.5	96.4
SRC	80.3	88.6	92.9	94.5	95.4	96.1	96.6
LLC+SRC	62.4	79.6	86.4	89.7	92.5	93.8	94.5
NN	45.1	56.0	63.3	66.8	69.4	73.2	76.7
OURS	90.5	94.3	95.8	96.2	97.1	97.5	97.5

different methods. All experiments run ten times and the average results are shown in Table1-3.

It is clear from those Tables mentioned above that the proposed method consistently achieves higher recognition rates than other NN and SRC-based approaches. For example, at dimension = 100, our method achieves a better recognition rate at 96.2%, and those for LR, SRC, LLC, and NN are 95.4%, 94.5%, 89.7%, and 66.8%, respectively (see Table3). Repeating the above experiments using different training images for each person, we can confirm from these empirical results that the use of LR method alleviates the problem of severe illumination variations even when such noise is presented in both training and testing data. Furthermore, when discrimination is taken into account as proposed in the paper, LR method exhibits enhanced classification capability and thus outperforms the standard LR algorithm.

On AR Database. In the experiment, a subset of the dataset consisting of 50 male subjects and 50 female subjects was chosen. The images are cropped with dimension 165 × 120. Different from [4], for each subject, both neutral (four neutral faces with different lighting conditions and three faces with different expressions) and corrupted images (three faces with sunglasses and three faces with scarfs) taken at session 1 are used for training, and session 2 for testing. Specifically, we consider the following sample selection for training: 7 neutral images plus 3 sunglass images; 7 neutral images plus 3 scarf images; 7 neutral images plus 3 sunglass images and 3 scarf images. We vary the dimension of the eigenspace as 25, 50, 75, 100, 150, 200, 300 and 400 to compare the recognition performance between different methods. The experimental results are visualized in Fig.3.



(a) (b) (c)

Fig. 3. Recognition rate on AR. (a)7 neutral + 3 sunglass images. (b)7 neutral + 3 scarf images. (c)session 1 as training set.

From these three figures, we see that the proposed method outperforms all other algorithms across different dimensions. It is worth noting that with the increase of occlusion (from sunglass ro scarf), the recognition rates of all the approaches are severely degraded, which can be seen from Fig.3(a) and Fig.3(b). In addition, with the increase of occluded images in the training set, the performances of all the approaches are also severely degraded which can be seen from Fig.3(c). These two cases indicate that the direct use of corrupted training image data will remarkably make the recognition results worse.

5 Conclusions

In this paper, a low-rank matrix recovery algorithm with discriminant regularization is proposed. The discrimination regularizer is motivated by Fisher criterion which plays an important role in classification tasks. The introduction of this kind of regularizer into low-rank matrix recovery promotes the discrimination power in the learned representation basis. We also show that the proposed optimization algorithm can be formulated by augmented Lagrange multipliers. When applied to face recognition problem, the proposed algorithm demonstrates robustness to severe occlusions of face images even in the training set. The experiments has shown that our method achieves the state-of-the-art recognition results.

Acknowledgments. The authors confirm that the research was supported by National Natural Science Foundation (No.61170109, No.61100119 and No.61272468), and Science and Technology Planning Project of Zhejiang Province (No.2012C21021), China.

References

1. Olshausen, B.A., Field, D.J.: Sparse coding with an over-complete basis ser: a strategy employed by v1? Vision Research 37(23), 3311–3325 (1997)
2. Vinje, W.E., Gallant, J.L.: Sparse coding and decorrelation in primary visual cortex during natural vision. Science 287(5456), 1273–1276 (2000)
3. Wright, J., Ma, Y., Mairal, J., Spairo, G., Huang, T., Yan, S.C.: Sparse representation for computer vision and pattern recognition. Proceedings of the IEEE 98(6), 1031–1044 (2010)
4. Wright, J., Yang, A.Y., Sastry, A.G.S.S., Ma, Y.: Robust face recognition via sparse representation. IEEE PAMI 31(2), 210–227 (2009)
5. Yang, M., Zhang, L., Yang, J., Zhang, D.: Robust sparse coding for face recognition. In: CVPR (2011)
6. Huang, K., Aviyente, S.: Sparse representation for signal classification. In: NIPS (2006)
7. Wagner, A., Wright, J., Ganesh, A., Zhou, Z.H., Ma, Y.: Towards a practical face recognition system: Robust registration and illumination by sparse representation. In: CVPR (2009)
8. Wagner, A., Wright, J., Ganesh, A., Zhou, Z.H., Ma, Y.: Towards a practical face recognition system: Robust registration and illumination by sparse representation. IEEE PAMI 34(2), 372–386 (2012)
9. Candes, E., Li, X., Ma, Y., Wright, J.: Robust principal component analysis? Journal of ACM 58(1), 1–37 (2009)

10. Candes, E., Recht, B.: Exact low rank matrix completion via convex optimization. In: Allerton (2008)
11. Lin, Z., Chen, M., Ma, Y.: The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrix. UIUC Technical Report UILU-ENG-09-2215 (2009)
12. Belhumeur, P.N., Hespanha, J.P., Kriegman, D.J.: Eigenfaces vs fisherfaces: recognition using class specific linear projection. IEEE PAMI 19(7), 711–720 (1997)
13. Li, Z., Lin, D., Tang, X.: Nonparametric discriminant analysis for face recognition. IEEE PAMI 31(4), 755–761 (2009)
14. Lu, J., Tan, Y., Wang, G.: Discriminaive multi-manifold analysis for face recognition from a single trainning sample per person. IEEE PAMI pp(99), 1 (2012)
15. Turk, M., Pentland, A.: Eigenfaces for recognition. Journal of Cognitive Neuroscinces 3, 72–86 (1991)
16. He, X., Yan, S., Hu, Y., Niyogi, P., Zhang, H.: For recognition using laplacianfaces. IEEE PAMI 27(3), 328–340 (2005)
17. He, X., Cai, D., Niyogi, P.: Locality preserving projections. In: NIPS (2003)
18. Cai, D., He, X., Han, J., Zhang, H.: Orthogonal laplacianfaces for face recognition. IEEE TIP 15(11), 3608–3614 (2006)
19. Yan, S., Xu, D., Zhang, B., Zhang, H., Yang, Q., Lin, S.: Graph embedding and extension: A general framework for dimensionality reduction. IEEE PAMI 29(1), 40–51 (2007)
20. Hua, G., Viola, P., Drucker, S.: Face recognition using discriminatively trained orthogonal rank one tensor projections. In: CVPR (2007)
21. Xue, H., Chen, S., Yang, Q.: Discriminatively regularized least-squares classification. Pattern Recognition 42(1), 93–104 (2009)
22. Si, S., Tao, D., Geng, B.: Bregman divergence-dased regularization for transfer subspace learning. IEEE TKDE 22(7), 929–942 (2010)
23. Lu, J., Tan, Y.: Cost-sensitive subspace learning for face recognition. In: CVPR (2010)
24. Lu, J., Tan, Y.: Regularized locality preserving projections and its extensions for face recognition. IEEE SMCB 40(3), 958–963 (2010)
25. Yuan, X., Yan, S.: classification with multi-task joint sparse representation. In: CVPR (2010)
26. Jenatton, R., Mairal, J., Obozinski, G., Bach, F.: Proximal methods for hierarchical sparse coding. Journal of Machine Learning Research 12, 2297–2334 (2011)
27. Cabral, R., Costeira, J., Torre, F., Bernardino, A.: Fast incremental method for matrix completion: an application to trajectory correction. In: ICIP (2011)
28. Wright, J., Ganesh, A., Rao, S., Ma, Y.: Robust principal component analysis: exact recovery of corrupted low rank matrices by convex optimization. In: NIPS (2009)
29. Peng, Y., Ganesh, A., Wright, J., Xu, W., Ma, Y.: Robust alignment by sparse and low-rank decomposition for linearly correlated images. In: CVPR (2010)
30. Liu, G., Lin, Z., Yu, Y.: Robust subspace segmentation by low-rank representation. In: ICML (2010)
31. Ji, H., Liu, C., Shen, Z., Xu, Y.: Robust video denoising using low rank matrix completion. In: CVPR (2010)
32. Wang, J., Yang, J., Yu, K., Lv, F., Huang, T., Gong, Y.: Locality constrained linear coding for image classification. In: CVPR (2010)
33. Georghiades, A., Belhumeur, P., Kriegman, D.: From few to many: Illumination cone models for face recognition under variable lighting and pose. IEEE PAMI 23(6), 643–660 (2001)
34. Martinez, A., Benavente, R.: The ar face database. CVC Technical Report 24 (1998)

Multi-Manifold Ranking: Using Multiple Features for Better Image Retrieval

Yang Wang[1,2], Muhammad Aamir Cheema[1], Xuemin Lin[1], and Qing Zhang[2,1]

[1] The University of New South Wales, Sydney, Australia
{wangy,macheema,lxue}@cse.unsw.edu.au
[2] Australian E-Health Research Center
Qing.Zhang@csiro.au

Abstract. Manifold Ranking (MR) is one of the most popular graph-based ranking methods and has been widely used for information retrieval. Due to its ability to capture the geometric structure of the image set, it has been successfully used for image retrieval. The existing approaches that use manifold ranking rely only on a single image manifold. However, such methods may not fully discover the geometric structure of the image set and may lead to poor precision results. Motivated by this, we propose a novel method named **Multi-Manifold Ranking** (MMR) which embeds multiple image manifolds each constructed using a different image feature. We propose a novel cost function that is minimized to obtain the ranking scores of the images. Our proposed multi-manifold ranking has a better ability to explore the geometric structure of image set as demonstrated by our experiments. Furthermore, to improve the efficiency of MMR, a specific graph called anchor graph is incorporated into MMR. The extensive experiments on real world image databases demonstrate that MMR outperforms existing manifold ranking based methods in terms of quality and has comparable running time to the fastest MR algorithm.

Keywords: Image retrieval, integrated features, manifold ranking.

1 Introduction

Traditional image retrieval techniques rely on the semantic labels attached to the images such as image annotations [13] and tags [7]. However, a severe drawback of such techniques is that the manual labelling is laborious, expensive and time-consuming. Another disadvantage is that such techniques do not consider the content of the images and this may lead to poor results especially if the quality of the labelling is poor.

To address the issues mentioned above, content-based image retrieval (CBIR) [10,5,12] may be used which utilizes the low-level features (e.g., color, shape, texture) for image retrieval. These low-level features can be extracted automatically and remain consistent for each image in contrast to the manually attached labels. However, it is difficult to choose an ideal descriptor for the images because the low-level features may not represent the same semantic concepts. For example, two images having similar color visualization may have totally different

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 449–460, 2013.

semantic meanings (e.g., a green apple and a tennis ball as shown in Fig. 1). This is one of the main challenges CBIR needs to address.

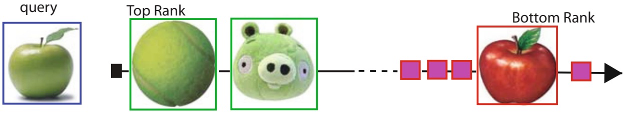


Fig. 1. If only the color feature is used, the most relevant results include a green tennis ball and a green angry bird instead of the red apple. Hence, a single feature may not provide desired results.

To address this challenge, He *et al.* [4] used manifold ranking that uses low-level features as well as the intrinsic structure of the images. The basic idea behind the manifold ranking is as follows. A weighted graph is constructed where the vertices represent the images and, for each vertex, its near by vertices are connected to it by weighted edges. The queries are assigned a positive ranking and the remaining vertices are ranked with respect to the queries. The vertices spread their ranking scores to their neighbors via the weighted graph. The spread process is repeated until convergence. This approach has been shown to yield better retrieval results because it utilizes the intrinsic structure of the image set. Xu *et al.* [18] proposed a faster manifold ranking approach that uses anchor graphs [8] to approximate the original graph and provides the results of similar quality.

The above mentioned manifold ranking techniques use a single feature. In other words, these techniques utilize the intrinsic structure of the images based only on a single feature. The ranking based on the single manifold may have low precision especially if the selected feature is not very representative. Motivated by this, in this paper, we propose a technique called multi-manifold ranking (MMR) that ranks the images by considering multiple manifolds each constructed using a different feature. MMR demonstrates excellent ability to retrieve relevant images because it considers multiple intrinsic structures of the images. We propose a novel cost function that is minimzed to obtain the ranking scores of the images. Our proposed approach provides better results than the existing techniques. Furthermore, we present efficient techniques to create the multiple manifolds.

We remark that Huang *et al.* [6] also utilizes more than one low-level features. However, they construct only one manifold by using average manifold distance of multiple features. Since only a single manifold is used, the proposed approach does not preserve the original geometric structure of any of the features. In contrast, our approach constructs multiple manifolds and utilizes the geometric structure of each feature. This enables our approach to yield better results as demonstrated in our experiments. Furthermore, we show that our proposed approach is more efficient and can be used on large image databases.

Our contributions in this paper are summarized below.

- We propose multi-manifold ranking (MMR) that utilizes multiple intrinsic structures of the images to provide a better ranking of the images.
- To handle large image databases, we improve the efficiency of MMR by using singular value decomposition [1] as well as anchor graphs.

• Our extensive experimental results on real world image databases demonstrate that our algorithm provides better retrieval results than state of the art existing techniques (MR [4], ADF [6] and EMR [18]) that use a single manifold for image retrieval. Furthermore, the running time of our algorithm is similar to that of EMR and is significantly lower than those of MR and ADF. We also present simple extensions of MR and EMR that use more than one manifolds. Although these extended versions demonstrate better retrieval results than the original versions, our proposed multi-manifold ranking performs significantly better.

The rest of the paper is organized as follows. Related work is reviewed in Section 2. The details of multi-manifold ranking are presented in Section 3. Extensive experimental study on real world image databases is presented in Section 4. Section 5 concludes the paper.

2 Related Work

Zhou *et al.* [20] explored the importance of intrinsic geometrical structure of the data. They propose manifold ranking [21] that considers the intrinsic structure of the data for the ranking. Manifold ranking has been successfully used on various data types such as text [15], image [4] and video [19]. He *et al.* [4] are the first to use manifold ranking for image retrieval. While the proposed approach demonstrates good quality results, it is computationally expensive. Xu *et al.* [18] propose a more efficient approach that can efficiently handle large image databases. They replace the original image graph with anchor graph [8] which is significantly smaller in size but provides the results of similar quality. Huang *et al.* [6] use a probabilistic hypergraph for image retrieval. They construct a single manifold using the average manifold distance of multiple features.

All of the above manifold ranking based approaches consider geometric structure of a single image manifold which may not precisely represent the image content. Motivated by this, we propose a multi-manifold ranking based method for image retrieval which exploits the geometric structure of multiple manifolds each constructed using a different feature. Our idea for MMR is inspired by [3], which addresses the problem of video annotation through multi-graph using different video features.

3 Multi-Manifold Ranking

3.1 Preliminaries

Let X be a set containing n images, i.e. $X = \{x_1, x_2, \cdots, x_n\}$. Multi-manifold ranking assigns each image x_i a ranking score F_i. $F = \{F_1, F_2, \cdots, F_n\}$ is the ranking score vector containing the score of each image. $L = \{L_1, L_2, \cdots, L_n\}$ is an indicator label vector where $L_i = 1$ if x_i is the query image, otherwise $L_i = 0$.

Multi-manifold ranking (MMR) constructs N graphs each using a different feature. G^k denotes a s-NN graph constructed on X using k^{th} feature. Specifically, G^k is constructed by connecting every two vertices x_i and x_j if one is among the s nearest neighbors of the other. Here, the nearest neighbors are computed using Euclidean distance between the k^{th} feature vectors of the images. The Euclidean distance between the k^{th} feature vectors of x_i and x_j is denoted as $||x_i, x_j||_k$.

W^k denotes the edge affinity matrix of G^k. Each entry W^k_{ij} in W^k represents the similarity between x_i and x_j according to the k^{th} feature vector. W^k_{ij} is defined by a Gaussian kernel and is set to $exp(-||x_i, x_j||^2_k/2\sigma^2)$ if there is an edge in G^k between x_i and x_j. Otherwise, W^k_{ij} is zero. D^k is the diagonal matrix of G^k where each element D^k_{ii} is defined as $D^k_{ii} = \sum_{j=1}^n W^k_{ij}$.

3.2 Objective Cost Function

In this section, we propose a novel cost function, inspired by [3], to obtain the ranking scores of the images in X. The cost function $O(F)$ considers N image manifolds each constructed using a different feature. The ranking score vector F is obtained by minimizing the cost function $O(F)$ given in Eq. 1.

$$O(F) = \frac{1}{2}\sum_{k=1}^{N}(\sum_{i,j=1}^{n} W^k_{ij}(\frac{1}{\sqrt{D^k_{ii}}}F_i - \frac{1}{\sqrt{D^k_{jj}}}F_j)^2 + \lambda\sum_{i=1}^{n}(F_i - L_i)^2) \tag{1}$$

The first term ensures that nearby points (i.e., similar images in the multiple image manifolds) are assigned similar ranking scores. The second term is the fitting constraint which ensures that the ranking results should fit the initial label assignment. λ is the regularization trade-off parameter for the fitting constraint.

We minimize $O(F)$ by setting $\frac{\partial O(F)}{\partial F} = 0$, which leads to the following equation.

$$\sum_{k=1}^{N}((I-(D^k)^{-\frac{1}{2}}W^k(D^k)^{-\frac{1}{2}})F+\lambda(F-L)) = \sum_{k=1}^{N}((1+\lambda)F-S^kF-\lambda L) = 0 \tag{2}$$

where $S^k = (D^k)^{-\frac{1}{2}}W^k(D^k)^{-\frac{1}{2}}$. Note that Eq. 2 is equivalent to the following equation.

$$\sum_{k=1}^{N}(F - \frac{S^k}{1+\lambda}F - \frac{\lambda}{1+\lambda}L) = 0 \tag{3}$$

Let $\alpha = \frac{1}{1+\lambda}$. Eq. 3 is equivalent to $\sum_{k=1}^{N}(I - \alpha S^k)F = N(1-\alpha)L$. Hence, the final optimal ranking score vector denoted by F^* can be obtained as follows.

$$F^* = (\sum_{k=1}^{N}(I - \alpha S^k))^{-1}N(1-\alpha)L \tag{4}$$

where I is the identity matrix. Since both (1 - α) and N remain the same for all the images, they do not affect the retrieval results. Therefore, F^* can be obtained as follows.

$$F^* = (\sum_{k=1}^{N}(I - \alpha S^k))^{-1}L \tag{5}$$

Eq. 5 is the closed form for the optimal solution F^*. In large scale problems, the *iteration scheme* is preferred [18]. Therefore, we also consider the iterative form which is given below.

$$F(t+1) = F(t) + \mu \sum_{k=1}^{N} (F(t) - S^k F(t) + \lambda(F(t) - L)) \tag{6}$$

where $F(t)$ is the ranking score vector at time stamp t. By setting $\mu = -\frac{1}{N(1+\lambda)}$, the following equation can be obtained.

$$F(t+1) = F(t) - \frac{1}{N(1+\lambda)} \sum_{k=1}^{N} ((1+\lambda)F(t) - S^k F(t) - \lambda L) = \frac{\sum_{k=1}^{N} (\alpha S^k F(t) + (1-\alpha)L)}{N} \tag{7}$$

Since N remains constant for all images, it is sufficient to consider the following equation which omits N.

$$F(t+1) = \sum_{k=1}^{N} (\alpha S^k F(t) + (1-\alpha)L) \tag{8}$$

The above iterative form can be used in the iterative scheme. During each iteration, each vertex (i.e., image) receives information from its neighbors (the first term) and retains its initial information (the second term). The iteration process is repeated until convergence. By following the arguments similar to [21], it can be shown that Eq. 8 is converged to the following equation when $F(0)$ is initialized to L.

$$F^* = \lim_{t \to \infty} F(t) = N(1-\alpha)(I - \alpha S)^{-1} L \tag{9}$$

Note that both N and $(1-\alpha)$ can be omitted from Eq. 9 without changing the final retrieval results because they are constant for all images. Therefore, the optimal ranking results can be obtained as follows.

$$F^* = \lim_{t \to \infty} F(t) = (I - \alpha S)^{-1} L \tag{10}$$

We remark that although Eq. 10 may assign negative scores to some of the images, the relative ranking order of the images is preserved. Nevertheless, if desired, the scores of all the images may be normalized (e.g., by shifting) such that each image gets a positive score.

3.3 Improving the Efficiency of MMR

The approach we mentioned in the previous section has two major limitations. Firstly, the time complexity for constructing the affinity matrix for n data points using s nearest neighbors is $\mathcal{O}(sn^2)$ [8]. Secondly, the inverse matrix computation in Eq. 5 requires $\mathcal{O}(n^3)$. Clearly, the cost of constructing the affinity matrix and inverse matrix computation is prohibitive for large image databases. Hence, this approach is not suitable for the large image databases.

The first limitation can be addressed by using anchor graphs [8] in a similar way as used in [18]. This reduces the cost from $\mathcal{O}(sn^2)$ to $\mathcal{O}(dmn)$ where $m \ll n$ and $d \ll n$. Next, we use singular decomposition to address the second limitation and reduce the cost from $\mathcal{O}(n^3)$ to $\mathcal{O}(m^3)$ where $m \ll n$.

Let I_r denote an identity matrix of size $r \times r$. Before we present the details of efficient matrix inversion, we prove that the following equation holds.

$$(NI_n - \alpha HH^T)^{-1} = \frac{I_n - H(H^TH - \frac{N}{\alpha}I_m)^{-1}H^T}{N} \tag{11}$$

Proof. We prove the correctness of the equation by showing that R.H.S. divided by L.H.S. equals to an identity matrix.

$$\begin{aligned} &(NI_n - \alpha HH^T)(\tfrac{I_n - H(H^TH - \frac{N}{\alpha}I_m)^{-1}H^T}{N}) \\ &= \tfrac{NI_n - \alpha HH^T - (NH - \alpha HH^TH)(H^TH - (\frac{N}{\alpha}I_m))^{-1}H^T}{N} \\ &= \tfrac{NI_n - \alpha HH^T + \alpha H(-\frac{N}{\alpha}I_m + H^TH)(H^TH - \frac{N}{\alpha}I_m)^{-1}H^T}{N} \\ &= \tfrac{NI_n - \alpha HH^T + \alpha HH^T}{N} = \tfrac{NI_n}{N} = I_n \end{aligned} \tag{12}$$

■

Based on Eq. 11, we show that the cost of the matrix operation can be reduced. Let H^k be defined as following.

$$H^k = (D^k)^{-\frac{1}{2}} Z^k (\Lambda^k)^{\frac{1}{2}} \tag{13}$$

The following equation can be verified.

$$H^k(H^k)^T = (D^k)^{-\frac{1}{2}} W^k (D^k)^{-\frac{1}{2}} \tag{14}$$

Recall that R.H.S. of Eq. 14 equals to S^k (see Eq. 2 in Section 3.2).

$$S^k = H^k(H^k)^T \tag{15}$$

We replace S^k in Eq. 5 with its value in Eq. 15 which yields the following.

$$F^* = (NI - \alpha \sum_{k=1}^{N} H^k(H^k)^T)^{-1} L \tag{16}$$

Note that each $H^k(H^k)^T$ is a symmetric matrix. Hence, $\sum_{k=1}^{N} H^k(H^k)^T$ is also a symmetric matrix. Without loss of generality, we set $S = \sum_{k=1}^{N} H^k(H^k)^T$ which is a $n \times n$ gram matrix.

$$F^* = (NI - \alpha S)^{-1} L \tag{17}$$

We decompose S by using singular value decomposition [1] $S = U\Lambda U^T$ such that $U^TU = I_n$. Note that the decomposition takes $\mathcal{O}(n^3)$ but it can be efficiently approximated in $\mathcal{O}(m^3)$ by using the techniques presented in [16]. Assume that we have obtained the approximate decomposition of S as follows.

$$S = U_m \Lambda_m U_m^T \tag{18}$$

where U_m is a $n \times m$ matrix formed by the first m normalized eigenvectors of U and m equals to the number of anchor images. Λ_m is the diagonal matrix with m diagonal elements (sorted in decreasing order from left to right) and correspond to the m largest eigenvalues of S. Eq. 18 is equivalent to the following equation.

$$S = U_m \Lambda_m^{\frac{1}{2}} \Lambda_m^{\frac{1}{2}} U_m^T = YY^T \tag{19}$$

where $Y = U_m \Lambda_m^{\frac{1}{2}}$. By combining Eq. 17 and Eq. 19, we obtain the following equation.

$$F^* = (NI_n - \alpha S)^{-1} L = (NI_n - \alpha YY^T)^{-1} L = \frac{I_n - Y(Y^T Y - \frac{N}{\alpha} I_m)^{-1} Y^T}{N} L \tag{20}$$

Since N remains constant for all of the images, the optimal ranking score vector F^* can be obtained as follows.

$$F^* = (I_n - Y(Y^T Y - \frac{N}{\alpha} I_m)^{-1} Y^T) L \tag{21}$$

Note that Eq. 21, requires the inversion of a $m \times m$ matrix in contrast to Eq. 5 that requires the inversion of a $n \times n$ matrix. Hence, Eq. 21 reduces the cost from $\mathcal{O}(n^3)$ to $\mathcal{O}(m^3)$.

4 Experimental Results

In this section, we evaluate the performance of our proposed approach (MMR) by using several real world image databases. All the experiments are implemented in Matlab R2009a and C++. First, we present the experimental setup in Section 4.1. Then, in Section 4.2, we evaluate the performance of our proposed approach.

4.1 Experimental Setup

Data Sets. We evaluated the performance of MMR on the following data sets.
- COREL: It is composed of 7700 images divided into 77 categories.
- Caltech101: This image database contains 8677 images from 101 different categories.
- MSRC: The data set contains 18 different categories and consists of approximately 4300 images.

Competitors. We compare our proposed approach with several manifold ranking based algorithms. Below are the details.
- **MR**. This is the first work [4] that applied manifold ranking (MR) for image retrieval.
- **EMR**. This is the algorithm proposed by Xu *et al.* [18]. While MR demonstrated good quality results, it is not suitable for large scale image databases because of its high computational cost. EMR proposes interesting techniques to improve the efficiency of MR and demonstrates that it retrieves the results of similar quality.

•**ADF**. This algorithm is proposed in [6]. ADF uses multiple features to construct a single image manifold.

Recall that our proposed approach uses multiple features to construct multiple image manifolds. We argue that using multiple image manifolds yield better results than the previous techniques. A natural question is whether previous approaches (e.g., MR and EMR) can perform better if they also utilize more than one features. To answer this question, we extend the previous techniques such that they utilize multiple features. Below are the details of how each technique is extended.

•**MR**$^{+N}$. N denotes the total number of features used by the algorithm. Let F_i^k be the ranking score of x_i computed by MR [4] using k^{th} feature. The final score of each image x_i is $\sum_{k=1}^{N} F_i^k$. MR^{+N} ranks the images according to the final scores. Note that MR^{+1} is the same as original MR algorithm proposed in [4].

•**EMR**$^{+N}$. Similar to MR^{+N}, EMR^{+N} computes the score of each image according to each feature. The images are then ranked according to their final scores. We remark that EMR^{+1} is the original EMR algorithm proposed in [18].

Later, we show that these extended versions retrieve better results than their respective original versions. Moreover, the quality of the retrieved results improves as the value of N increases. Similar to the notations used for extended version of MR and EMR, we use MMR^{+N} to denote that our algorithm MMR was run using N features. Similarly, ADF^{+N} denotes that ADF was run using N features.

Features used by the algorithms. We use some of the most popular features in the algorithms. More specifically, we use DoG-SIFT (Scale-Invariant Feature Transform) [9], HOG (Histogram of Oriented Gradients) [2], LBP (Local Binary Patterns) [11], Centrist [17] and RBG-SIFT [14]. Table 1 shows these features in a particular order. Any algorithm using N features uses the first N features shown in Table 1. For example, MMR^{+3} is our algorithm and uses the first three features (DoG-SIFT, HOG and LBP). Similarly, EMR^{+2} denotes that EMR was run using first two features (DoG-SIFT and HOG). ADF^{+5} denotes that ADF was run using all of the features. We remark that this order of the features best suits EMR^{+N} which is our main competitor.

Table 1. Features used by the algorithms

	1	2	3	4	5
Feature	DoG-SIFT	HOG	LBP	Centrist	RGB-SIFT

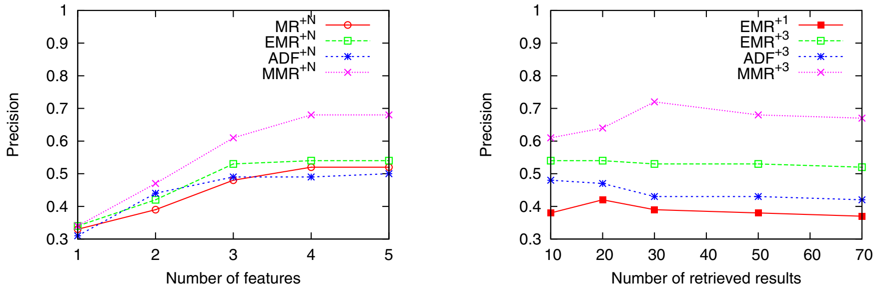
Evaluation metric. Each image in the image databases has its own category label (e.g., car, aeroplane etc.). A query is randomly selected from these databases and a retrieved result is considered correct if its label matches with the query label. For each query, we retrieve top-K results where the default value of K is 10 unless mentioned otherwise. We use precision as the main evaluation metric which corresponds to the number of correct results in the top-K retrieved results divided by K. Since K is fixed for all competitors, the recall value is directly related to the precision, i.e., if precision is high then the recall is also high and vice versa. Hence, we use precision as the only evaluation metric.

Recall that our algorithm (MMR) samples m anchor points and, for each image x_i, x_i is connected to its d nearest neighbors. We set m to 500 and d to 5 because our preliminary experimental evaluation demonstrated that these values of m and d give a reasonable trade-off between the precision and efficiency of the algorithm.

4.2 Performance Comparison

In this section, we compare the performance (precision and efficiency) of our algorithm with the other competitors. At the end, we present a case study where we show the top-10 results returned by MMR, EMR and ADF for three queries.

Precision. In Fig. 2, we increase the number of features used by each algorithm and study its affect on the precision. Note that the performance of each algorithm improves as it uses more features. However, the precision obtained by our algorithm (MMR^{+N}) is the highest. This is because our algorithm constructs multiple manifolds and minimizes the cost function to obtain the ranking scores in contrast to the other algorithms that use multiple features (manifolds) somewhat trivially. Note that the improvement in precision is less significant when $N > 3$. Since the running time increases with the increase in N, we choose $N = 3$ for rest of the experiments (unless mentioned otherwise).



(a) Effect of number of features on precision (b) Effect of K on precision

Fig. 2. Effect of number of features and K on precision.

As noted in [18] and observed from Fig. 2 (a), the precision of EMR^{+N} and MR^{+N} is quite similar. Furthermore, EMR^{+N} is more efficient than MR^{+N} as we demonstrate later. Therefore, for a clearer illustration of results, in the rest of the experiments we exclude MR^{+N}. In Fig. 2 (b), we issue top-K queries and vary K from 10 to 70 and study its affect on the precision. We observe that the precision of each of the algorithms remain unaffected with the increase in K. Also, note that our algorithm consistently gives better results than the other competitors.

In Fig. 3, we study the precision at a more detailed level. More specifically, we randomly choose 90 categories from the three image databases. For each category, we randomly choose one image as the query. For each query, we obtain top-10 results and record the precision. Fig. 3 shows the precision of each algorithm for the queries selected from each of the 90 categories. It can be observed that

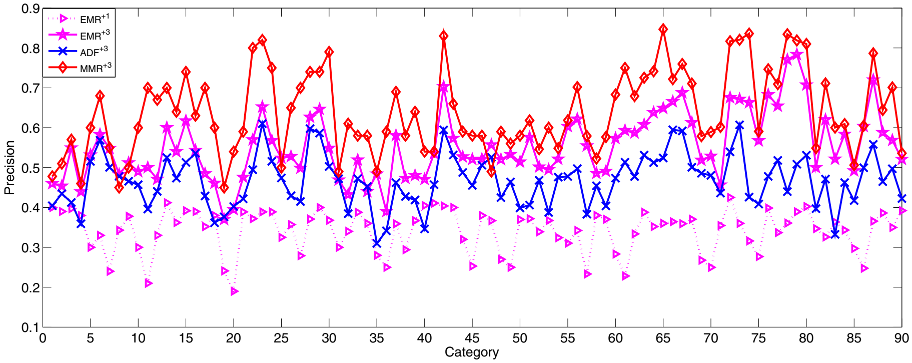
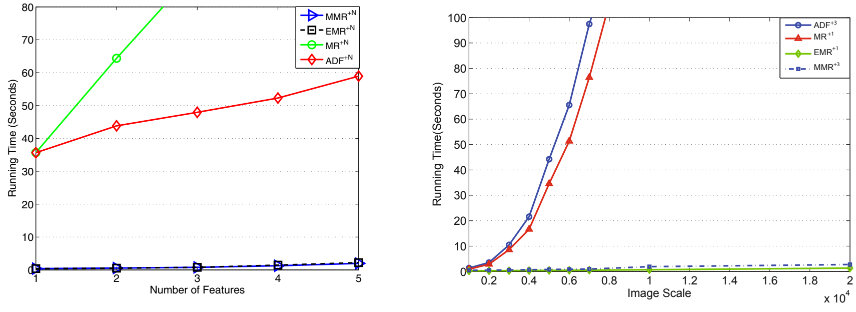


Fig. 3. Precision of top-10 images for randomly selected queries from each category

our approach MMR^{+3} consistently performs better than the other methods. The EMR proposed in [18] (shown as EMR^{+1}) has the lowest precision. However, EMR^{+3} that uses three manifolds has better retrieval performance than ADF^{+N}.

Running Time. In Fig. 4(a), we increase the number of features used by each algorithm and study its affect on the running time. Note that the running times of ADF^{+N} and MR^{+N} are much higher than the running time of our algorithm (MMR^{+N}) and EMR^{+N}. This is because MMR^{+N} and EMR^{+N} present efficient techniques for matrix inversion and use the anchor graphs to approximate the large image graphs. Also, note that EMR^{+N} and MMR^{+N} scale better as the number of features increases. The cost of MR^{+N} is the highest. In order to better illustrate the performance of other approaches, we do not display the cost of MR^{+N} when $N > 2$.

In Fig. 4(b), we increase the size of image databases and study its affect on the running times of all algorithms. It can be observed that ADF^{+N} and MR^{+N} cannot handle large scale databases (e.g., the running time is more than 80 seconds when the image database contains 8000 images). On the other hand,



(a)

(b)

Fig. 4. Effect of number of features and image database size on running time

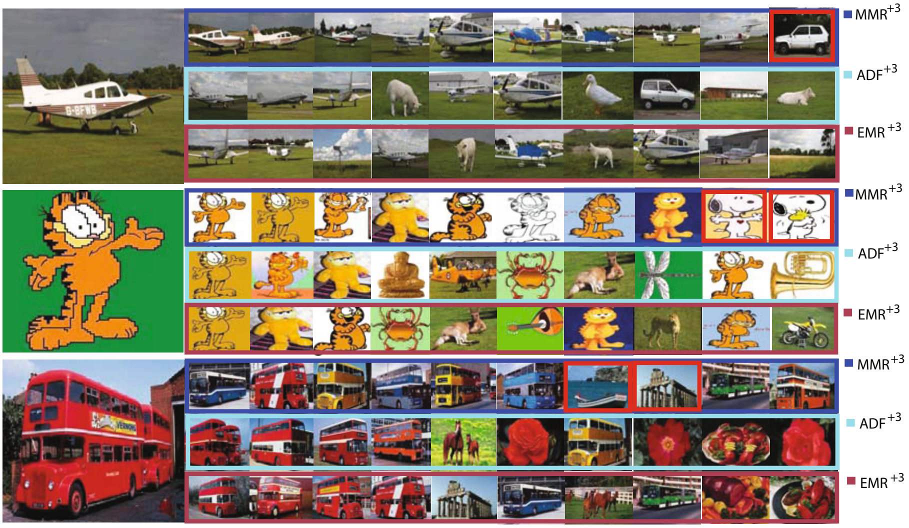


Fig. 5. Three queries are issued and top-10 results returned by MMR^{+3}, ADF^{+3} and EMR^{+3} are displayed. The irrelevant images retrieved by our algorithm are marked with red square. It can be noted that MMR^{+3} returns more relevant results than the other two algorithms.

our proposed algorithm scales better and can handle large scale image databases. The cost of EMR^{+1} is the lowest. This is because it uses a single image manifold whereas our algorithm MMR^{+3} uses three image manifolds. Nevertheless, the running times of both of the algorithms are quite close to each other.

A case study. In this section, we display the top-10 results returned by MMR^{+3}, ADF^{+3} and EMR^{+3} for three different queries. Fig. 5 displays the results returned by each of the algorithms. Irrelevant results returned by our algorithm are denoted by red square. Note that our algorithm returns more relevant results than the other two algorithms.

5 Conclusion

In this paper, we propose a novel method name multi-manifold ranking (MMR) which uses multiple image manifolds for image retrieval. We conduct extensive experimental study on real world image databases and demonstrate that MMR provides better retrieval results than state of the art techniques. Our experimental results demonstrate that our algorithm is much more efficient than two existing algorithms and is comparable to the most efficient existing approach.

Acknowledgments. Muhammad Aamir Cheema is supported by ARCDP130103405 and ARCDE130101002; Xuemin Lin is supported by ARC DP0987557, ARC DP110102937, ARC DP120104168 and NSFC61021004.

References

1. Christopher, P.R., Manning, D., Schütze, H.: An introduction to information Retrieval. Cambridge University Press (2009)
2. Dalal, N., Triggs, B.: Histogram of oriented gradients for human detection. In: CVPR, pp. 886–893 (2005)
3. Wang, M., et al.: Unified video annotation via multigraph learning. IEEE Trans. Circuits Syst. Video Techn. 19(5), 733–746 (2009)
4. He, J., Li, M., Zhang, H., Tong, H., Zhang, C.: Manifold-ranking based image retrieval. In: ACM Multimedia, pp. 9–16 (2004)
5. Huang, J., Kumar, S.R., Mitra, M., Jing Zhu, W.: Spatial color indexing and applications. International Journal of Computer Vision 35(3), 245–268 (1999)
6. Huang, Y., Liu, Q., Zhang, S., Metaxas, D.N.: Image retrieval via probabilistic hypergraph ranking. In: CVPR, pp. 3376–3383 (2010)
7. Liu, D., Hua, X.-S., Yang, L., Wang, M., Zhang, H.-J.: Tag ranking. In: ACM WWW, pp. 351–360 (2009)
8. Liu, W., He, J., Chang, S.-F.: Large graph construction for scalable semi-supervised learning. In: ICML, pp. 679–686 (2010)
9. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60(2), 91–110 (2004)
10. Manjunath, B.S., Rainer Ohm, J., Vasudevan, V.V., Yamada, A.: Color and texture descriptors. IEEE Transactions on Circuits and Systems for Video Technology 11, 703–715 (1998)
11. Ojala, T., Pietikainen, M., Maenpaa, T.: Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Transactions on Pattern Analysis and Machine Intelligence 24(7), 971–987 (2002)
12. Swain, M.J., Ballard, D.H.: Color indexing. International Journal of Computer Vision 7(1), 11–32 (1991)
13. Ulges, A., Worring, M., Breuel, T.M.: Learning visual contexts for image annotation from flickr groups. IEEE Transactions on Multimedia 13(2), 330–341 (2011)
14. van de Sande, K.E.A., Gevers, T., Snoek, C.G.M.: Evaluating color descriptors for object and scene recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 32(9), 1582–1596 (2010)
15. Wan, X., Yang, J., Xiao, J.: Manifold-ranking based topic-focused multi-document summarization. In: IJCAI, pp. 2903–2908 (2007)
16. Williams, C.K.I., Seeger, M.: Using the nyström method to speed up kernel machines. In: NIPS, pp. 682–688 (2000)
17. Wu, J., Rehg, J.M.: Centrist: A visual descriptor for scene categorization. IEEE Transactions on Pattern Analysis and Machine Intelligence 33(8), 1489–1501 (2011)
18. Xu, B., Bu, J., Chen, C., Cai, D., He, X., Liu, W., Luo, J.: Efficient manifold ranking for image retrieval. In: ACM SIGIR, pp. 525–534 (2011)
19. Yuan, X., Hua, X.-S., Wang, M., Wu, X.: Manifold-ranking based video concept detection on large database and feature pool. In: ACM Multimedia, pp. 623–626 (2006)
20. Zhou, D., Bousquet, O., Lal, T.N., Weston, J., Schölkopf, B.: Learning with local and global consistency. In: NIPS, pp. 592–602 (2003)
21. Zhou, D., Weston, J., Gretton, A., Bousquet, O., Schölkopf, B.: Ranking on data manifolds. In: NIPS (2003)

One Pass Concept Change Detection for Data Streams

Sripirakas Sakthithasan[1], Russel Pears[1], and Yun Sing Koh[2]

[1] School of Computing and Mathematical Sciences, Auckland University of Technology,
{ssakthit,rpears}@aut.ac.nz
[2] Department of Computer Science, University of Auckland
ykoh@cs.auckland.ac.nz

Abstract. In this research we present a novel approach to the concept change detection problem. Change detection is a fundamental issue with data stream mining as models generated need to be updated when significant changes in the underlying data distribution occur. A number of change detection approaches have been proposed but they all suffer from limitations such as high computational complexity, poor sensitivity to gradual change, or the opposite problem of high false positive rate. Our approach, termed OnePassSampler, has low computational complexity as it avoids multiple scans on its memory buffer by sequentially processing data. Extensive experimentation on a wide variety of datasets reveals that OnePassSampler has a smaller false detection rate and smaller computational overheads while maintaining a competitive true detection rate to ADWIN2.

Keywords: Data Stream Mining, Concept Drift Detection, Bernstein Bound.

1 Introduction

Data stream mining has been the subject of extensive research over the last decade or so. The well known CVFDT [1] algorithm is a good example of an early algorithm that proposed an incremental approach to building and maintaining a decision tree in the face of changes or concept drift that occur in a data stream environment. Since then there has been a multitude of refinements to CVFDT (such as [2]) and to other methods [3] [4] that perform other types of mining such as a clustering and association rule mining.

The fundamental issue with data stream mining is to manage the sheer volume of data which grows continuously over time. A standard method of coping with this issue is to use a fixed size window of width w, where only the most recent w instances are used to update the model built [5]. While this method is conceptually appealing, the major limitation is that concept change can occur at intervals that are quite distinct from the window boundaries. If rapid changes occur within a window, then these multiple changes will be undetected by the mining algorithm thus reducing the effectiveness of the model generated. Ideally a data stream algorithm should use long periods of stability to build a more detailed model whereas in time of rapid change the window needs to be shrunk at each change, the data representing the old concept be purged and the model updated with the new concept. Concept change detection with variable-sized adaptive windows has received very little attention compared to the well established area of algorithm development for data stream mining.

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 461–472, 2013.

The methods proposed for concept change detection with adaptive windows all suffer from limitations with respect to one or more key performance factors such as high computational complexity, poor sensitivity to gradual change or drift, or the opposite problem of high false positive rate. In this research we propose a novel concept change detection method called OnePassSampler and compare it with the state-of-the-art concept change detector, ADWIN2 [6]. Our empirical results show that OnePassSampler has a lower false positive rate and significantly lower computational overheads than ADWIN2. OnePassSampler, as its name suggests makes only a single pass through its memory buffer and employs a simple and efficient array structure to maintain data about the current window. With ADWIN2 every new data block that arrives triggers a reassessment of candidate cut points previously visited, thus making it a multi-pass algorithm with respect to its internal memory buffer.

The major contributions made by this research are: a robust one pass algorithm for concept drift detection that has low memory and run time overheads while offering a rigorous guarantee on the false positive rate. The rest of the paper is as follows. Section 2 reviews the major research relating to concept drift detection. In Section 3 we describe our novel approach to drift detection with the formulation of a model, the derivation of a test statistic and the one pass algorithmic approach that is the key to low overheads. Section 4 presents a conceptual comparison between OnePassSampler and ADWIN2. Section 5 presents our empirical results and we conclude in Section 6 with a summary of the research achievements and some thoughts on further work in the area of concept change detection.

2 Related Work

The concept drift detection problem has a classic statistical interpretation: given a sample of data, does this sample represent a single homogeneous distribution or is there some point in the data (i.e the concept change point) at which the data distribution has undergone a significant shift from a statistical point of view? All concept change detection approaches in the literature formulate the problem from this viewpoint but the models and the algorithms used to solve this problem differ greatly in their detail.

Sebastiao and Gama [7] present a concise survey on change detection methods. They point out that methods used fall into four basic categories: Statistical Process Control (SPC), Adaptive Windowing, Fixed Cumulative Windowing Schemes and finally other classic statistical change detection methods. Early Drift Detection Method (EDDM) [8] works on the same basic principle as the authors earlier work but uses different statistics to detect change. More recently Bifet et al [6] proposed an adaptive windowing scheme called ADWIN that is based on the use of the Hoeffding bound to detect concept change. The ADWIN algorithm was shown to outperform the SPC approach and has the attractive property of providing rigorous guarantees on false positive and false negative rates. ADWIN maintains a window (W) of instances at a given time and compares the mean difference of any two sub windows (W_0 of older instances and W_1 of recent instances) from W. If the mean difference is statistically significant, then ADWIN removes all instances of W_0 considered to represent the old concept and only carries W_1 forward to the next test.

An improved version of ADWIN called ADWIN2[6] was also proposed by the same author which used a variation of exponential histograms and a memory parameter, to limit the number of hypothesis tests done on a given window. ADWIN2 was shown to be superior to Gama's method and fixed size window with flushing [9] on performance measures such as the false positive rate, false negative rate and sensitivity to slow gradual changes [6]. Despite the improvements made in ADWIN2, some issues remain namely, the fact that multiple passes on data are made in the current window and an improvement in the false positive rate for noisy data environments.

3 The One Pass Sampler Concept Change Detector

We start by defining in formal terms the problem that we address in this research. We then describe some generic principles that govern our change detector model. A test statistic is then derived that will be used in the change detector algorithms that we propose. We present a memory management strategy that supports incremental sampling with the use of a fixed size buffer in the form of a reservoir.

3.1 Change Detection Problem Definition

Concept Change Detection. Let $S_1 = (x_1, x_2, ..., x_m)$ and $S_2 = (x_{m+1}, ..., x_n)$ with $0 < m < n$ represent two samples of instances from a stream with population means μ_1 and μ_2 respectively. Then the change detection problem can be expressed as testing the null hypothesis H_0 that $\mu_1 = \mu_2$ that the two samples are drawn from the same distribution versus the alternate hypothesis H_1 that they arrive from different distributions with $\mu_1 \neq \mu_2$. In practice the underlying data distribution is unknown and a test statistic based on the sample means needs to be constructed by the change detector. If the null hypothesis is accepted incorrectly when a change has occurred then a false negative is said to have taken place. On the other hand if H_1 is accepted when no change has occurred in the data distribution then a false positive is said to have occurred. Since the population mean of the underlying distribution is unknown, sample means need to be used to perform the above hypothesis tests. The hypothesis tests can be restated as: Accept hypothesis H_1 whenever $Pr(|\hat{\mu_{S_1}} - \hat{\mu_{S_2}}|) \geq \epsilon) > \delta$, where δ lies in the interval $(0, 1)$ and is a parameter that controls the maximum allowable false positive rate, while ϵ is a function of δ and the test statistic used to model the difference between the sample means.

Detection Delay. Due to the use of sample data to infer changes in the population, detection delay is inevitable in any concept change detector and is thus an important performance measure. Detection Delay is the distance between $(m + 1)$ and $m^{'}$, where $m^{'}$ is the instance at which change is detected. In other words, detection delay equals: $\left(m^{'} - (m + 1)\right)$.

3.2 OnePassSampler Conceptual Change Detection Model

Our change detector is designed to widen its applicability to streams with different characteristics while yielding comparable performance, accuracy and robustness to methods such as ADWIN2. OnePassSampler has the following properties, as illustrated in our experimentation: (1) is oblivious to the underlying data distribution, and (2) is inexpensive in terms of computational cost and memory.

Core Algorithm Overview. We first provide a basic sketch of our algorithm before discussing details of hypothesis testing. We use a simple example to illustrate the working of the algorithm. OnePassSampler accumulates data instances into blocks of size b. When attached to a classifier that uses OnePassSampler to detect change points, input data instances consists of a binary sequence of bits where binary 1 denotes a misclassification error and binary 0 denotes a correct classification decision. We use a block of data instances as the basic unit instead of instances as it would both be very inefficient and unnecessary from a statistical point of view to test for concept changes at the arrival of every instance.

Suppose that at time t_1 blocks B_1 and B_2 have arrived. OnePassSampler then checks whether a concept change has occurred at the $B_1|B_2$ boundary by testing H_1 above. If H_1 is rejected then blocks B_1 and B_2 are concatenated into one single block B_{12} and H_1 is next tested on the $B_{12}|B_3$ boundary. In this check the sample mean of sub-window B_{12} is computed by taking the average value of a random sample of size b from the sub-window of size $2b$. This sample mean is then compared with the sample mean computed from block B_3, also of size b. This process continues until H_1 is accepted, at which point a concept change is declared; instances in the left sub-window are removed and the instances in the right sub-window are transferred to the left. At all testing points equal sized samples are used to compare the sample means from the two sides of the window. The use of random sampling accelerates the process of the computation of the sample mean while maintaining robustness. The use of the averaging function as we shall see from our experimentation helps to smooth variation in the data and makes OnePassSampler more robust to noise than ADWIN2. In essence, OnePassSampler does a single forward scan through its memory buffer without the use of expensive backtracking as employed ADWIN2. While the use of random sampling ensures that sample means can be computed efficiently, a memory management strategy is required to ensure efficient use of memory as the left sub-window has the potential to grow indefinitely during periods of long stability in the stream.

Use of Bernstein Bound. Our approach relies on well established bounds for the difference between the true population and sample mean. A number of such bounds exist that do not assume a particular data distribution. Among them are the Hoeffding , Chebyshev , Chernoff and Bernstein inequalities [10]. The Hoeffding inequality has been widely used in the context of machine learning but has been found to be too conservative [6], over estimating the probability of large deviations for distributions of small variance. In contrast , the Bernstein inequality provides a tighter bound and is thus adopted in our work.

The Bernstein inequality states the following:

$$Pr\left(\left|\frac{1}{n}\sum_{i=1}^{n} X_i - E[X]\right| > \epsilon\right) \leq 2\exp\Big(\frac{-n\epsilon^2}{2\hat{\sigma}^2 + \frac{2}{3}\epsilon(c-a)}\Big)$$

where $X_1, ..., X_n$ are independent random variables, $E[X]$ is the expected value or population mean, $X_i \in [a, c]$ and $\hat{\sigma}$ is the sample variance.

Memory Management in OnePassSampler. As OnePassSampler never re-examines previous candidate cut points it does not need to maintain a history of such cut-points and thus does not need to store memory synopses in the form of exponential histograms as ADWIN2 does. Instead, OnePassSampler only requires the means of its left and right sub-windows. In order to efficiently support the computation of sample averages a random sampling strategy is employed.

In addition to improving efficiency, random sampling is also necessary to satisfy the independence requirement for data used in the computation of the Bernstein bound. In a data stream environment independence between data instances in the same locality may not always be true as changes in the underlying data causes instances arriving after such a change to have very similar data characteristics, thus violating the independence property. One simple and effective method of addressing this dependence effect is to perform random sampling.

Our memory management strategy is based on the use of arrays to store blocks of data. An array enables fast access to specific data blocks that are sampled via the use of random sampling. The array is used to capture data in OnePassSampler's memory buffer. The memory buffer is divided into a left sub-window and a right sub-window, each of which uses an array for storage. When a new data block arrives, the block is temporarily inserted into the right sub-window and the sample means from the two sub-windows are compared to check for statistically significant differences. If no such difference exists, data in the right sub-window is copied into the left sub-window and is then removed from the right sub-window. Essentially this means that the left sub-window consists of a set of largely homogeneous blocks. In this context, it is more efficient from a memory point of view to slide the oldest $\frac{w}{b}$ block from the sub-window, where w is the width of the window and b is the data block size.

In certain circumstances the right sub-window may hold more than one data block. This happens when OnePassSampler enters a warning state after which newly arriving data blocks are added to the right sub-window instead of the left sub-window. A warning state is triggered when the mean of the data block in the right sub-window is not significantly different from the mean in the left sub-window on the basis of the drift confidence value $1 - \delta_{drift}$ but is significantly different with respect to a warning confidence value $1 - \delta_{warning}$. In cases when a warning state is entered a sliding window scheme is used for the right sub-window as well.

Given the OnePassSampler's worst case memory requirements are bounded above by $2w$ as two memory buffers are allocated of size w for each of the two sub-windows. We experimented with different values of w and show that the quality of change detection (false positive rate, false negative rate and detection delay) is largely insensitive to the size of w, provided that w exceeds the block size b.

3.3 Computation of Cut Point Threshold ϵ

We now establish the value of the cut threshold against a null hypothesis that the data in the left and right sub-windows are drawn from the same population. Our null hypothesis is expressed as: H_0 is $H_0 : \mu_l = \mu_r = \mu$ and the alternate hypothesis as $H_1 : \mu_l \neq \mu_r$. Let S_l = a random sample $\{z_1, z_2, \ldots, z_l\}$of size l from $\{x_1, x_2, ..., x_m\}$ which comprise the m blocks in the left sub-window and let S_r = a random sample $\{z_1, z_2, \ldots, z_r\}$ of size r from $\{x_{m+1}, x_{m+2}, \ldots, x_n\}$ which comprises the (n-m) blocks in the right sub-window. With the application of the union bound on expression (1), we derive the following for every real number $k \in (0, 1)$:

$$Pr\left[|\hat{\mu_l} - \hat{\mu_r}| \geq \epsilon\right] \leq Pr\left[|\hat{\mu_l} - \mu| \geq k\epsilon\right] + Pr\left[|\mu - \hat{\mu_r}| \geq (1-k)\,\epsilon\right] \tag{1}$$

Applying the Bernstein inequality on the R.H.S of Equation 1, we get:

$$\begin{aligned} Pr\left[|\hat{\mu_l} - \hat{\mu_r}| \geq \epsilon\right] \quad & \leq 2\exp\Big(\tfrac{-b(k\epsilon)^2}{2\sigma_s^2 \frac{2}{3} k\epsilon(c-a)}\Big) \\ & +2\exp\Big(\tfrac{-b((1-k)\epsilon)^2}{2\sigma_s^2 + \frac{2}{3}(1-k)\epsilon(c-a)}\Big) \end{aligned} \tag{2}$$

In the classification context, the bounds a and c for the Bernstein bound take values $a = 0$, $c = 1$. Substituting this in (2) we get:

$$Pr\left[|\hat{\mu_l} - \hat{\mu_r}| \geq \epsilon\right] \leq 2\exp\Big(\frac{-b(k\epsilon)^2}{2\sigma_s^2 + \frac{2}{3}k\epsilon}\Big) + 2\exp\Big(\frac{-b((1-k)\epsilon)^2}{2\sigma_s^2 + \frac{2}{3}(1-k)\epsilon}\Big) \tag{3}$$

The probability $Pr\left[|\hat{\mu_l} - \hat{\mu_r}| \geq \epsilon\right]$ represents the false positive rate δ and hence we have:

$$\delta = Pr\left[|\hat{\mu_l} - \hat{\mu_r}| \geq \epsilon\right] \leq 2\exp\Big(\frac{-b(k\epsilon)^2}{2\sigma_s^2 + \frac{2}{3}k\epsilon}\Big) + 2\exp\Big(\frac{-b((1-k)\epsilon)^2}{2\sigma_s^2 + \frac{2}{3}(1-k)\epsilon}\Big) \tag{4}$$

We now need to minimize the RHS of (4) in order to minimize the upper bound δ for the false positive rate. Given the two exponential terms, the RHS of (4) can be minimized when:

$$\frac{-b(k\epsilon)^2}{2\sigma_s^2 + \frac{2}{3}k\epsilon} = \frac{-b((1-k)\epsilon)^2}{2\sigma_s^2 + \frac{2}{3}(1-k)\epsilon} \tag{5}$$

The variable k above represents the proportion of instances among the left and right sub-windows. OnePassSampler uses equal sized samples across the sub-windows, giving $k = \frac{1}{2}$. We note that $k = \frac{1}{2}$ satisfies (5) above. Substituting $k = \frac{1}{2}$ in (4) gives:

$$\delta \leq 2\exp\Big(\frac{-b\frac{1}{4}\epsilon^2}{2\sigma_s^2 + \frac{2}{3}.\frac{1}{2}\epsilon}\Big) + 2\exp\Big(\frac{-b\frac{1}{4}\epsilon^2}{2\sigma_s^2 + \frac{2}{3}.\frac{1}{2}\epsilon}\Big) \tag{6}$$

Solving (6) to find ϵ gives:

$$\epsilon = \frac{2}{3b}\left\{p + \sqrt{p^2 + 18\sigma_s^2 bp}\right\} \tag{7}$$

where $p = ln\left(\frac{4}{\delta}\right)$. If $|\hat{\mu_l} - \hat{\mu_r}| \geq \epsilon$, concept change is declared at instance $(m+1)$ and S_l, S_r can be considered to be from different distributions with probability $(1-\delta)$, otherwise, hypothesis H_0 is accepted that there is no concept change in the window of instances S_n.

A change detection algorithm by its very nature needs to test multiple cut points before an actual change point is detected. Each test involves a hypothesis test applied at a certain confidence level. The effect of multiple tests is to reduce the confidence from δ to $\delta^{'}$ which represents the effective (overall) confidence after n successive hypothesis tests have been carried. However, we note that the hypothesis tests in the change detection scenario are not independent of each other as the probability of a false positive (i.e incorrectly accepting hypothesis H1 that the means across the left and and right sub-windows are different) at a particular test has an influence on whether a false positive occurs at subsequent tests and hence methods such as Bonferroni do not apply. We use our own error correction factor, $\delta^{'} = 2\frac{\delta}{(1-\frac{1}{2}^{n})}$. The derivation of $\delta^{'}$ is omitted due to space constraints. Thus, in our model the change and warning significance levels, δ_{Change} and $\delta_{Warning}$ are set to $\delta^{'}_{Change}$ and $\delta^{'}_{Warning}$ respectively to control the false positive probability. We observe that the the correction factor above converges to 2δ for large values of n.

3.4 OnePassSampler Change Detection Algorithms

This section presents the core algorithms used in our change detector system. S_r and S_l denote the right and left sub windows. Algorithm (1) decides the change type given the mean values $\hat{\mu_r}$ and $\hat{\mu_l}$ of S_r and S_l respectively, ϵ_{change} (the threshold mean difference for δ_{change}) and $\epsilon_{warning}$ (the warning threshold mean difference for $\delta_{warning}$). ϵ_{change} and $\epsilon_{warning}$ are calculated using the equation (7). Though OnePassSampler detects drifts in any variation in the mean, algorithm (1) only reports the change when mean increases ($\hat{\mu_r} > \hat{\mu_l}$). In the event of a concept change Algorithm (2) transfers the contents of the right sub-window into the left. When a warning state is triggered it increases the sample size, in expectation of a subsequent concept change. This increase has the effect of increasing precision in sampling and the algorithm may become more sensitive to slow gradual change.

```
Input: $\hat{\mu_l}, \hat{\mu_r}, \epsilon_{Change}, \epsilon_{Warning}$
Output: $Change \; || \; Warning \; || \; Internal$
if $\epsilon_{Warning} \leq |\hat{\mu_l} - \hat{\mu_r}|$ then
    if $\epsilon_{Change} \leq |\hat{\mu_l} - \hat{\mu_r}|$ then
        if $\hat{\mu_r} > \hat{\mu_l}$ then
            return $Change$;
        end
        return $Internal$;
    end
    return $Warning$;
end
return $None$;
```

Algorithm 1. GetDriftType()

```
Input: $An\ instance(Ins),\ BlockSize,\ S_l,\ S_r$
Output: True/False
Increment the instance counter;
$S_l = S_r \cup \{Ins\}$;
if At the block boundary then
    ChangeType = GetDriftType();
    if (DriftType is $Change$ or $Internal$ ) then
        Remove all elements from $S_l$;
        Copy all elements of $S_r$ to $S_l$;
        Remove all elements from $S_r$;
        Set SampleSize to BlockSize;
        if (DriftType is $Change$) then
            return $True$;
        end
        return $False$;
    end
    else if (DriftType is $Warning$) then
        Double the sample size;
        return $False$;
    end
    Copy all elements of $S_r$ to $S_l$;
    $SampleSize = BlockSize$;
    return $False$;
end
```

Algorithm 2. IsDrift()

4 OnePassSampler versus ADWIN2: Similarities and Differences

Two major design differences exist between the two change detectors. The first lies in the policy used in determining cuts. When new data arrives, ADWIN2 creates a new bucket and adds it to its memory buffer. It then searches through all buckets currently stored in its memory buffer for a possible cut point. A cut point in ADWIN2 lies on the boundary between buckets. With N buckets currently in storage, ADWIN2 will examine a total of $(N - 1)$ possible cut points. Furthermore, as each new bucket arrives previous bucket boundaries that were examined before will be re-examined for possible cuts. Effectively, ADWIN2 makes multiple passes through its memory buffer. In contrast, OnePassSampler never re-examines previous block (equivalent of ADWIN2's bucket) boundaries for cuts and only examines the boundary between the newly arrived block and the collection of blocks that arrived previously for a possible cut. In this sense, OnePassSampler can be said to do a single pass through its memory buffer when searching for cuts, and hence its name.

The second major difference lies in the estimation strategy for assessing means of data segments. ADWIN2 relies on exponential histograms for estimating mean values, whereas OnePassSampler uses random sampling base on an efficient array structure to estimate means. The problem with exponential histograms is that some of the buckets, typically the more recent ones may be too small in size to yield accurate estimations for mean values. This is due to the fact that in ADWIN2 a bucket is created whenever a 1 appears in the stream, and when data has high variation bucket size will vary widely. For buckets that are too small in size to support accurate estimation, ADWIN2 will end up overestimating the true mean and false positives may then result.

5 Empirical Study

Our empirical study had two broad objectives. Firstly, we conducted a comparative study of OnePassSampler with ADWIN2 on key performance criteria such as the true positive rate, the false positive rate, the time delay in detecting changes and the execution time overheads involved in change detection. We used Bernoulli distribution in all experiments to simulate classifier outputs though OnePassSampler is a general drift detector for any distribution.

In the second part of our experimentation we conducted a sensitivity analysis of the effects of block size, warning level and sliding window size on the delay detection time for OnePassSampler.

5.1 Comparative Performance Study

One first experiment was designed to test OnePassSampler's false positive rate vis-a-vis ADWIN2. We used a stationary Bernoulli distribution for this and tested the effect of various combinations of mean values (μ) and confidence values (δ) as shown in Table 1. For this experiment the block size for OnePassSampler was set to its default value of 100 and ADWIN2's internal parameter M was also set to its default value. We conducted a total of 100 trials for each combination of μ and δ and the average false positive rate for each combination was recorded.

Table 1. False Positive Rate for stationary Bernoulli distribution

	One Pass Sampler			ADWIN2		
μ	δ =0.05	δ =0.1	δ =0.3	δ =0.05	δ =0.1	δ =0.3
0.01	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.1	0.0000	0.0000	0.0000	0.0001	0.0002	0.0018
0.3	0.0000	0.0000	0.0001	0.0008	0.0017	0.0100
0.5	0.0000	0.0000	0.0001	0.0012	0.0030	0.0128

Table 1 shows that both OnePassSampler and ADWIN2 have good false positive rates that are substantially lower than the confidence level set. However, we observe that as the variance in the data increases with the increase in the μ value (for a Bernoulli distribution, the variance is $\mu \times (1-\mu)$) that ADWIN2 starts to register false positives. The ADWIN2 false positive rate increases progressively with the increase in the variance as well as the lowering of confidence (ie higher δ values). On the other hand OnePassSampler retains a virtually zero false positive rate except when the confidence is low at 0.3 when it registers a rate of 0.01%, compared to the ADWIN2 rate of 1.28% at $\mu = 0.5$ and $\delta = 0.3$. As the confidence becomes lower the ϵ value decreases and this results in an increase in the false positive rate for ADWIN2. However, OnePassSampler is virtually insensitive to the decrease in ϵ due to the fact that the mean value can be estimated more accurately through the combined use of random sampling and the use of the aggregated running average mechanism.

The second experiment was designed to test the true positive (detection) rates of OnePassSampler and ADWIN2 over data that was also generated from a Bernoulli distribution. We generated four different data streams of length $L = 10,000$, 50,000, 100,000 and 1,000,000 bits from a Bernoulli distribution. The data generated was stationary with mean 0.01 in the first $L - 2300$ time steps and we then varied the distribution in a linear fashion with different gradients in the last 2300 time steps. A total of 100 trials were conducted for each combination of data length and slope values. We tracked key performance indicators such as the true detection rate, average execution time and the detection delay time. Both OnePassSampler and ADWIN2 managed to achieve a true detection rate of 100% for all combinations of data length and change gradients.

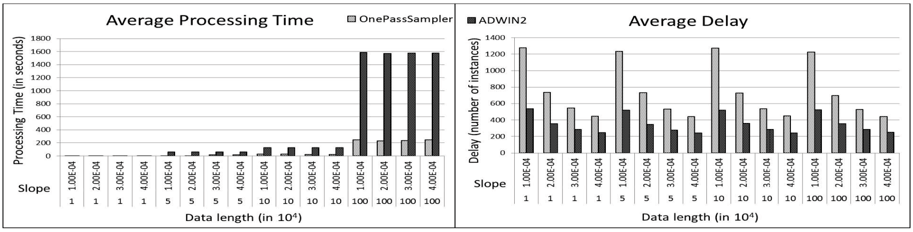


Fig. 1. Comparative Change Detection Performance of OnePassSampler and ADWIN2

Figure 1 also illustrates that ADWIN2 was much slower in stream processing than OnePassSampler. Furthermore, the gap between the two processing times becomes wider as the length of the stable segment of the stream becomes longer. This was expected as ADWIN2 spends much time doing repeated scans through the histogram and examines every possible combination of cuts defined by the buckets. OnePassSampler, on the other hand does a single pass through the window segment and at each block of data it assesses whether the newly arrived block is sufficiently different from the previous blocks in its memory buffer.

However, it is clear from Figure 1 that ADWIN2 has better mean detection delay when compared to OnePassSampler. OnePassSampler needs a relatively larger window segment before it can decide whether a newly arrived block is sufficiently different due to the sampling strategy that it uses. As expected, the delay times reduced with increasing gradient of change, although we observe that OnePassSampler reduces at a faster rate than ADWIN2 with the gap between the two closing for higher gradients of change. Section 5.2 shows that OnePassSampler's detection delay can be reduced with proper use of warning level and particularly block size on which it is most sensitive with respect to delay.

The final part of our experimentation involved an investigation of the sensitivity of OnePassSampler's key parameters on detection delay time. From previous experimentation with Bernoulli data it was observed that OnePassSampler had a higher detection delay time than ADWIN2 and thus the motivation was to determine parameter settings that minimize OnePassSampler's detection delay time.

5.2 Sensitivity Analysis on OnePassSampler

In the first experiment we investigated the effect of block size on Bernoulli data streams with different gradients. Section 5.1. Figure 2 shows that as block size increases, delay time initially decreases, reaches a minimum value and then starts to rise once again. In order to detect changes in data distribution a sample of sufficient size is required, which in turn is determined by the block size. If the size of the block (sample size) is too low, then in common with other statistical tests of significance, a statistical difference cannot be determined until a greater change occurs with time, thus delaying the detection. On the other hand, if the block size is too large then the probability increases that a change occurs too late within a given block for the change to be detected and so the change will go undetected until at least a new block arrives, thus giving rise to an increased detection delay. A block size of 200 appears to be optimal across a range of different change gradients, except when the change is very gradual , in which case 500 gives a slightly lower delay.

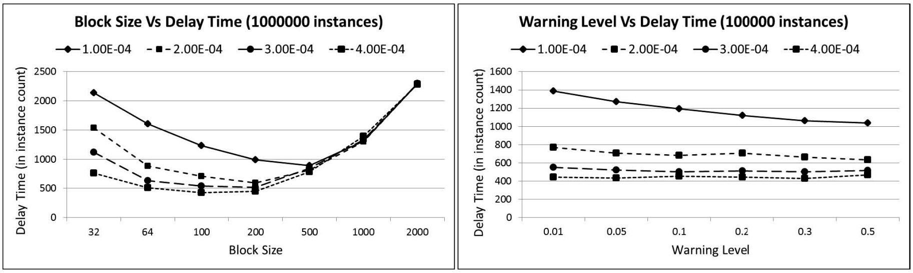


Fig. 2. Effects of Block Size and Warning Level on Detection Delay Time for OnePassSampler

We next checked the effect of warning level on delay. Figure 2 shows that warning level has a much smaller effect on delay than block size. With a slope of $1.00E - 04$ the warning level setting has a negligible effect on delay and thus a pragmatic setting that is twice the significance level should suffice in most cases to reduce the delay.

Next, we assessed the effect of sample size increment. Whenever the warning level is triggered the sample size is incremented in the hope of trapping an impending change earlier. We investigated a range of increments and as Figure 3 shows, a doubling of sample size produces optimal results across the entire spectrum. As with the warning level, too large an increase results in an increase in the detection delay.

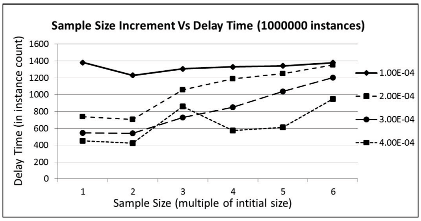


Fig. 3. Effects of Sample Size Increment on Detection Delay Time for OnePassSampler

Overall, it appears that block size is of prime importance in minimizing delay time; a block size of 200 works well for a range of datasets with different change dynamics. The other two parameters have a much smaller effect in general but can also contribute to smaller delay times with settings that we discussed above, especially in the case of slowly varying data.

Finally, we assessed the effects of the sliding window size on true positive rate, false positive rate and delay time. We varied the sliding window size in the range 500 to 10,000. For each window size, 30 trials were conducted on data from a Bernoulli distribution and the average for each of the performance measures were recorded. Due to space constraints we show the detection delay for the smallest change gradient of $1.00E - 4$; the results for the other change gradients followed very similar trends. As Table 2 shows, the detection delay is largely insensitive to window size. In addition, all window sizes recorded a true positive rate of 100%. The false positive rate was in line with the other two measures, virtually no change in rate was observed across the entire range of window sizes used. Once again space constrains prevent us from showing the entire set of results; we only show the case with mean value 0.3 and delta 0.3. All other combinations of mean and delta returned virtually identical results. These results indicate that window size when set at a reasonable multiple of block size has no significant effect on key factors such as the true positive rate and delay time. These results are to be expected as data that is slid out of the window consists of a set of homogeneous instances from OnePassSampler's left sub-window.

Table 2. Detection delay for varying window sizes

Sliding Window Size	500	1000	2000	4000	6000	8000	10000
True Positive Rate	100	100	100	100	100	100	100
Delay Time	1300	1330	1350	1320	1350	1370	1330
False Positive Rate	0.00000	0.00001	0.00000	0.00000	0.00000	0.00000	0.00000

6 Conclusions and Future Work

This research has shown that a concept change detector based on a sequential hypothesis testing strategy based on use of the Bernstein bound as a test statistic yields excellent performance in terms of false positive rate, true positive rate and processing time. Our comparative study with ADWIN2 clearly shows that a single pass strategy can produce competitive false positive and true positive rates to ADWIN2, with much lower computational overheads.

The use of sequential hypothesis testing combined with an efficient incremental strategy that updates statistics on the memory buffer were the two major factors behind the greatly reduced computational overheads over ADWIN2. Despite lower computational overheads, OnePassSampler has a higher detection delay time in certain cases and our future work will focus on improving this aspect. By means of a mechanism that monitors change in the running average of data arriving in the window an alternate candidate cut point can be defined at a point further downstream than the current block boundary. The system would then check both the current block boundary as well as the alternate point. This modification would result in trading off computational overhead with an improvement in detection delay for datasets with small gradients of change.

References

1. Hulten, G., Spencer, L., Domingos, P.: Mining time-changing data streams. In: Proc. of the 2001 ACM SIGKDD, pp. 97–106 (2001)
2. Hoeglinger, S., Pears, R., Koh, Y.: Cbdt: A concept based approach to data stream mining. In: Theeramunkong, T., Kijsirikul, B., Cercone, N., Ho, T.-B. (eds.) PAKDD 2009. LNCS, vol. 5476, pp. 1006–1012. Springer, Heidelberg (2009)
3. Koh, Y.S., Pears, R., Yeap, W.: Valency based weighted association rule mining. In: Zaki, M.J., Yu, J.X., Ravindran, B., Pudi, V. (eds.) PAKDD 2010, Part I. LNCS, vol. 6118, pp. 274–285. Springer, Heidelberg (2010)
4. Widiputra, H., Pears, R., Serguieva, A., Kasabov, N.: Dynamic interaction networks in modelling and predicting the behaviour of multiple interactive stock markets. Int. J. Intell. Syst. Account. Financ. Manage. 16, 189–205 (2009)
5. Wang, H., Fan, W., Yu, P.S., Han, J.: Mining concept-drifting data streams using ensemble classifiers. In: Proceedings of the 9th ACM SIGKDD, KDD 2003, pp. 226–235 (2003)
6. Bifet, A., Gavaldà, R.: Learning from time-changing data with adaptive windowing. In: SDM. SIAM (2007)
7. Sebastiao, R., Gama, J.: A study on change detection methods. In: 4th Portuguese Conf. on Artificial Intelligence (2009)
8. Jose, M.B., Campo-Ávila, J.D., Fidalgo, R., Bifet, A., Gavaldà, R., Morales-bueno, R.: Early Drift Detection Method. In: Proc. of the 4th ECML PKDD Int. Workshop on Knowledge Discovery from Data Streams, pp. 77–86 (2006)
9. Kifer, D., Ben-David, S., Gehrke, J.: Detecting change in data streams. In: Proceedings of the Thirtieth International Conference on VLDB, vol. 30, pp. 180–191. VLDB Endowment (2004)
10. Peel, T., Anthoine, S., Ralaivola, L.: Empirical bernstein inequalities for U-statistics. In: NIPS, pp. 1903–1911 (2010)

Incremental Mining of Significant URLs in Real-Time and Large-Scale Social Streams

Cheng-Ying Liu[1,2], Chi-Yao Tseng[2], and Ming-Syan Chen[1,2]

[1] Dept. of Electronic Engineering National Taiwan University, Taipei, Taiwan, R.O.C.
[2] Research Center for IT Innovation, Academia Sinica, Taipei, Taiwan, R.O.C.
{bermuda,cytseng,mschen}@citi.sinica.edu.tw

Abstract. Sharing URLs has recently emerged as an important way for information exchange in online social networks (OSN). As can be perceived from our investigation toward several social streams, the percentage of messages with URL embedded ranges from 54% to 92%. Due to the extremely high volume of evolving messages in OSN, finding interesting and significant URLs from social streams possesses numerous challenges, such as the real-time need, noisy contents, various URL shortening services, etc. In this paper, we propose the Significant URLs MINing algorithm, abbreviated as SURLMINE, to produce the up-to-date ranking list of significant URLs without any pre-learning process. The key strategy of SURLMINE is to incrementally update the significance coefficients of all collected URLs by four pivotal features, including Follower-Friend ratio, language distribution, topic duration and period and decay model. Moreover, its capability of incremental update enables SURLMINE to achieve the real-time processing. To evaluate the effectiveness and efficiency of SURLMINE, we apply the proposed framework to Twitter platform and conduct experiments for 30 days (over 75 million tweets). The experimental results show that the precision of SURLMINE can reach up to 92%, and the execution performance can also satisfy the real-time requirements in large-scale social streams.

Keywords: Significant URLs mining, incremental scheme, large-scale social streams, real-time processing.

1 Introduction

Due to the exploding popularity of online social networks (OSN) and microblogging platforms, such as Facebook, Twitter, LinkedIn, and Google+, spreading information with URLs has become a general phenomenon in social interactions. According to our observation by randomly sampling 140 million Twitter messages related to "YouTube", as high as 91.8% of messages contain at least one URL. Moreover, other similar experiments also indicate a high frequency of URL attachment in messages with certain keywords, such as 75.8% in "Google", 60.7% in "News", and 54.2% in "Obama". Since the data generation rate on OSN is very high, it is challenging to efficiently deal with real-time social streams. As far as Facebook is concerned, there are more than 900 million users, and in each

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 473–484, 2013.

day, over 3 billion comments and 300 million photos are added and uploaded in this platform[1]. On the other hand, from the report of mediabistro.com[2], over 500 million accounts have registered on Twitter in March 2012, and the number of tweets per day has reached 400 million[3]. It can be perceived that social streams are now a large-scale data warehouse with a great wealth of real-time information [1], such as news, blog articles, interesting facts, comments, and multimedia content. Although there are several existing services (e.g., Twitter Search, Google, and Bing) offering the social streams searching function based on the similarity between text content and query keywords, the results are often not well-organized and thus cannot provide users concise and meaningful information. For example, if a user intends to query social messages about specific breaking news, it is impossible for him/her to explore all related contents which are unstructured and generated rapidly. Furthermore, users may desire to know what time-sensitive news and hot topics are widely discussed at this moment. These functionalities cannot be provided by existing systems and are not yet fully explored in the literature.

In this paper, we focus on mining significant URLs which attract much attention and are highly discussed on OSN. We aim to monitor social streams containing a specific keyword and return the top-k up-to-date ranking list of significant URLs. The motivations of focusing on URLs are as follows. First, with the text length restriction, people are only allowed to write limited characters in a social message. Thus, to provide more complete information and share news with friends, attaching URLs has become a common approach in OSN and micro-blogging platforms [2, 3]. Second, URL is a universal locator that is language-independent, which means that people using different languages may still share identical URL. Third, since URL is a convenient cross-platform linker, it is helpful for celebrities or companies to find out which mediums are often included and highly correlated to them. However, although URL provides many advantages, discovering significant URLs possesses numerous challenges. The first challenge comes from the wide use of URL shortening services, which greatly increases complexity and difficulty when dealing with various kinds of URLs. Second, some URL shortening services have the time limit, which means URL shorteners could be invalid after a period of time. Third, since social streams are dynamic and ever-increasing, designing an efficient and real-time mining scheme for dealing with such large-scale and noisy data is a challenging task.

To the best of our knowledge, there is no existing mechanism which considers all above issues and fully explores the discovery of significant URLs on large-scale and real-time social streams. In this paper, we propose the Significant URLs MINing algorithm (abbreviated as SURLMINE) that incorporates a variety of social features to determine the significance and popularity of URLs. In addition, each post will just be analyzed once so as to reduce computation time and enable real-time processing to users. To verify the effectiveness of the proposed

[1] http://twopcharts.com/twitter500million.php

[2] http://www.mediabistro.com/alltwitter/twitter-active-total-users_b17655

[3] http://techglimpse.com/index.php/facebook-901-million-user-accounts.php

algorithm, we collect more than 75 million tweets in 30 days from May 6th to June 6th in year 2012 with the specific keywords. The experimental results show that the proposed scheme not only is able to extract significant URLs with high precision, but also satisfies real-time need in large-scale social streams. The rest of this paper is outlined as follows. Section 2 reviews the related studies on searching and recommendation techniques in micro-blogging platforms. We elaborate the details of our data crawling scheme and observation in Section 3, while Section 4 introduces the proposed SURLMINE algorithm. Extensive experiments and performance evaluation are presented in Section 5. Finally, concluding remarks are given in Section 6.

2 Related Work

Mining in OSN has been widely discussed in recent years due to its rapid growth. One major advantage of discovering knowledge from OSN is that social messages generally reflect most recent news and topics, and it is hardly to be accomplished by directly applying traditional algorithms, such as PageRank. The reason is that several related algorithms typically suffer the cold start problem, indicating that time-sensitive and relevant contents cannot be discovered in time to provide latest information [4]. Therefore, some researchers aim to extract interesting contents and rank them based on certain query inputs. In [5], Duan et al. used Rank-SVM technique to obtain critical features for selecting the candidate set, and the tweets are ranked according to the relevance of topics. The three pivotal features in [5] are: whether a tweet contains one or more URLs, the length of a tweet (number of characters), and the authority of user accounts. Phelan et al. [6] proposed a novel news recommendation technique called Buzzer, which harnesses real-time Twitter data as the basis for ranking and recommending articles from the collection of RSS feeds. Dong et al. [7] exploited Gradient Boosted Decision Tree algorithm to improve receny ranking from real-time Twitter streams. Another research direction is the burst detection. In [8], Mathioudakis et al. attempted to catch the vocabularies which suddenly appear with an unusually high rate. They also showed that the proposed approach has good performance in extracting trending topics from real-time information streams.

Note that most prior studies have all encountered the challenge of not being able to process large-scale datasets [9, 10] due to the high cost of computation time. In order to reduce the amount of data returned by each query, some researches proposed to narrow down search range by considering a small group of people among friends and the friends of friends [3, 10]. However, the main weaknesses of those methods are that results may be affected by preconception, and the quality of results also highly depends on the user's social communities. Moreover, even if bursty events can be precisely detected, further description and details are still unable to be given just by a set of words. Considering the period of presidential election, the names of president candidates will appear very frequently in social discussion, but further analysis is required to find out what do users talk about and which articles are representative ones. The most important thing is that the information diffusion patterns and behaviors of participants toward various topics have been confirmed as different [3]. Therefore, it is challenging to design a universal mining scheme which covers various topics in such large-scale data streams.

Our work builds on earlier contributions in three key respects. First, due to the diversity of URL shortening services and the time limitation of URL shorteners, most previous works only tracked one format of URL shorteners. For example, in [7] and [11], the authors focus on tinyURL and bit.ly respectively. In this paper, we consider involving all kinds of URL shorteners, and thus more complete information is covered for mining significant URLs. Second, micro-blogging is a constantly evolving medium where users often leave and join, and relationships between users may also change anytime. Thus, in order to obtain more information, our crawling has included a more wide range of global real-time streams that could be spread from any person and in any language, which is much bigger than crawling from user communities. Third, since social stream is a Big Data with high volume of noise and fast data generation rate, in order to be applicable in such context, each post will be regarded as an impact to certain URLs. Even if a post is a spam message, our algorithm is still able to determine the negative influence and incrementally updates the significant coefficients toward certain URL without any prior learning process.

3 Data Crawling and Pre-processing

In this paper, we evaluate our scheme with Twitter platform, since it is the most popular micro-blogging website with more than 140 million active users[4]. After gathering a large amount of Twitter data, URLs can be directly extracted and expanded from shortened forms to original forms. In Section 3.1, we first introduce two common approaches for crawling social messages from Twitter information streams. Next, we explain the procedures of expanding various URL shorteners and provide detailed analysis in Section 3.2.

3.1 Real-Time Crawling

There are two main approaches for collecting Twitter data, which are REST API[5] and streaming API[6]. Although both methods allow developers to access Twitter data, they are still several different properties between them. REST API provides simple interfaces for most Twitter functionality, and up to a maximum of 100 tweets will be returned per query. Twitter also applies a rate limitation to REST API where at most 350 requests are permitted per hour[7]. On the other hand, streaming API requires keeping a permanent HTTP connection open, and it randomly returns tweets containing a specific search keyword with the total

[4] http://blog.twitter.com/2012/03/twitter-turns-six.html

[5] https://dev.twitter.com/docs/api

[6] https://dev.twitter.com/docs/streaming-apis

[7] https://dev.twitter.com/docs/faq#6861

Table 1. The number of tweets crawled by streaming API and REST API

Keyword	Streaming API		REST API		Multiple
	Total	TPS*	Total	TPS	(S/R)*
YouTube	143,869,821	30.28	6,306,355	1.33	22.81
News	41,482,108	8.73	7,906,215	1.66	5.25
Google	28,720,525	6.04	7,474,687	1.57	3.84
Obama	8,503,834	1.79	5,271,187	1.11	1.61

*__TPS__: Tweets Per Second *__S/R__: Streaming/REST

quantity never exceeding 1% of all public data streams. Without the overhead and duplication issues caused by polling REST API at endpoint, streaming API is able to crawl a larger number of tweets. Table 1 shows the information of the data quantity by both APIs from May 6th to June 30th (55 days) in year 2012.

3.2 URL Statistics

By employing the above-mentioned crawling mechanism, we are able to determine the percentage of URLs in Twitter data and the proportional distributions of various URL shortening services. As shown in Table 2, among the specified four keywords, more than 54.22% of tweets attach at least one URL, in particular for tweets that mention "YouTube", where the URL attachment rate is as high as 90.80%. This statistic indicates that the popularity of URL shorting services may vary with different topics. Moreover, there are still many other services which are not so well-known and are infeasible to enumerate all of them. Therefore, developing a universal expanding method is desirable for covering more complete information.

Table 2. The distribution of various URL shortening services

Keyword	original	bit.ly	tinyurl	ow.ly	goo.gl	others	URL%
YouTube	96.49%	0.95%	0.14%	0.10%	0.12%	2.20%	90.80%
News	37.92%	17.92%	1.10%	0.00%	2.17%	40.89%	75.77%
Google	54.49%	16.30%	0.98%	2.28%	4.12%	21.83%	60.67%
Obama	30.20%	23.33%	2.27%	2.62%	2.87%	38.71%	54.22%

To resolve this difficulty, we devise a subroutine to expand all kinds of URL shorteners to their original forms by recursively tracking their redirections. To enable the real-time processing, this subroutine will be executed immediately whenever a URL is detected. With original URLs expanded by the devised procedure, we are able to calculate the average number of URLs embedded in a tweet and the frequency of URL occurrence more accurately. As can be seen in Figure 1, most tweets tend to attach only one URL, and tweets that contain more than three URLs are less than 0.1%. In addition, Figure 2 shows the cumulative distribution function (CDF) of the frequency of URL occurrence, which

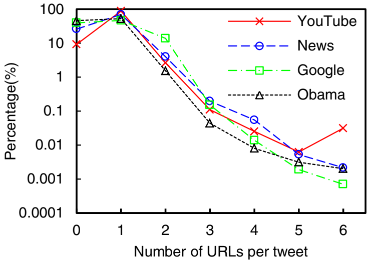


Fig. 1. Distribution of URL quantity in each tweet with different keywords

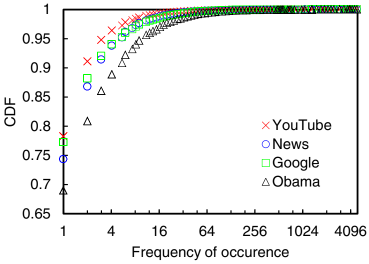


Fig. 2. Frequency of URL occurrence expressed in CDF with different keywords

addresses that only a very small number of URLs are posted frequently, and most URLs are just attached in one message. It is worthy to notice that most frequently attached URLs do not always imply they are significant or popular ones that interest users. It is because in order to gain more focus, spammers usually attempt to continuously post a large number of advertise links or phishing links to defraud audience. Moreover, some ordinary URLs with very high frequency is just owing to their inherent function, such as www.google.com. Table 3 gives instances of top-5 URLs with the most frequency of occurrence in messages containing keywords "Google" and "News", respectively.

Table 3. An example of top-5 most frequent URLs related "Google" and "News"

Keyword	URL

Google	`http://itunes.apple.com/app/rage-of-bahamut/id506944493?mt=8`
	`https://play.google.com/store/apps/details?id=com.ruckygames.gunmaapps`
	`https://play.google.com/store/apps/details?id=com.ruckygames.otherjp`
	`http://www.google.com`
	`http://www.google.com/intl/en/ipv6/`

News	`http://www.billboard.com/bbma/news/justin-bieber-usher-billboard-music-awards-cover-story`
	`http://www.goal.com/`
	`http://mobile.gungho.jp/news/sengoku/root.html`
	`http://www.thedailymash.co.uk/news/international/greeks-apologise-with-huge-horse-20120515`
	`http://news-discussions.com/`

4 Significant URLs Mining

As mentioned previously, the frequency of URL occurrence cannot be directly exploited to guarantee the quality of links due to some malicious operations and inherent characteristics. To better identify the significance of each URL, we devise SURLMINE algorithm to estimate the significance coefficients of URLs by measuring several characteristic features of social messages (i.e. tweets). In Section 4.1, we introduce four pivotal features that are considered in SURLMINE algorithm, and the details of SURLMINE are described in Section 4.2.

4.1 Characteristic Features of Social Messages

There are four characteristic features used to estimate the significance coefficients of URLs. They are (1) Follower-Friend ratio, (2) language distribution ratio, (3) duration and period, and (4) decay model, which are explained as follows.

Follower-Friend Ratio. Nowadays most of OSNs support non-reciprocal relationships to manage the social circles of users. The characteristic of non-reciprocal relationships is that it allows users to add anyone into their circles without their approval. For instance, users on Twitter and Google+ are allowed to directly add celebrities, such as Barack Obama or Lady Gaga, into their social circles without any permission. Thus, we can determine the ratio of followers to friends to quantify the user popularity. Let $\mathcal{S} = \{u_1, u_2, u_3, ...\}$ denote the set of distinct URLs in real-time social streams. For each URL $u_i \in \mathcal{S}$, the Follower-Friend ratio ε_i is defined as follows.

$$\varepsilon_i = \frac{C^i_{follower}}{C^i_{friend}} \tag{1}$$

In above equation, $C^i_{follower}$ and C^i_{friend} respectively represent the number of followers and friends of the author who spreads the URL u_i. As discuss in [12], the ratio ε directly reflect an author's popularity, for a user who has $\varepsilon \geq 2$, it means that he/she is a popular person, and lots of people want to hear what he/she said. Oppositely, if $\varepsilon < 1$, it shows this person is a knowledge seeker but not getting much attention. Therefore, in our scheme, so as to mining those URL that concerned by most of people, if a URL posted by a high Follower-Friend ratio author, the URL will be regarded as more significant.

Language Distribution. In SURLMINE, for a URL u_i, let $l_1, l_2, l_3, ..., l^i_{m_i}$ represent the number of messages used in m_i kinds of languages. We first use m_i for comparing global popularity. Once the m_i is equal to other URLs, the language distribution ratio ξ_i will be determined as follow by using average standard deviation.

$$\xi_i = \frac{\sum_{j=1}^{m_i} |m_i \cdot l^i_j - \sum_{j=1}^{m_i} l^i_j|}{{m_i}^2 \cdot \sum_{j=1}^{m_i} l^i_j} \tag{2}$$

It can be noticed from above equation that if the language distribution of URL u_i is balanced, the value of ξ_i will approach zero. Oppositely, if the distribution is uneven, the value of ξ_i will be much larger. Since each URL definitely links to a web page, if users attach an identical URL with different languages, it can be inferred that this URL probably contains international insights. Moreover, owing to the fact that most spam messages are spread in one particular language, the probability of a spam URL mentioned by several different languages must be very low. In this way, we perceive that language distribution not only reflects global popularity, but also simultaneously contributes to resolve the problem of spam links.

Duration and Period. In our scheme, we use both duration and period to differentiate various URL influences and behaviors. By enforcing the limitation of both period and duration, we can rapidly eliminate outdated URLs and the URLs just discussed in a flash of time. Let $\mathcal{T} = \left\{u_1^i, u_2^i, u_3^i, ..., u_{k_i}^i\right\}$ denote the set of URLs that point to identical web page with u_i in $\mathcal{S}$. For each URL $u_i \in \mathcal{S}$, the duration d_i and period p_i are formulated as follows.

$$p_i = \frac{d_i}{k_i} = \frac{\Sigma_{j=1}^{k_i}(t_j^i - t_{j-1}^i)}{k_i} \tag{3}$$

In Eq. 3, t_j^i represents the timestamp of URL u_j^i in $\mathcal{T}_i$, and k_i is the frequency of occurrence of URL u_i in $\mathcal{S}$ at that time. That is, the duration d_i is the time between t_1^i to t_k^i, and the period p_i is the specific value of duration d_i to the frequency of occurrence k_i.

Decay Model. In addition, a decay function model has been applied to determine the decay ratio λ_k^i for each u in $\mathcal{T}_i$, which is defined as follows.

$$\lambda_k^i = \begin{cases} 1 & , i < \rho \\ 1 - \frac{\rho^2 (t_k^i - t_{k-1}^i)^2}{\eta (t_\rho^i - t_1^i)^2} & , \lambda_k^i \geq minDecay \\ minDecay & , otherwise \end{cases} \tag{4}$$

In above equation, ρ is a constant value used to determine the average period of first ρ URLs in $\mathcal{T}_i$, and η is a threshold that determines the multiple of average period for decay cycles. Note that the minimum decay λ_k^i is experimentally set as 0.01 since if a URL is unfrequented for a long time (more than η times as many as the average period), the value of $t_k^i - t_{k-1}^i$ will be large. In such situation, the decay ratio λ_k^i will be very small or negative, and this situation must be prevented. Oppositely, if URL has just been attached, the difference of t_k^i and t_{k-1}^i will be small, and the decay ratio λ_k^i will approach to 1, indicating a weak decay.

4.2 SURLMINE Algorithm

$$\delta_k^i = \prod_{j=1}^{k} \varepsilon_j \cdot \lambda_j^i = \varepsilon_k \cdot \lambda_k^i \cdot \prod_{j=1}^{k-1} \varepsilon_j \cdot \lambda_j^i = \varepsilon_k \cdot \lambda_k^i \cdot \delta_{k-1}^i \tag{5}$$

By considering above-mentioned features, the goal of SURLMINE is to incrementally update the significance coefficient of a URL so as to immediately output the results. Let $\delta_1^i, \delta_2^i, \delta_3^i, ..., \delta_k^i$ denote the significance coefficients for any URL u_i in $\mathcal{T}$. The significance coefficient δ_k^i of URL u_i is formulated in Eq. 5. For any significance coefficient δ_k^i, it only requires the Follower-Friend ratio (i.e. ε) and the decay ratio (i.e. λ_k^i) to determine significance coefficient δ_k^i, where δ_{k-1}^i is the previous state of δ_k^i that is kept in memory.

Algorithm 1. Significant URLs MINing Algorithm (SURLMINE)

```
Input: $u$, A URL extracted from social streams;
Result: $\mathcal{R}$, Update the set of the URL significance coefficients;
Data: $\mathcal{S}$, The set of existing distinct URLs;
      $\mathcal{K}$, The set of the frequency of all URL occurrence;
$\tilde{u}$ = Expand URL $u$;
$\varepsilon$ = Compute Follower-Friend ratio of user $\alpha$ by Eq. 1;
if $\tilde{u} \notin \mathcal{S}$ then /* If $\tilde{u}$ has not been quoted before*/
    Insert URL $\tilde{u}$ to set $\mathcal{S}$;
    Insert the value 1 as the frequency of occurrence of $\tilde{u}$ in $\mathcal{K}$;
    Set significance coefficient $\delta$ of URL $u$ as $\varepsilon$;
else
    $\alpha$ = Get user ID who attaches URL u;
    $\mathcal{A}$ = Get the user list which has attached URL $\tilde{u}$ before;
    if $\alpha \notin \mathcal{A}$ then /* If $\tilde{u}$ is mentioned by a new user */
        Insert user $\alpha$ to set $\mathcal{A}$;
        Add the value 1 to the frequency of occurrence of $\tilde{u}$ in $\mathcal{K}$;
        $\lambda$ = Compute decay ratio of $\tilde{u}$ with its creation time by Eq. 4;
        $\delta$ = Get previous state significance coefficient of URL $\tilde{u}$ from $\mathcal{R}$;
        Set significance coefficient $\delta$ as $\delta \cdot \varepsilon \cdot \lambda$ by Eq. 5;
    end
end
$\xi$ = Compute language distribution ratio of $\tilde{u}$ based on Eq. 2;
Insert both $\delta$ and $\xi$ of URL $\tilde{u}$ to $\mathcal{R}$ in decreasing order;
Compute period $p$ and duration $d$ of $\tilde{u}$ with its creation time by Eq. 3;
if $d < minDuration$ or $p > maxPeriod$ then Set URL $\tilde{u}$ as unavailable;
```

For each incoming URL u, SURLMINE first expands u to the original form $\tilde{u}$, and next identifying whether URL $\tilde{u}$ has been quoted before by examining if URL $\tilde{u}$ is an element of $\mathcal{S}$. If it is not, this implies $\tilde{u}$ is a new URL and the significance coefficient δ of $\mathcal{S}$ will be directly assigned as its Follower-Friend ratio. Otherwise, if it is the first time that author α attaches URL $\tilde{u}$, the significance will be incrementally determined with the previous state of significance coefficient by Eq. 5. Finally, significance coefficient will be inserted into the set $\mathcal{R}$ in decreasing order. When doing insertion, once the significance coefficients are equal, URL that has higher language distribution ratio will be regarded more significant. The two thresholds of *minDuration* and *maxPeriod* are separately set as 1 and 0.5 to filter those immature URL. It indicates even a URL u has a high significance coefficient, the URL can be outputted if and only if duration d_i is larger than *minDuration* and period p_i is less than *maxPeriod.* Overall, with the incremental characteristic, SURLMINE is able to maintain an up-to-date ranking list of significant URLs in the environment of fast-pacing social streams. Moreover, since SURLMINE is unnecessary to scan past data for updating significance coefficients, and thus large storage space and computation time can be saved.

5 Experiments

In this paper, we conduct a series of experiments to verify the effectiveness of SURLMINE on a personal computer with 3.4 GHz CPU and 4 GB main memory. The implementation and experimental design are described in Section 5.1. We then present the performance evaluation of precision and efficiency in Section 5.2.

5.1 Experimental Design

In order to make experiments more extensible and modular, we divide the implementation into two parts. The first part is the data crawling through Twitter streaming API. The major task of this data crawler is maintaining a connection with Twitter servers to continuously access tweets that contain the specified keywords. The second part is responsible for significant URLs mining that outputs up-to-date top-k URLs, where k is defined as 0.01% of URLs collected in that day. In our experiments, we additionally implement Boolean Spreading Activation algorithm and Burst Detection algorithm [8] (abbreviated as BSA and BD respectively) for comparing purpose. The BSA has been widely used in areas such as information retrieval and epidemic models, where its ranking strategy is mainly based on the frequency of occurrence. On the other hand, the BD outputs the URLs that suddenly appear with an unusually high rate by continuously tracking period of URLs. To judge the precision of different algorithms, since the preference for information may differ from person to person, evaluating URL significance through manual study by a specific group of people may be biased. To better solve this problem, we focus on the social streams of Twitter that mention the keyword "YouTube" and attach at least one URL. YouTube is well-known for being the busiest video sharing site, where the total number of views in each day has exceeded 4 billion. Moreover, since each video on YouTube has its own statistics about audience rating, we can validate video significance by considering following conditions (1) more than 500,000 of view counts; (2) more than 1,000 of views per day after 24 hours with present view count growing rate; (3) the ratio of like and dislike is more than 100. If one of the above conditions is satisfied, a video will be identified as a significant video, and the precision is the percentage of significant video URLs.

5.2 Precision and Efficiency Issues

The experiments have been continuously executed for 30 days from May 6th to June 6th in year 2012. During this period, we snapshot our data warehouse at several time points with different durations. As shown in Table 4, after 30 days, there are totally 41 million videos mentioned in 75 million tweets. Moreover, since we only focus on social messages with URL attachment, an additional search term "http" will be automatically appended for any querying topics before crawling. Thus, around 99% of tweets contain at least one valid URL, except some special cases, such as the tweets containing the term "http" but with no URL attached. The results of precision evaluation are shown in Figure 3. It

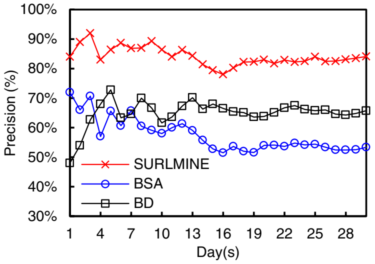


Fig. 3. Day-to-day precision in 30 days from May 6th to June 6th in year 2012

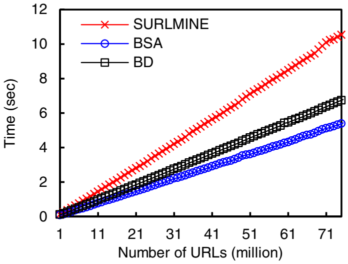


Fig. 4. Cumulative computation time for around 75 million URLs

Table 4. Summary information of dataset with different durations

Duration	Tweet	URL	Video	URL%
6 hours	651,815	645,792	353,119	99.08%
12 hours	1,228,422	1,217,244	681,778	99.09%
1 day	2,782,752	2,754,090	1,508,690	98.97%
7 days	19,830,243	19,637,890	11,020,784	99.03%
15 days	37,506,988	37,195,680	20,505,978	99.17%
30 days	75,500,079	74,865,879	41,408,318	99.16%

can be seen that SURLMINE is more precise than BSA and BD algorithm. On average, the precision can reach up to 92%. By further analyzing the output URLs of these three algorithms, we notice that most advertising and noisy URLs are unable to be excluded by BSA algorithm. In general, these tweets are posted frequently and swiftly accumulate a high frequency of occurrence in a short time. This situation causes that based only on the frequency of occurrence, unwanted URLs always rank high and are hardly to be replaced by new ones. The similar problem has occur in BD algorithm as well. Although BD algorithm has better precision than BSA algorithm, the main weakness of BD algorithm is that it needs more time to become stable. This is because if the frequency of URL occurrence is not large enough for detecting bursty events, the precision of BD algorithm could be lower.

On the other hand, regarding the efficiency issue, SURLMINE only takes about 140 nanoseconds to incrementally include a new URL. Note that the time for the URL expansion step is not covered. Moreover, Figure 4 shows the comparison of cumulative computation time with the number of URLs increasing. It can be seen that the total computation time for analyzing the whole data warehouse (up to 75 million URLs) is only about 11 seconds. Furthermore, although SURLMINE considers more characteristic features and employs more advanced processing, the execution time is not significantly larger than that of BSA, which is solely based on the frequency of occurrence. These evaluations verify that SURLMINE can deal with large-scale and real-time social streams and can be applicable to real applications.

6 Conclusion

In this paper, to enable the real-time processing of significant URLs extraction from OSNs, we proposed an efficient and effective algorithm named SURLMINE. The up-to-date ranking list of significant URLs are produced by incrementally updating the significance coefficients of all collected URLs with four pivotal features, including Follower-Friend ratio, language distribution, topic duration and period and decay model. In our experiments, the collected datasets with over 75 million messages from Twitter cover various kinds of languages, and URLs. With such general settings and such a large quantity of tweets, the precision of SURLMINE can still reach up to 92%, which verifies the effectiveness of the proposed scheme. Moreover, the experimental results also validate that the incremental capability of SURLMINE greatly enhances the efficiency performance. Consequently, these evidences indicate that SURLMINE can be applicable to large-scale and real-time social streams.

References

[1] Kwak, H., Lee, C., Park, H., Moon, S.: What Is Twitter, a Social Network or a News Media? In: 19th ACM International Conference on WWW, pp. 591–600 (2010)

[2] Nagpal, A., Hangal, S., Joyee, R.R., Lam, M.S.: Friends, Romans, Countrymen: Lend Me Your URLs. Using Social Chatter to Personalize Web Search. In: ACM International Conference on CSCW, pp. 461–470 (2012)

[3] Chen, J., Nairn, R., Nelson, L., Bernstein, M., Chi, E.: Short and Tweet: Experiments on Recommending Content from Information Streams. In: 28th ACM International Conference on CHI, pp. 1185–1194 (2010)

[4] Schein, A.I., Popescul, A., Ungar, L.H., Pennock, D.M.: Methods and Metrics for Cold-Start Recommendations. In: 25th ACM International Conference on SIGIR, pp. 253–260 (2002)

[5] Duan, Y., Jiang, L., Qin, T., Zhou, M., Shum, H.Y.: An Empirical Study on Learning to Rank of Tweets. In: 23rd ACM International Conference on COLING, pp. 295–303 (2010)

[6] Sankaranarayanan, J., Samet, H., Teitler, B.E., Lieberman, M.D.: TwitterStand: News in Tweets. In: 17th ACM International Conference on GIS, pp. 42–51 (2009)

[7] Dong, A., Zhang, R., Kolari, P., Bai, J., Diaz, F., Chang, Y., Zheng, Z., Zha, H.: Time Is of The Essence: Improving Recency Ranking Using Twitter Data. In: 19th ACM International Conference on WWW, pp. 331–340 (2010)

[8] Mathioudakis, M., Koudas, N.: TwitterMonitor: trend detection over the twitter stream. In: ACM International Conference on SIGMOD, pp. 1155–1158 (2010)

[9] Rashid, A.M., Lam, S.K., Karypis, G., Riedl, J.: ClustKNN: A Highly Scalable Hybrid Model- &. Memory-Based CF Algorithm. In: 12th ACM International Conference on WebKDD (2006)

[10] Sarwar, B.M., Karypis, G., Konstan, J., Riedl, J.: Recommender Systems for Large-scale E-Commerce: Scalable Neighborhood Formation Using Clustering. In: 5th IEEE International Conference on CIT (2002)

[11] Antoniades, D., Polakis, I., Kontaxis, G., Athanasopoulos, E., Ioannidis, S., Markatos, E.P., Karagiannis, T.: we.b: The Web of Short Urls. In: 20th ACM International Conference on WWW, pp. 715–724 (2011)

[12] Cha, M., Haddadi, H., Benevenuto, F., Gummadi, K.P.: Measuring user influence in twitter: The million follower fallacy. In: 4th International AAAI Conference on ICWSM (2010)

A Concept-Drifting Detection Algorithm for Categorical Evolving Data

Fuyuan Cao[1,2] and Joshua Zhexue Huang[1]

[1] Shenzhen Key Laboratory of High Performance Data Mining,
Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences,
Shenzhen 518055, China

[2] Key Laboratory of Computational Intelligence and Chinese Information Processing
of Ministry of Education, the School of Computer and Information Technology,
Shanxi University, Taiyuan 030006, China

cfy@sxu.edu.cn, zx.huang@siat.ac.cn

Abstract. In data streams analysis, detecting concept-drifting is a very important problem for real-time decision making. In this paper, we propose a new method for detecting concept drifts by measuring the difference of distributions between two concepts. The difference is defined by approximation accuracy of rough set theory, which can also be used to measure the change speed of concepts. We propose a concept-drifting detection algorithm and analyze its complexity. The experimental results on a real data set with a half million records have shown that the proposed algorithm is not only effective in discovering the changes of concepts but also efficient in processing large data sets.

Keywords: Categorical Data, Evolving, Concept-drifting.

1 Introduction

Many real world applications generate continuously arriving data, such as business transactions, web logs, sensors networks, etc. This type of data is known as data streams [1]. Generally speaking, a data stream can be considered as a sequence of items of structural information in which each item is stamped with a time point. As the arrival items change with time, the data distribution of the underlying structural information may change as well. Usually, the cause of the change is unknown. To understand the behaviors of data streams, it is important to investigate the changes of the distributions and the causes of the changes.

Semantically, the distribution of the structural information at a particular time point in a data stream is referred to as representation of a concept. A concept is defined by its intension and extension. Intension is the representation schema of structural information while extension refers to the set of objects represented by the schema. A concept often contains a set of sub-concepts. In machine learning, we can learn the intensions of concepts or sub-concepts from a set of objects. In supervised learning, every object is labeled with a class in the target variable. The set of objects in the same class is referred to as a sub-concept. In unsupervised learning, the classes of objects can be obtained with a clustering algorithm. In this case, a cluster is a sub-concept.

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 485–496, 2013.

As the arrival items change over time, the change of data distribution can be used to induce the change of a concept. In real applications, the change of a concept is mainly caused by emerging new sub-concepts or fading old sub-concepts or both. A radical change of a concept is often known as concept drift [2]. Two kinds of concept drift are illustrated in literature [3]. One is sudden (abrupt) concept drift and the other is gradual concept drift. Sudden concept drift is described as that the data distribution is dramatically changed in a short time period. Gradual concept drift is considered that the change of a concept occurs gradually over time. For example, in social network analysis, different groups of people are interested in different topics. Some people may gradually change their interests from one topic to another over time and some may suddenly change their interests to new topics.

To investigate the behaviors of such data streams, we concern whether the concept at time t_2 has drifted from the concept at time t_1, where $t_2 > t_1$. Meantime, we are interested in the change speed of concepts.

In this paper, we propose a new method to measure the difference between two concepts at the different time points. This difference is defined by approximation accuracy of rough set theory. Based on the new measure, we propose a concept-drifting detection algorithm to detect whether a concept has drifted or not. We have conducted a series of experiments on the KDD-CUP'99 data. The experimental results have demonstrated the proposed algorithm is not only effective in discovering the changes of concepts but also efficient in processing large data sets.

2 Preliminaries

In this section, we first review the basic concepts in rough set theory [4], such as indiscernibility relation, lower and upper approximations, approximation accuracy that are used to define the measures of concept change. We then define the problem of concept-drifting in the categorical time-evolving data.

2.1 Some Basic Concepts of Rough Set Theory

In a relational database, the structural data is stored in a table, where each row represents an object and each column represents an attribute that describes the objects. Formally, a data table can be defined as a quadruple $DT = (U, A, V, f)$, where U is a nonempty set of objects called the universe and A is a nonempty set of attributes such that for any $x \in U$ and $a \in A$, $f(x, a) \in V_a$. $V = \bigcup_{a \in A} V_a$ is the union of all attribute domains. If V is represented by continuous values, then DT is called a numerical data table. For any $a \in A$, if V_a is finite and unordered, then DT is called a categorical data table. Unless otherwise specified, DT represents a categorical data table in this paper.

Let DT be a categorical data table defined on A and $P \subseteq A$. P defines an equivalence relation $IND(P)$ as:

$$IND(P) = \{(x, y) \in U \times U | \forall a \in P, f(x, a) = f(y, a)\}. \tag{1}$$

$IND(P)$ is also called the indiscernibility relation with respect to P. If $(x, y) \in IND(P)$, the objects x and y are said to be indiscernible from each other by the attributes from P. It is easy to show that $IND(P)$ is an equivalence relation on U and $IND(P) = \bigcap_{a \in P} IND(\{a\})$. The relation $IND(P)$ induces a partition of U, denoted by $U/IND(P) = \{[x]_P | x \in U\}$, where $[x]_P$ denotes the equivalence class determined by x with respect to P, i.e., $[x]_P = \{y \in U | (x, y) \in IND(P)\}$.

As any equivalence relation induces a partition of the universe, these partitions can be used to build new subsets of the universe. These notions can be formally expressed as follows.

Let $DT = (U, A, V, f)$ be a categorical data table, $P \subseteq A$ and $X \subseteq U$. One can approximate X using only the information in P by constructing the lower approximation and the upper approximation of X, denoted as $\underline{P}X$ and $\overline{P}X$ respectively, where $\underline{P}X = \{x | [x]_P \subseteq X\}$ and $\overline{P}X = \{x | [x]_P \bigcap X \neq \emptyset\}$.

The objects in $\underline{P}X$ can be classified with certainty as members of X on the basis of knowledge in P, while the objects in $\overline{P}X$ can only be classified as possible members of X. The set $BN_P(X) = \overline{P}X - \underline{P}X$ is called the P-boundary region of X, and consists of those objects that cannot be decisively classified into X on the basis of knowledge in P. The set $U - \overline{P}X$ is called the P-outside region of X and consists of those objects which can not belong to X certainly. A set is said to be rough if the boundary region is non-empty.

A rough set can be characterized numerically by the following term

$$\alpha_P(X) = \frac{|\underline{P}X|}{|\overline{P}X|}. \tag{2}$$

which is called the approximation accuracy, where $|X|$ denotes the cardinality of $X \neq \emptyset$. Obviously, $0 \leq \alpha_P(X) \leq 1$. If $\alpha_P(X) = 1$, X is said to be crisp with respect to P, i.e., X is precise with respect to P. Otherwise, if $\alpha_P(X) < 1$, X is said to be rough with respect to P, i.e., X is vague with respect to P.

2.2 Problem Statement

Similarly, a categorical time-evolving data can also be stored in a table. Formally, a categorical time-evolving data table [5] can be formulated as a quintuple $TDT = (U, A, V, f, t)$, where U, A and V are the same as those in DT. The information function $f : U \times A \times t \to V$ is a mapping such that for any $x \in U$ and $a \in A$, $f(x, a, t) \in V_a$, where t is the arriving time of object x. As the arrival objects change with time, concepts often change at different time points. In order to detect the change of concepts, we adopt the sliding window technique which is used in the numerical data streams [6–8] to partition a categorical time-evolving data table. Suppose that N is the sliding window size, then the TDT is separated into several continuous subsets $S^{T_i}(1 \leq i \leq \lfloor \frac{U}{N} \rfloor)$ and each subset S^{T_i} has N objects. Each subset can also be called a concept. The superscript number T_i is the identification number of the sliding window and T_i is also called timestamp. In this work, our goal is to detect the difference between $S^{T_{i+1}}$ and S^{T_i} and analyze the speed of the difference.

3 Concept-Drifting Detecting

In this section, we define the lower approximation and upper approximation of a set Y with respect to a data set X instead of a universe U in rough set theory. To enable quantitative analysis of concept drifting for categorical evolving data, we formulate a set of measures for changes of concepts, including the degrees and speeds of a new concept emerging and a old concept emerging as well as the speed of change between two concepts.

3.1 Measures of Concepts Changes

To formulate the change of a concept, we define the lower approximation and upper approximation of a set as

Definition 1. *Let $TDT = (U, A, V, f, t)$ be a categorical time-evolving data table, $P \subseteq A$ and $X \subseteq U$. For any $Y \subseteq X$ and $x \in X$, the lower approximation and upper approximation of Y with respect to X are defined as*

$$\underline{P}Y = \{x|[x]_P \subseteq Y\} \tag{3}$$

and

$$\overline{P}Y = \{x|[x]_P \bigcap Y \neq \emptyset\}, \tag{4}$$

where $[x]_P = \{y \in X|(x, y) \in IND(P)\}$.

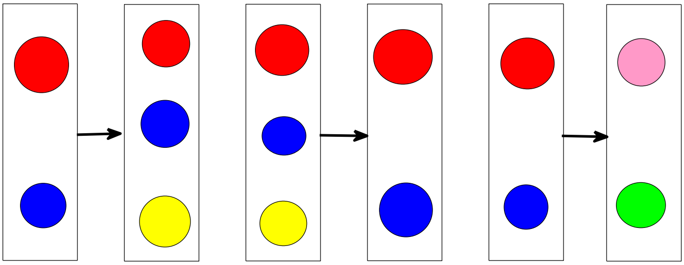
Here, the lower approximation and upper approximation of Y are defined with respect to X, not to the universe U.

Given a categorical data steam that carries a set of concepts at different time points, at a particular time point, a concept contains a set of sub-concepts and the concept changes as sub-concepts change over time. For example, in social media data streams, a topic may consist of several subtopics at a given time point and the topic changes as a new subtopic emerges or an old subtopic disappears at the following time points. We use an intuitive example in Fig.1 to illustrate three types of concept change.

Assume the two rectangles in each sub figure of Fig.1 represent a concept at two consecutive time points t_1 and t_2 from left to right. Each rectangle contains two or three sub-concepts described by circles in different colors. Fig.1(a) shows the yellow sub-concept emerged at t_2 after t_1. Fig.1(b) shows yellow sub-concept disappeared at t_2 from the concept. In Fig.1(c), two old sub-concepts faded and two new sub-concepts emerged.

Using the definitions of lower approximation and upper approximation in Definition 1, we define the measures for degrees and speeds of new concept emerging and old concept fading in categorical evolving data as follows.

Definition 2. *Let $TDT = (U, A, V, f, t)$ be a categorical time-evolving data table and $S^{T_i}, S^{T_j} \subseteq U$, where $S^{T_i} \bigcap S^{T_j} = \emptyset$ and $S^{[T_i,T_j]} = S^{T_i} \bigcup S^{T_j}$. The new*



(a) New concept emerging (b)Old concept fading (c) Dual occurring

Fig. 1. Three types of concept change

concepts emerging degree and old concepts fading degree from S^{T_i} to S^{T_j} with respect to A are defined as

$$NED_A < S^{T_i}, S^{T_j} > = \tfrac{1}{|A|} \textstyle\sum_{a \in A} NED_{\{a\}} < S^{T_i}, S^{T_j} > \tag{5}$$

and

$$OFD_A < S^{T_i}, S^{T_j} > = \tfrac{1}{|A|} \textstyle\sum_{a \in A} OFD_{\{a\}} < S^{T_i}, S^{T_j} >, \tag{6}$$

where

$$NED_{\{a\}} < S^{T_i}, S^{T_j} >= \frac{|\underline{\{a\}}S^{T_j}|}{|\overline{\{a\}}S^{T_j}|},$$

$$OFD_{\{a\}} < S^{T_i}, S^{T_j} >= \frac{|\underline{\{a\}}S^{T_i}|}{|\overline{\{a\}}S^{T_i}|}.$$

Here, $\underline{\{a\}}S^{T_m}(m = i, j)$ represents the lower approximation of S^{T_m} in $S^{[T_i,T_j]}$ with respect to attribute a, and $\overline{\{a\}}S^{T_m}(m = i, j)$ represents the upper approximation of S^{T_m} in $S^{[T_i,T_j]}$ with respect to attribute a.

$NED_A < S^{T_i}, S^{T_j} >$ and $OFD_A < S^{T_i}, S^{T_j} >$ are used to measure the degrees of concept change between two consecutive time points. The higher the value of $NED_A < S^{T_i}, S^{T_j} >$ or $OFD_A < S^{T_i}, S^{T_j} >$ is, the more dramatic the change of a concept from S^{T_i} to S^{T_j}, either a sub-concept emerged or faded.

According to Eq.(1), the degree of a concept change with respect to an attribute a, $NED_{\{a\}} < S^{T_i}, S^{T_j} >$ or $OFD_{\{a\}} < S^{T_i}, S^{T_j} >$ equals to 1 if $S^{[T_i,T_j]}/IND(\{a\}) = \{X | X = \{u\}, u \in S^{[T_i,T_j]}\}$. The degree of a concept change with respect to an attribute a equals to 0 if $S^{[T_i,T_j]}/IND(\{a\}) = \{X | X = S^{[T_i,T_j]}\}$. In other situations, $0 < NED_{\{a\}} < S^{T_i}, S^{T_j} >, OFD_{\{a\}} < S^{T_i}, S^{T_j} > < 1$. Therefore, we have $0 \leq NED_A < S^{T_i}, S^{T_j} >, OFD_A < S^{T_i}, S^{T_j} > \leq 1$.

The speed of concept drifting was used [9]. In this paper, we use speed to measure the amount of concept change from t_1 to t_2. The speeds of new concept emerging and old concept fading are defined as follows.

Definition 3. *Let $TDT = (U, A, V, f, t)$ be a categorical time-evolving data table and $S^{T_i}, S^{T_j} \subseteq U$, where $S^{T_i} \bigcap S^{T_j} = \emptyset$ and $S^{[T_i,T_j]} = S^{T_i} \bigcup S^{T_j}$. The new concepts emerging speed and old concepts fading speed from S^{T_i} to S^{T_j} with respect to A are defined as*

$$NES_A < S^{T_i}, S^{T_j} > = NED_A < S^{T_i}, S^{T_j} > \times \frac{|S^{T_j}|}{t_j} \tag{7}$$

and

$$OFS_A < S^{T_i}, S^{T_j} > = OFD_A < S^{T_i}, S^{T_j} > \times \frac{|S^{T_i}|}{t_i}. \tag{8}$$

where $\frac{|S^{T_m}|}{t_m}(m = i, j)$ represents the flowing speed of objects.

By considering the degrees of new concept emerging and old concept fading together, we define the degree and speed of change between two concepts as:

Definition 4. *Let $TDT = (U, A, V, f, t)$ be a categorical time-evolving data table and $S^{T_i}, S^{T_j} \subseteq U$, where $S^{T_i} \bigcap S^{T_j} = \emptyset$ and $S^{[T_i,T_j]} = S^{T_i} \bigcup S^{T_j}$. The degree and speed of change between S^{T_i} and S^{T_j} with respect to A are defined respectively as*

$$CD_A(S^{T_i}, S^{T_j}) = \frac{NED_A{<}S^{T_i}, S^{T_j}{>} + OFD_A{<}S^{T_i}, S^{T_j}{>}}{2} \tag{9}$$

and

$$CS_A(S^{T_i}, S^{T_j}) = CD_A(S^{T_i}, S^{T_j}) \times \frac{|S^{[T_i,T_j]}|}{t_i + t_j}. \tag{10}$$

It is easy to prove that $CD_A(S^{T_i}, S^{T_j})$ is a metric.

Property 1. Let $TDT = (U, A, V, f, t)$ be a categorical time-evolving data table. For any $S^{T_i}, S^{T_j}, S^{T_k} \subseteq U$, where $S^{T_i} \bigcap S^{T_j} \bigcap S^{T_k} = \emptyset$, we have

(1) Symmetry: $CD_A(S^{T_i}, S^{T_j}) = CD_A(S^{T_j}, S^{T_i})$;
(2) Nonnegativity: $CD_A(S^{T_i}, S^{T_j}) \geq 0$; and
(3) Triangle Inequality: $CD_A(S^{T_i}, S^{T_j}) + CD_A(S^{T_j}, S^{T_k}) \geq CD_A(S^{T_i}, S^{T_k})$.

Example 1. We use the simple categorical time-evolving data set in Table 1 to show the procedure of computing the degree of change between two concepts. The speed of change can be computed similarly.

In Table 1, data set is $X = \{x_1, x_2, \ldots, x_{20}\}$ and $A = \{A_1, A_2, A_3\}$ is the attribute set. Assume there are 5 records in each sliding window (i.e., the window size N=5), and totally 4 windows in X, i.e., $S^{T_1} = \{x_1, x_2, \cdots, x_5\}$, $S^{T_2} = \{x_6, x_7, \cdots, x_{10}\}$, $S^{T_3} = \{x_{11}, x_{12}, \cdots, x_{15}\}$ and $S^{T_4} = \{x_{16}, x_{17}, \cdots, x_{20}\}$.

Using Definition 1, we calculate

$$S^{[T_1,T_2]}/IND(\{A_1\}) = \{\{x_1, x_5, x_6, x_8, x_{10}\}, \{x_2, x_4, x_9\}, \{x_3, x_7\}\},$$

$$S^{[T_1,T_2]}/IND(\{A_2\}) = \{\{x_1, x_4, \cdots, x_{10}\}, \{x_2, x_3\}\},$$

$$S^{[T_1,T_2]}/IND(\{A_3\}) = \{\{x_1, x_6, x_{10}\}, \{x_2, x_3, x_4, x_7, x_9\}, \{x_5, x_8\}\}.$$

According to Definition 2, we calculate

Table 1. A categorical time-evolving data table

Object	A_1	A_2	A_3
x_1	A	M	C
x_2	Y	E	P
x_3	X	E	P
x_4	Y	M	P
x_5	A	M	D
x_6	A	M	C
x_7	X	M	P
x_8	A	M	D
x_9	Y	M	P
x_{10}	A	M	C
x_{11}	B	E	G
x_{12}	X	M	P
x_{13}	B	E	D
x_{14}	Y	M	P
x_{15}	B	F	D
x_{16}	Y	M	P
x_{17}	X	M	P
x_{18}	Z	N	T
x_{19}	X	M	P
x_{20}	Y	M	P

$$NED_{\{A_1\}} < S^{T_1}, S^{T_2} >= \frac{|\emptyset|}{|\{x_1, x_2, \cdots, x_{10}\}|} = 0,$$

$$NED_{\{A_2\}} < S^{T_1}, S^{T_2} >= \frac{|\emptyset|}{|\{x_1, x_4, \cdots, x_{10}\}|} = 0,$$

$$NED_{\{A_3\}} < S^{T_1}, S^{T_2} >= \frac{|\emptyset|}{|\{x_1, x_2, \cdots, x_{10}\}|} = 0,$$

$$OFD_{\{A_1\}} < S^{T_1}, S^{T_2} >= \frac{|\emptyset|}{|\{x_1, x_2, \cdots, x_{10}\}|} = 0,$$

$$OFD_{\{A_2\}} < S^{T_1}, S^{T_2} >= \frac{|\{x_2, x_3\}|}{|\{x_1, x_2, \cdots, x_{10}\}|} = \frac{1}{5},$$

$$OFD_{\{A_3\}} < S^{T_1}, S^{T_2} >= \frac{|\emptyset|}{|\{x_1, x_2, \cdots, x_{10}\}|} = 0,$$

Using Definition 3 and Definition 4, we obtain

$$\begin{aligned} CD_A(S^{T_1}, S^{T_2}) &= \tfrac{1}{2} \times (NED_A < S^{T_1}, S^{T_2} > + OFD_A < S^{T_1}, S^{T_2} >) \\ &= \tfrac{1}{2} \times \tfrac{1}{3}(0 + 0 + 0 + 0 + \tfrac{1}{5} + 0) \\ &= 0.0333 \end{aligned}$$

With similar computations, we obtain

$$CD_A(S^{T_2}, S^{T_3}) = 0.2507$$

$$CD_A(S^{T_3}, S^{T_4}) = 0.2381$$

We can compare the degrees of change at consecutive windows as

$$CD_A(S^{T_1}, S^{T_2}) < CD_A(S^{T_3}, S^{T_4}) < CD_A(S^{T_2}, S^{T_3}).$$

If we set 0.2 as a threshold, we can identify that concept has drifted from S^{T_2} to S^{T_3} and from S^{T_3} to S^{T_4}. S^{T_3} and S^{T_4} are considered as concept drifting windows.

If t_1, t_2, t_3, t_4 are the duration times of the 4 sliding windows, we can compute the speeds of changes NES_A, OFS_A and CS_A between consecutive sliding windows using Definition 3 and Definition 4, as shown in Table 2.

Table 2. The change speed between consecutive sliding windows

Sliding windows	NES_A	ODS_A	CS_A
$S^{T_1} \longrightarrow S^{T_2}$	0	$0.0667 \times \frac{5}{t_1}$	$0.0667 \times \frac{10}{t_1+t_2}$
$S^{T_2} \longrightarrow S^{T_3}$	$0.2845 \times \frac{5}{t_3}$	$0.2169 \times \frac{5}{t_2}$	$0.2507 \times \frac{10}{t_2+t_3}$
$S^{T_3} \longrightarrow S^{T_4}$	$0.1429 \times \frac{5}{t_4}$	$0.3333 \times \frac{5}{t_3}$	$0.2381 \times \frac{10}{t_3+t_4}$

3.2 Concept-Drifting Detecting Algorithm

From the above definitions, drifting of a concept can be detected by comparing the degree of change against a given threshold. As a result, a concept-drifting detection algorithm $CDDA$ is developed as shown in Algorithm 1. The key step of $CDDA$ is to compute the degree of change between two consecutive sliding windows $CD_A(S^{T_i}, S^{T_{i+1}})$. The complexity of this computation is $O(|S^{[T_i,T_{i+1}]}|^2|A|)$. Therefore, the time complexity of $CDDA$ algorithm is $O(\lfloor\frac{|X|}{N}\rfloor|S^{[T_i,T_{i+1}]}|^2|A|) = O(\lfloor\frac{|X|}{N}\rfloor 4N^2|A|) = O(|X|N|A|)$, where X is the data set, $|A|$ the number of attributes, and N the size of sliding windows. We can see that the time complexity of $CDDA$ is linear with respect to the number of the objects in X.

4 Experimental Results and Analysis

4.1 Data Set

We used the 10% subset version of the KDD-CUP'99 Network Intrusion Detection stream data set [10] to test the $CDDA$ algorithm. The Network Intrusion

Algorithm 1. The concept-drifting detection algorithm

```
Input:
- $TDT = (U, A, V, f, t)$ : the data set,
- $N$ : the size of sliding window,
- $\theta$ : the specified threshold value,
Output: Driftingwindow;
Method:
Driftingwindow=$\emptyset$;
for $i = 1$ to $\lfloor \frac{|U|}{N} \rfloor - 1$ do
    if $CD_A(S^{T_i}, S^{T_{i+1}}) \geq \theta$ then
        Driftingwindow=Driftingwindow $\bigcup\{i + 1\}$;
    end if
end for
```

Detection data set consists of a series of TCP connection records from two weeks of LAN network traffic data managed by MIT Lincoln Labs. Each record corresponded to either a normal connection or an intrusion (or attack). The attacks include 22 types. In the following experiments, all 22 attack-types are seen as “attack”. In this data set, there are 494,021 records and each record contains 42 attributes (class label is included). We discretized the 34 numerical attributes using the uniform quantization method and each attribute was quantized into 5 discrete values.

4.2 Concept-Drifting Detection

The size of the sliding windows and the given threshold are two parameters that affect the detection of concept drifting. We conducted a series experiments to investigate the settings of these two parameters. The experiment results are presented as follows.

Experiment 1. In this experiment, the threshold was set to 0.01 and the size of the sliding windows changed from 1000 to 30000 with a step length of 1000. The variations of the number of drifting-concepts with respect to the class label and the attribute set are shown in Fig.2.

From Fig.2, we can see that the number of drifting-concepts decreased with increase of the sliding window size.

Experiment 2. In this experiment, the size of the sliding window was set to 3000 and the threshold changed from 0.01 to 1 with a step length of 0.01. The number of drifting-concepts changed as threshold changed with respect to the class label and the attribute set. The result is shown in Fig.3.

From Fig.3, we can see that the change rate on the number of drifting-concepts over the threshold with respect to the attribute set is greater than that with respect to the class label. To make the number of drifting-concepts with respect to the class label as close as possible to the number with respect to the attribute set, the threshold with respect to the class label should be greater than the

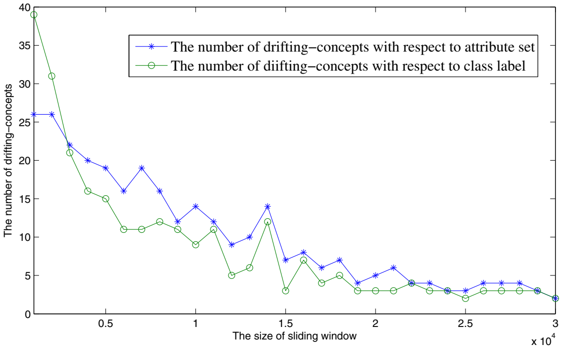


Fig. 2. The number of drifting-concepts varying with the size of the sliding windows

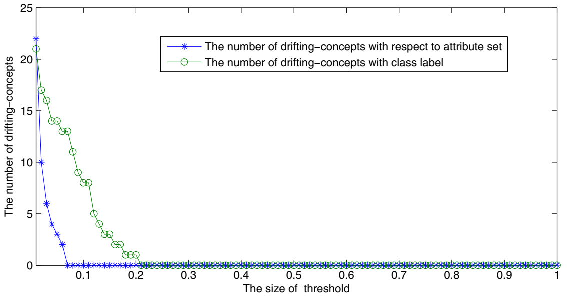


Fig. 3. The number of drifting-concepts varying with the values of the threshold

threshold with respect to the attribute set. In practice, a user can choose a threshold according to a prior knowledge or specific requirement.

Experiment 3. The duration of objects was assumed same in each sliding window and the evolving speeds of concepts in different sliding windows are shown in Fig.4. In this experiment, the size of the sliding window was set to 3000.

In Fig.4, the values of the change speed drop to zero in the range of 51 to 114, 134 to 149, and 155 to 160 because the records are same in these sliding windows of each interval.

5 Related Work

Detection of concept drifting has become an interesting research topic recently. The problem of detecting concept drifts in numerical data was explored in [11, 12]. As for detection of concept drifting in categorical data, a method was proposed to determine concept drifts by measuring the difference of cluster distributions between two continuous sliding windows from categorical data streams

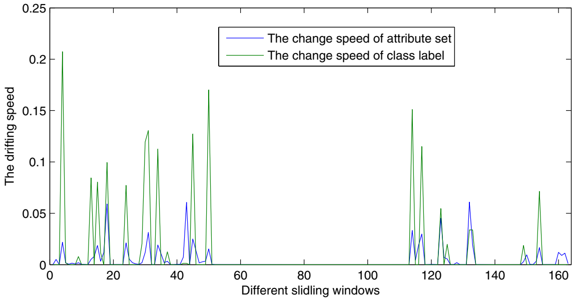


Fig. 4. The evolving speed on KDD-CUP'99 data set

[13]. The shortcoming of the method is the difficulty to set suitable system parameters for different applications. In [14], a framework was presented for detecting the change of the primary clustering structure which was indicated by the best number of clusters in categorical data streams. However, setting the decaying rates to adapt to different types of clustering structures is very difficult. Nasraoui [15] presented a framework for mining, tracking, and validating evolving multifaceted user profiles which summarize a group of users with similar access activities. In fact, two continuous sliding windows can be considered as two concepts. Cao [5] used rough set theory to define the distance between two concepts as the difference value of the degree of membership of each object belonging to two different concepts, respectively. This method only requires one parameter to set, so it is easy to use in real applications. However, the distance can only detect the change of concepts, and reasons that cause the change are not considered.

6 Conclusion

In this paper, based on sliding window techniques and approximation accuracy, the change degree and the change speed of concepts have been defined, and a concept-drifting detection algorithm has been proposed. The time complexity analysis and experimental results on a real data set have demonstrated the proposed algorithm is not only effective in detecting concept drifts from categorical data streams but also efficient in processing large data sets due to its linearity with respect to input data X.

Acknowledgements. This work is supported by Shenzhen Internet Industry Development Fund under Grant JC201005270342A, China Postdoctoral Science Foundation under Grant 2012M510046, the Natural Science Foundation of Shanxi under Grant 2010021016-2.

References

1. Babcock, B., Babu, S., Dater, M., Motwanti, R.: Models and Issues in data stream systems. In: Proc. PODS, pp. 1–16 (2002)
2. Widmer, G., Kubat, M.: Learning in the presence of concept drift and hidden context. Machine Learning 23, 69–101 (1996)
3. Guha, S., Meyerson, A., Mishra, N., Motwani, R., OCallaghan, L.: Clustering data streams: theory and practice. IEEE Transactions Knowledge and Data Engineering 15, 515–528 (2003)
4. Pawlak, Z.: Rough sets. International Journal of Computer and Information Sciences 11, 341–356 (1982)
5. Cao, F.Y., Liang, J.Y., Bai, L., Zhao, X.W., Dang, C.Y.: A framework for clustering categorical time-evolving data. IEEE Transactions on Fuzzy Systems 18, 872–885 (2010)
6. Aggarwal, C.C., Han, J., Wang, J., Yu, P.S.: A framework for clustering evolving data streams. In: Proc. Very Large Data Bases Conf. (2003)
7. Chakrabarti, D., Kumar, R., Tomkins, A.: Evloluationary clustering. In: Proc. ACM SIGKDD. Knowledge Discovery and Data Mining, pp. 554–560 (2006)
8. Gaber, M.M., Yu, P.S.: Detection and classification of changes in evolving data streams. International Journal of Information Technology and Decision Making 5, 659–670 (2006)
9. Minku, L.L., White, A.P., Yao, X.: The impact of diversity on online ensemble learning in the presence of concept drift. IEEE Transactions on Knowledge and Data Engineering 22, 730–742 (2010)
10. UCI Machine Learning Repository (2012), http://www.ics.uci.edu/~mlearn/MLRepository.html
11. Dai, B.-R., Huang, J.-W., Yeh, M.-Y., Chen, M.-S.: Adaptive clustering for multiple evolving steams. IEEE Transactions Knowledge and Data Engineering 18, 1166–1180 (2006)
12. Yeh, M.Y., Dai, B.R., Chen, M.S.: Clustering over multiple evolving streams by events and corrlations. IEEE Transactions Knowledge and Data Engineering 19, 1349–1362 (2007)
13. Chen, H.-L., Chen, M.-S., Lin, S.-C.: Catching the trend: A framework for clustering concept-drifting categorical data. IEEE Transactions Knowledge and Data Engineering 21, 652–665 (2009)
14. Chen, K.K., Liu, L.: HE-Tree:a framework for detecting changes in clustering structure for categorical data streams. The VLDB Journal 18, 1241–1260 (2009)
15. Nasraoui, O., Soliman, M., Saka, E., Badia, A., Germain, R.: A web usage mining framework for mining evolving user profiles in dynamic web sites. IEEE Transactions Knowledge and Data Engineering 20, 202–215 (2008)

Framework for Storing and Processing Relational Entities in Stream Mining[*]

Pawel Matuszyk and Myra Spiliopoulou

Otto-von-Guericke-University Magdeburg
Universitätsplatz 2
D-39106 Magdeburg, Germany
{pawel.matuszyk,myra}@iti.cs.uni-magdeburg.de

Abstract. Relational stream mining involves learning a model on relational entities, which are enriched with information from further streams that reference them. To incorporate such information into the entities in an efficient incremental way, we propose a multi-threaded framework with a weighting function that prioritizes the entities delivered to the learner for learning and adaption to drift. We further propose a generator for drifting relational streams, and use it to show that our framework reaches substantial reduction of computation time.

1 Introduction

Stream mining algorithms are gaining in importance on many research and application fields. However, traditional stream mining makes the limiting assumption that a data instance is seen only once. In relational learning, the same entity (a customer, a patient, etc.) is observed many times, each time associated with additional data that should be taken into account during learning. This raises new challenges: (a) how to enrich a relational entity with all information known about it, although space and time are limited? (b) how to choose the entities that are most important for learning, since we can never consider all of them? In this study, we propose a multi-threaded framework that prepares and prioritizes relational entities for stream mining.

Relational learning is a mature field, but has only recently been perceived in the stream learning context [SS09, FCAM09, SS10, SS11, IDDG11], although learning on a drifting stream of relational entities is demanded in many applications. Consider, for example, churn prediction as experienced e.g. in the telecommunications industry or in banking. Companies attempt to predict the likelihood that a customer will continue or discontinue a contract. Obviously, this prediction is regularly done for each customer, each time considering the new activities of the customer *and* combining them to past activities. Making the right prediction for a customer x allows to use the predictor on customers, who *later* behave similarly to x. We generalize this example to the *Stream Learning Problem for Relational Entities* as follows:

[*] Part of this work was funded by the German Research Foundation project SP 572/11-1 IMPRINT: Incremental Mining for Perennial Objects.

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 497–508, 2013.

> Let T be a stream of relational entities, which are in 1-to-n relation to instances from further streams $T_1, \ldots, T_m$. At timepoint t, learn/adapt a model $\zeta_T(t)$ on stream T, the *target stream*, exploiting the related instances from the other streams that have been seen thus far.

In the churn example, the customers constitute the target stream T; their service invocations, hotline requests, money transfers etc. constitute further streams of activities. These activities are exploited by $\zeta_T(t)$ to predict whether the label of customer x observed at t is CONTINUE or DISCONTINUE (i.e. quit the contract).

The stream learning problem for relational entities requires a tight and efficient coupling of stream preparation with stream mining. As mentioned at the beginning, space is limited and execution speed is an issue: how should a relational entity be expanded with all information seen on it thus far, so that it is made available to the learner? We propose a four-layer architecture with parallelizable components. Two layers of the architecture are responsible for (a) accommodating and (b) fetching relational entities from secondary storage, whenever activities referencing them are encountered; the other two layers are responsible for (c) queuing activities and (d) extracting information from the activities before discarding them. A core element of the framework is the weighting scheme responsible for the prioritization of relational entities that are expanded and delivered to the learner at each moment. The prioritization is motivated by the fact that the stream learner can never process all entities that have ever been seen, and by the fact that the model to be learned at a given moment should not consider entities that have not been referenced for a long while. Hence, our framework can be coupled with an arbitrary stream learner, whereby it handles the management and prioritization of past information for the relational stream.

In the next section we explain the context of this publication and related works. The new framework and its architecture are presented in section 3. Section 4 encompasses evaluation of the new framework and a description of a data generator, which has been developed for this purpose. In the last section we conclude and summarize the work and discuss a possible, further development.

2 Related Work

Learning on multiple interrelated streams is a very new problem, mostly called 'relational stream mining' or 'multi-relational stream mining'. In [SS09], Siddiqui et al. stress the fact that the relational entities re-appear, by calling them *perennial*, as opposed to the *ephemeral* transactions that reference these entities. We adopt this terminology, and use the terms 'perennial entity', 'perennial' and 'relational entity (of the target stream)' interchangeably hereafter; the terms 'instance' or 'ephemeral instance' we use for elements of the other streams.

First solutions to the Stream Learning Problem for Relational Entities, as we described it in the introduction, have been proposed by Siddiqui et al. in [SS09], where they studied stream clustering, and in [SS10], where they proposed a decision tree classifier for multi-relational streams. More recently, Ikonomovska et al. proposed following task: 'For each stream, determine the amount of facts that a relational incremental learner needs to observe at any point in time in order to be able to infer a correct model of the target function' [IDDG11, page 698]. Their goal is to minimize the information delivered to the learner, while our goal is to deliver only the entities relevant at some moment - but keep all information on these entities intact for the learner to exploit. Accordingly, we concentrate on lossless reconstruction of the relational entities.

However, the demand for recalling past entities does not always appear. For example, the method of Fumarola et al. on association rules discovery over multi-relational streams [FCAM09] only deals with the entities that are inside the window. The demand of recovering past entities emerges through the need to adapt the model on instances that reference entities seen earlier. For example, consider a decision tree based stream classifier that distinguishes between low-risk and high-risk customers. The customers constitute the *target stream*, the entities of which should be associated to the fastest transaction stream(s) of the customers[1] Whenever a new transaction for this customer appears, the customer's entity must be reconstructed and and its label must be predicted again, given the entire information known on this customer - the classifier may then predict for this customer a different label than it has predicted earlier. Multi-stream classification algorithms [IDDG11, SS10] either operate on the most recent data (at the cost of information loss, especially for entities that re-appear at very slow pace) or require an entity reconstruction algorithm.

Incremental entity reconstruction algorithms have been proposed in [SS09, SS11]. The former is an incremental version of a join-like operation called *propositionalization* [KRv$^+$03], which essentially extends the schema by turning values into columns/features.

The incremental propositionalization method of [SS09] ensures that the schema does not grow in an unbounded way by fixing the size of the feature space and clustering the values of similar entities into the fixed features. The CRMPES method of [SS11] rather identifies values that predict the label with classification rule mining, and uses these values as features. While this method requires much less space than the former, it is by nature sensitive to drift, and may lose important information on rarely re-appearing entities. In this study, we focus on entity reconstruction for multi-stream classification *without information loss.* We therefore build upon the former method, by establishing a database architecture that prepares the interrelated streams for adaptive learning and ranks entities on their expected importance for adaption.

In studies on database querying over streams, we find heuristics that prioritize the entities to compute approximate joins [DGR03, XYC05]. These heuristics predict which of the entities seen thus far will be needed for the join computation and which will not: the former are retained in the cache, the latter are kept in secondary storage. Their objective is to minimize the number of accesses to the secondary storage. Das et al. propose several heuristics that rank entities on the likelihood of being needed for the join computation [DGR03]: PROB assigns

[1] There may be more than one transaction streams: bank account transactions, transactions on loans, transactions with a credit card etc.

higher rank to those entities of the one stream that are frequently referenced from the other stream. LIFE [DGR03]computes for each entity the time it will stay inside the window[2]; LIFE chooses entities whose expected remaining 'lifetime' in the window is longer. The 'Heuristic of Estimated Expected Benefit' HEEB [XYC05] exploits past knowledge to compute the likelihood that an entity being observed now will be needed in near future. HEEB comes closest to the demands of our stream classification scenario: if an entity x of the target stream has been often referenced from the other streams in the past, it is likely that many instances refering to it will arrive in the future.

One pitfall of the HEEB weighting scheme is that it does not account for drift. In this study, we abstract drift as the *ageing* of some part of the model, until this part becomes completely obsolete. The anytime stream learner ClusTree uses an exponentially non-increasing function $\omega(\Delta t) = \beta^{-\lambda \Delta t}$ to assign an age-based weight to micro-clusters [KABS09]. One-stream classifiers based on VFDT [DH00] discard subtrees that have not received instances for a while [HSD01, GRM03], while the multi-stream classifier TrIP computes the age of a subtree on the basis of the age of the entities in it, since entities may re-appear [SS10]. We similarly take account of an entity's age with help of an exponentially non-increasing decay function, and re-juvenate entities as they re-appear.

3 Framework

Conventional stream mining algorithms learn a model by sliding a window over the arriving data instances. For the learning problem introduced in Section 1, the model is learned on the *permanent* relational entities and is updated as new instances referencing these entities arrive. Hence, instead of sliding a window over the arriving instances, we need to fetch the referenced entities for learning and adaption. We use a *cache* to accommodate the referenced entities: if a relational entity is referenced by an arrived instance, then this entity is fetched from the database to the cache, and is extended with the information of the arrived instance. This aggregation of new information is implemented as an incremental propositionalization mechanism [KRv$^+$03, SS09], which is part of our Four-Layer Architecture for relational entity preparation. This architecture and the functionalities of its components are depicted in Figure 1 and described below. In the following, we use the term 'perennial' for the relational entities of the target stream, to stress their permanent nature, and 'ephemeral' for the arriving instances, to stress that they are seen and forgotten, as is the case for conventional stream data [GMM$^+$03].

3.1 Four-Layer-Architecture

The Ephemeral layer consists of streams of ephemeral instances. It is responsible for supplying the framework with data. This layer can be physically distributed over many computers in a network. In the churn prediction scenario,

[2] As in stream classification, stream join is computed on the content of a window sliding over the streams.

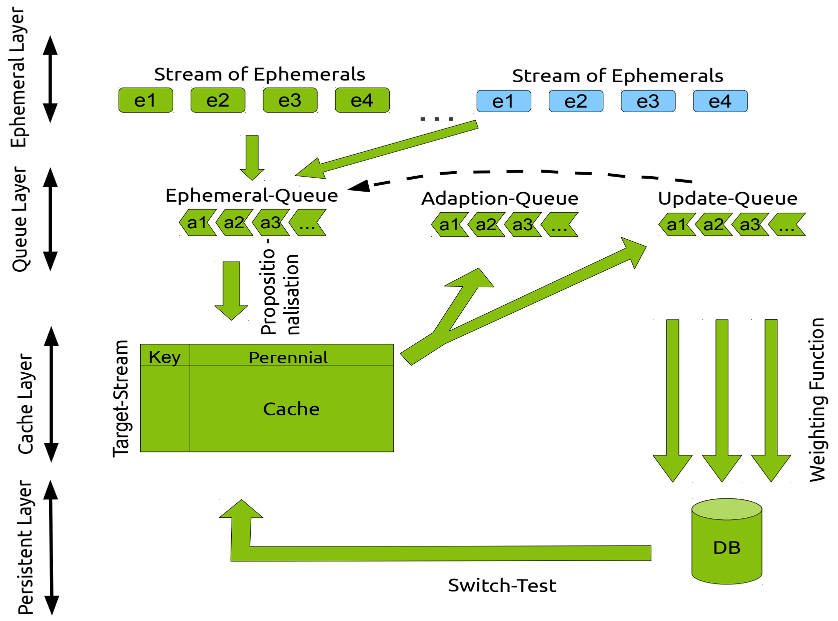


Fig. 1. Four-Layer-Architecture: Ephemerals (new data instances) are stored in the Ephemeral-Queue and propositionalised into perennials (entities). A subset of perennials is stored in the cache for faster access. It also serves as adaption mechanism. The Database as persistent storage is located in the persistent layer. The Queue layer is an intermediate layer that is responsible for paralellyzation and for decoupling the work of single components from each other.

for instance, the ephemerals represent money transfers, transactions or hotline requests recorded at different servers.

The Queue layer consists of three queues. The ephemerals delivered by the previous layer are stored in the Ephemeral-Queue, from where they are extracted by further threads of the framework. The extracted ephemerals are then propositionalised into the corresponding perennials using the mechanism of [SS09]. A selection of perennials is stored in the cache and used for model learning; all other perennials remain stored in the database, and fetched through an update action (carried out by a further thread group). The Ephemeral-Queue guarantees the separation of the mining algorithm from the stream management process. We have further queues, used for *actions for adaption and for updating*, which are performed by other threads:

- *Actions for adaption* encompass learning and forgetting. They process a perennial that should be presented to the mining algorithm and thus incorporated into a mining model, or one that should be forgotten by the algorithm and thus, excluded from the model. Such actions are launched every time a perennial enters or leaves the cache.

- *Update actions* serve the purpose of updating perennials stored in a database (cf. persistent layer). Therefore, they consist of a reference to a perennial and of an ephemeral that has to be aggregated into the referenced perennial.

The Queue Layer is responsible for the parallelization of processes within the framework. Each queue is served by its own thread group. The distribution of the tasks among many thread groups leads to the reduction of the computation time on multi-core processors.

The Cache layer serves following purposes. First, it speeds up the access to the perennials used for learning; these are selected with help of the weighting function presented thereafter. By delivering perennials to the learning algorithm, the cache layer is responsible for the process of model adaption to concept drift.

Second, the cache implements the operation of sliding a window over arriving stream instances. Since we study a learning problem upon relational entities, we cannot use a conventional window. Rather, as stream instances, i.e. ephemerals, arrive, the entities referenced by them (in the churn prediction example: already seen customers, or new ones) are presented to the mining algorithm. To prevent cache overflow and stick to the most important entities for learning, we define a weighting function that decides which perennials should enter the cache and which ones should be moved to the database.

Our framework allows to define an arbitrary weighting function. For example, in the churn prediction scenario, the human expert may decide that the most important customers for learning are not those observed most recently but those most active during the whole observation period. In this study, we propose following exponential function to compute the weight of a perennial p:

$$f_W(p) = (1 - \beta) \cdot e^{-p_a} + \beta \cdot (-e^{-p_s} + 1) \tag{1}$$

where p_a is the age of p, i.e. the elapsed time since the perennial was referenced for the last time, while p_s is the *support* of p, defined as the number of ephemerals referencing p thus far. The parameter β expresses the preferences of the analyst (or the decision maker) regarding relative importance of age versus support.

The Persistent layer is responsible for the management of perennials. The relational entities seen thus far may not be deleted, but it is not possible to keep all of them in main memory. Therefore, we use a database as a persistent storage. When a perennial is referenced by an ephemeral that has just arrived in the stream, then an "update action" is created and stored in the Update-Queue. From there the update actions are extracted by a thread group that is responsible for aggregating new information to the perennials, using the incremental propositionalization of [SS09]. After such an update, the weight of a perennial may change. If the new weight is larger than the lowest weight in the cache, then the perennial with the highest weight in the database replaces the perennial with the lowest weight in the cache. To avoid too frequent switches between database and cache, we perform a 'switch-test', where we check whether the difference between the two weights is significant.

Example 1. Consider the aforementioned churn prediction scenario, where the label of a customer has to be predicted. The challenge is here to combine all data that comes from multiple streams and relations belonging to this customer efficiently and learn upon those data in real time.

The process starts at the Ephemeral Layer, i.e. as a new transaction by the given customer arrives. Subsequently, this ephemeral object is stored in the Queue Layer, where it awaits the propositionalization. Hence, the corresponding perennial (the customer) has to be retrieved from the cache or from the database (Persistent Layer) and updated using the new transaction in course of the propositionalization. After that, the customer object can be stored back into the cache or into the database. The data mining model is kept consistent with the data in the cache, which plays the role of sliding window : every change of the data in the cache has to be followed by an update of the model. Therefore, an update action is created and stored in the Queue Layer. From there it is retrieved and performed by a different, parallel thread. Then, if the weight of the customer object (cf. Eq. 1) is high, the update action concludes by storing the object into the cache and incorporating it into the model. Finally, the object's label is derived.

3.2 Data Flow within the Framework

After explaining the internal structure and architecture of our framework, we now focus on the data flow and processes within the framework, as depicted in Figure 2. The work of the framework starts with the arrival of a new ephemeral instance, which is subsequently stored in the Ephemeral-Queue. From there it is extracted by another thread that carries out the propositionalisation. First, the location of the referenced perennial has to be determined. The thread checks whether the perennial is in the cache. If this is the case, the propositionalisation is performed immediately, the weight of the perennial is updated, and the perennial is saved back into the cache. If the referenced perennial is not in the cache, then it must be in the database. In this case, an update action has to be created and stored in the Update-Queue. Another thread group extracts the update actions from the queue and picks the referenced perennial(s) from the database.

Following case may occur: the perennial has been already moved into the cache, while the update action was waiting in the Update-Queue. Therefore, a further check is necessary. Thereafter, the perennial is updated using propositionalisation, and a switch-test is performed to check whether the perennial should be moved to the cache.

Cache overflow prevention: If the perennial has high weight (Eq. 1) and must be kept in the cache, we check whether the cache is overfilled. In such a case, we move the perennial with the lowest weight back to the database. Additionally, we launch a forgetting action (stored in the Adaption-Queue), so that this perennial is no more considered in the current model.

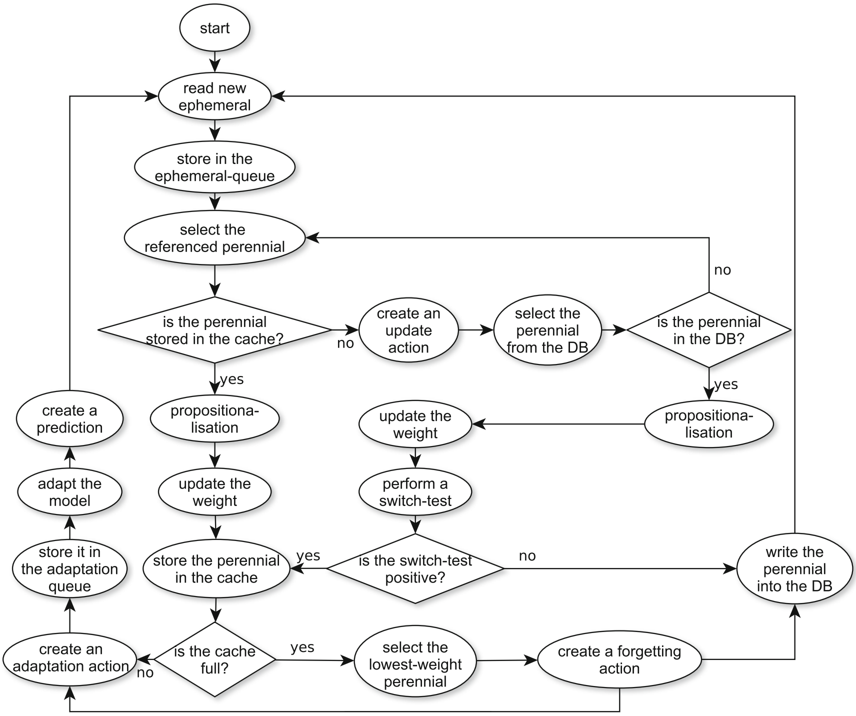

start
read new ephemeral
store in the ephemeral-queue
select the referenced perennial
is the perennial stored in the cache?
no
create an update action
select the perennial from the DB
is the perennial in the DB?
no
yes
propositiona-lisation
yes
propositiona-lisation
update the weight
update the weight
perform a switch-test
create a prediction
adapt the model
store it in the adaptation queue
store the perennial in the cache
yes
is the switch-test positive?
no
write the perennial into the DB
create an adaptation action
no
is the cache full?
yes
select the lowest-weight perennial
create a forgetting action

Fig. 2. Data flow within the architecture; processes can be parallelized (e.g. propositionalisation of perennials and database accesses can be performed simultaneously)

If a new perennial is saved in the cache, then also an adaption action is put into the Adaption-Queue. The last group of threads executes the actions in the Adaption-Queue and predicts the perennials' labels. This is repeated as long as there are new unprocessed ephemerals, whereby the threads run in parallel.

4 Experiments

To evaluate the performance of our framework in a controlled way, we developed a data generator that creates streams that change their speed and exhibit drift. The generator is presented first. We then present the experiments on execution speed and result quality, for which we coupled our framework with the relational stream classifier TrIP [SS10]. We used an Intel i5 with 2.4 GHz (2 cores and 4 parallel threads) and 4 GB RAM.

4.1 Data Generator

Our data generator takes as input a number of perennials and generates a stream of ephemerals which reference them. Although the streams are by definition infinite, an obviously finite number of ephemerals is specified in order to ensure that the generator terminates its work and results can be seen.

The number of classes of perennials and the number of feature space dimensions are also input parameters. The classes follow the Gaussian distribution. Furthermore, the generator simulates concept drift by shifting the μ parameter of the class distribution by a given value. We introduce several parameters to govern concept drift, and use a simple notation, which we call 'drift string', to set them. The parameters are: the time point s when the concept drift starts; the time point e when the concept drift ends; the velocity of the drift v (number of units a class center is shifted at each time point); the class c affected by the drift, and the attribute a affected by the drift. Hence, the drift string has the form: $s < s > e < e > v < v > c < c > a < a >$.

4.2 Reducing Computation Time through Multi-threading

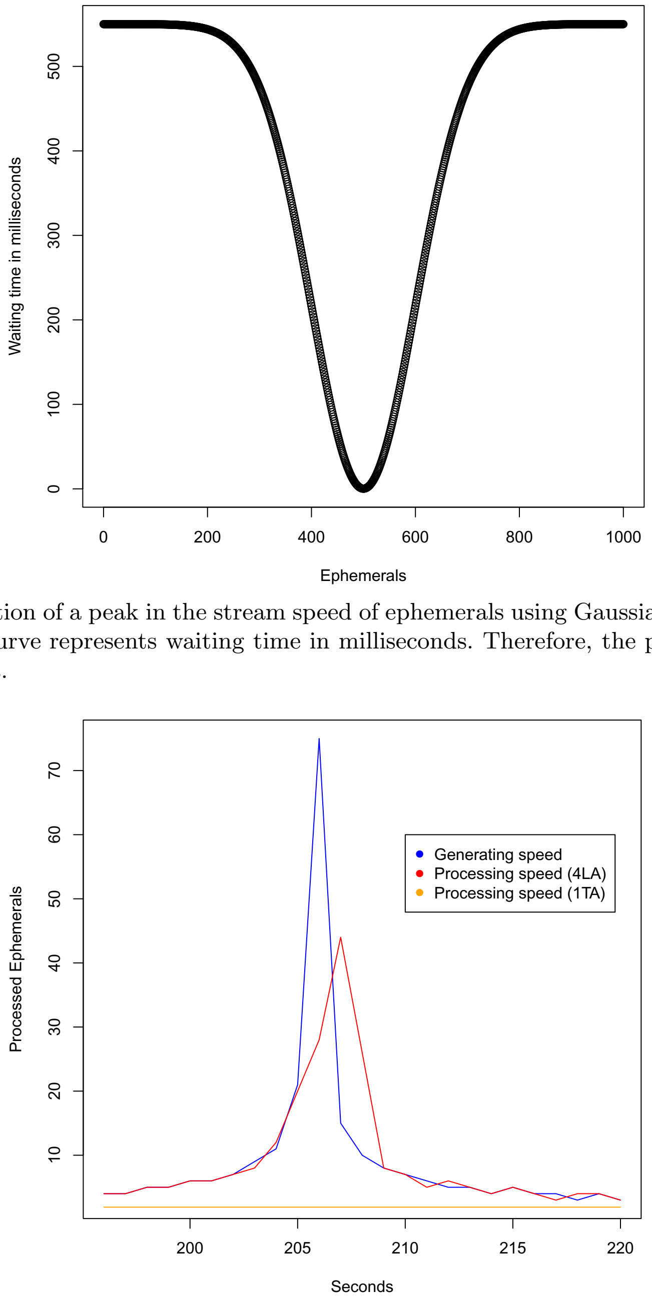
To measure how our four-layer-architecture (4LA) reduces computation time, we implemented a baseline one-thread-architecture (1TA) with the same functionality as 4LA.

The first parameter affecting the computation time of the framework is the cache size. When the cache is small, then many perennials have to be moved to the database, thus increasing computation time. The other important parameter is the number of perennials: if they are only few, then it is more likely that a new ephemeral will reference a perennial that is already stored in the cache.

In Table 1, we show the impact of the parameters on the computation time of 4LA and 1TA. Two cases have to be distinguished. In the first case, the cache is so large that it accommodates all perennials (black numbers, below the diagonal). In the second case, the number of perennials exceeds cache size (blue numbers, above the diagonal), so some perennials had to be moved to the database, increasing computation time. Red numbers denote the best relative

Table 1. Computation time of 4LA and 1TA in milliseconds: lower values are better, best improvements marked in red. When the cache is too small (above the diagonal), computation time includes data swapping.

Number of Perennials		2	10	100	1000	10 000
Cache Size (as number of perennials)	3 (1TA)	15803	525368	689051	666646	543933
	3 (4LA)	1253	26479	258546	605514	499776
	10 (1TA)	24298	75260	616953	676923	514007
	10 (4LA)	1510	9148	58591	569834	460868
	100 (1TA)	24537	25193	34744	579670	494098
	100 (4LA)	1340	1137	1602	249130	439244
	1000 (1TA)	18516	33919	29059	32481	382275
	1000 (4LA)	1080	972	1135	1124	308691
	10 000 (1TA)	20921	25042	26350	26307	38591
	10 000 (4LA)	887	888	1086	1103	1342



(a) Simulation of a peak in the stream speed of ephemerals using Gaussian distribution; the curve represents waiting time in milliseconds. Therefore, the peak points downwards.

(b) Generation and processing speed of 4LA (red) and 1TA (blue); the red curve has a lower peak and exhibits a time lag over the peak in the stream speed; the peak of the blue curve is at the same time point as the stream peak.

Fig. 3. Effect of smoothing a peak in stream speed

improvement of 4LA over 1TA. In the first case (below the diagonal), the improvement on computation time reached 97.13%. In the second, more realistic case, the relative improvement reached 94.96%, i.e. 4LA needed only 5.04% of 1TA's computation time.

4.3 Dealing with Speed-Up of the Stream

A further advantage of our 4LA framework is that it smooths temporal speedups of the ephemeral streams. We have simulated such a speedup by reducing the elapsed time between the arrival of two ephemerals, and compared the computation time of 4LA to 1TA. When the stream speed becomes higher than the processing speed of the miner, then further actions that cannot be carried out immediately, are cumulated in the queues. Following experiment shows that our new architecture can cope with a temporal speed-up of the stream.

A peak in the speed of the stream of ephemerals was simulated using normal distribution (cf. Figure 3a) over the waiting time between generating two ephemerals. Thus, the peak of the distribution points to the bottom of the page. Thereafter, the processing speed of the 4LA and 1TA were measured. The stream speed is represented by the blue curve in Figure 3b. It is apparent that the processing speed of the 4LA (red curve) is not as high as the stream speed. An advantage of the new framework shows at this stage - the framework does not collapse under the high load of ephemerals, but it rather starts to cumulate the ephemerals in queues. When the speed of the stream becomes lower again, the framework processes the ephemerals from the queues, what is apparent from the shift of the red curve to the right of the blue curve. For the contrast, the maximal processing speed of the 1TA has been depicted by the orange line in Figure 3b.

5 Conclusions

We described a novel framework for storing and processing relational entities in stream mining, based on a four-layer-architecture. Due to a partial parallelization of processes within the framework an essential reduction of computation time was possible. The computation time was reduced by up to 97.13%. Thanks to the usage of the queue layer the framework gained the ability to cope with streams witch changing speed - an issue of particular relevance for real-world scenarios. Furthermore, we used a new weighting function that prioritizes the entities to be used for learning, giving preference to those most recently referenced.

Our framework is scalable and appropriate for multi-core processors. As future work, we want to extend it so that it runs on many computers in a network. This will not only reduce computation time, but it also will allow us also to delegate part of the stream classification itself to a network of computers. While distributed data mining has been widely investigated for static data, stream mining and the incremental propositionalization step in the preprocessing phase incur additional coordination overhead that has yet to be investigated.

References

[DGR03] Das, A., Gehrke, J., Riedewald, M.: Approximate join processing over data streams. In: SIGMOD Conference, pp. 40–51. ACM (2003)

[DH00] Domingos, P., Hulten, G.: Mining high-speed data streams. In: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 71–80. ACM (2000)

[FCAM09] Fumarola, F., Ciampi, A., Appice, A., Malerba, D.: A sliding window algorithm for relational frequent patterns mining from data streams. In: Gama, J., Costa, V.S., Jorge, A.M., Brazdil, P.B. (eds.) DS 2009. LNCS, vol. 5808, pp. 385–392. Springer, Heidelberg (2009)

[GMM$^+$03] Guha, S., Meyerson, A., Mishra, N., Motwani, R., O'Callaghan, L.: Clustering data streams: Theory and practice. IEEE Transactions on Knowledge and Data Engineering, 515–528 (2003)

[GRM03] Gama, J., Rocha, R., Medas, P.: Accurate decision trees for mining high-speed data streams. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 523–528. ACM (2003)

[HSD01] Hulten, G., Spencer, L., Domingos, P.: Mining time-changing data streams. In: Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 97–106. ACM Press (2001)

[IDDG11] Ikonomovska, E., Driessens, K., Dzeroski, S., Gama, J.: Adaptive windowing for online learning from multiple inter-related data streams. In: Proc. of Int. Workshop on Learning and Data Mining for Robots (LEMIR 2011) at the 11th IEEE Int. Conf. on Data Mining Workshops Volume, Vancouver, Canada, pp. 697–704 (December 2011)

[KABS09] Kranen, P., Assent, I., Baldauf, C., Seidl, T.: Self-adaptive anytime stream clustering. In: Ninth IEEE International Conference on Data Mining, ICDM 2009, pp. 249–258. IEEE (2009)

[KRv$^+$03] Krogel, M.-A., Rawles, S., Železný, F., Flach, P.A., Lavrač, N., Wrobel, S.: Comparative Evaluation of Approaches to Propositionalization. In: Horváth, T., Yamamoto, A. (eds.) ILP 2003. LNCS (LNAI), vol. 2835, pp. 197–214. Springer, Heidelberg (2003)

[SS09] Siddiqui, Z.F., Spiliopoulou, M.: Combining multiple interrelated streams for incremental clustering. In: Winslett, M. (ed.) SSDBM 2009. LNCS, vol. 5566, pp. 535–552. Springer, Heidelberg (2009)

[SS10] Siddiqui, Z.F., Spiliopoulou, M.: Tree induction over perennial objects. In: Gertz, M., Ludäscher, B. (eds.) SSDBM 2010. LNCS, vol. 6187, pp. 640–657. Springer, Heidelberg (2010)

[SS11] Siddiqui, Z.F., Spiliopoulou, M.: Classification rule mining for a stream of perennial objects. In: Bassiliades, N., Governatori, G., Paschke, A. (eds.) RuleML 2011 - Europe. LNCS, vol. 6826, pp. 281–296. Springer, Heidelberg (2011)

[XYC05] Xie, J., Yang, J., Chen, Y.: On joining and caching stochastic streams. In: Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data, SIGMOD 2005, pp. 359–370. ACM, New York (2005)

Discovering Semantics from Multiple Correlated Time Series Stream

Zhi Qiao[1,2], Guangyan Huang[1], Jing He[1], Peng Zhang[2], Li Guo[2],
Jie Cao[3], and Yanchun Zhang[1]

[1] Victoria University, Melbourne, Australia
[2] Institute of Information Engineering, Chinese Academy of Science, Beijing, China
[3] Nanjing University of Finance and Economics, China
zhiqiao.ict@gmail.com, {zhangpeng,guoli}@iie.ac.cn,
{Guangyan.Huang,Jing.He,yanchun.zhang}@vu.edu.au, caojie690929@163.com

Abstract. In this paper, we study a challenging problem of mining data generating rules and state transforming rules (i.e., semantics) underneath multiple correlated time series streams. A novel Correlation field-based Semantics Learning Framework (CfSLF) is proposed to learn the semantic. In the framework, we use Hidden Markov Random Field (HMRF) method to model relationship between latent states and observations in multiple correlated time series to learn data generating rules. The transforming rules are learned from corresponding latent state sequence of multiple time series based on Markov chain character. The reusable semantics learned by CfSLF can be fed into various analysis tools, such as prediction or anomaly detection. Moreover, we present two algorithms based on the semantics, which can later be applied to next-n step prediction and anomaly detection. Experiments on real world data sets demonstrate the efficiency and effectiveness of the proposed method.

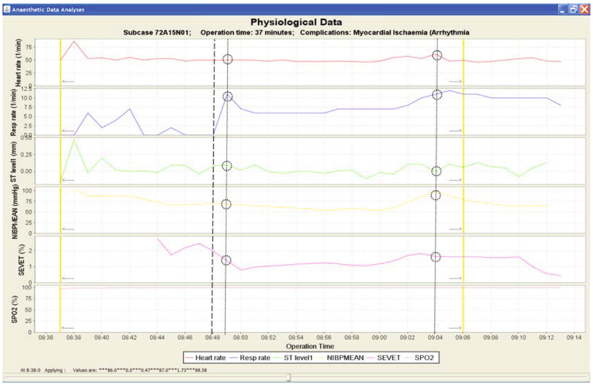
Keywords: Semantics, correlated time series streams, prediction, anomaly detection.

1 Introduction

Time series data have emerged in a wide range of applications from almost every domain. Examples include economic index data in stock markets, patient medical observation data, experimental biological data, to name a few. As a result, it is of utmost importance to find inherent semantics from time series data. Take the medical examination for example, doctors often estimate the physical status of a patient by monitoring and collecting multiple correlated time series data from electrocardiograms (ECG), electroencephalograms (EEG), heart beat rate (HR), and blood pressure observations. Obviously, it is very hard to make accurate estimation by only relying on a single time series data (e.g., HR increases does not mean the patient is suffering from severe illness, perhaps she/he just had exercises). Hence, it is necessary to combine all these time series data for estimation.

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 509–520, 2013.

Many methods have been proposed to analyze multiple time series in [3][5][11]. In addition, there are some popular time series models for time series forecasting, such as the Vector Auto-regression (VAR) and Linear-regression (LR) models. These approaches focus on frequent patterns in the time series, which cannot explain observations using the internal dynamics of systems. The dynamics of a system can be considered as a mechanism of system-work as equivalent to semantics. Here, the system is unseen and unknown, which determines observed time series. It can help us to know more about observation generating rules and state transforming rules underneath data by learning the mechanism. A previous work [12] also studies the semantics detection problem from time series data. The difference is that it uses the pattern-based Hidden Markov Model (pHMM) to describe the univariate time series data where a line segmentation method is used to obtain significant segment patterns. In our work, the patterns are irregularly summarized from multiple time series, and we use the Hidden Markov Random Field (MRF) as the solution.


Physiological Data

Subcase 72A15N01; Operation time: 37 minutes; Complications: Myocardial Ischaemia (Arrhythmia

At 8:38:0 Applying : Values are: ***86.0***0.0***0.47***87.0***1.73***98.58

Fig. 1. Real Medical Case (All signals are sampled in a minute unit)

Supposing all time series are observed synchronously at each time point, observations from all time series compose of a tuple at each moment and each tuple is regarded as an output generated by a certain latent state. Each state demonstrates a certain pattern of fluctuation. In this paper, the pattern of fluctuation is taken as a generating rule, conforming with generated observations. Semantics learning basically learns both observation value generating rules and transformation rules among latent states. It is widely admitted that the Hidden Markov Model (HMM) can be used to learn semantics. In HMM, state assignment is mainly determined by aligning observation value production and state transmission. However, a single observation tuple value contains little information. We illustrate this by a real medical example in Figure 1. Figure 1 demonstrates 6 vital body signals of a patient in an operating theatre. At the t_0 time point, the operation starts. Apparently, observation values at t_1 are very similar with observation values at t_2. Actually, they represent different situations of the patient. At t_1, the observation value represents a natural reaction of the patient to an outside emergency which can be regarded as the operation. At t_2, the observation value represents physical state of patient during the operation. If we only consider observation values of one tuple, we cannot obtain significant semantics for a practical problem. Actually, what is different from other common data is that time series data has natural temporal ordering. The temporal ordering of observation data is not directly considered in HMM. According to the temporal ordering of time series, each observation tuple is strongly correlated with the previous tuple. In Figure 1, it is obvious that correlation between t_1 tuple and $t_1 - 1$ tuple is different from correlation between t_2 tuple and $t_2 - 1$ tuple.

Hence, we propose Correlation field-based Semantics Learning Framework (CfSLF) to learn semantics underneath multiple correlated time series in this paper. In the framework, generating rules and state transforming rules will be learned. In order to comprehensively considering both temporal ordering and observation value of tuple, we use hidden Markov Random Field (HMRF) to obtain approximately optimal latent state assignment to learn generating rules. Then, state transforming rules are learned from label sequence in the framework.

The rest of the paper is structured as follows. Section 2 introduces a mathematical description of the problem. Section 3 describes detailed proposed CfSLF. Section 4 introduces two main applications: Observation Value Prediction & Anomaly Detection. Section 5 reports experimental results to show the advantage of our CfSLF model compared with some other algorithms. Section 6 introduces related works. We conclude the paper in Section 7.

2 Problem Setting

In this paper, we propose a Correlation field-based Semantics Learning Framework (CfSLF) to learn latent semantics underlying multiple time series which represent a mechanism of system work. It contains two parts: generating rules learning and state transforming rules learning.

Given multiple time series $X = \{X_1, ..., X_n\}$, $X_i = \{x_{i1}, ..., x_{im}\}$. X_i represents an observed tuple from the m time series at the i^{th} moment and x_{ij} represents observed data from the j_{th} time series at i_{th} moment. Assume that there exists a state set $S = \{s_1,..., s_k\}$. Each tuple is produced by one state of S. $Z = \{z_1, z_2, ..., z_n\}$ is a latent label set. z_i is a label variable, which is discrete and represents latent state of the i_{th} tuple. The value of the variable is in the range from 1 to k. Assume z_i=j, which represents that the latent state of the i_{th} tuple is the j_{th} state s_j. In order to learn data generating rules, we need to obtain optimal latent state assignment and the corresponding generating rules to maximize production probability of observation tuples. The problem can be described as follow:

$$\hat{Z} = argmax_Z P(X, Z) = argmax_Z P(X|Z)P(Z) \tag{1}$$

which is maximized to obtain state assignment. According to the continuity of tuples sequence, the label variable set constituted of finite states can be seen as a latent state sequence. After we obtain optimal state assignment, transforming rules among latent states are learned from the state sequence based on Markov chain characteristics.

3 CfSLF Learning

In this section, we discuss how to learn semantics by our proposed CfSLF.

3.1 Construction of Dependence Relationship

In time series, temporal ordering can be seen as the relationship between the current and previous tuples. In this paper, we simply consider temporal ordering as changing the trend to represent the relationship. We introduce a definition of a local trend as follow,

Definition 1. *Local Trend: In time series, we consider the direction and volume of changing from the last node to the current node as the local trend along time axis.*

Suppose that the current tuple is x_t with observation value o_t. We directly obtain local trend of by $T_{d_t} = o_t - o_{t-1}$. For the local trend of the tuple, we apply the attribute to represent temporal ordering of the current tuple. Then, we simply use the cosine distance to measure the similarity among local trends of different tuples. Intuitively, continuity of time series can be considered as integrating temporal ordering of all tuples. We use the local trend of tuple to represent temporal ordering, which can be seen as an independent attribute of each tuple. In doing so, each tuple has two attributes: its observation value and its local trend. For each tuple, the observation value represents its individual character and local trend represents sequence character. As a result, the tuple series with size n is divided into n independent observation tuples

3.2 Latent State Assignment

In the procedure of state assignment, tuples with similar local trends and observation values are more likely to have the same latent state. Therefore, we use HMRF to learn the rules. A correlation field is built based on local trend similarity. In the correlation field, the assignment of labels depends on corresponding brotherhood set. In this paper, the similarity matrix is considered as a correlation network describing dependency relationship among latent labels of tuples in the correlation field. In the network, similarity is seen as a weight of dependency relationship.

Suppose that we have a correlation network denoted as M. Here, M is the symmetric $n \times n$ matrix, where w_{ij} is the link weight between labels z_i and z_j.

The links in M induce dependence relationships among latent labels, with the rationale that if the link weight is higher between labels z_i and z_j, then they are more likely to have the same value equivalent to the same state.

We define a brotherhood set of the i_{th} label as a label set consisting of labels which have the same latent state value, B_i=$\{z_j$,$i \neq j$ & $z_i = z_j\}$. The random field defined over hidden label variable Z is a Markov random field, where the Markov property is satisfied by $p(z_i|z_{B_i})$. It indicates the probability of z_i depending on z_i's brotherhood set. By introducing the HMRF model, latent labels of tuples are mapped to a correlation field, where assignment of labels depends on a corresponding brotherhood set without considering tuple value.

Because the observation value of tuple is generated by the corresponding state, it is irrelevant to other states and only depends on its latent state. Thus, the values of tuples are conditional independent given their labels.

$$P(X|Z) = \prod_{i=1}^{n} p(x_i = o_i|z_i) \tag{2}$$

We assume that the observation value of the i_{th} tuple generated by the k_{th} latent state is characterized by a set of parameters θ_k, i.e., as we mainly consider multiple time series as real data, we propose to model observation data by Gaussian distribution, because of its flexibility in approximating a wide range of continuous distributions. Therefore, we use the parameter mean vector μ and variance matrix $\sum$ to describe the kth latent state, $\theta_k = (\mu_k, \varSigma_k)$.

We first assume model parameters λ=$\{$ $\varTheta_i$, i from 1 to k$\}$ are known a prior. In order to obtain approximately optimal assignment of latent variables for each observation tuple, we transform to find the optimal configuration that maximizes the posterior distribution given λ.

As discussed in Eq. (1), the probability distribution of Z is given by $P(Z) = exp(\gamma \sum \omega_{ij} \delta(z_i - z_j))/H$.

In the above equation, H is a constant value, which can be neglected. We use the Iterated Conditional Modes (ICM) algorithm [5] to estimate the maximum a posteriori probability (MAP). The greedy algorithm can be used by calculating local minimization iteratively, which converges after a few iterations. The basic idea is to sequentially update the label of each object, keeping the labels of the other objects fixed. At each step, the algorithm updates z_i given x_i and the label by maximizing the conditional posterior probability, $p(z_i|x_i = o_i, Z_{I-\{i\}})$.

$$p(z_i|x_i = o_i, Z_{I-\{i\}}) = p(x_i|z_i = s) \times \frac{exp(\gamma \sum_{z_i=s} \omega_{ij} \delta(z_i - z_j))}{H_2} \tag{3}$$

Actually, H_2 can be considered as a constant variable. In doing so, we take the logarithm of the posterior probability, and transform the MAP estimation problem into the minimization of the conditional posterior energy function as shown in the following equation,

$$U_i(k) = -ln(p(x_i|z_i = k)) - \gamma \sum_{j \in B_i} \omega_{ij} \delta(z_i - z_j) \tag{4}$$

where γ is a predefined parameter that represents the importance of the temporal ordering correlation. $\gamma > 0$ represents the confidence of the temporal ordering correlation network. To minimize $U_i(k)$, we find the latent state k of the tuple i by $k = argmax_k U_i(k)$.

3.3 Parameter Estimation

In this part, we consider the problem of estimating unknown λ in order to iteratively learn optimal state assignment. λ describes the pattern conformed with the time stamp that x is generated. We first seek to find λ to maximize $P(X|\lambda)$, which can be considered as the maximal likelihood estimation for λ. However, since both the hidden label and the parameter are unknown and inter-dependent, it is intractable to directly maximize the data likelihood. We view it as an "incomplete-data" estimation problem, and use the Expectation-Maximization (EM) algorithm as the solution.

The basic procedure is as follows. We start with an initial estimate λ_0. Assume that there exist k latent states, where λ_0 is obtained by a simple K-Means algorithm. In the E-step, we calculate the conditional expectation $Q(\Theta|\Theta(t))$,

$$Q(\lambda|\lambda_t) = ElnP(X, Z|\lambda_t) = \sum_Z \{P(Z|X, \lambda_t) \times ln(X, Z|\lambda_t)\} \tag{5}$$

Next, in the M-step, we find λ_{t+1} by computing the derivation of the maximizing function $Q(\lambda|\lambda_t)$.

$$\mu_j^{t+1} = \frac{\sum_Z \sum_{i=1}^n \{p(z_i = s_j))|x_i, \lambda_t \times x_i\}}{\sum_Z \sum_{i=1}^n \{p(z_i = s_j|x_i, \lambda_t)} \tag{6}$$

$$\Sigma_j^{t+1} = \frac{\sum_Z \sum_{i=1}^n \{p(z_i = s_j))|x_i, \lambda_t \times (x_i - \mu_j^{t+1})(x_i - \mu_j^{t+1})^T\}}{\sum_Z \sum_{i=1}^n \{p(z_i = s_j|x_i, \lambda_t)} \tag{7}$$

In each iteration, we have obtained the optimal latent state assignment from the last step. Assume the latent state of tuple i is state j, given λ_t. Thus, $p(z_i|x_i, \lambda_t)$ is 1 when z_i is at state j, else 0. As a result, the E-step and M-step can be recursively computed until $Q(\lambda|\lambda_t)$ converges to a local optimal solution.

3.4 State Transforming Learning

After we obtain the optimal latent state assignment, each observation tuple is assigned a label corresponding to the latent state set of tuples. The labels can be seen as a sequence consisting of limit states along the time axis. We then model correlation among states to reveal system dynamics. Here we regard correlation as the transforming probability $p(s_i|s_j)$ representing the probability from the state j to state i. A Markov chain model can be used to approximately estimate the transforming probability among states by using $p(s_j|s_i) = N(s_i s_j)/N(s_i)$, where $N(s_j)$ represents the amount of labels with s_j value in label series and $N(s_j s_i)$ represents the amount of adjacent labels with s_j and s_i values in label series.

4 Applications of the Model

Observation Value Prediction. Time series semantics can be used to make the following value prediction. We first introduce the next 1-step value prediction. Assume that we have learned the semantics from the training data, we then have λ and the state transformation rule. In test step, we consider time series X, as we care about the local trend of tuples, let $x_{t-1}$and x_t are current tuples. Our task is to predict x_{t+1}. We first compute the current latent state of x_t. When we assign a label to the current tuple, we need to predict the label of the next tuple according to the Markov chain characteristics. Assume the current label z_t is s_c, the next label z_{t+1} can be obtained by maximizing $p(z_{t+1} = i|z_t = s_c)$. Additionally, according to time series continuity, we can estimate that the next tuple close to the current tuple with high probability. Therefore, it is safe to say that the next state maintains the continuity with high probability. That is, next state can be predicted by,

$$\hat{Z}_{t+1} = argmax_i p(\dot{x}_{t+1} = x_t) \times p(z_{t+1} = i|z_t = s_c) \tag{8}$$

Because observation tuples produced by a state have similar values and similar trends, we approximately predict observation values of the state by,

$$\hat{x}_{t+1} = argmax_x |x - E_{z_{t+1}}| + |x - x_t| \times |\frac{x - x_t}{|x - x_t|} - \frac{V_{z_{t+1}}}{|V_{z_{t+1}}|}| \tag{9}$$

where $E_{Z_{t+1}}$ represents the Expectation of the observation value of the predicted state, and $V_{Z_{t+1}}$ represents the exception of the local trend of the predicted state. We use the Euclidean distance to measure similarity. By computing the derivative of function, the prediction value x can be obtained. Then we extend next 1-step value prediction to the next n-step value prediction. We iteratively apply the predicted value as the new observation value to forecast the next value until n steps have been performed.

Anomaly Detection. Our proposed model also can be used to detect data anomaly. In time series, there is no apparent and definite label to represent which observation is normal or abnormal. So, it is not a classification problem. Generally speaking, we only know that anomaly occurs in a certain period. Take a finance application for example. The worldwide economical recessions have occurred several times in history. The recession always lasts for a period of time, which is regarded as recession date. It impacts all business activities. Compared with economic affairs in other periods, economical affairs in recessions can be seen as anomaly. Hence, we propose the method based on a semantics model only qualitatively to indirectly reflect anomalies.

According to time series continuity and semantics rules, we know that the current tuple is similar to the last tuple with high probability, and the current state has high transformation probability from the last state. Considering both rules, we can measure the probability with which the current tuple normally is generated by the following equation: $f(x_t) = p(x_t|z_{t-1}) \times p(z_t|z_{t-1})$. A logarithm function is generally used to obtain a degree of the energy. Thus, we compute the

Table 1. Runtime and Accuracy Comparison

N	0.05	0.1	0.2	0.5	0.8	1
H-Runtime	2.223	4.727	5.82	13.413	30.568	49.377
H-Error	0.0259	0.0152	0.0129	0.0113	0.0112	0.0111
V-Runtime	1.915	4.9590	3.785	10.65	19.315	26.024
V-Error	0.115	0.0117	0.0092	0.0078	0.0064	0.0059
T-Runtime	0.845	16.433	25.635	73.180	136.077	180.367
T-Error	0.0389	0.0265	0.0211	0.0169	0.0163	0.0152

H(Humid data) V(Volt data) and T(Temp data)

anomaly score of the current tuple by $-logf(x_t)$, which measures the significance the current tuple deviates from that emanating from the normal producing rule. Intuitively, high scores indicate anomalous data with high probability.

5 Empirical Evaluation

In this section, we present extensive experiments on real-world multiple time series data to validate performance of our proposed approach. All experiments are conducted on a 3.0GHZ CPU with 2 GB RAM. The experimental environment is windows XP with Matlab.

Benchmark Data. We adopt four multiple time series data as our test-bed: 1) Price data[1] 2) Mote data sets[2] 3) Medical data[3] 4) Financial data[4]

Baseline Methods. As discussed above, our proposed semantics learning method can be applied to value prediction. We compare our method with following value prediction algorithms: (1) Multivariate Autoregression Model. (2) Hidden Markov Model (HMM).

5.1 Experiment Results

We first consider the time cost of model learning and hyper parameter λ sensitivity for data. Then we compare our proposed model with benchmark method on multiple step prediction accuracy. Additionally, our proposed model can be used to indirectly reflect latent anomalies, which are hard to see from the original data. Here we mainly analyze financial data to discover financial mark depression.

[1] The data set consists of the Reference Price Data (RPD) for APX Power UK Spot market, which can be downloaded from Website `http://www.apxendex.com/index.php?id=466`.

[2] Mote data sets are collected using Berkeley Mote sensors, at several different locations in a lab, over a period of a month. For each category of data, we just select four time series.

[3] It consists of 11 medical time series of patient from an Australian Hospital.

[4] It is constituted of 11 economical time series GS1, DTB3, TB3MS, WTB3MS, GS5, GS10, MPRIME, WPRIME, FEDFUNDS, AAA and BAA respectively, which are obtained from the Website of the Federal Reserve Bank of St. Louis.

We use relative error to measure accuracy of prediction. We compute the relative error by $|\hat{x}_i - x_i|/|x_i|$ where $|x_i|$ is the estimated value, and x_i is the real value. Thus, the lower relative error is, the higher the accuracy is.

Time Complexity. Suppose the number of tuples is N in multiple correlated time series. In M-step,the time complexity is O(N). In E-step, the time complexity is O(N_2) because of aggregating the effect of the labels of brotherhood set of v_i to compute $P(Z)$. Actually, semantics rules seen as latent pattern repeatedly exist in the multiple time series. Hence, we do not need to learn a model based on entire training set. We can approximately obtain the semantics from part of the data set. In the experiment, we conduct our proposed model on all of benchmark data set, and compare the runtime and next 1-step value prediction accuracy under different N, the selection ratio of boundary points. For example, assume size of train-ing is 2000, and 0.05 means we choose 0.05*2000 =100 time points to train the model. The results are shown in Table 1. In Table 1, the error is average relative error for applying learned CfSLF to 200 testing tuples sampled from testing set. It can be seen that runtime gets longer and accuracy gets higher when ratio N gets bigger. We can see that in every data set, when the percentage is equal to or larger than a certain value, the accuracy is not affected much.

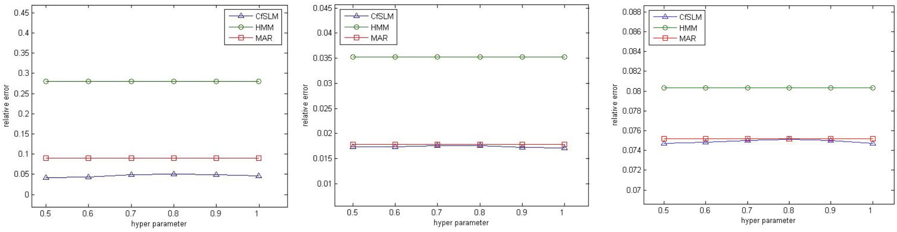


Fig. 2. Hyper parameter λ sensitive

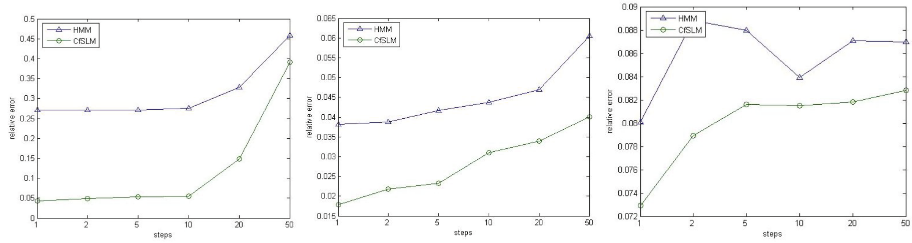


Fig. 3. Next n-step prediction comparison

Hyper Parameter Sensitivity. In our proposed model, parameter λ represents the confidence of the temporal ordering correlation network. Different values of λ determine different effects of the corre-lation field. We separately conduct experiments on all of the data sets to demonstrate hyper parameter λ setting and effect. In each experiment, we vary λ from 0.1 to 2 separately, and compute the corresponding relative error of the next 1-step value prediction under the predefined λ. HMM is used as baseline method to compare with our proposed method. The outcome is shown in Figure 2. It can be seen that there are slight changes in performance when parameters are varied and CfSLF has better performance than the HMM and Multivariate Autoregression models.

Application of the Model. In the former section, we have tested performance on the next 1-step value prediction. In the following, we will discuss additional next n-step value prediction and anomaly detection.

Next n-step Value Prediction. In this experiment, we test the accuracy of CfSLF for the next n-step value prediction. The experiments are conducted for all data sets. We select 10 points randomly. For each selected point, we predict the values after 1, 2, 5, 10, 20, and 50 steps respectively. The CfSLF is compared with a Hidden Markov Model (denoted by HMM). The HMM can be used to learn system-work mechanisms underlying time series. In the experiment, we suppose that each tuple is produced by a latent state and the producing procedure conforms to Gaussian distribution. We first find latent state of the tuple at a selected time point, then make state predictions at 6 future time points. Prediction value is corresponding expectation of predicted state. We use relative error as the measurement. The results are shown in Figure 3. It can be seen that on all data sets, CfSLF is more accurate than HMM.

Anomaly Detection. In the experiment, we use CfSLF on financial data to verify its performance for anomaly detection. The experimental results indicate that the proposed method detected deviations from that emanating from the normal producing rule as anomalies and these corresponded to actual economic events. The degree of anomaly in these time series is shown in Figure 4. In Figure 4, we see that two apparently peak deviated from other scores. Each peak corresponded to big economic events occurring in corresponding month. The first peak appeared on January 2008, where the Federal Reserve lowered its federal funds rate, which impacts how much consumers pay on credit card debt, home equity lines of credit and auto loans, to 3.5 percent from 4.25 percent, which was the biggest rate cut by the Fed since October 1984. The second peak appeared on September 2008, where Lehman Brothers announced its bankruptcy. The second peak indicates that the proposed method detected the depression whictarted in September 2008, as anomalies.

6 Related Works

There are many works on analysing time series, such as summary learning, time series segmentation, forecasting and so on, which have always been popular topics [8][9][11][12][13][14][15][16]. However, they just can be used to analyse single time series. Additionally, pattern learning from time series based on sliding windows has attracted more and more attention [1][2][3][4][5][6][7][17]. However, these methods cannot reveal global system-work rules. In recent years, semantics mining has been always a popular topic. In time series analysis, semantics can

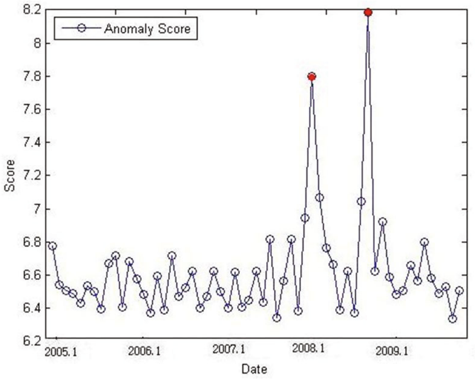


Fig. 4. Anomaly score time series of multiple correlated financial time series data

mainly be seen as system-work mechanisms. While, there are few studies on it. In [12], pHMM is proposed to learn time series semantics. However, it is just used to analyze a single time series. Generally, a Hidden Markov Model can be used to learn the semantics rules. Some other improved methods based on HMM are applied to learn latent system rules [16]. However, in multiple time series, a single observation value contains little information. Compared with these methods, our proposed model introduces local trends to extend information of tuple, and learn semantics from multiple time series based on both observation values and local trend correlation.

7 Conclusions

In this paper, we present a new Correlation field-based Semantics Learning Framework (CfSLF) to model multiple correlated time series. Our model aims to find semantics underneath multiple time series, by detecting data generating rules and transforming rules. Experiments have demonstrated the utility of the proposed method. The contribution of the study is three folder: (1) The Hidden Markov Random Field (HMRF) is used to model the data observations and corresponding states, by which the irregular patterns can be summarized from multiple correlated time series. (2) A value prediction method is presented based on semantics learned by CfSLF. (3) An anomaly detection method is proposed based on data semantics.

Acknowledgments. This research was supported by the ARC Linkage Project (LP100200682) named "Real-time and Self- Adaptive Stream Data Analyser for Intensive Care Management"; Internation S&T Cooperation Program of China No.2011DFA12910; National Science Foundation of China (NSFC) Grant (71072172; 61103229; 61272480; 61003167); the "Strategic Priority Research Program" of the Chinese Academy of Sciences Grant (XDA06030200); 863 Program Grant (2011AA01A103).

References

1. Zhang, C., Weng, N., Chang, J., Zhou, A.: Detecting Abnormal Trend Evolution over Multiple Data Streams. In: Chen, L., Liu, C., Zhang, X., Wang, S., Strasunskas, D., Tomassen, S.L., Rao, J., Li, W.-S., Candan, K.S., Chiu, D.K.W., Zhuang, Y., Ellis, C.A., Kim, K.-H. (eds.) WCMT 2009. LNCS, vol. 5731, pp. 285–296. Springer, Heidelberg (2009)
2. Zhang, P., Gao, B.J., Liu, P., Shi, Y., Guo, L.: A framework for application-driven classification of data streams. Neurocomputing 92, 170–182 (2012)
3. Papadimitriou, S., Sun, J., Faloutsos, C.: Streaming Pattern Discovery in Multiple Time-Series. In: Proceedings of VLDB 2005 (2005)
4. Chan, P.K., Mahoney, M.V.: Modeling Multiple Time Series for Anomaly Detection. In: Proceedings of ICDM
5. Hirose, S., Yamanishi, K., Nakata, T., Fujimaki, R.:]Network Anomaly Detection based on Eigen Equation Compression. In: Proceedings of SIGKDD 2009 (2009)
6. Qiao, Z., He, J., Cao, J., Huang, G., Zhang, P.: Multiple Time Series Anomaly Detection Based on Compression and Correlation Analysis: A Medical Surveillance Case Study. In: Sheng, Q.Z., Wang, G., Jensen, C.S., Xu, G. (eds.) APWeb 2012. LNCS, vol. 7235, pp. 294–305. Springer, Heidelberg (2012)
7. Fujimaki, R., Nakata, T., Tsukahara, H., Sato, A., Yamanishi, K.: Mining Abnormal Patterns from Heterogeneous Time-Series with Irrelevant Features for Fault Event Detection. Statistical Analysis and Data Mining 2 (2009)
8. Zhang, P., Gao, B.J., Zhu, X., Guo, L.: Enabling Fast Lazy Learning for Data Streams. In: Proceedings of ICDM (2011)
9. Zhang, P., Zhu, X., Shi, Y., Guo, L., Wu, X.: Robust ensemble learning for mining noisy data streams. Decision Support Systems 50(2), 469–479 (2011)
10. Stock, J.H., Watson, M.W.: Vector Autoregressions. Journal of Economic Perspectives 15(4), 101–115
11. Yves, N.: Total Least Squares: State-of-the-Art Regression in Numerical Analysis. SIAM Review 36 (2), 258–264
12. Wang, P., Wang, H., Wang, W.: Finding Semantics in Time Series. In: Proceedings of SIGMOD 2011 (2011)
13. Duncan, G., Gorr, W., Szczypula, J.: Forecasting Analogous Time Series, pp. 15213–13890. Carnegie Mellon University, Pittsburgh
14. Pang, C., Zhang, Q., Hansen, D.P., Maeder, A.J.: Unrestricted wavelet synopses under maximum error bound. In: Proceedings of EDBT 2009 (2009)
15. Keogh, E., Chu, S., Hart, D., Pazzani, M.: An online algorithm for segmenting time series. In: Proceedings of ICDM 2001 (2001)
16. Wang, Y., Zhou, L.: Mining complex time-series data by learning the temporal structure using bayesian techniques and markovian models. In: Proceedings of ICDM 2006 (2006)
17. Zhang, P., Li, J., Wang, P., Gao, B., Zhu, X., Guo, L.: Enabling Fast Prediction for Ensemble Models on Data Streams. In: Proceedings of SIGKDD 2011 (2011)

Matrix Factorization With Aggregated Observations

Yoshifumi Aimoto and Hisashi Kashima

Department of Mathematical Informatics, The University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan
{Yoshifumi_Aimoto,Kashima}@mist.i.u-tokyo.ac.jp

Abstract. Missing value estimation is a fundamental task in machine learning and data mining. It is not only used as a preprocessing step in data analysis, but also serves important purposes such as recommendation. Matrix factorization with low-rank assumption is a basic tool for missing value estimation. However, existing matrix factorization methods cannot be applied directly to such cases where some parts of the data are observed as aggregated values of several features in high-level categories. In this paper, we propose a new problem of restoring original micro observations from aggregated observations, and we give formulations and efficient solutions to the problem by extending the ordinary matrix factorization model. Experiments using synthetic and real data sets show that the proposed method outperforms several baseline methods.

1 Introduction

In many real data analysis applications, we often face datasets with missing values due to various reasons such as sensor failures and biased sampling. Since most of the existing data analysis methods are not directly applicable to them, we first need to estimate the missing values before analysis, or we need to develop new methods that can handle data with missing values. With its ubiquitous needs, missing value estimation [9,1,12] has been placed as one of the fundamental tasks in the field of machine learning and data mining, and it has been studied extensively. A typical dataset looks can be represented as a table with missing values (see Table 1). The table shows the numbers of beers of various brands purchased by four customers, where missing values are indicated by "-". The

Table 1. An example of purchase data. Typical data are given as a matrix-shaped table. The table shows the numbers of beers of various brands purchased by four customers, and missing values are indicated with "-".

items\users	Alice	Bob	Carol	Dave
Budweiser	5	-	2	3
Heineken	1	3	2	-
Carlsberg	1	-	1	2
Miller	3	1	3	2

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 521–532, 2013.

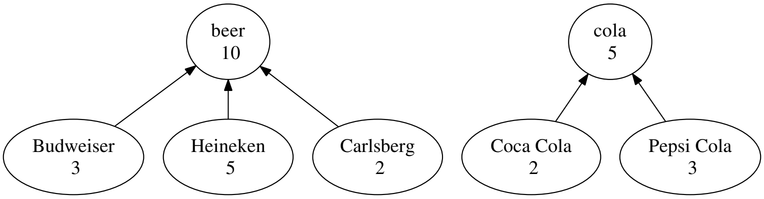


Fig. 1. Some portion of micro-level purchase data (e.g., the number of purchased bottles of a particular beer brand) are observed in an aggregated category (e.g., "beer")

table-structured data is mathematically considered as a matrix; hence, matrix analysis techniques are useful for missing value estimation. Matrix factorization (MF), which decomposes matrices by using the low- rank assumption [3,4], is one of the effective approaches to restore missing values in such matrix-shaped data. Missing value estimation using low-rank matrix factorization does not arise only as a preprocessing step, but also as a primary purpose of the analysis. Typical examples include recommender systems [6] and relational learning [10].

In this study, we consider a more complex situation where some parts of data are not completely missing, but are observed at a more abstract category level as aggregated values. Figure 1 shows examples of such cases. In each category (such as "beer" and "cola"), several micro-level counts (such as "Budweiser" and "Heineken") belonging to the category are observed as an aggregated count. To address such situations, we introduce a new variant of the missing value estimation problem, which we call *restoration of micro-level observations from aggregated observation*, where some parts of data are observed as aggregated values of several features. Since the existing techniques for missing value estimation including matrix factorization cannot be applied directly to such cases, we extend the existing low-rank matrix factorization formulation for missing value estimation to our case. We also devise iterative algorithms for solving the optimization problems, where each step consists of the standard singular value decomposition or closed form updates. Finally, using synthetic and real datasets, we show some experimental results on micro-observation restoration, which demonstrates that the proposed approach performs better than baseline methods.

The remainder of this paper is organized as follows. In Section 2, we introduce the *restoration of micro-level observations from aggregated observation* with a motivating example of purchase data analysis. We formulate matrix factorization problems with aggregated observations in Section 3, and give an efficient algorithm to solve the optimization problems in Section 4. In Section 5, we demonstrate the advantage of our approach over baseline approaches. Section 6 summarizes the related work, and Section 7 concludes the paper.

2 Problem Definition

In this section, we introduce a new problem that we refer to as the *restoration of micro observations from aggregated observations* using a motivating example

Table 2. An example of purchase data with aggregated observations. In addition to micro-level observations $\boldsymbol{Z}$, we have aggregated observations $\boldsymbol{Y}$, whose allocations to micro-level observations are not known. We have to restore the micro-level observation $\boldsymbol{X}$ from $\boldsymbol{Y}$ to obtain $\boldsymbol{Z}+\boldsymbol{X}$ which is the true sales data.

micro-level observations $\boldsymbol{Z}$					aggregated observations $\boldsymbol{Y}$				
items\users	Alice	Bob	Carol	Dave	items\users	Alice	Bob	Carol	Dave
Budweiser	5	0	2	3	beer (aggregated)	3	3	4	1
Heineken	1	3	2	1					
Carlsberg	1	0	1	2					
Miller	3	1	3	2					

of purchase data. We consider two cases that differ according to the assumption we make on categorical structures.

2.1 Motivating Example

Let us assume that we have purchase data represented as a matrix $\boldsymbol{X}$, where each column corresponds to a customer, each row corresponds to a product (such as a particular brand of beer), and element X_{ij} indicates the number of i-th products the j-th customer purchased. In many cases, the data has many missing values; for some (i, j), X_{ij} is completely missing, or a part of X_{ij} is missing (for example, only five of eight actual purchases are recorded). Restoration of the true purchases is quite important in sales management and analysis, and various missing value imputation methods [5] are employed for the purpose.

Let us now imagine a more complex situation where a part of X_{ij} is not missing, but is observed at a more abstract category level. In each category, several micro-level counts belonging to the category are observed as an aggregated count. For example, among eight actual purchases of the "Budweiser" brand of beer, only five are observed at the micro level (as five purchases of Budweiser), and the other three are observed in the more abstract "beer" category. The "beer" category might have ten purchases, including other beer brands such as five "Heineken" bottles and two "Carlsberg" bottles (See Figure 1). Now our goal is to restore the original micro level purchases (such as ten Budweiser purchases) from the aggregated observations.

2.2 Restoration of Aggregated Observations

General Problem Definition. Let us assume that we have two data matrices $\boldsymbol{Z}$ and $\boldsymbol{Y}$. $\boldsymbol{Z}$ is an $I \times J$ matrix that represents micro-level observations. In the previous example, I is the number of product brands, and J is the number of customers. $\boldsymbol{Y}$ is an $L \times J$ matrix which represents category-level observations, where L indicates the number of categories. An example with purchase data is given in Table 2. In addition to $\boldsymbol{Z}$ and $\boldsymbol{Y}$, we also have a correspondence matrix $\boldsymbol{C}$ as side information about product-category relationships. $\boldsymbol{C}$ is an $L \times I$ binary

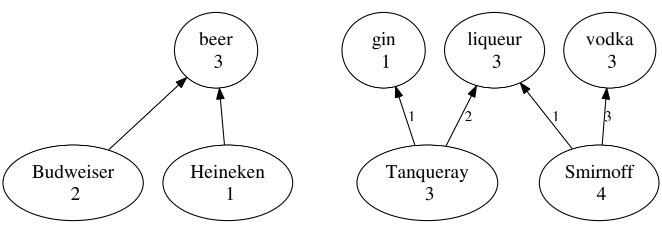


Fig. 2. (left) Case 1: each micro-level dimension belong to at most one category. (right) Case 2: each dimension can belong to more than one category.

matrix, whose (ℓ, i)-th element is 1 if the i-th product is included in the ℓ-th category. Our goal is to restore the hidden micro-level observation matrix $\boldsymbol{X}$ (of size $I \times J$) from $\boldsymbol{Y}$ with the help of $\boldsymbol{C}$ and $\boldsymbol{Z}$.

Two Different Assumptions on Product-Category Relationships. In our problem setting, we consider two different assumptions on the correspondence matrix $\boldsymbol{C}$, which results in slightly different formulations of the problem.

The first case is when each dimension of column vectors belongs to only one category (Figure 2 (left)), and the other case is when each dimension can belong to more than one category (Figure 2 (right)). Figure 2 (right) shows that a micro-level product "Tanqueray" belongs to two possible categories "gin" and "liqueur", and another micro-level product "Smirnoff" belongs to both "vodka" and "liqueur". We denote the former case as Case 1, and the latter as Case 2. The difference between the two cases is reflected by the definition of the correspondence matrix $\boldsymbol{C}$. In Case 1, each column of $\boldsymbol{C}$ has at most one value as "1" value and the rest are "0". On the other hand, in Case 2, each column of $\boldsymbol{C}$ can have multiple values as 1.

3 Formulation

In this section, we formulate our problem as optimization problems, where we restore micro observations $\boldsymbol{X}$ from aggregated observations $\boldsymbol{Y}$. Our model is an extension of the matrix factorization approach for missing value estimation.

3.1 Matrix Factorization Approach for Missing Value Estimation

We first review the existing matrix factorization approach for missing value estimation, where the observed (micro-level) data matrix $\boldsymbol{Z}$ has missing values, i.e., Z_{ij} are missing for some (i, j). Let us assume an observation matrix $\boldsymbol{E}$, where $E_{ij} = 1$ if Z_{ij} is observed; otherwise, $E_{ij} = 0$. To impute the missing values of the matrix, the low-rank assumption is often employed. We consider the following optimization problem of rank-k approximation of the observed matrix.

$$\text{minimize}_{\boldsymbol{A}} \quad \|\boldsymbol{E} * (\boldsymbol{Z} - \boldsymbol{A})\|_{\mathrm{F}}^2 \quad \text{s.t. } \operatorname{rank}(\boldsymbol{A}) \leq k,$$

where the Frobenius norm of a matrix $\boldsymbol{X} \in \mathbb{R}^{I \times J}$ is defined as $\|\boldsymbol{X}\|_{\mathrm{F}} = \sqrt{\sum_{j=1}^{J} \sum_{i=1}^{I} X_{ij}^2}$, and $*$ indicates the element-wise product. If all of the elements of $\boldsymbol{Z}$ are observed, i.e., $E_{ij} = 1$ for $\forall(i,j)$, the optimal solution is obtained by singular value decomposition (SVD). However, since we have missing elements in $\boldsymbol{Z}$, SVD cannot be applied. Furthermore, the optimization problem is not convex, hence numerical optimization methods do not guarantee optimal solutions. Recently, instead of using the rank constraint, the trace-norm constraint is often used, because the trace-norm constraint of a matrix is a convex set (whereas the rank constraint is not) [11,2]. Using the trace-norm constraint, we can formulate the low-rank matrix approximation problem as a convex optimization problem as

$$\mathrm{minimize}_{\boldsymbol{A}} \quad \|\boldsymbol{E} * (\boldsymbol{Z} - \boldsymbol{A})\|_{\mathrm{F}}^2 \;\; \text{s.t.} \;\; \|\boldsymbol{A}\|_{\mathrm{Tr}} \leq \tau,$$

where the trace norm of a matrix $\boldsymbol{X}$ is defined as $\|\boldsymbol{X}\|_{\mathrm{Tr}} = \mathrm{Tr}(\sqrt{\boldsymbol{X}\boldsymbol{X}^{\top}})$.

3.2 Matrix Factorization with Aggregated Observations

Now we extend the previous formulation to address our problem setting. Similar to the matrix factorization problem for missing value estimation, we also employ the low-rank assumption that our micro-level observations are of low-rank. We consider two slightly different formulations for the two cases we mentioned in the previous section.

Case 1. When each row of the micro-observation matrix can belong to at most one category, we need that the linear constraint $\boldsymbol{C}\boldsymbol{X} = \boldsymbol{Y}$, where each column of $\boldsymbol{C}$ has at most one value as "1" and the rest are "0". For example, let us assume that John bought several bottles of beer and cola as in Figure 1, the corresponding column in the constraint $\boldsymbol{C}\boldsymbol{X} = \boldsymbol{Y}$ looks like

$$\begin{array}{c} \text{beer} \\ \text{cola} \end{array} \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} 3 \\ 5 \\ 2 \\ 2 \\ 3 \end{bmatrix} \begin{array}{c} \text{Budweiser} \\ \text{Heineken} \\ \text{Carlsberg} \\ \text{Coca Cola} \\ \text{Pepsi Cola} \end{array} = \begin{bmatrix} 10 \\ 5 \end{bmatrix} \begin{array}{c} \text{beer} \\ \text{cola} \end{array}$$

With the constraint $\boldsymbol{C}\boldsymbol{X} = \boldsymbol{Y}$, we formulate the optimization problem as follows.

$$\begin{aligned} \mathrm{minimize}_{\boldsymbol{A},\boldsymbol{X}} \quad & \|\boldsymbol{A} - (\boldsymbol{X} + \boldsymbol{Z})\|_{\mathrm{F}}^2 \\ \text{s.t.} \quad & \|\boldsymbol{A}\|_{\mathrm{Tr}} \leq \tau, \quad \boldsymbol{C}\boldsymbol{X} = \boldsymbol{Y} \end{aligned} \tag{1}$$

Note that we assume that the "true" micro-observations $\boldsymbol{X} + \boldsymbol{Z}$ are of low-rank.

Case 2. When each row of the micro-observation matrix can belong to more than one category, aggregation from micro-level observations to category-level observations is not unique; therefore, we divide the micro-level observation matrix $\boldsymbol{X}$ into a sum of multiple matrices $\{\boldsymbol{X}^{(\ell)}\}_{\ell=1}^{L}$ so that $\sum_{\ell=1}^{L}\boldsymbol{X}^{(\ell)} = \boldsymbol{X}$ is satisfied.

In this case, we need that the linear constraints

$$\boldsymbol{C}_{\ell:}\boldsymbol{X}^{(\ell)} = \boldsymbol{Y}_{\ell:} \text{ for } \ell = 1, 2, \ldots, L \tag{2}$$

are satisfied. Note that one constraint is made for each of the L categories. If John bought several bottles of alcoholic beverage as in Figure 2 (right), one column in the constraint (2) looks like

$$\text{liqueur}\begin{bmatrix}0 \; 0 \; 1 \; 1\end{bmatrix}\left[\begin{array}{c}0\\0\\2\\1\end{array}\right]\begin{array}{c}\text{Budweiser}\\\text{Heineken}\\\text{Tanqueray}\\\text{Smirnoff}\end{array} = \begin{bmatrix}3\end{bmatrix}\text{ liqueur.}$$

The optimization problem is defined as follows.

$$\begin{aligned}\text{minimize}_{\boldsymbol{A},\boldsymbol{X}} \quad & \|\boldsymbol{A} - (\boldsymbol{X} + \boldsymbol{Z})\|_{\mathrm{F}}^{2} \\ \text{s.t.} \quad & \|\boldsymbol{A}\|_{\mathrm{Tr}} \leq \tau \\ & \boldsymbol{C}_{\ell:}\boldsymbol{X}^{(\ell)} = \boldsymbol{Y}_{\ell:} \text{ for } \ell = 1, \ldots, L, \quad \sum_{\ell=1}^{L}\boldsymbol{X}^{(\ell)} = \boldsymbol{X}\end{aligned} \tag{3}$$

Table 3 summarizes the ordinary formulation of matrix factorization, our formulation for Case 1, and one for Case 2.

4 Algorithms

Our optimization problems (1) and (3) are minimization problems of convex functions with respect to both $\boldsymbol{A}$ and $\boldsymbol{X}$. However, the number of variables involved is large, and it is time-consuming to minimize the objective functions with respect to them at once. Therefore, we devise iterative optimization procedures, each of whose step optimizes either of $\boldsymbol{A}$ and $\boldsymbol{X}$. We elaborate the concrete implementations of the estimation steps for both Case 1 and Case 2 below.

4.1 Case 1

Our proposed optimization procedure for Case 1 starts with initializing $\boldsymbol{X}$ so that the current $\boldsymbol{X}$ satisfies $\boldsymbol{C}\boldsymbol{X} = \boldsymbol{Y}$. The initialization is discussed in the Experiments section in detail. Then, we iterate the following updates of $\boldsymbol{A}$ and $\boldsymbol{X}$ until convergence.

When we update $\boldsymbol{A}$, we need to solve the optimization problem

$$\boldsymbol{A}^{\mathrm{NEW}} = \ \text{argmin}_{\boldsymbol{A}}\|\boldsymbol{A} - (\boldsymbol{X} + \boldsymbol{Z})\|_{\mathrm{F}}^{2} \ \text{ s.t. } \|\boldsymbol{A}\|_{\mathrm{Tr}} \leq \tau. \tag{4}$$

Table 3. Comparison of the formulation of the ordinary formulation of matrix factorization for missing value imputation and our formulations of restoration of micro observations from aggregated observations (Case 1 and Case 2). The constraints $\boldsymbol{X} \geq \boldsymbol{0}$ and $\boldsymbol{X}^{(\ell)} \geq \boldsymbol{0}$ are the additional non-negativity constraints we employ in Section 4.3.

	The existing MF	Proposed MF (Case 1)	Proposed MF (Case 2)
Inputs	$\boldsymbol{Z} \in \mathbb{R}^{I \times J}, \tau \in \mathbb{R}^{+}$	$\boldsymbol{Z} \in \mathbb{R}^{I \times J}, \tau \in \mathbb{R}^{+}, \boldsymbol{C} \in \mathbb{R}^{L \times I}, \boldsymbol{Y} \in \mathbb{R}^{L \times J}$	
Outputs	$\boldsymbol{A} \in \mathbb{R}^{I \times J}$	$\boldsymbol{A}$, $\boldsymbol{X} \in \mathbb{R}^{I \times J}$	
Objective function	$\lVert \boldsymbol{E} * (\boldsymbol{A} - \boldsymbol{Z}) \rVert_{\mathrm{F}}^2$ w.r.t. $\boldsymbol{A}$	$\lVert \boldsymbol{A} - (\boldsymbol{X} + \boldsymbol{Z}) \rVert_{\mathrm{F}}^2$ w.r.t. $\boldsymbol{A}, \boldsymbol{X}$	
Constraints	$\lVert \boldsymbol{A} \rVert_{\mathrm{Tr}} \leq \tau$	$\lVert \boldsymbol{A} \rVert_{\mathrm{Tr}} \leq \tau$ $\boldsymbol{C}\boldsymbol{X} = \boldsymbol{Y}$ $(\boldsymbol{X} \geq \boldsymbol{0})$	$\lVert \boldsymbol{A} \rVert_{\mathrm{Tr}} \leq \tau$ $\boldsymbol{C}_{\ell:}\boldsymbol{X}^{(\ell)} = \boldsymbol{Y}_{\ell:}$ $\sum_{\ell=1}^{L} \boldsymbol{X}^{(\ell)} = \boldsymbol{X}$ $(\boldsymbol{X}^{(\ell)} \geq \boldsymbol{0})$

This optimization problem can be solved by applying SVD to $\boldsymbol{X} + \boldsymbol{Z}$ and thresholding the singular values. Let the SVD of $\boldsymbol{X} + \boldsymbol{Z}$ be $\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\top}$, where $\boldsymbol{U}$ and $\boldsymbol{V}$ are orthogonal matrices, and $\boldsymbol{\Sigma}$ is a diagonal matrix with the singular values as its diagonals. We eliminate the singular values less than the threshold τ, and denote $\boldsymbol{\Sigma}'$ as the diagonal matrix with diagonal elements greater than or equal to τ. The optimal solution $\boldsymbol{A}^{\mathrm{NEW}}$ of Eq. (4) is obtained as

$$\boldsymbol{A}^{\mathrm{NEW}} = \boldsymbol{U}\boldsymbol{\Sigma}'\boldsymbol{V}^{\top}. \tag{5}$$

Optimization with respect to $\boldsymbol{X}$ is casted as the minimization problem

$$\boldsymbol{X}^{\mathrm{NEW}} = \operatorname{argmin}_{\boldsymbol{X}} \lVert \boldsymbol{X} - (\boldsymbol{A} - \boldsymbol{Z}) \rVert_{\mathrm{F}}^2 \text{ s.t. } \boldsymbol{C}\boldsymbol{X} = \boldsymbol{Y}. \tag{6}$$

This is generally a convex quadratic programming problem; however, the optimal solution is given in a simple closed form in this case. Since this problem can be seen as minimization of the Euclidean distance between $\boldsymbol{X}$ and $\boldsymbol{M}$ with the hyper-plane constraint $\boldsymbol{C}\boldsymbol{X} = \boldsymbol{Y}$, the optimal solution is given as the projection of $\boldsymbol{M}$ onto the hyper-plane. Assuming that the micro-level feature i belongs to the aggregated category ℓ, the optimal solution of i-th row $\boldsymbol{X}_{i:}^{\mathrm{NEW}}$ is obtained as

$$\boldsymbol{X}_{i:}^{\mathrm{NEW}} = \boldsymbol{M}_{i:} - \frac{1}{\sum_{i=1}^{I} C_{\ell i}} (\boldsymbol{C}_{\ell:}\boldsymbol{M} - \boldsymbol{Y}_{\ell:}), \tag{7}$$

where we defined $\boldsymbol{M} = \boldsymbol{A} - \boldsymbol{Z}$.

4.2 Case 2

In Case 2, noting that $\boldsymbol{X} = \sum_{\ell=1}^{L} \boldsymbol{X}^{(\ell)}$, the update of $\boldsymbol{A}$ is the same as that for Case 1. However, in contrast to Case 1, the update of $\boldsymbol{X}$ cannot be given in a closed form solution anymore in Case 2, and we have to solve the optimization problem

$$\boldsymbol{X}^{\mathrm{NEW}} = \mathrm{argmin}_{\boldsymbol{X}} \ \|\boldsymbol{X} - (\boldsymbol{A} - \boldsymbol{Z})\|_{\mathrm{F}}^2 \tag{8}$$
$$\text{s.t.} \ \ \boldsymbol{C}_{\ell:}\boldsymbol{X}^{(\ell)} = \boldsymbol{Y}_{\ell:} \quad (\ell = 1, \ldots, L), \quad \boldsymbol{X} = \sum_{\ell=1}^{L} \boldsymbol{X}^{(\ell)}.$$

Although it is a quadratic programming problem, the number of variables involved is rather large; hence, we again resort to iterative optimization, that is, we iterate updates with respect to one of $\{\boldsymbol{X}^{(\ell)}\}_{\ell=1}^{L}$ at once. The optimization problem with respect to only $\boldsymbol{X}^{(\ell)}$ with the other $\{\boldsymbol{X}^{(j)}\}_{j \neq \ell}$ fixed, the problem (8) is written as

$$\boldsymbol{X}^{(\ell)^{\mathrm{NEW}}} = \mathrm{argmin}_{\boldsymbol{X}^{(\ell)}} \ \|\boldsymbol{X}^{(\ell)} - (\boldsymbol{M} - \sum_{j \neq \ell} \boldsymbol{X}^{(j)})\|_{\mathrm{F}}^2 \ \text{ s.t. } \boldsymbol{C}_{\ell:}\boldsymbol{X}^{(\ell)} = \boldsymbol{Y}_{\ell:}.$$

This has the same form as that in Case 1; hence, the closed form update becomes

$$\boldsymbol{X}_{i:}^{(\ell)^{\mathrm{NEW}}} = (\boldsymbol{M}_{i:} - \sum_{j \neq \ell} \boldsymbol{X}^{(j)}) - \frac{1}{\sum_{i=1}^{I} C_{\ell i}} (\boldsymbol{C}_{\ell:}(\boldsymbol{M} - \sum_{i \neq j} \boldsymbol{X}^{(j)}) - \boldsymbol{Y}_{\ell:}). \tag{9}$$

4.3 Non-negativity Constraints

Since our original motivation came from the purchase data example, it is sometimes more reasonable to make a non-negativity assumption on the micro-level observations.

In Case 1, we make an additional constraint that $\boldsymbol{X}$ is non-negative. The resultant optimization problem for Case 1 with respect to $\boldsymbol{X}$ becomes

$$\boldsymbol{X}^{\mathrm{NEW}} = \ \mathrm{argmin}_{\boldsymbol{X}} \ \|\boldsymbol{X} - \boldsymbol{M}\|_{\mathrm{F}}^2 \ \text{ s.t. } \boldsymbol{C}\boldsymbol{X} = \boldsymbol{Y}, \ \ \boldsymbol{X} \geq 0.$$

Accordingly, the previous closed form solution (7) is modified to

$$X_{ij}^{\mathrm{NEW}} = \frac{(X_{ij} - s_j)Y_{\ell j}}{(Y_{\ell j} - \sum_{i=1}^{I} C_{\ell i} s_j)}, \tag{10}$$

where $s_j = \min_{1 \leq i \leq I} X_{ij}$.

In Case 2, we make the assumption that each $\boldsymbol{X}^{(\ell)}$ is non-negative. The update (10) is similarly obtained as

$$X_{ij}^{(\ell)^{\mathrm{NEW}}} = \frac{(X_{ij} - s_j)Y_{\ell j}}{Y_{\ell j} - \sum_{i=1}^{I} C_{\ell i} s_j}.$$

Note that the modified optimization problems are still convex; therefore, we obtain optimal solutions when converged.

5 Experiments

We show some experimental results using synthetic and real datasets that demonstrate the reasonable performance of the proposed methods to restore micro-level observations from category-level observations. We compare the restoration errors by the proposed methods with those by four baseline methods, and show the advantage of the proposed methods over them.

5.1 Datasets

Synthetic Dataset. The first dataset is a set of randomly generated matrices. The size of matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ is $1,000 \times 5$, and each element $U_{ir} \in \{0,1,2,3\}$ and $V_{jr} \in \{0,1,2\}$ is generated uniformly at random over the ranges. The true micro-level observation matrix is generated as $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{V}^{\top}$, where 5% of the elements of $\boldsymbol{A}$ are randomly missing.

A $100 \times 1,000$ correspondence matrix $\boldsymbol{C}$ is generated so that the (ℓ, i)-th element is 1 if and only if i is in $\{10 \times (\ell - 1) + 1, \dots, 10 \times \ell\}$ for Case 1. For Case 2, starting from the $\boldsymbol{C}$ we created for Case 1, we further sample 300 (ℓ, i) pairs to make additional "1" values. To create category-level observations, we employ binomial distributions to divide the true micro-level observations $\boldsymbol{A}$ into the hidden part $\boldsymbol{X}$ and the observed part $\boldsymbol{Z}$. Namely, each element Z_{ij} is determined by $\Pr(Z_{ij} = k) = \binom{A_{ij}}{k} p^k (1-p)^{A_{ij}-k}$, where p controls the likeliness of the observation of each micro-observation at its superordinate category. For example, $p = 1$ corresponds to the perfect observation case with no category-level observations. In our experiments, we varied p in $\{0.1, 0.4, 0.7\}$.

Once the hidden part $\boldsymbol{X}$ is determined, the corresponding category-level observations $\boldsymbol{Y}$ are created using the correspondence relationship $\boldsymbol{C}\boldsymbol{X} = \boldsymbol{Y}$ for Case 1. For Case 2, we set $\boldsymbol{X}^{(\ell)}_{i:} = \boldsymbol{X}_{i:} / \sum_{i=1}^{I} C_{\ell i}$ if $C_{\ell i} = 1$, and aggregate $\boldsymbol{X}^{(\ell)}_{i:}$ to $\boldsymbol{Y}_{\ell:}$ with $\boldsymbol{C}_{\ell:}\boldsymbol{X}^{(\ell)} = \boldsymbol{Y}_{\ell:}$.

Purchase Dataset for Internet Stores. Another dataset is a real cross-store purchase dataset collected from 6 internet stores to include for 494 customers, and 150 product brands belonging to 11 categories (such as electronic devices, undergarments, and magazines). Since the granularities of the input sales logs differ from store to store, not all of them provided detailed product names, and gave only category-level information. One product can belong to more than one category in this dataset; hence, this dataset belongs to Case 2. Since we had no ground truth micro-observations, we simulated category-level observations again from the micro-observed data; we assumed that some stores did not provide micro-level sales, and their sales were given as category-level observations.

5.2 Comparison Methods

Although there have not been any existing methods that address the restoration problem to the best of our knowledge, we consider four baseline methods as comparison methods to evaluate the proposed methods. The first method (that we call "Equal" method) divides each category-level observation into its descendant micro-observation equally. The second method (that we call "Prop" method) divides the category-level observations in proportion to the observed micro-level values (of $\boldsymbol{Z}$) as $X_{ij} = Z_{ij}Y_{\ell j}/\boldsymbol{C}_{\ell :}\boldsymbol{Z}_{:j}$ in Case 1, and $X_{ij}^{(\ell)} = Z_{ij}C_{\ell i}Y_{\ell j}/\boldsymbol{C}_{\ell :}\boldsymbol{Z}_{:j}$ in Case 2. In addition, we applied the matrix factorization method (SVD) to the matrices obtained using the above methods, which results in two additional baseline methods (which we call "Equal+MF" and "Prop+MF").

The micro-level estimations obtained by the simple methods are also used for initialization of $\boldsymbol{X}$ in the proposed method. Although our formulations are convex optimization problems and the solutions do not depend on the initial estimates, our preliminary experiments suggest that initialization with the "Equal" method shows better numerical stability.

5.3 Results

Table 4 and 5 show comparison of errors under different methods with varied p among $\{0.1, 0.4, 0.7\}$, where Table 4 shows the results for the synthetic data in Case 1, Table 5 for that in Case 2. Table 6 shows the results for the purchase dataset (in Case 2) where one, two, or four stores out of six are assumed not to provide micro-level sales. As the evaluation metric, we used the difference between the estimated micro-observations $\hat{\boldsymbol{X}}$ and the true micro-observations $\boldsymbol{X}$ defined as $\mathrm{Error}_{\boldsymbol{X}}(\hat{\boldsymbol{X}}) = \|\hat{\boldsymbol{X}} - \boldsymbol{X}\|_{\mathrm{F}}/\|\boldsymbol{X}\|_{\mathrm{F}}$. The ranks for matrix factorization were determined so that the simple MF method (Equal-MF or Prop-MF) performed the best (we reused it for the proposed method); they were 20 for the synthetic dataset (Case 1) and the purchase dataset, and 36 for the synthetic dataset (Case 2). The difference of the error between each comparison method and proposed method is significant in the Wilcoxon signed-rank test at a 0.05 significance level. The results show that the proposed matrix factorization method is superior to the baseline methods. Interestingly, the simple application of matrix factorization (Equal-MF and Prop-MF) sometimes made the results worse than those by Equal and Prop. The simple MF methods roughly correspond to stopping the iterations of the proposed algorithm at the first iteration, and the results show it improved the performance after several iterations.

Finally, we mention the computational cost of the proposed method; the computational cost depends approximately on the number of calls of the SVD routine, which was about five calls to converge.

6 Related Work

Dealing with incomplete data has been studied extensively, and widely applied in various fields including machine learning and data mining. Zhu *et al.* [12] categorized strategies to handle missing data into three categories, that are, case deletion, learning without handling of missing data, and data imputation. Case

Table 4. Comparison of the micro-observation reconstruction errors by the proposed method and the baseline methods with the artificial dataset in Case 1. p controls how likely each micro-observation is observed at its superordinate category. The proposed method achieves the lowest error for all p.

p	Equal	Prop	Equal+MF	Prop+MF	MFAO
0.1	0.384	0.966	0.399	0.382	**0.374**
0.4	0.429	0.551	0.499	0.430	**0.419**
0.7	0.521	0.552	0.812	0.772	**0.519**

Table 5. Comparison of the micro-observation reconstruction errors by the proposed method and the baseline methods with the artificial dataset in Case 2. p controls how likely each micro-observation is observed at its superordinate category. The proposed method achieves the lowest error for all p.

p	Equal	Prop	Equal+MF	Prop+MF	MFAO
0.1	0.540	1.104	0.543	0.646	**0.535**
0.4	0.570	0.708	0.592	0.561	**0.560**
0.7	0.615	0.669	0.771	0.747	**0.611**

Table 6. Comparison of the micro-observation reconstruction errors by the proposed method and the baseline methods with the purchase dataset (in Case 2). We assumed that some stores (out of six stores) did not provide the micro-level sales, and their sales were given as category-level observations. The proposed method achieves the lowest error regardless of the number of stores not providing micro-level sales.

Number of stores not providing micro-level sales	Equal	Prop	Equal+MF	Prop+MF	MFAO
1	**0.947**	1.308	1.006	1.340	**0.947**
2	0.964	1.100	1.461	1.529	**0.939**
4	0.975	1.151	1.712	1.800	**0.947**

deletion, which ignores missing values, is the simplest method. These kinds of approaches require robust methods to counter incomplete data [9]. Methods in the second category directly work with missing data. Data imputation approaches estimate unobserved values from the observed ones, and this method includes the matrix factorization approach we employed in this research.

Missing value imputation approaches can be classified into two categories, that are, data-driven approach and model-based approach [7]. Our method is categorized into the latter, and employs the matrix factorization model. There are several studies to impute missing values using matrix factorization techniques such as SVD and non-negative matrix factorization [8].

7 Conclusion

Missing value estimation is an unavoidable problem in real data analysis. In this study, we introduced an extended matrix factorization for a new missing value estimation problem, that is, restoration of micro-level observations from category-level aggregated observation. Since the existing methods cannot directly be applied to this problem, we formulated an extended low-rank matrix factorization problem, and devised efficient iterative algorithms for solving the optimization problems. The experimental results using synthetic and real datasets showed that our approach performed better than baseline methods.

Acknowledgment. The authors are grateful to Naonori Ueda, Hiroshi Sawada, and Katsuhiko Ishiguro of NTT Communication Science Laboratories, and Noriko Takaya of NTT Cyber Solution Laboratory.

References

1. Brand, M.: Incremental singular value decomposition of uncertain data with missing values. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002, Part I. LNCS, vol. 2350, pp. 707–720. Springer, Heidelberg (2002)
2. Candes, E.J., Tao, T.: The power of convex relaxation: Near-optimal matrix completion. IEEE Transactions on Information Theory 56(5), 2053–2080 (2010)
3. Eckart, C., Young, G.: The approximation of one matrix by another of lower rank. Psychometrika 1(3), 211–218 (1936)
4. Eriksson, A., Hengel, A.V.D.: Efficient computation of robust low-rank matrix approximations in the presence of missing data using the L_1 norm. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 771–778. IEEE, San Francisco (2010)
5. Han, J., Kamber, M., Pei, J.: Data Mining: Concepts and Techniques, 3rd edn. Morgan Kaufmann (2011)
6. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer 42(8), 30–37 (2009)
7. Lakshminarayan, K., Harp, S.A., Samad, T.: Imputation of missing data in industrial databases. Applied Intelligence 11, 259–275 (1999)
8. Lee, L., Seung, D.: Algorithms for non-negative matrix factorization. In: Advances in Neural Information Processing Systems 13, pp. 556–562 (2001)
9. Little, R.J., Rubin, D.B.: Statistical Analysis with Missing Data. Wiley (1987)
10. Singh, A.P., Gordon, G.J.: Relational learning via collective matrix factorization. In: ACM SIGKDD, Las Vegas, USA, pp. 650–658 (2008)
11. Srebro, N., Rennie, J., Jaakkola, T.: Maximum-margin matrix factorization. In: Advances in Neural Information Processing Systems 17 (2005)
12. Zhu, X., Zhang, S., Jin, Z., Zhang, Z., Xu, Z.: Missing value estimation for mixed-attribute data sets. IEEE Transactions on Knowledge and Data Engineering 23(1), 110–121 (2011)

An Approach to Identifying False Traces in Process Event Logs

Hedong Yang, Lijie Wen, and Jianmin Wang

School of Software, Tsinghua University, Beijing 100084, China
{yanghd06,wenlj00}@mails.tsinghua.edu.cn, jimwang@tsinghua.edu.cn

Abstract. By means of deriving knowledge from event logs, the application of process mining algorithms can provide valuable insight into the actual execution of business processes and help identify opportunities for their improvement. The event logs may be collected by people manually or generated by a variety of software applications, including business process management systems. However logging may not always be done in a reliable manner, resulting in events being missed or interchanged. Consequently, the results of the application of process mining algorithms to such "polluted" logs may not be so reliable and it would be preferable if *false traces*, i.e. polluted traces which are not possibly valid as regards the process model to be discovered, could be identified first and removed before such algorithms are applied. In this paper an approach is proposed that assists with identifying false traces in event logs as well as the cause of their pollution. The approach is empirically validated.

Keywords: process mining, event log, business process management, noise identification.

1 Introduction

Process mining provides a bridge between data mining and traditional model-driven Business Process Management (BPM) [10,11]. A business process (e.g. a purchase order, an insurance claim, etc.) is a sequence or network of tasks performed by humans or by machines to purposefully achieve a specific business goal. BPM provides supports, by affording methods, techniques, and softwares etc., for (re)design, deployment (system configuration and process enactment), and analysis of operational business processes as well as concerned resources (humans, machines, data, etc.) [9]. Generally speaking, a process-aware information system (PAIS) plays a key role during the whole life cycle of BPM as depicted in Fig. 1. By means of deriving knowledge from event logs manually collected or auto-generated by a PAIS [5], process mining, which behaves like the traditional data mining as depicted by the red arrow in Fig. 1, is generally seen as a critical tool to improve operational business processes iteratively. The first of three main classical applications of process mining is *model discovery*, which objective is to extract process models, the most important data of BPM, from event logs [10,11]. An example is to mine a process model shown in Fig. 3 given

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 533–545, 2013.

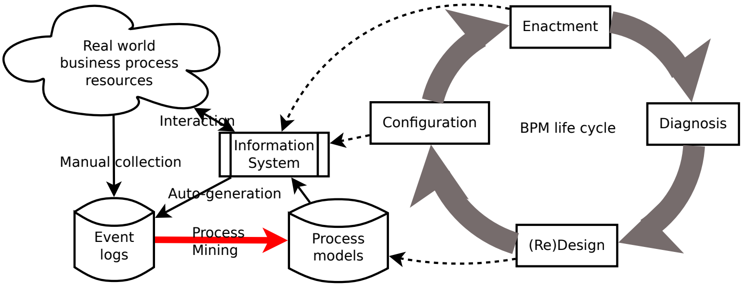


Fig. 1. Process mining and life cycle of Business Process Management

an event log containing four traces $\{ACDG, ADCG, BEH, BFH\}$. An overview of various mining algorithms for model discovery can be found in [9,14], and their implementations can be found in the open source platform, ProM[1], which has been used in industrial applications.

Generally speaking, the quality of process mining results is not determined only by the algorithm used but also by the quality of the concerned data, i.e. event logs which record the executions of process businesses. Of the criteria to judge the quality of an event log, one is *trustworthy* which requires recorded events and their orders being exactly same to what they happened [11]. However since event logs are often not treated as the key business data in real life, there are seldom policies to guarantee the quality of event logs. Consequently when a trace, a sequence of events recording an execution of a business process, was written into an event log, sometimes it was not recorded as it should be, namely it was *polluted*. For example, when deploying a PAIS in an organization for the first time, people have to describe the business processes of interest precisely and formally for configuring the system, which is often based on event logs collected manually where one or more events of a trace may be missed for some reasons. Another example happens daily in a hospital. Blood chemistry tests for patients are often carried out in groups rather than one by one instantly. And the test results are not available until some hours later, which are stamped with a date only. So do X-ray tests. Thus the sequence of such two tests may sometimes be messed up in the log for a patient since the granularity of timestamps is coarse-grained. Such traces that would not describe the actual executions of business processes, are called occurrences of *noise* [2] or *polluted traces*. The original sequence of events is transformed into another sequence of events by means of missing some events or interchanging the order of two events. In this paper we focus on these two types of *pollution*, which are widely accepted as the most common pollution of event logs (e.g., [3,6,15]). Although most problems in process mining have had satisfactory solutions, *noise identification* of logs is one of those unsolved which present impediments to advancement of the field [11].

Most of model discovery algorithms cannot guarantee the correctness of their mining results if the given event log is polluted. Mining algorithms can be classified

[1] `http://www.promtools.org`

into two categories. The first category consists of algorithms which assume the log to be noise-free (e.g., the most famous $\alpha-$algorithm [13]). For these algorithms it is necessary to identify occurrences of noise in the log and remove them before starting process mining. The second category consists of algorithms which have their own ways of dealing with occurrences of noise in the logs. These algorithms roughly treat low frequent traces as polluted ones directly or indirectly, no matter whether they are polluted or not, before process mining (e.g. [15]), during process mining (e.g. [1]) or after process mining (e.g. [2]). However, setting up a convincing threshold value is still a challenging problem.

Noise identification in process mining is similar to but not same as the data clean or outlier detection in data mining and the de-noising in signal processing. Traditional approaches for data clean make full use of relations among attributes and records of data [8], while there are unstructured traces only in event logs. Although a polluted trace is not the same as the normal trace which it should be, it may be by chance the same as another trace that is normal. Hence traditional approaches for outlier detection cannot be used [16]. Approaches for de-noising in signal processing focus on the Gaussian white noise, the widely accepted pollution type in the field, and are hard to be applied to deal with the pollution in process mining (e.g.,[4,7]). To summarize, algorithms available in these fields cannot be applied to identify polluted traces in an event log directly because of the characteristics of event logs and pollution concerned.

This then leads to the demand for a separate approach for noise identification. As a polluted trace may appear as another normal trace, we focus only on *false traces* in the paper, i.e. polluted traces which are not possibly valid as regards the process model to be discovered. Given a polluted log, as we do not have access to that process model that generated the log, we propose an approach, FATILP (FAlse Trace Identification based on Latent Probability), to helping find out false traces in a probabilistic manner, based on the occurrence frequencies of the observed traces and their transformation relations presented as a conditional probability matrix. The matrix describes the possible pollution type of the log, which itself can be obtained interactively by applying the approach.

It is important to note that our method for identifying false traces in a polluted log is not dependent on the choice of a specific mining algorithm. Our results can directly be used for those algorithms that are sensitive to false traces in logs (e.g. [6,13]). Beyond the field of process mining, the approach may be applied in the field of data provenance (e.g. to find out the origin of data), social network (e.g. to estimate the evolution of a social network), or traditional data mining (e.g. to mine the occurrence patterns of hot topics on the web).

The remainder of this paper is organized as follows. Section 2 describes basic concepts needed to define the problem and to describe our approach, explains three reasonable assumptions needed by our approach and formulates the problem of false trace identification of event logs for process mining. Then the proposed approach for the false trace identification problem is outlined in Section 3. In Section 4 the results obtained are evaluated and examined in an experimental manner. Section 5 concludes the paper and outlines future work.

2 Problem Characterization

2.1 Basic Definitions

A *task* is an activity to be performed in the context of a business process. A *process model* provides an abstraction of a business process capturing its tasks and all possible execution orders of these tasks in a formal manner. A *process instance* represents an actual execution of a business process. A *trace* is the result of the successful completion of a process instance and consists of a sequence of events, where each *event* corresponds to the execution of a task and all events are totally ordered typically on the basis of the timestamps that they were recorded. An *event log* is a set of traces, which records executions of a process model [12].

Two traces are *equivalent* if and only if their lengths are equal and every event of the first trace refers to the same task as the corresponding event at the same position of the second trace. A *trace class* consists of traces equivalent to each other. For simplicity, we refer to *a trace as a sequence of task names* to which the events of the trace correspond respectively, and thus a log as a *bag* of traces generated by a process model. As mentioned before, a trace is referred to as a *polluted trace* if it does not describe the actual execution of a business process, and as a *normal trace* otherwise. As a polluted trace may appear as a normal one, we define a special kind of polluted trace as follows.

Definition 1 (False Trace). *Given a process model P and a log L. A trace σ of L is referred to as a* false trace *if and only if it is not equivalent to any normal trace of P.*

A trace is referred to as a *true trace* if it is not a false trace. All normal traces are true traces and all false traces are polluted traces. Some polluted traces may be same as true traces. As illustrated in Fig. 2, a normal trace T_6 may be transformed into some polluted traces, and an observed trace T_2 in a log may originate from some normal traces. Traces T_0, T_1, T_8 and T_9 are false traces. Obviously the concepts of event log and false trace are quite different from those of *trajectory data* and *outlier* in the field of data mining respectively.

2.2 Assumptions

In this subsection the assumptions, which precisely characterise the event log and pollution type on the one hand and underpin the proposed approach on the other hand, are made explicit. Each assumption is described in detail and it is argued that the assumption is reasonable, why it is needed, and what would go wrong if the assumption was not made.

Assumption 1. *Normal traces occur randomly and independently.*

By observing the execution log, it is not possible to determine what the next trace will be recorded, based on the observed traces. It is reasonable to assume that traces appear randomly and independently.

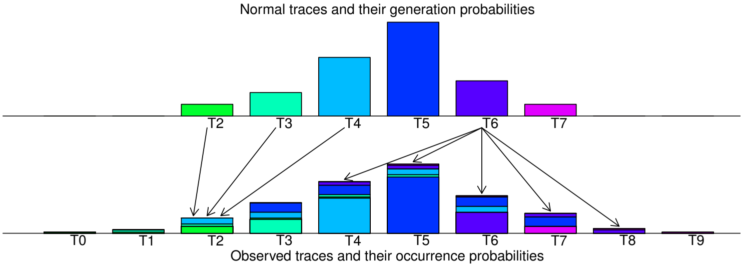


Fig. 2. Generation probabilities and occurrence probabilities of traces

If the occurrence of a new trace depends on an observed trace, the new trace and the observed trace are correlated. We treat them as different occurrences of the same trace as they can be (partially) deduced from existing ones.

Assumption 2. *A normal trace occurs with a constant but unknown probability, which may vary across different traces.*

A trace represents a particular application scenario of a process model. When a business process has been running for years, the same scenario may appear periodically. As time goes by the *occurrence frequencies* of the traces become relatively stable and in the long run they may converge to constant values, i.e. to their *latent generation probabilities.* Note that because of pollution, the generation probability of a trace is generally different from its occurrence probability.

If this assumption does not hold, we cannot solve the problem of noise identification of an event log without further information about the occurrence of traces. It is worthwhile noting that this means that our approach does not work so well for logs that result from processes that have not been running for a very long time as trace occurrence frequencies may not have sufficiently stabilized.

Assumption 3. *The pollution occurs randomly and independently, and given a normal trace the conditional probability of transforming the normal trace to another polluted trace because of pollution is a constant value, which may vary across different polluted traces.*

According to our observations, it is general that all traces in a log are not polluted, and that the pollution of a normal trace seldom depends on previous occurrences of pollution. Thus it is reasonable to assume the random and independent occurrence of pollution. Note all possible polluted traces of a normal trace are determinate because of its determinate conditional probabilities.

The assumption reflects the key idea of the proposed approach, i.e. trying to mimic the process of pollution and then to identify false traces by making full use of the relationships between false traces and their corresponding normal traces, which can be presented as a conditional probability matrix, i.e. so-called a *pollution matrix.* Such relationship may be various, yet we here require its conditional transformation probability to be constant. Without detail information of pollution, it is typically assumed that all possible polluted traces of a normal trace have the same conditional transformation probability. A priori knowledge of pollution may help set up the probability value for a specific transformation.

2.3 Problem Formulation

In this paper we are concerned with finding answers to the following problems related to a polluted event log.

Problem 1 (False trace identification problem). Given a polluted log L of an unknown process model, and a pollution matrix $\mathbf{M}$, which traces among all traces in L are most likely to be false traces?

Problem 2 (False trace discovery problem). Given a polluted log L of an unknown process model. Among all traces in L which are most likely to be false traces?

3 Approach

3.1 Key Idea

Given an event log L, for all observed traces $T_1, T_2, \cdots, T_M$ their *occurrence frequencies* $\mathbf{F} = \{f_1, f_2, \cdots, f_M\}$ are defined by $f_i = n_i/N$ for all $1 \leq i \leq M$, where $N = \sum_{i=1}^{M} n_i$ and n_i is the occurrence time of T_i. Based on all assumptions presented in subsection 2.2, suppose there are all W possible traces. Let $\mathbf{G} = \{g_1, g_2, \cdots, g_M\}$ be the latent generation probabilities of the observed traces, $p_{i,j} = \mathit{Prob}(T_j|T_i)$ the conditional probability of transforming trace T_i to T_j where $1 \leq i \leq M$ and $1 \leq j \leq W$, and $\mathbf{P} = \{p_1, p_2, \cdots, p_W\}$ the occurrence probability of the traces either observed or unobserved. Especially the combined conditional transformation probability of all unobserved traces of T_i is $u_i = \sum_{j=M+1}^{W} p_{i,j} = 1 - \sum_{j=1}^{M} p_{i,j}$. All these conventions are presented in Table 1, and the probabilities are in gray since they are unknown.

Formally, the relationship between $\mathbf{G}$ and $\mathbf{P}$ can be described as

$$p_j = \sum_{i=1}^{M} g_i * p_{i,j}, \text{ where } 1 \leq j \leq W \ . \tag{1}$$

The start point of the proposed approach, FATILP (FAlse Trace Identification based on Latent Probability), is the occurrence frequencies of traces $\mathbf{F}$, which will converge to the occurrence probabilities of the traces $\mathbf{P}$ respectively according to the law of large numbers in probability theory. If $\mathbf{P}$ can be estimated based on $\mathbf{F}$, the $\mathbf{G}$ can be calculated by means of the equation (1). As we know a process model does not generate a false trace, which implies the latent generation probability of a false trace should be *zero*. Thus the false traces can be identified based on their latent generation probabilities. Precisely, the FATILP consists of three steps as follows.

1. Process the given log to derive occurrence frequencies of observed traces.
2. Estimate the latent generation probabilities of observed traces by means of minimizing a distance function. This is the key step of the approach.
3. Perform a χ^2 test on the estimation result, and identify false traces.

Table 1. Pollution matrix and event log information

Trace	T_1	T_2	$\cdots$	T_M	T_{M+1}	$\cdots$	T_W	prob.unobserv	Gen. prob.
T_1	$p_{1,1}$	$p_{1,2}$	$\cdots$	$p_{1,M}$	$p_{1,M+1}$	$\cdots$	$p_{1,W}$	u_1	g_1
T_2	$p_{2,1}$	$p_{2,2}$	$\cdots$	$p_{2,M}$	$p_{2,M+1}$	$\cdots$	$p_{2,W}$	u_2	g_2
$\vdots$	$\vdots$	$\vdots$	$\ldots$	$\vdots$	$\vdots$	$\ldots$	$\vdots$	$\vdots$	$\vdots$
T_M	$p_{M,1}$	$p_{M,2}$	$\cdots$	$p_{M,M}$	$p_{M,M+1}$	$\cdots$	$p_{M,W}$	u_M	g_M
Occur. prob.	p_1	p_2	$\cdots$	p_M	p_{M+1}	$\cdots$	p_W	p_U	
Occur. freq.	f_1	f_2	$\cdots$	f_M	f_{M+1}	$\cdots$	f_W	0	
Occur. times	n_1	n_2	$\cdots$	n_M	n_{M+1}	$\cdots$	n_W	0	

3.2 False Traces Identification

Since there is often an error between the **P** and **F**, it is not appropriate to replace **P** with **F** directly. Here we define as follows a distance function Q^2 between **P** with **F** . The minimization of the distance function would force **P** of false traces to be zero or be very near to zero since the unobserved traces have higher weights, and consequently **G** of false traces would be zero or very near to zero since **P** and transform probability are non-negative (refer to equation (1) for detail).

$$\underset{\mathbf{G}}{\operatorname{argmin}}\, Q^2 = \underset{\mathbf{G}}{\operatorname{argmin}}\, p_U^2 N^2 + \sum_{i=1}^{M}(f_i - p_i)^2 \frac{N}{f_i} = \underset{\mathbf{G}}{\operatorname{argmin}}(\sum_{j=1}^{M} g_j * u_j)^2 N^2 + \sum_{i=1}^{M}(f_i - \sum_{j=1}^{M} g_j * p_{j,i})^2 \frac{N}{f_i} \qquad (2)$$

subject to $\sum_{i=1}^{M} g_i = 1$ and $g_i \geq 0$.

Now the false traces identification problem has been modeled as a quadric optimization problem, whose computation complexity is determined by the number of variables and the number of constraints. From equation (2), it is known that there are M variables and $M + 1$ constraints. To best of our knowledge, 32,000 and 16,000 are the limits of numbers of variables and constraints for a non-linear optimization problem respectively, achieved by the Lingo System (version 12.0).[2] Those are enough for almost all false trace identification problems we believe.

Once **G** are obtained, observed traces with latent generation probabilities lower than one tenth of the smallest occurrence frequency among all observed traces, an objective threshold we proposed, may be false traces. The acceptance of the identification result depends on the result of a χ^2 test with a specified confidence level $1 - \alpha$. If the test fails, which indicates that **F** cannot reflect **P** of traces, the identification result should be rejected. It is necessary to note that passing χ^2 test is not a sufficient condition but a necessary condition.

3.3 False Trace Discovery

It is general that only partial information about pollution is known. For example, an observed trace is found being polluted by chance, but it is unknown what

[2] `http://www.lindo.com/`

the ratio of polluted traces versus observed traces is. Yet the FATILP can still help discover possible false traces by trying all types of known pollution and valid pollution ratios with the partial information. Two heuristic rules are used to find a better pollution description: 1) The smaller the distance, the better the pollution description. 2)The smaller the distance, the better the pollution ratio.

We believe that the false traces discovery is an interactive process. At first the FATILP is run with transformation matrix, elements of which are initialized either based on partial pollution information or with equal transformation probability. After the estimated results have been analyzed, the information about the pollution improved, and the element values of the transformation matrix revised, the FATILP will be run again. Iteratively the most possible pollution type of the log, the appropriate pollution ratio and all possible false traces will be found out at last. The FATILP is an indispensable tool during the interaction.

4 Experiments

Experiments were carried out to 1) validate the proposed approach, 2) demonstrate how to discover an appropriate pollution ratio as well as 3) the most possible pollution type for a given polluted log.

4.1 Experiment Design

An experiment consists of two steps, 1) generating logs according to the specified generation probabilities of normal traces, pollution type, pollution ratio and log length, and 2) identifying false trace as well as evaluating experiment results.

For a process model shown in Fig. 3, there are four normal traces $T_1(ACDG)$, $T_2(ADCG)$, $T_3(BEH)$, and $T_4(BFH)$. For these traces, we here define three typical generation probability distributions of normal traces as shown in Table 2.

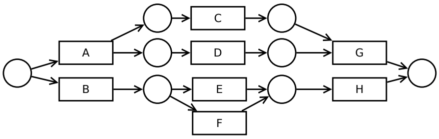


Fig. 3. A simplified business process model

Table 2. Generation probability distributions

Type	$P(T_1)$	$P(T_2)$	$P(T_3)$	$P(T_4)$
B(balanced)	0.25	0.25	0.25	0.25
N(unbalanced)	0.59	0.35	0.05	0.01
I(ext-unbalanced)	0.6999	0.25	0.05	0.0001

Two types of pollution are simulated, pollution D that $R\%$ of traces are polluted by missing an event and every event of a trace may be missed with the same probability and pollution E that $R\%$ of traces are polluted by exchanging orders of two adjacent events and every pair of adjacent events of a trace may be exchanged with the same probability. It is necessary to note that although complicated pollution, e.g. the two elementary types being combined together and/or repeated some times, may lead to diverse element values of the transformation matrix, this diversity can be approximated by means of elementary pollution along with various generation probability distributions of traces (refer to the Q^2). That is the reason why pollution types D and E are selected.

To evaluate the performance of approaches for the false trace identification problem, the correct identification rate, $h = \frac{tp+tn}{tp+fn+tn+fp}$, is defined, where tp and fn are the numbers of false traces being identified as false traces and as true traces respectively, tn and fp are the numbers of true traces being identified as true traces and as false traces respectively. Each experiment is repeated 100 times on 100 logs and the average values are used for evaluation.

To distinguish one experiment from the others, we name an experiment with a code $XYZK$, where X is a pollution type (D or E), Y is a pollution ratio ($Y \times 10\%$), Z is a generation distribution type (B,N or I), and K is sample size, i.e. log length. The K may be omitted when the sample size is $5k$.

4.2 Experiment Results

In Fig. 4, both sub-figures (a) and (b) depict that the performance of the proposed approach decreases from on balanced logs to on extreme unbalanced logs, but values of best performance of all experiments are greater than 0.9. The approach is so sensitive to the pollution ratio that the best identification rates can only be achieved around the real pollution ratio, 50%, no matter what the pollution type is and what the distribution is. Both sub-figures (c) and (d) show that the performance of the approach gets better when the length of the extreme unbalanced log increases. We can conclude that if the log length is big enough, the performance of the approach on the extreme unbalanced logs would be as good as that on balanced logs, and there is no significant difference between performance of the proposed approach on logs with E and that on logs with D. Thus to illustrate the performance of the approach, experiments on logs with any type of probability distributions and any type of pollution are acceptable.

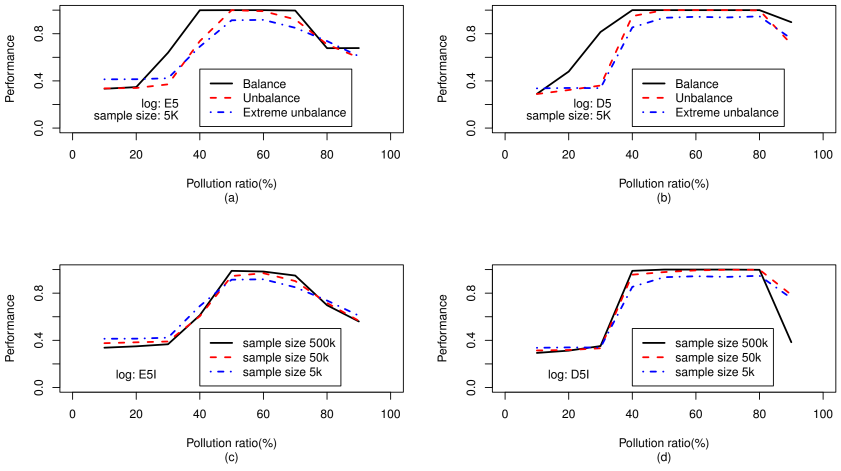


Fig. 4. Approach performance

Table 3. Performance comparison

	D5B	D5N	D5I	D5I50k	D5I500k	E5B	E5N	E5I	E5I50k	E5I500k
FATILP	1.00	1.00	0.94	0.98	1.00	1.00	1.00	0.91	0.95	0.99
Thr	1.00	0.87	0.88	0.86	0.87	1.00	0.83	0.87	0.83	0.83

The traditional approach for noise identification in process mining is denoted as “Thr” in Table 3, which depends on an empirical threshold [15]. The results of “Thr” are based on the most appropriate threshold values respectively. Although the FATILP works as well as “Thr” on balanced logs, it works better than “Thr” on unbalanced logs. It is interesting that the performance of FATILP increases when the log length increases, while the “Thr” does not. The main reason, we believe, is that the proposed approach considers the nature of pollution by means of modeling the process of pollution of event logs in a probabilistic manner.

Table 4. Average Q^2/h of each experiment on E polluted balanced logs

	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
$R_A = .1$	0.00/1.00	0.01/1.00	0.01/0.67	0.02/0.37	0.02/0.33	0.03/0.33	0.03/0.33	0.04/0.33	0.05/0.16
$R_A = .2$	0.10/1.00	0.00/1.00	0.01/1.00	0.03/0.64	0.04/0.33	0.06/0.33	0.07/0.33	0.08/0.33	0.10/0.16
$R_A = .3$	0.37/1.00	0.05/1.00	0.00/1.00	0.02/0.98	0.05/0.63	0.08/0.34	0.11/0.33	0.14/0.23	0.18/0.16
$R_A = .4$	0.84/1.00	0.21/1.00	0.04/1.00	0.00/1.00	0.03/1.00	0.08/0.65	0.14/0.37	0.19/0.18	0.29/0.16
$R_A = .5$	1.44/1.00	0.46/1.00	0.15/1.00	0.03/1.00	0.00/1.00	0.03/1.00	0.13/0.91	0.25/0.49	0.41/0.16
$R_A = .6$	1.65/0.50	0.70/0.50	0.34/0.99	0.13/1.00	0.03/1.00	0.00/1.00	0.03/1.00	0.16/1.00	0.43/0.66
$R_A = .7$	1.38/0.18	0.67/0.50	0.43/0.62	0.26/0.67	0.12/0.99	0.03/1.00	0.00/1.00	0.04/1.00	0.23/1.00
$R_A = .8$	1.00/0.00	0.65/0.50	0.48/0.66	0.34/0.66	0.22/0.67	0.11/0.88	0.03/1.00	0.00/1.00	0.06/1.00
$R_A = .9$	0.82/0.12	0.68/0.48	0.54/0.66	0.41/0.66	0.30/0.68	0.20/0.73	0.11/0.83	0.04/1.00	0.00/1.00

Table 5. Discovering pollution type on D polluted logs

	h_E	Q_E^2	χ_E^2	h_D	Q_D^2	χ_D^2
$R_A = 0.1$	0.27	0.09	$4.7E + 2$	0.29	0.05	$2.4E + 2$
$R_A = 0.2$	0.27	0.20	$1.0E + 3$	0.47	0.09	$4.4E + 2$
$R_A = 0.3$	0.27	0.33	$1.8E + 3$	0.82	0.10	$5.1E + 2$
$R_A = 0.4$	0.28	0.46	$2.7E + 3$	1.00	0.04	$2.2E + 2$
$R_A = 0.5$	0.28	0.62	$4.1E + 3$	1.00	0.00	$1.1E + 1$
$R_A = 0.6$	0.32	0.78	$6.4E + 3$	1.00	0.04	$2.2E + 2$
$R_A = 0.7$	0.34	0.95	$1.1E + 4$	1.00	0.16	$9.7E + 2$
$R_A = 0.8$	0.54	1.10	$2.9E + 4$	1.00	0.37	$2.9E + 3$
$R_A = 0.9$	0.60	1.19	$2.E + 22$	0.89	0.63	$1.0E + 4$

Table 4 contains average least distances (Q^2 values) and average performance (h values) of experiments on E polluted balanced logs. The upper line lists pollution ratios used to generate polluted logs. The left column lists assumed pollution ratios used to identify false traces. From the values in the table, we know that both the minimal value of distance is obtained and the best performance is achieved when the assumed pollution ratio equals the real ratio. This property can help discover the correct pollution ratio among assumed ratios, with which the approach performs best on a log given correct pollution type.

An example of looking for the exact pollution type as well as pollution ratio of a polluted log is presented in Table 5. A balanced log is polluted by means of pollution D with ratio 50%. To find out the real pollution, first the pollution E is assumed with pollution ratio increasing from 10% to 90%. The pollution ratio 10% seems a good choice since values of the distance Q_E^2 and the statistic χ_E^2 are minimal respectively. Second the pollution D is tried with, where both Q_D^2 and χ_D^2 reach their minimal values at ratio 50%. And both Q_D^2 and χ_D^2 with ratio 50% are less than Q_E^2 and χ_E^2 with ratio 10%, and the D pollution with ratio 50% may be a good option. Furthermore, the value of χ_D^2 with ratio 50% is 11, which is much smaller than the critical value $43.82 (= \chi^2(19))$ with a confidence level 99.9%. Therefore the pollution D with ratio 50% is acceptable. Thus the proposed approach help find out the real pollution type of a polluted log.

5 Conclusions and Future work

In this paper, the noise identification problem of event logs for process mining was discussed. We distinguished the concept of false trace, i.e. the invalid traces as regards the process model to be discovered, from that of the polluted trace, i.e. noise, and focused on the false trace identification problem. On some natural and reasonable assumptions, we characterised the problem and modeled it as a quadric optimization problem of estimating a probability distribution. Then we proposed a common approach, FATILP, to estimate the latent generation probability distribution of normal traces given a polluted log and a description of pollution, and then to identify false traces at a user-specified confidence level. Experiment results show that the proposed approach works better than traditional approaches, and it can be applied not only to identify false traces in a polluted logs but also to discover the most possible pollution type of the log as well as an appropriate pollution ratio interactively.

The work presented in this paper may be extended in several directions. First, the approach may be improved by taking informative completeness of event logs into consideration. Second, it may be possible to improve the precision of the estimation of latent generation probabilities of observed traces. Third, the approach may be extended to deal with new types of pollution, e.g. duplicate records of an event.

Acknowledgements. The authors are indebted to Professor Wil M. P. van der Aalst for his suggestions on the motivation of the topic, to Professor Arthur H. M. ter Hofstede for his advice on the topic, constructive suggestions, and continuous encouragement, and to Dr. Moe T. Wynn for her suggestions on the presentation of the paper.

The work reported in this paper is partially supported by the 973 Plan Program of China (No. 2009CB320700), the 863 Plan Program of China (No. 2012AA040911 and No. 2012AA040904) and NSF Projects of China (No. 61073005 and No. 61003099).

References

1. Aires da Silva, G., Ferreira, D.R.: Applying hidden markov models to process mining. In: Rocha, A., Restivo, F., Reis, L.P., Ao, S.T. (eds.) Sistemas e Tecnologias de Informação: Actas da 4a. Conferência Ibérica de Sistemas e Tecnologias de Informação, pp. 207–210. AISTI/FEUP/UPF (2009)
2. Alves de Medeiros, A.K., Weijters, A.J.M.M., van der Aalst, W.M.P.: Genetic process mining: an experimental evaluation. Data Mining and Knowledge Discovery 14(2), 245–304 (2007)
3. Cook, J., Du, Z., Liu, C., Wolf, A.: Discovering models of behavior for concurrent workflows. Computers in Industry 53(3), 297–319 (2004)
4. Donoho, D.: De-noising by soft-thresholding. IEEE Transactions on Information Theory 41(3), 613–627 (1995)
5. Dumas, M., van der Aalst, W.M.P., ter Hofstede, A.H.M. (eds.): Process-Aware Information Systems: Bridging People and Software through Process Technology. Wiley Interscience, Hoboken (2005)
6. Maruster, L., Weijters, A.J.M.M., van der Aalst, W.M.P., Bosch, A.: A Rule-Based Approach for Process Discovery: Dealing with Noise and Imbalance in Process Logs. Data Min. Knowl. Discov. 13(1), 67–87 (2006)
7. Portilla, J., Strela, V., Wainwright, M.J., Simoncelli, E.P.: Image denoising using scale mixtures of gaussians in the wavelet domain. IEEE Transactions on Image Processing 12(11), 1338–1351 (2003)
8. Rahm, E., Do, H.H.: Data cleaning: Problems and current approaches. IEEE Data Eng. Bull. 23(4), 3–13 (2000)
9. van der Aalst, W.M.P.: Process Mining: Discovery, Conformance and Enhancement of Business Processes. Springer, Berlin (2011)
10. van der Aalst, W.M.P.: Process mining: Overview and opportunities. ACM Trans. Management Inf. Syst. 3(2), 1–17 (2012)
11. van der Aalst, W., Adriansyah, A., de Medeiros, A.K.A., Arcieri, F., Baier, T., Blickle, T., Bose, J.C., van den Brand, P., Brandtjen, R., Buijs, J., Burattin, A., Carmona, J., Castellanos, M., Claes, J., Cook, J., Costantini, N., Curbera, F., Damiani, E., de Leoni, M., Delias, P., van Dongen, B.F., Dumas, M., Dustdar, S., Fahland, D., Ferreira, D.R., Gaaloul, W., van Geffen, F., Goel, S., Günther, C., Guzzo, A., Harmon, P., ter Hofstede, A., Hoogland, J., Ingvaldsen, J.E., Kato, K., Kuhn, R., Kumar, A., La Rosa, M., Maggi, F., Malerba, D., Mans, R.S., Manuel, A., McCreesh, M., Mello, P., Mendling, J., Montali, M., Motahari-Nezhad, H.R., zur Muehlen, M., Munoz-Gama, J., Pontieri, L., Ribeiro, J., Rozinat, A., Seguel Pérez, H., Seguel Pérez, R., Sepúlveda, M., Sinur, J., Soffer, P., Song, M., Sperduti, A., Stilo, G., Stoel, C., Swenson, K., Talamo, M., Tan, W., Turner, C., Vanthienen, J., Varvaressos, G., Verbeek, E., Verdonk, M., Vigo, R., Wang, J., Weber, B., Weidlich, M., Weijters, T., Wen, L., Westergaard, M., Wynn, M.: Process mining manifesto. In: Daniel, F., Barkaoui, K., Dustdar, S. (eds.) BPM Workshops 2011, Part I. LNBIP, vol. 99, pp. 169–194. Springer, Heidelberg (2012)
12. van der Aalst, W.M.P., van Hee, K.M.: Workflow Management: Models, Methods, and Systems. MIT Press, Cambridge (2004)
13. van der Aalst, W.M.P., Weijters, A.J.M.M., Maruster, L.: Workflow Mining: Discovering Process Models from Event Logs. IEEE Transactions on Knowledge and Data Engineering 16(9), 1128–1142 (2004)

14. van Dongen, B.F., Alves de Medeiros, A.K., Wen, L.: Process Mining: Overview and Outlook of Petri Net Discovery Algorithms. In: Jensen, K., van der Aalst, W.M.P. (eds.) ToPNoC II. LNCS, vol. 5460, pp. 225–242. Springer, Heidelberg (2009)
15. Weijters, A.J.M.M., van der Aalst, W.M.P.: Rediscovering Workflow Models from Event-Based Data Using Little Thumb. Integrated Computer-Aided Engineering 10(2), 151–162 (2003)
16. Xiong, H., Pandey, G., Steinbach, M., Kumar, V.: Enhancing Data Analysis with Noise Removal. IEEE Transactions on Knowledge and Data Engineering 18(3), 304–319 (2006)

Split-Merge Augmented Gibbs Sampling for Hierarchical Dirichlet Processes

Santu Rana, Dinh Phung, and Svetha Venkatesh

Center for Pattern Recognition and Data Analytics
Deakin University
Waurn Ponds, VIC 3216

Abstract. The Hierarchical Dirichlet Process (HDP) model is an important tool for topic analysis. Inference can be performed through a Gibbs sampler using the auxiliary variable method. We propose a split-merge procedure to augment this method of inference, facilitating faster convergence. Whilst the incremental Gibbs sampler changes topic assignments of each word conditioned on the previous observations and model hyper-parameters, the split-merge sampler changes the topic assignments over a group of words in a single move. This allows efficient exploration of state space. We evaluate the proposed sampler on a synthetic test set and two benchmark document corpus and show that the proposed sampler enables the MCMC chain to converge faster to the desired stationary distribution.

1 Introduction

The hierarchical Dirichlet process (HDP) [1] is an important tool for Bayesian nonparametric topic modelling, particularly when mixed-membership exists, such as in a document collection. Each document is modelled as a group of words, generated from the underlying latent topic. It is an extension of Latent Dirichlet Allocation (LDA) [2], allowing unbounded latent dimensionality, with capacity to automatically infer the number of topics in a document set. The HDP is a hierarchial version of the Drichilet process (DP) clustering model, where a corpus of documents are assumed to be generated from a set of top-level topics with independent mixing distribution. In contrast to the DP mixture model for which the metaphor is a Chinese Resturant Process (CRP), a HDP can be expressed using a metaphor of Chinese Restaurant Franchise (CRF), where a set of dishes is shared across a collection of franchise restaurants, each having tables generated using a CRP from the customers arriving at that franchise.

As with Bayesian nonparametric models, exact posterior inference is not tractable. MCMC or variational approximation are used for approximate posterior inference. In this paper, we focus on MCMC sampling, wherein posterior is computed from the empirical distribution of samples from a Markov chain, whose stationary distribution is the posterior of interest. [1] propose two MCMC sampling schemes, one based on the CRF and the other on the auxiliary variable

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 546–557, 2013.

method. In many cases the use of auxiliary variable sampling method is preferred to keep the sampling simple and easily extendable to elaborate models such as iHMM [1]. The basic MCMC sampler for the HDP is an incremental Gibbs sampler - the topic is sampled for a single observation, one at a time, conditioned on preceding observations and model hyperparameters. Since, only one state change takes place at a time, mixing may be slow, requiring many Gibbs iterations for the MCMC chain to converge to its stationary distribution. Whereas CRF based sampling is staightforward in formulation, the implementation is tedious, requiring tracking of individual table assignments for each restaurant, and then tracking the dish preference for each table. The auxiliary variable split-merge sampler is based on directly sampling the topic assignment (dish) of the words (customers) in the documents and thus straightforward in implementation.

Split-merge MCMC samplers have been proposed for Bayesian nonparametric models, such as for DPM to accelerate mixing [3]. In a split-merge setting, a group of observations are moved together in the state space based on whether splitting or merging of topics are accepted based on a Metropolis-Hastings ratio. In practice, each sampling run consists of a Gibbs sampling followed by a split-merge proposal evaluation. Since, the state change may occur for a group of points at each iteration, the MCMC chain can quickly traverse the state-space and potentially converge faster than if only the Gibbs sampler is used.

Motivated by this, we propose a split-merge procedure for the HDP to accelerate the mixing of the MCMC chain for the auxiliary variable sampling scheme called the *Split-Merge Augmented Gibbs sampler*. Assuming each word (customer) in the document corpus has been assigned to a topic(dish) at the higher level, we evaluate a split-merge proposal on the customer-dish relationship i.e. we either propose to split all the customers in all the franchise restaurants who share the same dish into two different dishes or propose merging the set of all customers sharing two different dishes. In contrast to the CRF based split-merge sampling scheme [4], we do not worry about the lower-level customer-table assignments and thus the proposed split-merge scheme is effective at both levels of the HDP.

We evaluate and analyze the proposed algorithm on synthetic data and two benchmark document corpus, - NIPS abstracts and 20 News Group data. In synthetic experiments, we generate topics with low separability and show that the incremental Gibbs sampler is unable to recover all the correct topics; however, our split-merge augmented Gibbs sampler is able to recover all topics correctly. For the document corpus, we evaluate the performance of Gibbs vs our sampler based on the perplexity of held-out data and show that our proposed method is able to produce lower perplexity in similar time.

The layout is as follows: Related background on HDP and inference techniques is described in the section 2; in the section 3, we detail the split-merge procedure after briefly reviewing the Gibbs sampling procedure based on the auxiliary variable scheme. Experimental results are discussed in section 4 and finally, section 5 concludes our discussion.

2 Related Background

Dirichlet Proess (DP) mixture model for clustering with theoretically unbounded mixture component has been first studied in [5] with [6] giving a stick-breaking construction for the DP prior. Hierarchical Dirichlet Process (HDP) extends the DP in two level where the bottom level DP uses the top-level DP as the base measure was first proposed in [1]. This is a mixed-membership model where a group is sampled from a mixture of topics and has been used extensively for document analysis [7], multi-population haplotype phasing [8], image/object retrieval [9] etc.

Split-merge sampling for DP mixture model was first proposed in [3] and splitting of a single cluster by running a Restricted Gibbs sampler on the subset of points belonging to that topic is described. Whilst a merge proposal is easy to generate, generating a split proposal takes some work as a random split will most likely to be a bad proposal and they would be rejected. Hence, the need for the Restricted Gibbs sampler. Using the same framework [10] proposed a slightly different split-merge algorithm by having a simpler split routine using a sequential allocation scheme. In contrast to running a Gibbs sampler to generate a split proposal they proposed a single run sequential allocation scheme to generate the split, thus reducing the overhead cost. Split-merge sampler for HDP based on the Chinese Restaurant Franchise sampling scheme has been proposed in [4]. This perform splitting or merging only at the top level assignments using the similar procedure for the DP with additional factors coming from the bottom level when computing the prior clustering probability.

3 Framework

3.1 Hierarchical Dirichlet Process

The hierarchical Dirichlet process is a distribution over a set of random probability measure over $(\Theta, \mathcal{B})$. It is a hierarchical version of the DP, where a set of group level random probability measures $(G_j)_{j=1}^{J}$ are defined for each group which shares a global random probability measure G_0 at the higher level. The global measure G_0 is a draw from a DP with a base measure H and a concentration parameter γ. The group specific random measures G_j are subsequently drawn from a DP with G_0 as its base measure,

$$G_0 \sim DP(\gamma, H) \tag{1}$$

$$G_j / G_0 \sim DP(\alpha_0, G_0) \tag{2}$$

with j denoting the group. Since G_j are drawn from the almost surely discrete distribution of G_0, it ensures that the top level atoms are shared across the groups. In the topic model context each document is a group of words and the

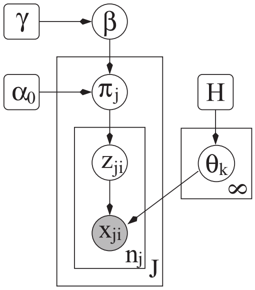


Fig. 1. The HDP model

atoms (topics) are the distribution over words. The stick-breaking representation of G_0 can be expressed as,

$$G_0 = \sum_{k=1}^{\infty} \beta_k \delta_{\theta_k} \tag{3}$$

where $\theta_k \sim H$ independently and $(\beta_k)_{k=1}^{\infty}$ admitting stick-breaking construction such that $(\beta_k)_{k=1}^{\infty} \sim Stick(\gamma)$. Since, G_0 is used as the base measure for G_j, it can be expressed as,

$$G_j = \sum_{k=1}^{\infty} \pi_{jk} \delta_{\theta_k} \tag{4}$$

where it can be shown that [1] $\pi_j \sim DP(\alpha_0, \beta)$. The stick-breaking representation for HDP is given below,

$$\begin{aligned} \beta|\gamma &\sim Stick(\gamma) && \\ \pi_{j|\alpha_0,,\beta} &\sim DP(\alpha_0, \beta) && z_{ji}|\pi_j \sim \pi_j \\ \theta_k|H &\sim H && x_{ji}|z_{ji}, (\theta_k)_{k=1}^{\infty} \sim F(\theta_{z_{ji}}) \end{aligned} \tag{5}$$

3.2 Posterior Inference with Auxiliary Variable

With the stick breaking representation of 5, the state space consists of $(\mathbf{z}, \pi, \beta, \theta)$. Since $\mathbf{z}$ and π forms a conjugate pair, π can be integrated out giving the conditional probability of $\mathbf{z}$ given β as.

$$P(\mathbf{z}|\beta) = \prod_{j=1}^{J} \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_0 + n_j)} \prod_{k=1}^{K} \frac{\Gamma(\alpha_0\beta_k + n_{jk})}{\Gamma(\alpha_0\beta_k)} \tag{6}$$

From 6 the prior probability for z_{ji} given $\mathbf{z}^{-\mathbf{ji}}$ and β can be expressed as,

$$p(z_{ji} = k|\mathbf{z}^{-ji}, \beta) = \frac{(\alpha_0\beta_k + n_{jk}^{-ji})}{\alpha_0 + n_j} \quad for \quad k = 1, ...K, u \tag{7}$$

where β =$[\beta_1\beta_2...\beta_u]$ such that $\beta_u = \sum_{k=K+1}^{\infty} \beta_k$. Adding the likelihood term we can have the sampling formula of z_{ji} as,

$$p(z_{ji} = k|\mathbf{z}^{-ji}, \beta) \propto (\alpha_0\beta_k + n_{jk}^{-ji}) f(x_{ji}/\theta_k) \quad for \quad k = 1, ...K, u \tag{8}$$

where θ_u is sampled from its prior H. If a new topic $(K + 1)$ is created then we set $\beta_{K+1} = b\beta_u$, where $b \sim Beta(1, \gamma)$. To sample β we use the auxiliary variable method as outlined in [Teh]. We first sample the auxiliary variable $\mathbf{m}$ from,

$$q(m_{jk} = m|\mathbf{z}, \mathbf{m}^{-jk}, \beta) \propto s(n_{jk}, m)(\alpha_0\beta_k)^m \tag{9}$$

where $s(n_{jk}, m)$ are the unsigned Stirling numbers of the first kind. Subsequently, β is sampled from,

$$q(\beta|\mathbf{z}, \mathbf{m}) \propto \beta_u^{\gamma-1} \prod_{k=1}^{K} \beta_k^{\sum_j m_{jk}-1} \tag{10}$$

Eq 8910 completes the Gibbs sampling formula for HDP inference. For elaboration please refer to [1,7].

3.3 Split-Merge procedure

The split-merge proposal is a form of Metropolis-Hasting algorithm where the algorithm draws a new candidate state C^* from a distribution with density $\pi(C)$ according to a proposal density $q(C^*/C)$ and then evaluation of the proposal based on the Metropolis-Hasting ratio of

$$a(C^*, C) = min[1, \frac{q(C|C^*)\pi(C^*)}{q(C^*|C)\pi(C)}]$$

The proposal $C^*$is accepted with the probability $a(C^*, C)$. If it is accepted the state changes to C^* or it remains at C. For HDP mixture model the above formula takes the form of

$$a(C^*, C) = min[1, \frac{q(C|C^*)P(C^*)L(C^*|\mathbf{x})}{q(C^*|C)P(C)L(C|\mathbf{x})}$$

From this prior distribution of $P(\mathbf{z}|\beta)$ (Eq. 7) we can use the Polya's urn metaphor to create a sequence $[z_{ji_1}z_{ji_2}...z_{jn_j}]$ for a particular document j given β as,

$$P(z_{ji} = k|c_1, c_2, ..., c_k; z_{j1}, z_{j2}, ..., z_{ji-1}; \beta) = \frac{\alpha_0\beta_k + n_{jc_k}^{<i}}{\alpha_0 + i - 1} \, for \; k <= K$$
$$= \frac{\alpha_0\beta_u}{\alpha_0 + i - 1} \, for \; k = K + 1$$

where c_k is the $k'th$ topic. Given this assignement scheme, the probability of a particular configuration of word assignments $\mathbf{C} = \{n_{jc_1}, n_{jc_2}, ..., n_{jc_K}\}_{j=1}^{J}$ to the topic set $\{c_k\}_{k=1}^{K}$ can be expressed as,

$$P(\mathbf{C}|\beta) = \frac{\alpha_0^K \beta_{u_1}\beta_{u_2}...\beta_{u_k} \prod_{j=1}^{J}\prod_{k=1}^{K} < \alpha_0\beta_k >_{n_{jk}}}{\prod_{j=1}^{J}\prod_{i=1}^{n_j}(\alpha_0 + i - 1)} \tag{11}$$

where $\beta_{u_k} = \sum_{l=k}^{\infty}\beta_l$ and $< \alpha_0\beta_k >_{n_{jk}} = \alpha_0\beta_k(\alpha_0\beta_k + 1)...(\alpha_0\beta_k + n_{jk} - 1)$ denotes the rising factorial and can be computed as the ratio of two gamma functions. For a split proposal a particular topic k is splitted in k_1 and k_2 and the new configuaration is denoted as $\mathbf{C}^{split}$. After we generate new latent assignements $\mathbf{z}^{split}$ corresponding to $\mathbf{C}^{split}$, we resample βusing 9 and 10 to obtain β^{split}.The configuration probability of $P(\mathbf{C}^{split}/\beta^{split})$ can now be computed as,

$$P(\mathbf{C}^{split}|\beta^{split}) = \frac{\alpha_0^{K+1} \beta_{u_1}\beta_{u_2}...\beta_{u_{k+1}} \prod_{j=1}^{J}\prod_{k=1}^{K+1} < \alpha_0\beta_k >_{n_{jk}}}{\prod_{j=1}^{J}\prod_{i=1}^{n_j}(\alpha_0 + i - 1)} \tag{12}$$

Now we can compute $\frac{P(\mathbf{C}^{split}|\beta^{split})}{P(\mathbf{C}|\beta)}$ as the ratio of 12 and 11. Similarly for merge proposal when topics k_1 and k_2 are merged into a single topic k and β_k^{merge} is sampled with the new $\mathbf{z}^{merge}$, then we have,

$$P(\mathbf{C}^{merge}|\beta^{merge}) = \frac{\alpha_0^{K-1} \beta_{u_1}\beta_{u_2}...\beta_{u_{k-1}} \prod_{j=1}^{J}\prod_{k=1}^{K-1} < \alpha_0\beta_k >_{n_{jk}}}{\prod_{j=1}^{J}\prod_{i=1}^{n_j}(\alpha_0 + i - 1)} \tag{13}$$

from which the ratio $\frac{P(\mathbf{C}^{split}|\beta^{split})}{P(\mathbf{C}|\beta)}$ can be computed from 13 and 11. In our proposed method we will use the conditional configuration probability ratio in place of $\frac{P(C^*)}{P(C)}$ as our target distribution is $\pi(\mathbf{C}|\beta)$.

The likelihood term $L(C/\mathbf{x})$ is computed over all the words of all the documents and is given as,

$$L(C|\mathbf{x}) = \prod_{j=1}^{J}\prod_{i=1}^{n_j}\int f(x_{ji}, \theta)dH_{ji,c_{ji}}(\theta)$$

where $H_{ji,c_{ji}}$is the posterior distribution of θ based on the prior G_0 and all the observations $x_{j',i'}$ such that $j' < j$ and $i' < i$. The above integral is analytically tractable if G_0 is conjugate prior. We can express the above likelihood equation as a product over topics such that,

$$L(C|\mathbf{x}) = \prod_{j=1}^{J}\prod_{c=1}^{K}\prod_{i:C_{ji}=c}\int f(x_{ji},\theta)dH_{ji,c}(\theta)$$

Expressing this way now we can compute the ratio of likelihoods between a split proposal C^{split} and the existing configuration C as,

$$\frac{L(C^{split}|\mathbf{x})}{L(C|\mathbf{x})} = \frac{\prod_{j=1}^{J}\prod_{i:C_{ji}^{split}=k1}\int f(x_{ji},\theta)dH_{ji,k1}(\theta)\prod_{j=1}^{J}\prod_{i:C_{ji}^{split}=k2}\int f(x_{ji},\theta)dH_{ji,k2}(\theta)}{\prod_{j=1}^{J}\prod_{i:C_{ji}=k}\int f(x_{ji},\theta)dH_{ji,k}(\theta)} \tag{14}$$

Similarly, for merge proposal the ratio of likelihood is,

$$\frac{L(C^{merge}|\mathbf{x})}{L(C|\mathbf{x})} = \frac{\prod_{j=1}^{J}\prod_{i:C_{ji}^{merge}=k}\int f(x_{ji},\theta)dH_{ji,k}(\theta)}{\prod_{j=1}^{J}\prod_{i:C_{ji}=k1}\int f(x_{ji},\theta)dH_{ji,k1}(\theta)\prod_{j=1}^{J}\prod_{i:C_{ji}=k2}\int f(x_{ji},\theta)dH_{ji,k2}(\theta)} \tag{15}$$

To evaluate the proposal density $q(C^*|C)$ we need to create an algorithm for creating C^* from the existing configuaration C. Here we use sequential assignment method similar to [10] for that. Let us assume that we are generating a split proposal for the topic k into two topics k_1 and k_2. We need to divide the words $S = \{n_{jc_k}\}_{j=1}^{J}$ into two sets $S_{k_1} = \{n_{j_{k_1}}\}_{j=1}^{J}$ and $S_{k_2} = \{n_{j_{k_2}}\}_{j=1}^{J}$. We start with a random word from a random document as the seed for the topic k_1 and similarly for topic k_2 i.e. $S_{k_1} = \{x_{jr_1,ir_1}\}_{(jr_1,ir_1)\in S}$ and $S_{k_1} = \{x_{jr_2,ir_2}\}_{(jr_2,ir_2)\in S}$ such that $(jr_1, ir_1) \neq (jr_2, ir_2)$. The rest of the words from the set S can be assigned by sampling from,

$$\begin{aligned}P(k_{ji} = k_1|S_{k_1}, S_{k_2}, \theta, x_{ji}) \\ = \frac{(\alpha_0\beta_{k_1}^{split}+|S_{k_1}^{j}|-1)p(x_{ji}|\theta_{S_{k_1}})}{(\alpha_0\beta_{k_1}^{split}+|S_{k}^{j}|-1)p(x_{ji}|\theta_{S_{k_1}})+(\alpha_0\beta_{k_2}^{split}+|S_{k_2}^{j}|-1)p(x_{ji}|\theta_{S_{k_2}})}\end{aligned} \tag{16}$$

$$\begin{aligned}P(k_{ji} = k_2|S_{k_1}, S_{k_2}, \theta, x_{ji}) \\ = \frac{(\alpha_0\beta_{k_2}^{split}+|S_{k_2}^{j}|-1)p(x_{ji}|\theta_{S_{k_2}})}{(\alpha_0\beta_{k_1}^{split}+|S_{k}^{j}|-1)p(x_{ji}|\theta_{S_{k_1}})+(\alpha_0\beta_{k_2}^{split}+|S_{k_2}^{j}|-1)p(x_{ji}|\theta_{S_{k_2}})}\end{aligned} \tag{17}$$

where, a simple allocation of $\beta_{k_1}^{split}$and $\beta_{k_2}^{split}$ can be assigned as $\beta_{k_1}^{split} = \beta_{k_2}^{split} = \beta_k/2$. The proposal probability $q(\mathbf{C}^{split}|\mathbf{C})$ is computed as a product of the above probabilities based on the actual assignment. The reverse proposal probability $q(\mathbf{C}|\mathbf{C}^{split}) = 1$ since the set of two sets of words can only be combined in a single way. We propose merge proposal as combining the two topics k_1 and $k_2$into a single topic k. In this case $q(\mathbf{C}^{merge}|\mathbf{C}) = 1$, however, to compute the reverse proposal probability $q(\mathbf{C}|\mathbf{C}^{merge})$ we need to create a dummy split proposal and compute $q(\mathbf{C}|\mathbf{C}^{merge}) = q(\mathbf{C}^{dummysplit}|\mathbf{C}^{merge})$ following the previously described split procedure.

Our split-merge procedure runs after each Gibbs iteration and at each run of aplit-merge procedure we either select to perform a split or merge. Till now we have not discussed whether a split or a merge proposal is to be evaluated. The simplest way to determine that by way of sampling two random words from the document corpus and then depending on whether they belong to the same topic or not we evaluate a split or merge proposal respectively. Whilst this scheme works fine it is understood that with the increasing number of topics we may encounter more merge proposal being evaluated than split proposals. To circumvent that we propose sampling from a binary random variable with equal probability of selecting a merge or split proposal at each run. When a split proposal has to be created we first select a topic at random and then proceed with splitting that topic, similarly when a merge proposal has to be created we select two topics at random and then proceed with the merging. From our experience this provides faster convergence than the naive method.

4 Experiments

We evaluate our proposed split-merge algorithm for HDP topic models for both synthetic and real world data. In all experiments, we run the normal conditional Gibbs sampler and the proposed split-merge augmented Gibbs sampler for the HDP model, with identical initialization of state space and variables. The normal Gibbs sampler visits each document and all words within it sequentially, assigning each to one to an existing topic or creating a new one based on the predictive likelihood of the word. The split-merge augmented Gibbs sampler runs a Gibbs iteration followed by the split-merge procedure. A split or a merge is proposed based on user-defined selection probability (a simple scheme is to have equal probability of acceptance). Depending on whether a split or merge has been selected, we pick two words randomly from a single topic or from two different topics for split and merge respectively. We then propose the split or the merge and accept them based on its acceptance probability.

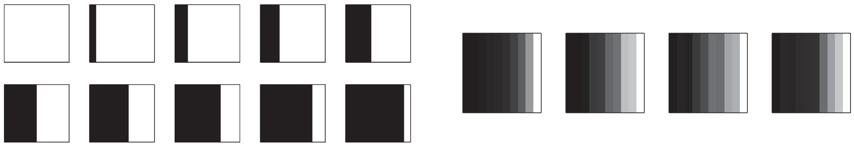
4.1 Synthetic Data

We use synthetic data to demonstrate the performance of our proposed split-merge augmented sampler in comparison to the simple conditional Gibbs sampler. We generate 10 topics from a vocabulary size of 10. The topics are created such that the first topic uses all the words with equal probability, and the rest use lesser number of words, with the last topic using only a single word, as shown in the Fig 2a. Fig 2b shows the extracted four groups. The topic mixture for each group has been generated as a random simplex.

Both the Gibbs sampler and the split-merge augmented Gibbs samplers are run for 1000 iterations and the posterior for the cluster number is shown in the Fig 3b and Fig 3a respectively. Whilst the naive conditional sampler fails to recover exact topics even after 1000 iterations, the split-merge augmented Gibbs Sampler is able to find the correct number of topics within the first 25

Algorithm 1. Split-merge augmented Gibbs sampler for HDP

```
For each iteration:
  – Perform Gibbs sampling using auxiliary variable scheme (Eq. 8,9, and10).
  – Choose a split or merge decision by sampling $t \sim Bern(0.5)$ with $t = 0$ indicating a split and $t = 1$ indicating a merge.
  – If split:
      • Randomly select a topic to split.
      • Split the chosen topic into two and generate $\mathbf{z}^{split}$ using Eq. 16 and 17.
      • Resample $\beta^{split}$ using Eq. 9 and 10.
      • Compute the proposal likelihood ratio $(\frac{P(\mathbf{C}^{split}|\beta^{split})}{P(\mathbf{C}|\beta)})$ from Eq. 12 and 11.
      • Compute likelihoods ratio $(\frac{L(C^{split}|\mathbf{x})}{L(C|\mathbf{x})})$ from Eq. 14.
      • Set $q(\mathbf{C}|\mathbf{C}^{split}) = 1$ and compute $q(\mathbf{C}^{split}|\mathbf{C})$ from Eq. 16 and 17 by multiplying the assignment probabilities.
      • Compute the Metropolis Hasting ratio
          $$a(\mathbf{C}^{split}, \mathbf{C}) = min[1, \frac{q(\mathbf{C}|\mathbf{C}^{split})P(\mathbf{C}^{split}|\beta^{split})L(\mathbf{C}^{split}|\mathbf{x})}{q(\mathbf{C}^{split}|\mathbf{C})P(\mathbf{C}|\beta)L(\mathbf{C}|\mathbf{x})}$$
      • Accept the split proposal with probability $a(\mathbf{C}^{split}, \mathbf{C})$.
      • Set $\mathbf{z} = \mathbf{z}^{split}$ and $\beta = \beta^{split}$.
  – if merge:
      • Randomly select two topics.
      • Merge them into two and generate $\mathbf{z}^{merge}$ and resample $\beta^{merge}$.
      • Create a dummy split following the split algorithm as outlined above to obtain
          $$a(\mathbf{C}^{merge}, \mathbf{C}^{dummysplit}) = min[1, \frac{q(\mathbf{C}^{dummysplit}|\mathbf{C}^{merge})P(\mathbf{C}^{merge}|\beta^{merge})L(\mathbf{C}^{merge}|\mathbf{x})}{q(\mathbf{C}^{merge}|\mathbf{C}^{dummysplit})P(\mathbf{C}^{dummysplit}|\beta^{dummysplit})L(\mathbf{C}^{dummysplit}|\mathbf{x})}$$
      • Accept the merge proposal with probability $a(\mathbf{C}^{merge}, \mathbf{C}^{dummysplit})$.
      • Set $\mathbf{z} = \mathbf{z}^{merge}$ and $\beta = \beta^{merge}$.
```

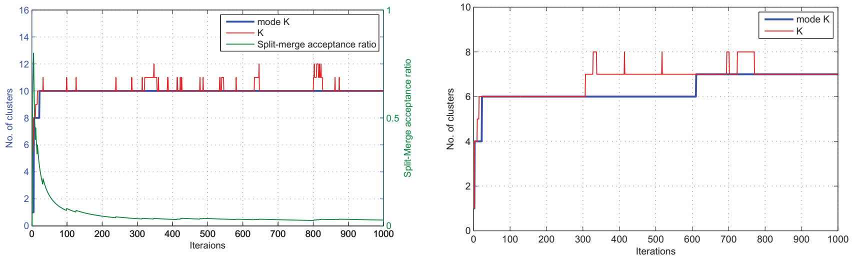


(a) The 10 topics

(b) The four groups.

Fig. 2. Synthetic experimental set up (a) the 10 topics, (b) the 4 groups represented as a bag of words

iterations. This is a significant speed up. The reason the naive Gibbs sampler fails to separate the topic is because they are not easily separable, however, our algorithm is able to split topics that are hard to separate. Fig 3a shows the split-merge acceptance ratio after each iteration. As expected the ratio falls with increasing number of samples, once all 10 topics have been recovered correctly. The confusion matrix for the topics as recovered by the two sampling algorithms is shown in Fig 4. Since the first few topics have a higher overlap, they are hard to separate.Thus it is nor surprising that the naive Gibbs sampling fails to separate them, however, our algorithm, with its capability to explore state-space in an efficient way, is able to separate the topics.



(a) Combined Gibbs and Split-Merge sampler

(b) Gibbs sampler

Fig. 3. Posterior K on synthetic data for (a) combined Gibbs and Split-Merge sampler, and (b) only the Gibbs sampler

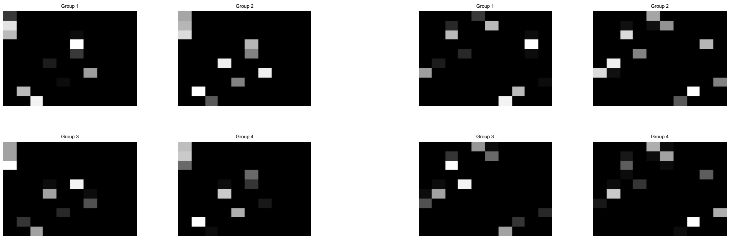
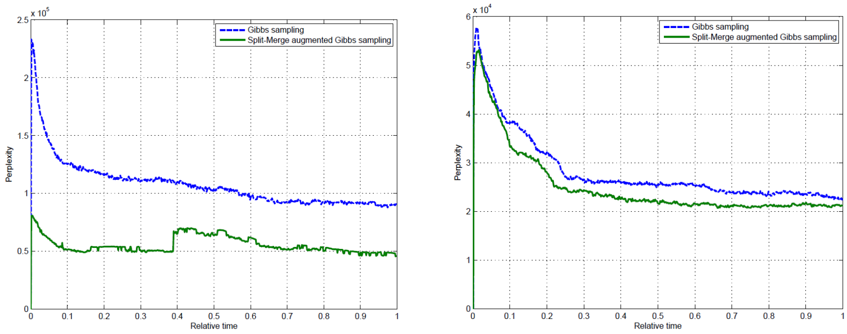


Fig. 4. Confusion matrix for topic mixtures for the four synthetic groups. Naive Gibbs sampler is in left and the split-merge augmented sampler is in right

4.2 Document Corpus

We used NIPS abstract data and 20 News Group data to study the convergence of our proposed method. NIPS0-12 data is a collection of abstracts published in the NIPS conference from the year 1988-1999. We select 1392 abstracts consisting of 263K words. The Dirichlet prior is set at $Dir(0.5)$. Both the Gibbs sampler and our sampler was initialized with the same initial topic distribution. We used random 80% of the data for topic modelling and the rest 20% data for perplexity computation. We run them for the same time and plot the the perplexity at each iteration in Fig 5a.



(a) NIPS corpus

(b) 20 News Group corpus

Fig. 5. Perplexity on the held-out data between the Split-Merge augmented Gibbs sampler and the Gibbs sampler on (a) NIPS corpus and (b) on 20 News Group corpus

The 20 News Group data contains 16242 documents with vocabulary size of 100. The Dirichlet parameter is set at 0.05. Similar to above setting, we learn our model with a random set of 80% of documents and the remaining 20% are used for perplexity computation. Both the Gibbs and our algorithm are run with the same initialization. Perplexity at each iteration is reported in Fig 5b. Superior perplexity is observed, although the algorithms ran for the same time.

5 Conclusion

In this paper we proposed a novel split-merge algorithm for HDP based on the direct conditional assignement of words-to-topics. The incremental Gibbs sampler can often be slow to mix and may often fail to provide a good posterior estimate in a limited time. The split-merge sampler with its ability to make a bigger move across the state-space mixes faster and often lead to very good posterior estimates. We experimented on both synthetic and real world data and demonstrate the convergence speedup of the proposed combined Gibbs and split-merge sampler over the plain Gibbs sampling method.

References

1. Teh, Y., Jordan, M., Beal, M., Blei, D.: Hierarchical Dirichlet processes. Journal of the American Statistical Association 101(476), 1566–1581 (2006)
2. Blei, D., Ng, A., Jordan, M.: Latent Dirichlet allocation. Journal of MachineResearch 3, 993–1022 (2003)
3. Jain, S., Neal, R.: A split-merge Markov chain Monte Carlo procedure for the Dirichlet process mixture model. Journal of Computational and Graphical Statistics 13(1), 158–182 (2004)
4. Wang, C., Blei, D.: A split-merge mcmc algorithm for the hierarchical dirichlet process. Arxiv preprint arXiv:1201.1657 (2012)
5. Antoniak, C.: Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. The Annals of Statistics 2(6), 1152–1174 (1974)
6. Sethuraman, J.: A constructive definition of Dirichlet priors. Statistica Sinica 4(2), 639–650 (1994)
7. Teh, Y., Jordan, M.: Hierarchical Bayesian nonparametric models with applications. In: Hjort, N., Holmes, C., Müller, P., Walker, S. (eds.) Bayesian Nonparametrics: Principles and Practice, p. 158. Cambridge University Press (2009)
8. Xing, E., Sohn, K., Jordan, M., Teh, Y.: Bayesian multi-population haplotype inference via a hierarchical dirichlet process mixture. In: Proceedings of the 23rd International Conference on Machine Learning, pp. 1049–1056. ACM (2006)
9. Li, L., Fei-Fei, L.: Optimol: automatic online picture collection via incremental model learning. International Journal of Computer Vision 88(2), 147–168 (2010)
10. Dahl, D.: Sequentially-allocated merge-split sampler for conjugate and nonconjugate dirichlet process mixture models. Journal of Computational and Graphical-Statistics (2005)

Adaptive Temporal Entity Resolution on Dynamic Databases*

Peter Christen[1] and Ross W. Gayler[2]

[1] Research School of Computer Science, The Australian National University,
Canberra ACT 0200, Australia
peter.christen@anu.edu.au

[2] Veda, Melbourne VIC 3000, Australia
ross.gayler@veda.com.au

Abstract. Entity resolution is the process of matching records that refer to the same entities from one or several databases in situations where the records to be matched do not include unique entity identifiers. Matching therefore has to rely upon partially identifying information, such as names and addresses. Traditionally, entity resolution has been applied in batch-mode and on static databases. However, increasingly organisations are challenged by the task of having a stream of query records that need to be matched to a database of known entities. As these query records are matched, they are inserted into the database as either representing a new entity, or as the latest embodiment of an existing entity. We investigate how temporal and dynamic aspects, such as time differences between query and database records and changes in database content, affect matching quality. We propose an approach that adaptively adjusts similarities between records depending upon the values of the records' attributes and the time differences between records. We evaluate our approach on synthetic data and a large real US voter database, with results showing that our approach can outperform static matching approaches.

Keywords: Data matching, record linkage, dynamic data, real-time matching.

1 Introduction

Entity resolution is the process of identifying, matching, and merging records that correspond to the same entities from several databases [1]. The entities under consideration commonly refer to people, such as patients or customers. The databases to be matched often do not include unique entity identifiers. Therefore, the matching has to be based on the available attributes, such as names, addresses, and dates of birth. Entity resolution is employed in many domains, the most prominent being health, census statistics, national security, digital libraries, and deduplication of mailing lists [2–4].

* This research was funded by the Australian Research Council (ARC), Veda Advantage, and Funnelback Pty. Ltd., under Linkage Project LP100200079.

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 558–569, 2013.

RecID	EntID	Given name	Surname	Street address	City	Postcode	Time-stamp
$r1$	$e1$	Gale	Miller	13 Main Road	Sydney	2000	2006-01-21
$r2$	$e2$	Peter	O'Brian	43/1 Miller Street	Sydney	2010	2006-02-21
$r3$	$e1$	Gail	Miller	11 Town Place	Melbourne	3003	2007-01-28
$r4$	$e1$	Gail	Smith	42 Ocean Drive	Perth	6010	2007-07-12
$r5$	$e2$	Pete	O'Brien	43 Miller Street	Sydney	2610	2008-01-11
$r6$	$e1$	Abigail	Smith	42 Ocean Drive	Perth	6010	2008-06-30
$r7$	$e2$	Peter	OBrian	12 Nice Terrace	Brisbane	7011	2009-01-01
$r8$	$e1$	Gayle	Smith	11a Town Place	Sydney	2022	2009-04-29

Fig. 1. Example data set of eight records representing two entities

While entity resolution has traditionally been applied in batch mode and on static databases, an increasingly common scenario in many application domains is the model of data streams [5], where query records (potentially from several sources) need to be matched with (and potentially inserted into) a database that contains records of known entities.

An example application of entity resolution in such a dynamic environment is the verification of identifying information provided by customers at the time of a credit card or loan application. In many countries, the financial institutions where customers apply will send the customers' identifying information to a central credit bureau. The responsibility of this bureau is to match the query to a pre-existing credit file to retrieve that customer's credit history, and to determine that the customer is the person they claim to be [6]. The credit bureau receives a stream of query records that contain identifying information about people, and the bureau's task is to match these records (in real-time) to a large database of known validated entities. In this application, accurate entity resolution is crucial to avoid inappropriate lending and prevent credit application fraud [6].

Temporal and dynamic aspects in the context of entity resolution have only been investigated in the past three years [7–11]. Most similar to our work is the technique proposed by Li et al. [8, 9] on linking temporal records. Their approach assumes that all records in a database have a time-stamp attached, as illustrated in Fig. 1. These time-stamps are used to calculate *decay* probabilities for combinations of individual attributes and time differences. Based on these probabilities the similarities between records (calculated using approximate string comparison functions on individual attributes [2]) are adjusted. The *agreement decay* was defined by Li et al. [8] as the probability that two different entities, represented by two records with a time difference of Δt, have the same value in an attribute. The *disagreement decay* was defined as the probability that the value in an attribute for a given entity changes over time Δt. Such a change occurs if, for example, a person moves to a new address.

In Fig. 1, for the 'City' attribute and $\Delta t \in [0, 1)$ year, there are three pairs of records by the same entity where the attribute value has changed: *(r3,r4)* and *(r6,r8)* for *e1*, and *(r5,r7)* for *e2*, and one pair where the value did not change: *(r4,r6)*. The disagreement probability for this attribute and $\Delta t \in [0, 1)$ year is therefore 75%. As both entities live in 'Sydney' in 2006, we can calculate an agreement probability for this value. However, due to the sparseness of this data set, we cannot calculate agreement probabilities for other 'City' values.

Li et al. [8] calculated both agreement and disagreement probabilities such that they increase monotonically as Δt gets larger. However, as Fig. 2 shows, this is not necessarily the case. Their approach also assumes that the databases from which records are matched are static; that the decay probabilities are learned once in an off-line process using a supervised learning technique (learning disagreement probabilities is of linear complexity in the number of entities, while learning agreement probabilities is of quadratic complexity); and that these decay probabilities are independent of the frequencies of individual attribute values. The experiments conducted on a bibliographic data set showed that taking such temporal information into account can improve matching quality [8].

In contrast, our approach is aimed at facilitating an efficient adaptive calculation of weights that are used to adjust similarities between records. While our approach to calculate disagreement probabilities is similar to Li et al. [8], we calculate agreement probabilities that incorporate the frequency distributions of the attribute values. This is similar to frequency-based weight adjustments as applied in traditional probabilistic record linkage [4]. As an example, if two records have an agreeing surname value ‘Smith’, which is common in many English speaking countries, then it is more likely that they correspond to two different entities compared to two records that have the uncommon surname value ‘Dijkstra’. As our experimental study on a large real-world database shows, taking these frequencies into account can lead to improved matching accuracy.

Our contributions are (1) an adaptive matching approach for dynamic databases that contain temporal information; (2) an efficient temporal adjustment method that takes the frequencies of attribute values into account; and (3) an evaluation of our approach on both synthetic data (with well controlled characteristics) and a large real voter database from North Carolina in the USA.

2 Related Work

Most research in entity resolution over the past decade has focused on improving quality and scalability when matching databases. Several recent surveys provide overviews of the research field [2–4]. Besides the work by Li et al. [8, 9] on linking temporal records (described above), several other recent approaches have investigated temporal or dynamic aspects in entity resolution.

Whang et al. [10] developed an approach to matching databases where matching rules can evolve over time, and where a complete re-run of an entity resolution process is not required when new data become available. The assumption is that a user provides an initial set of matching rules and over time refines these rules. While the rules in this approach are evolving, no temporal information in the records that are matched is taken into account in the matching process.

Ioannou et al. [7] proposed an entity query system based on probabilistic entity databases, where records and their attributes are assigned probabilities of uncertainty that are incorporated into the matching process. These probabilities correspond to the confidence one has that an attribute value has been recorded correctly for an entity. During the matching process a dynamic index data structure is maintained which contains sub-sets of entities that are connected through common attribute values. Related to this work, Christen et al. [12, 13] investigated techniques to facilitate the real-time matching of query records to a large static database by incorporating the similarities calculated between attribute values into a novel index data structure. Both of these approaches however do not consider temporal aspects of the databases that are matched.

Yakout et al. [11] developed a matching approach for transactional records that correspond to the behaviour of entities (such as shopping baskets of individuals) over time, rather than the actual entity records. Their approach converts the transactions of an entity into a behaviour matrix. Entities are then matched by calculating similarities between condensed representations of their behaviour matrices. While temporal information is used to generate sequences of transactions for an entity, this information is not used in the matching process.

Pal et al. [14] recently presented an approach to integrate Web data from different sources where temporal information is unreliable and unpredictable (such as updates of changes are missing or incomplete). The temporal aspects of updates of entities are modelled with a hidden semi-Markov process. Results on a diverse set of data, from Twitter to climate data, showed encouraging results. The focus of this work is however to compute the correct value of an entity's attributes, not to identify and match records that refer to the same entity.

A large body of work has been conducted in the areas of stream data mining [5], where temporal decays are commonly used to give higher weights to more recent data, and temporal data mining [15], where the aim is to discover patterns over time in temporal data. To summarise, while either temporal information or dynamic data have been considered in recent approaches to entity resolution, our approach is a first to consider both, with the aim to achieve an entity resolution approach that adapts itself to changing data characteristics.

3 Modelling Temporal Changes of Entities

We assume a stream of query records q, which are to be matched (as detailed in Sect. 4) to a database $\mathbf{R}$ that contains records about known entities. We denote records in $\mathbf{R}$ with r_i, $1 \leq i \leq N$, with N being the number of records in $\mathbf{R}$. At any point in time N is fixed, but over time new records (such as matched query records) are added to $\mathbf{R}$ while the existing records are left unchanged. The records in $\mathbf{R}$ consist of attributes, $r_i.a_j$, $1 \leq j \leq M$, with M the number of attributes. Examples of such attributes include name and address details, phone numbers, or dates of birth. The records also contain a time-stamp $r_i.t$, and an entity identifier, $r_i.e$. All records that correspond to the same entity are assumed to have the same unique value in $r_i.e$. In a supervised setting, the $r_i.e$ are the true known entity identifiers, while in an unsupervised setting the values of $r_i.e$ are based on the match classification, as will be described further in Sect. 4.

Assume we want to calculate the similarity between query record q and record r_i in $\mathbf{R}$. These two records have a difference in their time-stamps of $\Delta t = |q.t - r_i.t|$. For a single attribute a_j, we classify the attribute to be *agreeing* if both records have the same attribute value ($q.a_j = r_i.a_j$), or if the two attribute

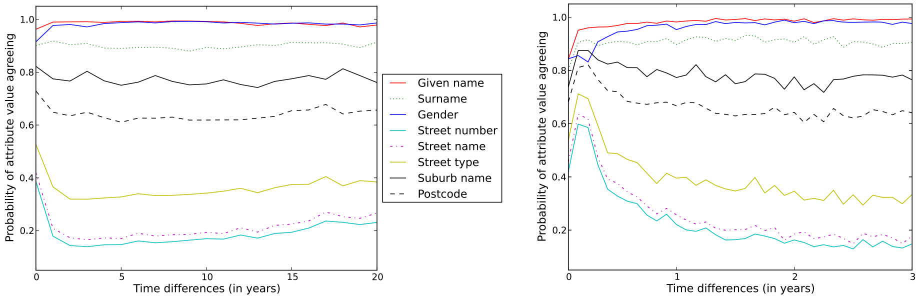


Fig. 2. The probabilities of an entity not changing its attribute values over time (i.e. the attribute values are agreeing), $P(A_j, \Delta t|S)$, as calculated from over 2.4 million entities in a North Carolina (NC) voter database. The x-axes show the time-differences Δt between two records that refer to the same entity for up-to 20 years, with the right-hand side plot zooming into the time differences up-to three years. As can be seen, address values are more likely to change over time than name values.

values are highly similar, i.e. $sim_j(q.a_j, r_i.a_j) \geq s_{same}$, with $sim_j(\cdot,\cdot)$ being the similarity function used to compare values for attribute a_j, and s_{same} a minimum similarity threshold. If, on the other hand, $sim_j(q.a_j, r_i.a_j) < s_{same}$, then we classify the attribute to be *disagreeing*. Similarity values are assumed to be normalised, with $sim_j(\cdot,\cdot) = 0$ for two attribute values that are completely different, and $sim_j(\cdot,\cdot) = 1$ for two values that are the same. Similarity functions vary by attribute and may be domain specific. For string attributes, such as names, approximate string similarity functions are commonly used [2].

To take temporal aspects into account, we need to consider the following events. We denote the event that q and r_i actually refer to the same entity with S, and the event that they actually refer to two different entities with $\neg S$. Furthermore, we denote the event that q and r_i have an agreeing value in attribute a_j with A_j, and the event that they have a disagreeing value in attribute a_j with $\neg A_j$.

We now consider the following two probabilities. As discussed in Sect. 3.1 below, they are used to adjust the similarities $sim_j(\cdot,\cdot)$ according to the time difference Δt between records q and r_i.

$$P(A_j, \Delta t|S) = P(sim_j(q.a_j, r_i.a_j) \geq s_{same} \wedge |q.t - r_i.t| = \Delta t \mid q.e = r_i.e) \quad (1)$$

is the probability that a query and a database record that actually refer to the same entity have an agreeing value in attribute a_j over Δt (i.e. the value does not change). It holds that $P(A_j, \Delta t|S) = 1 - P(\neg A_j, \Delta t|S)$.

$$P(\neg A_j, \Delta t|\neg S) = P(sim_j(q.a_j, r_i.a_j) < s_{same} \wedge |q.t - r_i.t| = \Delta t \mid q.e \neq r_i.e) \quad (2)$$

is the probability that a query and a database record that actually refer to two different entities have disagreeing (i.e. different) values in attribute a_j over Δt. Clearly, $P(\neg A_j, \Delta t|\neg S) = 1 - P(A_j, \Delta t|\neg S)$.

The probability $P(A_j, \Delta t|S)$ can be learned for individual attributes and different Δt by counting the number of agreeing and disagreeing attribute values for pairs of records of the same entity that have a time difference of Δt.

Calculating the probability $P(\neg A_j, \Delta t|\neg S)$ is more difficult because it requires the comparison of attribute values across records that have a time difference of Δt and where the records refer to different entities. Such an approach, which is of quadratic complexity in the number of entities, was employed by Li et at. [8, 9]. Our approach, described in Sect. 3.2, is aimed at efficiently calculating the two probabilities (1) and (2) in an adaptive fashion. First we present how these probabilities are used to adjust the similarities between records.

3.1 Adjusting Similarities between Records

Assume the attributes of a query record q and a database record r_i have been compared using a set of similarity functions $s_j = sim_j(q.a_j, r_i.a_j)$, $1 \leq j \leq M$, such as approximate string comparison functions [2], with s_j the similarity value calculated for attribute a_j, and M the number of compared attributes.

Without taking temporal effects into account, we use s_j as an estimate of the probability that two attribute values are the same, and $(1-s_j)$ as the probability they are different [8] (remember we assume $0 \leq s_j \leq 1$). In a temporal setting, we aim to assign weights to individual attributes that indicate their importance according to the likelihood of a change of value in this attribute, as discussed above. Following Li et al. [8], we use (3) to adjust and normalise similarities.

$$sim(q, r_i) = \frac{\sum_j^M w_j(s_j, \Delta t) \cdot s_j}{\sum_j^M w_j(s_j, \Delta t)}, \tag{3}$$

where $s_j = sim_j(q.a_j, r_i.a_j)$ and $\Delta t = |q.t - r_i.t|$. This equation is a heuristic that attempts to adjust the similarity in a qualitatively reasonable manner. We calculate the weights $w_j(s_j, \Delta t)$ using the minimum similarity threshold s_{same}:

- If $s_j \geq s_{same}$ we set $w_j(s_j, \Delta t) = s_j \cdot (1 - P(A_j, \Delta t|\neg S)) = s_j \cdot P(\neg A_j, \Delta t|\neg S)$. This means that the more likely it is that two entities have the same value in attribute a_j over time difference Δt, the less weight should be given to the similarity value s_j of this agreement.
- If $s_j < s_{same}$ we set $w_j(s_j, \Delta t) = s_j \cdot (1 - P(\neg A_j, \Delta t|S)) = s_j \cdot P(A_j, \Delta t|S)$. This means that the more likely it is that a value in attribute a_j changes over time difference Δt, the less weight should be given to this disagreement.

For example, as can be seen from Fig. 2, almost nobody in the NC voter database has changed their given name even over long periods of time, compared to changes in address attributes (such as street, suburb and postcode). Therefore, agreeing given name values are a strong indicator in this database that two records refer to the same entity. On the other hand, a low similarity in an address attribute is a weak indicator that two records refer to different entities.

During the matching process a query record is compared with one or more database records, and the resulting adjusted similarities $sim(q, r_i)$ as calculated with (3) are ranked. Details of this matching process are presented in Sect. 4.

3.2 Learning Agreement and Disagreement Probabilities

In a dynamic setting, the aim is to efficiently learn the probabilities given in (1) and (2) in an adaptive way. The agreement probability $P(A_j, \Delta t|S)$ can be learned from data if it is known which records correspond to the same entity. To facilitate an efficient calculation of this probability, we discretise the time differences Δt into equal sized intervals Δt_k, of durations such as weeks, months, or years (depending on the overall expected time span in an application). We keep two arrays for each attribute a_j, $\mathbf{A}_j[\Delta t_k]$ and $\mathbf{D}_j[\Delta t_k]$. The first array, $\mathbf{A}_j[\Delta t_k]$, keeps track of the number of times a value in attribute a_j is agreeing for two records of the same entity that have a discretised time difference of Δt_k, while $\mathbf{D}_j[\Delta t_k]$ keeps track of the number of times the values in a_j are disagreeing for two records of the same entity. Using these two counters, we calculate:

$$P(A_j, \Delta t_k|S) = \frac{\mathbf{A}_j[\Delta t_k]}{(\mathbf{A}_j[\Delta t_k] + \mathbf{D}_j[\Delta t_k])} \tag{4}$$

for $\Delta t_1, \ldots, \Delta t_{max}$, with Δt_{max} the maximum discretised time difference encountered between two records of the same entity in $\mathbf{R}$. The update of the counters $\mathbf{A}_j[\Delta t_k]$ and $\mathbf{D}_j[\Delta t_k]$, and calculating (4) for a certain discretised Δt_k, requires a single increase of a counter and one division for each query record that is matched to a database record, making this a very efficient approach.

It is possible that no pair of records of the same entity with a certain discretised time difference of Δt_k does occur in a data set, and therefore (4) cannot be calculated for this Δt_k. To overcome this data sparseness issue, we also calculate the average value for $P(A_j, \Delta t|S)$ over all Δt_k for each attribute a_j, and use this average value in case $P(A_j, \Delta t_k|S)$ is not available for a certain Δt_k.

To calculate $P(\neg A_j, \Delta t|\neg S)$, we use the probability that two records that refer to two different entities have the same value v in attribute a_j, which equals to $P(A_j, \Delta t|\neg S) = 1 - P(\neg A_j, \Delta t|\neg S)$. This probability depends upon how frequently a certain value v occurs in an attribute. The probability $P_j(v)$ that an entity has the value v in attribute a_j can be estimated from the count of how many records in $\mathbf{R}$ have this value: $P_j(v) = \frac{|r_i \in \mathbf{R}, r_i.a_j = v|}{|\mathbf{R}|}$. For example, only a few records in $\mathbf{R}$ might have the rare surname 'Dijkstra', and so the probability that two records from different entities both have this value is very low.

We keep an array $\mathbf{V}_j$ for each attribute a_j with a counter for each distinct value v of how many records in $\mathbf{R}$ have this value in a_j. We then calculate $P(\neg A_j, \Delta t|\neg S) = 1 - P_j(v)$ for all attributes values v. If a value in a query record q in attribute $q.a_j$ has not previously occurred in $\mathbf{R}$ (and thus $P_j(q.a_j) = 0$), we set $P(\neg A_j, \Delta t|\neg S) = 1.0$ for this value. Updating the counters $\mathbf{V}_j$ and probabilities $P_j(v)$ requires one integer increment and one division per attribute. For a single query record, the calculations of both agreement and disagreement probabilities are thus of complexity $O(1)$, making this approach very efficient.

The database record to which a query record is matched determines the calculation of $P(A_j, \Delta t_k|S)$. In a supervised setting, the true match status of (a subset of) record pairs is known, allowing the calculation of this probability based on these training pairs. For a query record q, if there is a previous record r_i of the

Algorithm 1. *Adaptive Temporal Matching Process*

```
Input:
- Initial database with known entity records: $\mathbf{R} = \mathbf{R}_{init}$
- Arrays with counters for agreements, disagreements, and values: $\mathbf{A}_j$, $\mathbf{D}_j$, and $\mathbf{V}_j$
- Attribute similarity functions: $sim_j(\cdot,\cdot)$; and thresholds: $s_{same}$ and $s_{match}$
- Stream of query records: $q$
Output:
- Identifiers of matched entities: $q.e$

1:  Set $count_{ent}$ to number of entities in $\mathbf{R}$
2:  while $q$ do:
3:      Get candidate records $\mathbf{C}$ from $\mathbf{R}$: $\mathbf{C} = get\_cand\_records(q, \mathbf{R})$
4:      for $c \in \mathbf{C}$ do:
5:          $\Delta t = |q.t - c.t|$; $s_j = sim_j(q.a_j, c.a_j), 1 \leq j \leq M$
6:          Calculate $sim(q, c)$ using Equation (3)
7:      $c_{best} = max(c_i : sim(q, c_i) > sim(q, c_k), c_i \in \mathbf{C}, c_k \in \mathbf{C}, c_i \neq c_k)$
8:      if $sim(q, c_{best}) \geq s_{match}$ then $q.e = c_{best}.e$
9:      else $q.e = count_{ent}$; $count_{ent} = count_{ent} + 1$
10:     Insert $q$ into $\mathbf{R}$: $insert(q, \mathbf{R})$
11:     Update $\mathbf{A}_j$, $\mathbf{D}_j$, and $\mathbf{V}_j$, and $P(A_j, \Delta t_k|S)$ and $P_j(v)$, $1 \leq j \leq M$.
12:     Pass $q.e$ to application
```

same entity in $\mathbf{R}$, we calculate $\Delta t = |q.t - r_i.t|$ and which attributes agree or disagree for these two records using $sim_j(q.a_j, r_i.a_j)$ and s_{same}, and we update the appropriate counters in $\mathbf{A}_j$ and $\mathbf{D}_j$ and the corresponding probabilities.

If no training data are available, then for a query record the best matching database record, i.e. the record with the highest adjusted similarity according to (3), is assumed to be the true match. We can then calculate the time differences and update the relevant counters and probabilities in the same way as in a supervised setting. Due to limited space, we only consider the supervised case.

4 Adaptive Temporal Matching

Algorithm 1 provides an overview of the main steps of our matching process. The required input is a database $\mathbf{R}_{init}$ of known entities. In a supervised setting, where (some of) the true entities are known, the entity identifiers $r_i.e$ in $\mathbf{R}_{init}$ allow us to generate for an entity a chain of records sorted according to the timestamps $r_i.t$. Using these entity chains, the counters $\mathbf{A}_j$ and $\mathbf{D}_j$ are populated, and the initial values for the $P(A_j, \Delta t_k|S)$, $1 \leq j \leq M$ are calculated. In an unsupervised setting, no such initial database is available, and thus $\mathbf{R} = \emptyset$.

The other input parameters required are a set of similarity functions $sim_j(\cdot,\cdot)$, one per attribute a_j used, and the two thresholds s_{same} and s_{match}. The former is used to decide if two attribute values are agreeing or not (as described in Sect. 3), while the latter is use to decide if a query record is classified as a match with a database record or not (lines 8 and 9 in Algo. 1).

The algorithm loops as long as query records q are to be matched. We assume the case where query records are inserted into the database once they are matched (line 10). Removing this line will mean that $\mathbf{R}$ and the counters in $\mathbf{A}_j$, $\mathbf{D}_j$ and $\mathbf{V}_j$ are not updated, and therefore our approach becomes non-adaptive (while still taking temporal aspects into account).

In line 3, a set of candidate records $\mathbf{C}$ is retrieved from $\mathbf{R}$ using an index technique for entity resolution. For example, $\mathbf{C}$ might be all records from $\mathbf{R}$ that have the same postcode value as q. In our implementation, we employ a similarity-aware index technique that has shown to be efficient for real-time matching [13]. For each candidate record $c \in \mathbf{C}$, its time difference and similarities with q are calculated in line 5 and adjusted in line 6, as described in Sect. 3.1.

The candidate record with the highest adjusted similarity, c_{best}, is identified in line 7 by finding the candidate record that has the maximum similarity with the query record q. If this similarity is above the minimum match similarity s_{match}, then q is assigned the entity identifier of this best match (line 8). Otherwise q is classified as a new entity and given a new unique entity identifier number in line 9. In line 10, the query record q is inserted into the database $\mathbf{R}$, and in line 11 the counters in $\mathbf{A}_j[\Delta t]$, $\mathbf{D}_j[\Delta t]$, and $\mathbf{V}_j$, as well as the relevant probabilities, are updated for all attributes a_j. Finally, the entity identifier of q is passed on to the application that requires this information, allowing the application to extract all records from $\mathbf{R}$ that refer to this entity.

5 Experiments and Discussion

We evaluate our adaptive temporal entity resolution approach on both synthetic and real data. Synthetic data allows us to control the characteristics of the data, such as the number of records per entity, and the number of modifications introduced [16]. We generated six data sets, each containing 100,000 records, with an average of 4, 8, or 16 records per entity, and either having a single modification per record (named 'Clean' data) or eight modifications per record (named 'Dirty' data). Modifications introduced were both completely changed attribute values, as well as single character edits (like inserts, deletes and substitutions).

As real data we used the North Carolina (NC) voter registration database [17], which we downloaded every two months since October 2011 to build a compound temporal data set. This data set contains the names, addresses, and ages of more than 2.4 million voters, as well as their unique voter registration numbers. Each record has a time-stamp attached which corresponds to the date a voter originally registered, or when any of their details were changed. This data set therefore contains realistic temporal information about a large number of people. There are 111,354 individuals with two records, 2408 with three, and 39 with four records in this data set. An exploration of this data set has shown that many of the changes in the given name attribute are corrections of nicknames and small typographical mistakes, while changes in surnames and address attributes are mostly genuine changes that occur when people get married or move address.

We sorted all data sets according to their time-stamps, and split each into a training and test set such that around 90% of the second and following records of an entity (of those that had more than one record) were included in the training set. We compare our proposed adaptive temporal matching technique with a static matching approach that does not take temporal information into account, and with a simple temporal matching approach where an additional temporal similarity value is calculated for a record pair as $sim_t(q.t, r_i.t) = \frac{|q.t - r_i.t|}{\Delta T_{max}}$, with ΔT_{max} being the difference between the time stamps of the earliest and latest

Table 1. Matching results for synthetic data sets shown as percentage of true matches correctly identified. The parameter pair given refers to s_{same} / s_{match}.

Parameters	None			Temp Attr			Adapt			Non Adapt		
Avrg rec per ent	4	8	16	4	8	16	4	8	16	4	8	16
Clean, 0.8 / 0.7	92.3	90.5	87.2	91.9	90.6	87.4	92.3	90.8	87.3	92.3	91.1	87.3
Clean, 0.9 / 0.8	63.8	57.9	50.7	63.2	57.9	50.7	64.2	59.1	50.9	63.9	59.6	50.9
Dirty, 0.8 / 0.7	47.1	35.1	25.5	46.5	38.6	29.9	53.5	40.5	29.2	53.6	40.5	29.2
Dirty, 0.9 / 0.8	16.7	10.3	5.9	14.7	10.5	6.6	21.9	13.5	7.4	21.9	13.5	7.4

record in **R**. Note that we cannot compare our approach to the temporal linkage method proposed by Li et al. [8, 9], because their method is only applicable on static databases, as was described in Sect. 1. We label the non-temporal matching approach with 'None'; the above described temporal attribute approach with 'Temp Attr'; our proposed adaptive approach described in Sect. 4 as 'Adapt'; and with 'Non Adapt' a variation of this adaptive approach that calculates the probabilities $P(A_j, \Delta t_k|S)$ and $P_j(v)$ only once at the end of the training phase, but that does not adjust these probabilities for each of the following query records (i.e. line 11 in Algo. 1 is not executed for query records in the test set).

In Table 1 and Fig. 3 we report matching accuracy as the percentage of true matches correctly identified, i.e. if a candidate record c_{best} in line 7 in Algo. 1 was a record of the same entity as the entity of query record q, $c_{best}.e = q.e$. For the NC voter data set we additionally report the percentage of true matches that occurred in the ten top-ranked candidate records. In a real-time query environment, returning the ten top-ranked results is a realistic assumption.

We implemented our approach using Python 2.7.3, and ran experiments on a server with 128 GBytes of memory and two 6-core CPUs running at 2.4 GHz. We used the Jaro-Winkler string comparison function [2] as $sim_j(\cdot, \cdot)$ to compare name and address attribute values. We ran four sets of experiments with parameter settings: $s_{same} = \{0.8, 0.9\}$ and $s_{match} = \{0.7, 0.8\}$. To facilitate repeatability, the programs and synthetic data sets are available from the authors.

The results shown in Table 1 for the synthetic data sets indicate that all four matching approaches are sensitive to the choice of parameter settings, and if the data are clean or dirty. With data that contain a larger number of variations, identifying true matches becomes much more difficult, as would be expected. Matching also becomes harder with more records per entity, because more attributes will have their values changed over time. The proposed approach to adjust the similarities between records is able to improve matching quality most when data are dirty. Such data are likely to occur in dynamic entity resolution applications, where it is not feasible to apply expensive data cleaning operations on query records prior to matching them to a database of entity records.

The experiments conducted on the NC voter database show quite different results (Fig. 3), with the proposed similarity adjustment approach outperforming the two baseline approaches. Taking top ten matches into account, our approach, which adjusts the similarities between records, is able to identify nearly twice as many true matches compared to the two baseline approaches. However, the

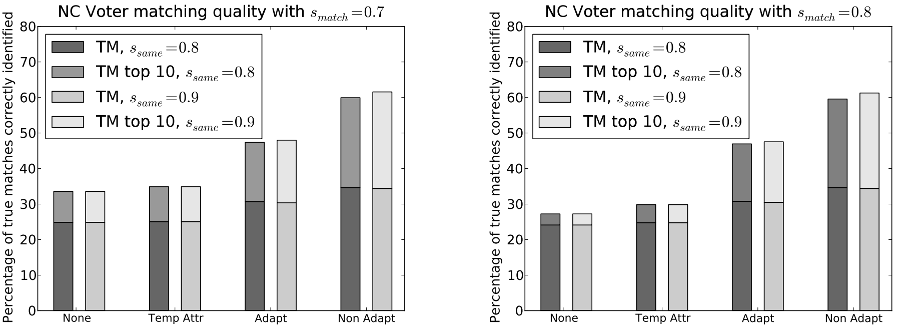


Fig. 3. Matching results for the NC voter database shown as percentage of true matches (TM) correctly identified. Different from the results in Table 1, these results are robust with regard to parameter settings, and they are also significantly better for the proposed adaptive approach compared to the baseline approaches 'None' and 'Temp Attr'.

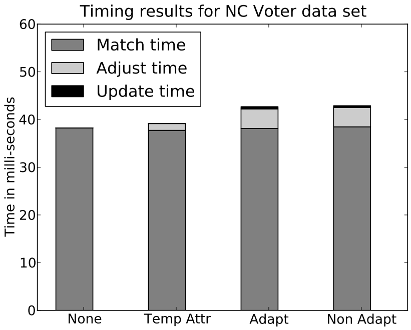


Fig. 4. Average time for matching a single query record to the North Carolina voter database. The adjustment of similarities using (3) adds only around 10% extra time.

adaptive approach does not perform as well as the non-adaptive approach. This is likely because the number of true matches compared to all query records is very small, which distorts the calculation of the probabilities in (4).

As Fig. 4 illustrates, our approach is very efficient, even on the large real database with over 2.4 million records. The time needed to adjust the similarity values between records, as done in (3), only adds around 10% to the total time required to match a query record, while the time needed to update the counters $\mathbf{A}_j$, $\mathbf{D}_j$, and $\mathbf{V}_j$, and $P(A_j, \Delta t_k|S)$, is negligible. This makes our approach applicable to adaptive entity resolution on dynamic databases.

6 Conclusions and Future Work

We have presented an approach to facilitate efficient adaptive entity resolution on dynamic databases by adaptively calculating temporal agreement and disagreement probabilities that are used to adjust the similarities between records. As shown through experiments on both synthetic and real data, our temporal matching approach can lead to significantly improved matching quality.

Our plans for future work include to conduct experiments for unsupervised settings as discussed in Sect. 3.2 and run experiments on other data sets, to conduct an in-depth analysis of our proposed algorithm, and to extend our approach to account for attribute and value dependencies. For example, when a person moves, most of their address related attribute values change, and the probability that a person moves also depends upon their age (young people are known to change their address more often than older people). We also plan to integrate our approach with traditional probabilistic record linkage [4], and we will investigate how to incorporate constraints, such as only a few people can live at a certain address at any one time.

References

1. Winkler, W.E.: Methods for evaluating and creating data quality. Elsevier Information Systems 29(7), 531–550 (2004)
2. Christen, P.: Data Matching. In: Data-Centric Systems and Appl., Springer (2012)
3. Elmagarmid, A., Ipeirotis, P., Verykios, V.: Duplicate record detection: A survey. IEEE Transactions on Knowledge and Data Engineering 19(1), 1–16 (2007)
4. Herzog, T., Scheuren, F., Winkler, W.: Data quality and record linkage techniques. Springer (2007)
5. Aggarwal, C.: Data Streams: Models and Algorithms. Database Management and Information Retrieval, vol. 31. Springer (2007)
6. Anderson, K., Durbin, E., Salinger, M.: Identity theft. Journal of Economic Perspectives 22(2), 171–192 (2008)
7. Ioannou, E., Nejdl, W., Niederée, C., Velegrakis, Y.: On-the-fly entity-aware query processing in the presence of linkage. VLDB Endowment 3(1) (2010)
8. Li, P., Dong, X., Maurino, A., Srivastava, D.: Linking temporal records. Proceedings of the VLDB Endowment 4(11) (2011)
9. Li, P., Tziviskou, C., Wang, H., Dong, X., Liu, X., Maurino, A., Srivastava, D.: Chronos: Facilitating history discovery by linking temporal records. VLDB Endowment 5(12) (2012)
10. Whang, S., Garcia-Molina, H.: Entity resolution with evolving rules. VLDB Endowment 3(1-2), 1326–1337 (2010)
11. Yakout, M., Elmagarmid, A., Elmeleegy, H., Ouzzani, M., Qi, A.: Behavior based record linkage. VLDB Endowment 3(1-2), 439–448 (2010)
12. Christen, P., Gayler, R.: Towards scalable real-time entity resolution using a similarity-aware inverted index approach. In: AusDM 2008, Glenelg, Australia (2008)
13. Christen, P., Gayler, R., Hawking, D.: Similarity-aware indexing for real-time entity resolution. In: ACM CIKM 2009, Hong Kong, pp. 1565–1568 (2009)
14. Pal, A., Rastogi, V., Machanavajjhala, A., Bohannon, P.: Information integration over time in unreliable and uncertain environments. In: WWW, Lyon (2012)
15. Laxman, S., Sastry, P.: A survey of temporal data mining. Sadhana 31(2) (2006)
16. Christen, P., Pudjijono, A.: Accurate synthetic generation of realistic personal information. In: Theeramunkong, T., Kijsirikul, B., Cercone, N., Ho, T.-B. (eds.) PAKDD 2009. LNCS (LNAI), vol. 5476, pp. 507–514. Springer, Heidelberg (2009)
17. North Carolina State Board of Elections: NC voter registration database, `ftp://www.app.sboe.state.nc.us/` (last accessed September 11, 2012)

Fuzzy Multi-Sphere Support Vector Data Description

Trung Le[1], Dat Tran[2], and Wanli Ma[2]

[1] HCM City University of Education, Vietnam
[2] University of Canberra, ACT 2601, Australia
dat.tran@canberra.edu.au

Abstract. Current well-known data description methods such as Support Vector Data Description and Small Sphere Large Margin are conducted with assumption that data samples of a class in feature space are drawn from a single distribution. Based on this assumption, a single hypersphere is constructed to provide a good data description for the data. However, real-world data samples may be drawn from some distinctive distributions and hence it does not guarantee that a single hypersphere can offer the best data description. In this paper, we introduce a Fuzzy Multi-sphere Support Vector Data Description approach to address this issue. We propose to use a set of hyperspheres to provide a better data description for a given data set. Calculations for determining optimal hyperspheres and experimental results for applying this proposed approach to classification problems are presented.

Keywords: Kernel Methods, Fuzzy Interference, Support Vector Data Description, Multi-Sphere Support Vector Data Description.

1 Introduction

Support Vector Machine (SVM) [2], [4] has been proven a very effective method for binary classification. However, it cannot render good performance for one-class classification problems where one of two classes is under-sampled, or only data samples of one class are available for training [9] . One-class classification involves learning data description of normal class to build a model that can detect any divergence from normality [11]. The samples of abnormal class if existed contribute to refining the data description. Support Vector Data Description (SVDD) was introduced in [14], [13] as a kernel method for one-class classification. SVDD aims at constructing an optimal hypersphere in feature space which includes only normal samples and excludes all abnormal samples with tolerances. This optimal hypersphere is regarded as a data description since when mapped back to input space it becomes a set of contours that tightly enclose the normal data samples [1].

Variations of SVDD were proposed to enhance this approach. In [9], density-induced information was incorporated to the samples so that the dependency of data description on support vectors can be less imposed when these support

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 570–581, 2013.

vectors cannot characterise well the data. To reduce the impact of less important dimensions, a single ellipse was learnt rather than a single hypersphere [10]. However, this work was not general since it was only proposed for the model in input space. Other approaches introduced better margins for SVDD such as [15] [7]. To reduce the chance of acceptance of outliers, in [15] a small sphere with large margin was proposed. However, this can induce side-effect which causes the interference of decision boundary into normal data region. To overcome this issue, an optimal sphere with two adjustable margins for reducing both true positive ($TP\%$) and true negative ($TN\%$) error rates was proposed [7].

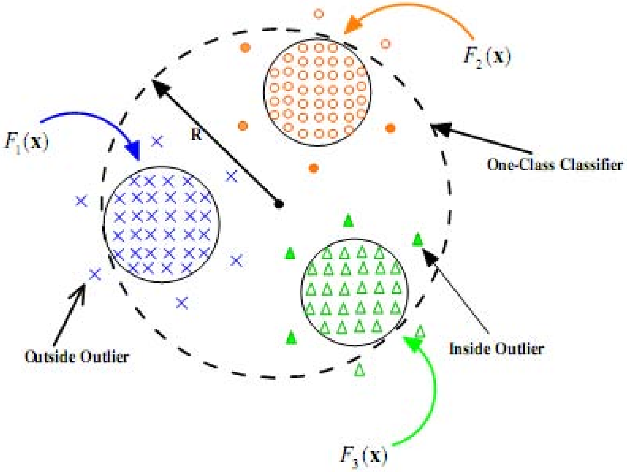


Fig. 1. Inside outliers would be improperly included if only one hypersphere is constructed [16]

SVDD assumes that all samples of the training set are drawn from a single uniform distribution [13]. However, this hypothesis is not always true since real-world data samples may be drawn from distinctive distributions [5]. Therefore, a single hypersphere cannot be a good data description. For example, in Figure 1, data samples are scattered over some distinctive distributions and one single hypersphere would improperly record the inside outliers. In [16], a multi-sphere approach to SVDD was proposed for multi-distribution data. The domain for each distribution was detected and for each domain an optimal sphere was constructed to describe the corresponding distribution. However, the learning process was heuristic and did not follow up learning with minimal volume principle [12]. In [6], a method was proposed to link the input space to the feature space. The dense regions (clusters) in the input space were identified and became a single sphere in the feature space. Again, this method was heuristic and did not abide by learning with minimum volume principle. To motivate learning with this principle, a hard multi-sphere support vector data description (HMS-SVDD) [8] was proposed. A set of hyperspheres was introduced to enclose all the data samples. A data sample will belong to only one hypersphere. This restriction should be relaxed to allow a data sample to belong to different hyperspheres if that sample has similar degrees of belonging to those hyperspheres. To address this issue, we propose fuzzy multi-sphere support vector data description (FMS-SVDD) in this paper. A fuzzy membership is assigned to each data sample to denote the degree of belonging of that sample to a hypersphere. We prove that classification error will be reduced after each iteration in the learning process. The set of hyperspheres will gradually converge to a stable configuration. To evaluate the proposed apprach, we performed classification experiments on 23 data sets in UCI repository. The experimental results showed that FMS-SVDD could provide better classification rates in comparison to other one-class kernel methods.

2 Fuzzy Multi-Sphere Support Vector Data Description (FMS-SVDD)

2.1 Problem Formulation

Let x_i, $i = 1, \ldots, p$ be normal data samples with label $y_i = +1$ and x_i, $i = p+1, \ldots, n$ be abnormal data samples with label $y_i = -1$. Consider a set of m hyperspheres $S_j(c_j, R_j)$ with center c_j and radius R_j, $j = 1, \ldots, m$. This hypershere set is a good data description of the data set $X = \{x_1, x_2, \ldots, x_n\}$ if each of the hyperspheres describes a distribution in this data set and the sum of all radii $\sum_{j=1}^{m} R_j^2$ should be minimised.

Let matrix $U = [u_{ij}]_{p \times m}$, $u_{ij} \in [0,1]$, $i = 1, \ldots, p$, $j = 1, \ldots, m$ where u_{ij} is the membership representing degree of belonging of sample x_i to hypersphere S_j. It is necessary to construct a set of hyperspheres so that these hyperspheres can include all normal data and exclude all abnormal data with tolerances. The optimisation problem of fuzzy multi-sphere SVDD can be formulated as follows

$$\min_{R,c,\xi,u} \left(\sum_{j=1}^{m} R_j^2 + \frac{1}{\nu_1 p} \sum_{i=1}^{p} \xi_i + \frac{1}{\nu_2 q} \sum_{i=p+1}^{n} \sum_{j=1}^{m} \xi_{ij} \right) \tag{1}$$

subject to

$$\begin{aligned} &\sum_{j=1}^{m} u_{ij}^d \|\phi(x_i) - c_j\|^2 \leq \sum_{j=1}^{m} u_{ij}^d R_j^2 + \xi_i,\ i = 1, \ldots, p \\ &\|\phi(x_i) - c_j\|^2 \geq R_j^2 - \xi_{ij},\ i = p+1, \ldots, n,\ j = 1, \ldots, m \\ &\sum_{j=1}^{m} u_{ij} = 1,\ i = 1, \ldots, p \end{aligned} \tag{2}$$

where $R = [R_j]_{j=1,\ldots,m}$ is vector of radii, ν_1 and ν_2 are constants, ξ_i and ξ_{ij} are slack variables, $c = [c_j]_{j=1,\ldots,m}$ is vector of centres, and $q = n - p$ is the number of abnormal (negative) samples.

In the FMS-SVDD model, we also introduce parameter $d > 1$ to adjust the relative ratios among the memberships u_{ij}, $i = 1, \ldots, p$, $j = 1, \ldots, m$.

Minimising the function in (1) over variables R, c and ξ subject to (2) will determine radii and centres of hyperspheres and slack variables if the matrix U is given. On the other hand, the matrix U will be determined if radii and centres of hyperspheres are given. Therefore an iterative algorithm will be applied to find a complete solution. The algorithm consists of two alternative steps: *1) Calculate radii and centres of hyperspheres and slack variables*, and *2) Calculate membership U*.

We present in the next sections the iterative algorithm and the proof of key theorem which states that the classification error in the current iteration will be smaller than that in the previous iteration. It means that the model is gradually refined and converged to a stable configuration.

For classifying a sample x, the following decision function is used:

$$f(x) = sign\Big(\max_{1 \le j \le m} \left\{ R_j^2 - ||\phi(x) - c_j||^2 \right\} \Big) \tag{3}$$

The unknown sample x is normal if $f(x) = +1$ or abnormal if $f(x) = -1$. This decision function implies that the mapping of a normal sample has to be in one of the hyperspheres and that the mapping of an abnormal sample has to be outside all of those hyperspheres.

The following theorem is used to consider the relation of slack variables to the classified samples:

Theorem 1. *Assume that (R, c, ξ_i, ξ_{ij}) is a solution of the optimisation problem (1), x_i, $i \in \{1, 2, \ldots, n\}$ is the i-th sample. The slack variable ξ_i or ξ_{ij} can be computed as*

$$\begin{aligned} \xi_i &= \max \left\{ 0, \sum_{j=1}^{m} u_{ij}^d \left(\|\phi\left(x_i\right) - c_j\|^2 - R_j^2 \right) \right\},\ i = 1, \ldots, p \\ \xi_{ij} &= \max \left\{ 0, R_j^2 - \|\phi(x_i) - c_j\|^2 \right\},\ i = p+1, \ldots, n,\ j = 1, \ldots, m \end{aligned} \tag{4}$$

Proof

For all i, from equation (2) we have

$$\begin{aligned} \xi_i &\ge \max \left\{ 0, \sum_{j=1}^{m} u_{ij}^d \left(\|\phi\left(x_i\right) - c_j\|^2 - R_j^2 \right) \right\},\ i = 1, \ldots, p \\ \xi_{ij} &\ge \max \left\{ 0, R_j^2 - \|\phi(x_i) - c_j\|^2 \right\}, \quad i = p+1, \ldots, n,\ j = 1, \ldots, m \end{aligned} \tag{5}$$

Moreover, (R, c, ξ) is minimal solution of (1). Hence, the *theorem 1* is proved.

It is natural to define *error*(i), i.e. the error at sample x_i, $1 \le i \le n$ as follows
If *x_i is normal data sample* **then**

$$error(i) = \begin{cases} 0 & if\, x_i\, is\, correctly\, classified \\ \min\limits_{1 \le j \le m} \left\{ \|\phi(x_i) - c_j\|^2 - R_j^2 \right\} & otherwise \end{cases} \tag{6}$$

Else

$$error(i) = \begin{cases} 0 & if\, x_i\, is\, correctly\, classified \\ \min\limits_{j \in J} \left\{ R_j^2 - \left\| \phi(x_i) - c_j \right\|^2 \right\} & otherwise \end{cases} \tag{7}$$

where $J = \{j : x_i \in S_j \; and \; 1 \leq j \leq m\}$.

We can prove that $\sum\limits_{i=1}^{p} \xi_i$ is an upper bound of $\frac{1}{m^{d-1}} \sum\limits_{i=1}^{p} error(i)$, and $\sum\limits_{i=p+1}^{n} \sum\limits_{j=1}^{m} \xi_{ij}$ is an upper bound of $\sum\limits_{i=p+1}^{n} error(i)$. The second inequality is trivial, therefore we primarily concentrate on the first one.

Theorem 2. $\sum\limits_{i=1}^{p} \xi_i$ *is an upper bound of* $\frac{1}{m^{d-1}} \sum\limits_{i=1}^{p} error(i)$*.*

Proof

Let us denote $d_{ij} = \left\| \phi(x_i) - c_j \right\|^2 - R_j^2, \; i = 1, \ldots, p, \; j = 1, \ldots, m$. Given $1 \leq i \leq p$, we will prove that $\xi_i \geq \frac{1}{m^{d-1}} error(i)$. This above inequality is trivial if x_i is correctly classified. We consider the case where x_i is misclassified, i.e., $d_{ij} > 0$ for all j. It means that $\xi_i = \sum\limits_{j=1}^{m} u_{ij}^d d_{ij}$. From definition of $error(i)$, we have: $\xi_i = \sum\limits_{j=1}^{m} u_{ij}^d d_{ij} \geq error(i) \sum\limits_{j=1}^{m} u_{ij}^d$. To fulfill this proof, we will show that $\sum\limits_{j=1}^{m} u_{ij}^d \geq \frac{1}{m^{d-1}} \left(\sum\limits_{j=1}^{m} u_{ij} \right)^d = \frac{1}{m^{d-1}}$. Indeed, this inequality can be rewritten as follows

$$\sum_{j=1}^{m} \left(\frac{m u_{ij}}{\sum\limits_{j'=1}^{m} u_{ij'}} \right)^d \geq m \tag{8}$$

By referring to Bernoulli inequality which says $(1 + x)^r \geq 1 + rx$ if $r \geq 1$ and $x > -1$, we have

$$\left(\frac{m u_{ij}}{\sum\limits_{j'=1}^{m} u_{ij'}} \right)^d = \left(1 - \left(1 - \frac{m u_{ij}}{\sum\limits_{j'=1}^{m} u_{ij'}} \right) \right)^d \geq 1 - d \left(1 - \frac{m u_{ij}}{\sum\limits_{j'=1}^{m} u_{ij'}} \right) \; for\; all\; j \tag{9}$$

It follows that

$$\sum_{j=1}^{m} \left(\frac{m u_{ij}}{\sum\limits_{j'=1}^{m} u_{ij'}} \right)^d \geq \sum_{j=1}^{m} \left(1 - d \left(1 - \frac{m u_{ij}}{\sum\limits_{j'=1}^{m} u_{ij'}} \right) \right) = m \tag{10}$$

2.2 Calculating Radii, Centres and Slack Variables

The Lagrange function for the optimisation problem (1) subject to (2) is as follows

$$\begin{aligned} L\left(R,c,\xi,\alpha,\beta\right) = &\sum_{j=1}^{m} R_j^2 + C_1\sum_{i=1}^{p}\xi_i + C_2\sum_{i=p+1}^{n}\sum_{j=1}^{m}\xi_{ij} \\ &+ \sum_{i=1}^{p}\alpha_i\left(\sum_{j=1}^{m} u_{ij}^d\left(\left\|\phi(x_i)-c_j\right\|^2 - R_j^2\right) - \xi_i\right) \\ &- \sum_{i=p+1}^{n}\sum_{j=1}^{m}\alpha_{ij}\left(\left\|\phi(x_i)-c_j\right\|^2 - R_j^2 + \xi_{ij}\right) \\ &- \sum_{i=p+1}^{n}\sum_{j=1}^{m}\beta_{ij}\xi_{ij} - \sum_{i=1}^{p}\beta_i\xi_i \end{aligned} \tag{11}$$

where $C_1 = \frac{1}{\nu_1 p}$, $C_2 = \frac{1}{\nu_2 q}$ and $q = n - p$, q is the number of abnormal data samples.

Setting derivatives of $L(R,c,\xi,\alpha,\beta)$ with respect to primal variables to 0, we obtain

$$\frac{\partial L}{\partial R_j} = 0 \to \sum_{i=1}^{p} u_{ij}^d\alpha_i - \sum_{i=p+1}^{n}\alpha_{ij} = 1,\ j = 1,\ldots,m \tag{12}$$

$$\frac{\partial L}{\partial c_j} = 0 \to c_j = \sum_{i=1}^{p} u_{ij}^d\alpha_i\phi(x_i) - \sum_{i=p+1}^{n}\alpha_{ij}\phi(x_i),\ j = 1,\ldots,m \tag{13}$$

$$\frac{\partial L}{\partial \xi_i} = 0 \Rightarrow \alpha_i + \beta_i = C_1,\ i = 1,\ldots,p \tag{14}$$

$$\frac{\partial L}{\partial \xi_{ij}} = 0 \to \alpha_{ij} + \beta_{ij} = C_2,\ i = p+1,\ldots,n \quad j = 1,\ldots,m \tag{15}$$

To get the dual form, we substitute equations (12)-(15) to the Lagrange function in (11) and obtain the following

$$\begin{aligned} &L(R,c,\xi,\alpha,\beta) = \sum_{i=1}^{p}\sum_{j=1}^{m}\alpha_i u_{ij}^d\left(\left\|\phi(x_i)-c_j\right\|^2\right) - \sum_{i=p+1}^{n}\sum_{j=1}^{m}\alpha_{ij}\left(\left\|\phi(x_i)-c_j\right\|^2\right) \\ &= \sum_{i=1}^{p}\sum_{j=1}^{m} u_{ij}^d\alpha_i K(x_i,x_i) - \sum_{j=1}^{m}\sum_{i=p+1}^{n}\alpha_{ij}K(x_i,x_i) \\ &-2\sum_{j=1}^{m} c_j\left(\sum_{i=1}^{p} u_{ij}^d\alpha_i\phi(x_i) - \sum_{i=p+1}^{n}\alpha_{ij}\phi(x_i)\right) + \sum_{j=1}^{m}\left\|c_j\right\|^2\left(\sum_{i=1}^{p} u_{ij}^d\alpha_i - \sum_{i=p+1}^{n}\alpha_{ij}\right) \\ &= \sum_{i=1}^{p}\alpha_i K(x_i,x_i)\sum_{j=1}^{m} u_{ij}^d - \sum_{j=1}^{m}\sum_{i=p+1}^{n}\alpha_{ij}K(x_i,x_i) - \sum_{j=1}^{m}\left\|c_j\right\|^2 \\ &= \sum_{i=1}^{p}\alpha_i s_i K(x_i,x_i) - \sum_{r,j}\alpha_{rj}K(x_r,x_r) - \sum_{j=1}^{m}\left\|\sum_{i=1}^{p} u_{ij}^d\alpha_i\phi(x_i) - \sum_{i=p+1}^{n}\alpha_{ij}\phi(x_i)\right\|^2 \\ &= \sum_{i=1}^{p}\alpha_i s_i K(x_i,x_i) - \sum_{r,j}\alpha_{rj}K(x_r,x_r) - \sum_{i,i'} u_i u_{i'} K(x_i,x_{i'})\alpha_i\alpha_{i'} \\ &- \sum_{rj,r'j}\alpha_{rj}\alpha_{r'j}K(x_r,x_{r'}) + 2\sum_{i,rj} u_{ij}^d K(x_i,x_r)\alpha_i\alpha_{rj} \end{aligned} \tag{16}$$

where $1 \leq i, i' \leq p$, $p+1 \leq r, r' \leq n$, $1 \leq j \leq m$, $u_i = [u_{i1}^d, u_{i2}^d, \ldots, u_{im}^d]$, $u_i u_{i'} = \sum\limits_{j=1}^{m} u_{ij}^d u_{i'j}^d$, and $s_i = \sum\limits_{j=1}^{m} u_{ij}^d$.

We come up with the following optimisation problem

$$\min_{\alpha} \left(\begin{array}{l} \sum\limits_{i,i'} u_i u_{i'} K(x_i, x_{i'}) \alpha_i \alpha_{i'} + \sum\limits_{rj, r'j} \alpha_{rj} \alpha_{r'j} K(x_r, x_{r'}) - 2 \sum\limits_{i,rj} u_{ij}^d K(x_i, x_r) \alpha_i \alpha_{rj} \\ - \sum\limits_{i=1}^{p} \alpha_i u_i K(x_i, x_i) + \sum\limits_{r,j} \alpha_{rj} K(x_r, x_r) \end{array} \right) \tag{17}$$

subject to

$$\begin{array}{l} \sum\limits_{i=1}^{p} u_{ij}^d \alpha_i - \sum\limits_{i=p+1}^{n} \alpha_{ij} = 1, \; j = 1, \ldots, m \\ 0 \leq \alpha_i \leq C_1, \; i = 1, \ldots, p \\ 0 \leq \alpha_{ij} \leq C_2, \; i = p+1, \ldots, n, \; j = 1, \ldots, m \end{array} \tag{18}$$

Note that the number of variables in solution of the optimisation problem (17) is $p + (n-p) \times m$. In practice, we apply the Interior Point (IP) method [3] to solve out the above optimisation problem. The complexity is dependent on double logarithmic of tolerance ϵ, i.e. $log(log(1/\epsilon))$.

2.3 Calculating Membership *U*

We are in position to describe how to evaluate matrix U after obtaining new (R, c). Given $1 \leq i \leq p$, let us denote

$$\begin{array}{l} d_{ij} = \|\phi(x_i) - c_j\|^2 - R_j^2 \quad and \quad D_{ij} = \left(\frac{1}{d_{ij}}\right)^{\frac{1}{d-1}} \\ j_0 = \arg\min\limits_{1 \leq j \leq m} d_{ij} \end{array} \tag{19}$$

The membership matrix can be updated as follows

If $d_{ij_0} \leq 0$ **then** $u_{ij_0} = 1$ **and** $u_{ij} = 0, \; j \neq j_0$
Else

$$u_{ij} = \frac{D_{ij}}{\sum\limits_{k=1}^{m} D_{ik}}, \; j = 1, \ldots, m \tag{20}$$

2.4 Iterative Learning Process

The proposed iterative learning process for FMS-SVDD will run two alternative steps until a convergence is reached as follows

```
Initialise U by clustering the normal data set in the input space
Repeat the following
    Calculate R, c and ξ using U
    Calculate U using R and c
Until convergence is reached
```

2.5 Theoretical Background of FMS-SVDD

In the objective function $\sum\limits_{j=1}^{m} R_j^2 + \frac{1}{\nu_1 p} \sum\limits_{i=1}^{p} \xi_i + \frac{1}{\nu_2 q} \sum\limits_{i=p+1}^{n} \sum\limits_{j=1}^{m} \xi_{ij}$, the first summand can be regarded as regularisation quantity and the rest can be considered as empirical risk (referred to Theorem 2). We will prove that structural risk $\sum\limits_{j=1}^{m} R_j^2 + \frac{1}{\nu_1 p} \sum\limits_{i=1}^{p} \xi_i + \frac{1}{\nu_2 q} \sum\limits_{i=p+1}^{n} \sum\limits_{j=1}^{m} \xi_{ij}$ gradually becomes smaller in Theorem 4.

Theorem 3. *Given a m-multivariate function $f(x_1, x_2, ..., x_m) = \sum\limits_{i=1}^{m} d_i x_i^d$ and $d > 1$. The following optimisation problem*

$$\min_x f(x) \tag{21}$$

subject to

$$\sum_{i=1}^{m} x_i = 1 \tag{22}$$

yields the solution as follows

If $d_{i_0} \le 0$ then $x_{i_0} = 1$ and $x_i = 0,\ i \ne i_0$
Else $x_i = \frac{\left(\frac{1}{d_i}\right)^{\frac{1}{d-1}}}{\sum\limits_{k=1}^{m} \left(\frac{1}{d_k}\right)^{\frac{1}{d-1}}}, \quad i = 1, \ldots, m$
where $i_0 = \arg\min\limits_{1 \le i \le m} d_i$.

Proof
Case 1: $d_{i_0} < 0$

$$\sum_{i=1}^{m} d_i x_i^d \ge d_{i_0} \sum_{i=1}^{m} x_i^d \ge d_{i_0} \sum_{i=1}^{m} x_i = d_{i_0} = f(0, ..., 1_{i_0}, ..., 0) \tag{23}$$

since $d > 1$.

Case 2: $d_{i_0} \ge 0$
The Lagrange function is of

$$L(x, \lambda) = \sum_{i=1}^{m} d_i x_i^d - \lambda \left(\sum_{i=1}^{m} x_i - 1 \right) \tag{24}$$

where λ is Lagrange multiplier.

Setting derivatives to 0, we gain

$$\begin{aligned} &\tfrac{\partial L}{\partial x_i} = 0 \Rightarrow d d_i x_i^{d-1} - \lambda = 0 \\ &\Rightarrow x_i = \left(\tfrac{\lambda}{d d_i}\right)^{\frac{1}{d-1}},\ i = 1, \ldots, m \end{aligned} \tag{25}$$

From $\sum\limits_{i=1}^{m} x_i = 1$, we have $x_i = \frac{\left(\frac{1}{d_i}\right)^{\frac{1}{d-1}}}{\sum\limits_{k=1}^{m}\left(\frac{1}{d_k}\right)^{\frac{1}{d-1}}}, \quad i = 1, \ldots, m.$

Theorem 4. *Let ($R^{(t)}$, $c^{(t)}$, $\xi_i^{(t)}$, $\xi_{ij}^{(t)}$, $U^{(t)}$) and ($R^{(t+1)}$, $c^{(t+1)}$, $\xi_i^{(t+1)}$, $\xi_{ij}^{(t+1)}$, $U^{(t+1)}$) be solutions at the previous iteration and current iteration, respectively. The following inequality holds*

$$\begin{aligned}\sum_{j=1}^{m}\left(R_j^{(t+1)}\right)^2 + \tfrac{1}{\nu_1 p}\sum_{i=1}^{p}\xi_i^{(t+1)} + \tfrac{1}{\nu_2 q}\sum_{i=p+1}^{n}\sum_{j=1}^{m}\xi_i^{(t+1)} \le \sum_{j=1}^{m}\left(R_j^{(t)}\right)^2 + \tfrac{1}{\nu_1 p}\sum_{i=1}^{p}\xi_i^{(t)} \\ + \tfrac{1}{\nu_2 q}\sum_{i=p+1}^{n}\sum_{j=1}^{m}\xi_i^{(t)}\end{aligned} \tag{26}$$

Proof

By referring to Theorem 3, it is easy to see that $u_i^{(t+1)} = (u_{i1}^{(t+1)}, ..., u_{im}^{(t+1)})$, $i = 1, \ldots, p$ is solution of the following optimisation problem

$$\min\left(\sum_{j=1}^{m} d_{ij}^{(t)} u_{ij}^{d}\right) \tag{27}$$

subject to

$$\sum_{j=1}^{m} u_{ij} = 1 \tag{28}$$

Therefore, we have

$$\sum_{j=1}^{m} d_{ij}^{(t)}(u_{ij}^{(t+1)})^d \le \sum_{j=1}^{m} d_{ij}^{(t)}(u_{ij}^{(t)})^d \tag{29}$$

It means that

$$\begin{aligned}&\sum_{j=1}^{m}\left(u_{ij}^{(t+1)}\right)^d\left(\left\|\phi(x_i) - c_j^{(t)}\right\|^2 - \left(R_j^{(t)}\right)^2\right) \\ &\le \sum_{j=1}^{m}\left(u_{ij}^{(t)}\right)^d\left(\left\|\phi(x_i) - c_j^{(t)}\right\|^2 - \left(R_j^{(t)}\right)^2\right) \le \xi_i^{(t)}\end{aligned} \tag{30}$$

or

$$\sum_{j=1}^{m}\left(u_{ij}^{(t+1)}\right)^d\left\|\phi(x_i) - c_j^{(t)}\right\|^2 \le \sum_{j=1}^{m}\left(u_{ij}^{(t+1)}\right)^d\left(R_j^{(t)}\right)^2 + \xi_i^{(t)} \tag{31}$$

It is certain that for $i = p+1, \ldots, n$ and $j = 1, \ldots, m$ we have

$$\left\|\phi(x_i) - c_j^{(t)}\right\|^2 \ge \left(R_j^{(t)}\right)^2 - \xi_{ij}^{(t)} \tag{32}$$

Hence, ($R^{(t)}$, $c^{(t)}$, $\xi_i^{(t)}$, $\xi_{ij}^{(t)}$, $U^{(t)}$) is feasible solution of optimisation problem (1) at time $t+1$. Since ($R^{(t+1)}$, $c^{(t+1)}$, $\xi_i^{(t+1)}$, $\xi_{ij}^{(t+1)}$, $U^{(t+1)}$) is minimal solution of this optimisation problem, Theorem 4 is proved.

3 Experiments

To show the performance of the proposed method, we established an experiment on 23 data sets in UCI repository as shown in Table 1. Most of them are two-class data sets and others are multi-class data sets. For each data set, we randomly selected one class and regarded its data samples as normal data samples. Data samples from the remaining class(es) were randomly selected to form a set of abnormal samples such that the ratio of normal samples and abnormal samples was kept to 12 : 1. We run cross validation with five folds and ten times and then take average of ten accuracies to obtain the final cross validation accuracy.

Table 1. Details of the data sets: #normal,#abnormal and d are number of normal, abnormal data and dimension of the input space, respectively

Datasets	*#normal*	*#abnormal*	*#d*
Astroparticle	2000	166	4
Australian	307	25	14
Bioinformatics	221	18	20
Breast Cancer	444	36	10
Diabetes	500	41	8
Dna	464	38	180
DelfPump	1124	93	64
Germany Number	300	24	24
Four class	307	25	2
Glass	70	5	9
Heart	164	13	13
Ionosphere	225	18	34
Letter	594	49	16
Liver Disorders	145	12	6
Sonar	97	8	60
Specf	254	21	44
Splice	517	43	60
SvmGuide1	2000	166	4
SvmGuide3	296	24	22
Thyroid	3679	93	21
Vehicle	212	17	18
Wine	59	5	13
USPS	1194	99	256

We compared the proposed method with SVDD [14] and HMS-SVDD [8]. The popular RBF Kernel $K(x, x') = e^{-\gamma\|x-x'\|^2}$ was applied and the parameter γ was varied in grid $\{2^i : i = 2j + 1, j = -8, \ldots, 2\}$. The parameter ν_1 and ν_2 were selected in grid $\{0.1, 0.2, 0.3, 0.4\}$. For FMS-SVDD, the number of spheres was chosen in grid $\{3, 5, 7, 9\}$ and parameter d was set to 1.5. To evaluate the classification rate, we employed the accuracy metric given by $acc = \frac{acc^+ + acc^-}{2}$ where

acc^+ and acc^- are the accuracies on positive (normal) and negative (abnormal) classes, respectively.

For most of the data sets, especially for the large data sets, the proposed method outperforms other kernel methods.

Table 2. Experimental results on 23 data sets in UCI repository

Datasets	*SVDD*	*HMS-SVDD*	*FMS-SVDD*
Astroparticle	91%	94%	**96%**
Australian	82%	82%	82%
Bioinformatics	69%	82%	**84%**
Breast Cancer	95%	98%	**99%**
Diabetes	65%	**71%**	70%
Dna	81%	97%	**97%**
DelfPump	69%	74%	**74%**
Germany Number	68%	70%	**72%**
Four class	93%	96%	**97%**
Glass	82%	88%	**90%**
Heart	83%	**86%**	85%
Ionosphere	88%	91%	**94%**
Letter	90%	95%	**96%**
Liver Disorders	62%	71%	**72%**
Sonar	63%	69%	**71%**
Specf	70%	**76%**	**76%**
Splice	57%	63%	**64%**
SvmGuide1	92%	**98%**	**98%**
SvmGuide3	63%	68%	**70%**
Thyroid	88%	92%	**94%**
Vehicle	58%	59%	**60%**
Wine	97%	**98%**	**98%**
USPS	93%	**95%**	94%

4 Conclusion

In this paper, we have presented a fuzzy approach to Multi-sphere Support Vector Data Description to provide a better description to data sets with mixture of distinctive distributions. Each sample is assigned a fuzzy membership function representing the degree of belonging of that sample to a hypersphere. We have theoretically proved that *structural risk* becomes smaller across iterations in the learning process. Experiments on 23 real data sets in UCI repository have showed a better performance of the proposed method in comparison to HMS-SVDD and SVDD.

References

[1] Ben-Hur, A., Horn, D., Siegelmann, H.T., Vapnik, V.: Support vector clustering. Journal of Machine Learning Research 2, 125–137 (2001)

[2] Boser, B.E., Guyon, I.M., Vapnik, V.: A training algorithm for optimal margin classifiers. In: Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory, pp. 144–152. ACM Press (1992)

[3] Boyd, S., Vandenberghe, L.: Convex Optimisation. Cambridge University Press (2004)

[4] Burges, C.J.C.: A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery 2, 121–167 (1998)

[5] Chen, Y., Zhou, X., Huang, T.S.: One-class svm for learning in image retrieval. In: ICIP (2001)

[6] Chiang, J.-H., Hao, P.-Y.: A new kernel-based fuzzy clustering approach:support vector clustering with cell growing. IEEE Transactions on Fuzzy Systems 11(4), 518–527 (2003)

[7] Le, T., Tran, D., Ma, W., Sharma, D.: An optimal sphere and two large margins approach for novelty detection. In: The 2010 International Joint Conference on Neural Networks (IJCNN), pp. 1–6 (2010)

[8] Le, T., Tran, D., Ma, W., Sharma, D.: A theoretical framework for multi-sphere support vector data description. In: Wong, K.W., Mendis, B.S.U., Bouzerdoum, A. (eds.) ICONIP 2010, Part II. LNCS, vol. 6444, pp. 132–142. Springer, Heidelberg (2010)

[9] Lee, K., Kim, W., Lee, K.H., Lee, D.: Density-induced support vector data description. IEEE Transactions on Neural Networks 18(1), 284–289 (2007)

[10] GhasemiGol, M., Monsefi, R., Yazdi, H.S.: Ellipse support vector data description. In: Palmer-Brown, D., Draganova, C., Pimenidis, E., Mouratidis, H. (eds.) EANN 2009. CCIS, vol. 43, pp. 257–268. Springer, Heidelberg (2009)

[11] Moya, M.M., Koch, M.W., Hostetler, L.D.: One-class classifier networks for target recognition applications, pp. 797–801 (1991)

[12] Scott, C.D., Nowak, R.D.: Learning minimum volume sets. Journal of Machine Learning Research 7, 665–704 (2006)

[13] Tax, D.M.J., Duin, R.P.W.: Support vector data description. Journal of Machine Learning Research 54(1), 45–66 (2004)

[14] Tax, D.M.J., Duin, R.P.W.: Support vector domain description. Pattern Recognition Letters 20, 1191–1199 (1999)

[15] Wu, M., Ye, J.: A small sphere and large margin approach for novelty detection using training data with outliers. IEEE Transactions on Pattern Analysis and Machine Intelligence 31(11), 2088–2092 (2009)

[16] Xiao, Y., Liu, B., Cao, L., Wu, X., Zhang, C., Hao, Z., Yang, F., Cao, J.: Multi-sphere support vector data description for outliers detection on multi-distribution data. In: ICDM Workshops, pp. 82–87 (2009)

Erratum: A Coupled Clustering Approach for Items Recommendation

Yonghong Yu[1], Can Wang[2], Yang Gao[1], Longbing Cao[2], and Xixi Chen[3]

[1] State Key Lab for Novel Software Technology, Nanjing University, P.R. China
[2] Advanced Analytics Institute, University of Technology Sydney, Australia
[3] Shandong Branch, Bank of Communications, P.R. China
{yuyh.nju,canwang613}@gmail.com, gaoy@nju.edu.cn,
LongBing.Cao@uts.edu.au

J. Pei et al. (Eds.): PAKDD 2013, Part II, LNAI 7819, pp. 365–376, 2013.

DOI 10.1007/978-3-642-37456-2_49

The name of the 5th author has been printed incorrectly in the paper. Instead of "Xixi Chen" it should be "Qianqian Chen".

The original online version for this chapter can be found at
http://dx.doi.org/10.1007/978-3-642-37456-2_31

Author Index