

Lecture Notes in Social Networks

Przemysław Kazienko
Nitesh Chawla *Editors*

Applications of Social Media and Social Network Analysis

 Springer

Lecture Notes in Social Networks

Series editors

Reda Alhadj, University of Calgary, Calgary, AB, Canada

Uwe Glässer, Simon Fraser University, Burnaby, BC, Canada

Advisory Board

Charu Aggarwal, IBM T.J. Watson Research Center, Hawthorne, NY, USA

Patricia L. Brantingham, Simon Fraser University, Burnaby, BC, Canada

Thilo Gross, University of Bristol, Bristol, UK

Jiawei Han, University of Illinois at Urbana-Champaign, IL, USA

Huan Liu, Arizona State University, Tempe, AZ, USA

Raúl Manásevich, University of Chile, Santiago, Chile

Anthony J. Masys, Centre for Security Science, Ottawa, ON, Canada

Carlo Morselli, University of Montreal, QC, Canada

Rafael Wittek, University of Groningen, The Netherlands

Daniel Zeng, The University of Arizona, Tucson, AZ, USA

More information about this series at <http://www.springer.com/series/8768>

Przemysław Kazienko · Nitesh Chawla
Editors

Applications of Social Media and Social Network Analysis

 Springer

Editors

Przemysław Kazienko
Department of Computational Intelligence
Wrocław University of Technology
Wrocław
Poland

Nitesh Chawla
Department of Computer Science
and Engineering
University of Notre Dame
Notre Dame, IN
USA

ISSN 2190-5428

Lecture Notes in Social Networks

ISBN 978-3-319-19002-0

DOI 10.1007/978-3-319-19003-7

ISSN 2190-5436 (electronic)

ISBN 978-3-319-19003-7 (eBook)

Library of Congress Control Number: 2015939432

Springer Cham Heidelberg New York Dordrecht London

© Springer International Publishing Switzerland 2015

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

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

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made.

Printed on acid-free paper

Springer International Publishing AG Switzerland is part of Springer Science+Business Media
(www.springer.com)

Preface

Social media and social networks are pervasive in the daily use as well as in a number of applications. Social media and social networks are also intertwined, as the social medial platforms also offer the opportunity to develop and analyze social networks. This book is an edited collection of a number of chapters that focus on topics from analysis of different nodal attributes of a social network, including node centrality, node reputation, and contributions of different nodes within the network; aspects of information and influence diffusion in a social network, including influence diffusion, rumor spreading, and control; sentiment analysis via opinion extraction and topic modeling; system level framework for decision making and cultural experience learning; and finally an application of social networks in a biometric field. The chapters are independent contributions of the authors.

“[A Node-Centric Reputation Computation Algorithm on Online Social Networks](#)” introduces an algorithm to capture how reputations of individuals spread within the network. There are two major contributions: (1) authors demonstrated that individual’s reputation is influenced by its position in the network and associated local structure; (2) the topological information of networks matters in computing individual’s reputation. In the algorithm, individual’s reputation is updated from its neighbors by considering the interaction history between this node and its neighbors.

“[Measuring Centralities for Transportation Networks Beyond Structures](#)” discusses centrality from the perspective of evaluating node’s importance within the network structure. Centrality measures are important to identify critical nodes in transportation networks, which are useful to improve the design of transportation networks. However, most centrality measures only consider the network topological information, and thus they are oblivious of transportation factors. This paper introduced a new centrality measure, which combines network topology and some external transportation factors, such as travel time delay and travel flux volume. The new centrality measure is demonstrated to be more efficient in identifying critical nodes in transportation networks.

“[Indifferent Attachment: The Role of Degree in Ranking Friends](#)” discusses the role of degree in ranking friends. The authors study whether the popularity of one’s

friends is the determining factor when ranking the order of all friends. They find that the popularity of two friends is essentially uninformative about which will be ranked as the more preferred friend. Surprisingly, there is evidence that individuals tend to prefer less popular friends to more popular ones. These observations suggest that positing individuals' tendency to attach to popular people—as in network-growth models like preferential attachment—may not suffice to explain the heavy-tailed degree distributions seen in real networks.

“[Analyzing the Social Networks of Contributors in Open Source Software Community](#)” analyzes the social networks of contributors in the open source community. The authors analyze the connectivity and tie structure in social networks where each user is one contributor of Open Source Software communities, and to investigate the network effects on developers' productivity. First, they find high degree nodes tend to connect more with low degree nodes suggesting collaboration between experts and newbie developers; second, they show positive correlations between in-degree, out-degree, betweenness, and closeness centrality and the developers' contribution and commitment in Open Source Software projects; third, in general, highly connected and strongly tied contributors are more productive than the low connected, weakly tied, and not connected contributors.

“[Precise Modeling Rumor Propagation and Control Strategy on Social Networks](#)” discusses various models for epidemic spreading of rumor and/or information. The authors also propose a novel epidemic model, namely the SPNR model. SPNR model has three states, infected states, positive infected states, and negative infected states. The introduction of positive infected states and negative infected states enable the SPNR model to better capture the rumor spreading process in real-world social systems. Additionally, a corresponding rumor control strategy is designed based on SPNR model. This novel rumor control strategy is demonstrated to be effective in fighting against rumor spreading, compared with several state-of-the-art control strategies.

“[Studying Graph Dynamics Through Intrinsic Time Based Diffusion Analysis](#)” focuses on time-based diffusion analysis. The authors aim to characterize the co-evolution of network dynamics and diffusion processes. They propose the notion of intrinsic time to record the formation of new links, and further use it to isolate the diffusion processes from the network evolution. The experimental results show significant differences in the analysis of diffusion in the extrinsic, extrinsic converted into intrinsic, and intrinsic times.

“[A Learning-Based Approach for Real-Time Emotion Classification of Tweets](#)” focuses on emotion recognition by analyzing tweets—a cross-section of social media and social networks. The authors discuss their computational framework as well as the machine learning-based approach. The reported experiments demonstrate that the authors' approach is competitive to the other lexicon-based methods. They also demonstrate that their computational framework can make emotion recognition and classification a lightweight task, enabling use on mobile devices.

“[A New Linguistic Approach to Assess the Opinion of Users in Social Network Environments](#)” focuses on linguistic methods to assess the opinion of users in social networks. The authors study the problem of differentiating texts expressing a

positive or a negative opinion. The major observation is that positive texts are statistically shorter than negative ones. The method is to generate a lexicon that is used to indicate the level of opinions of given texts. The resulting adaptability would represent an advantage with free or approximate expression commonly found in social networks environment.

“[Visual Analysis of Topical Evolution in Unstructured Text: Design and Evaluation of TopicFlow](#)” presents an application of topic modeling to group-related documents into automatically generated topics, as well as an interactive tool, TopicFlow, to visualize the evolution of these topics. The authors discuss their analysis technique, namely binned topic models and alignment. It is an application Latent Dirichlet Application (LDA) to time-stamped documents at independent time intervals. The TopicFlow tool provides interactive visualization capabilities to select topics between time slices as they develop or diverge over a time period.

“[Explaining Scientific and Technical Emergence Forecasting](#)” provides an infrastructure for enabling the explanation of hybrid intelligence systems using probabilistic models and providing corresponding evidence. This infrastructure is designed to assist users in the interpretation of results. There are two contributions: (1) enable to support transparency into indicator-based forecasting systems; (2) provide evidence underlying presented indicators to the analyst users. The application of such an infrastructure is to automate the prediction of trends in science and technology based on the information of scientific publications and published patents.

“[Combining Social, Audiovisual and Experiment Content for Enhanced Cultural Experiences](#)” presents an experimental system that enhances the experience of visitors in cultural centers by leveraging social networks and multimedia. The system offers a modern, immersive, richer experience to the visitors, and provides valuable instant, spontaneous, and comprehensive feedback to organizers. It has been deployed and used in the Foundation of the Hellenic World cultural center.

“[Social Network Analysis for Biometric Template Protection](#)” provides an application of social network analysis for biometric template protection. Its general goal is to provide biometric system with cancelability that can defend the biometric database. The authors applied social network analysis to transfer raw biometric features related to, e.g., face or ear images, into secure ones.

Acknowledgments

This work was partially supported by The National Science Centre, decision no. DEC-2013/09/B/ST6/02317 and the European Commission under the 7th Framework Programme, Coordination and Support Action, Grant Agreement Number 316097, the ENGINE project.

Wroclaw, Poland
Notre Dame, USA

Przemysław Kazienko
Nitesh Chawla

Contents

A Node-Centric Reputation Computation Algorithm on Online Social Networks	1
JooYoung Lee and Jae C. Oh	
Measuring Centralities for Transportation Networks Beyond Structures	23
Yew-Yih Cheng, Roy Ka-Wei Lee, Ee-Peng Lim and Feida Zhu	
Indifferent Attachment: The Role of Degree in Ranking Friends	41
David Liben-Nowell, Carissa Knipe and Calder Coalson	
Analyzing the Social Networks of Contributors in Open Source Software Community	57
Mohammad Y. Allaho and Wang-Chien Lee	
Precise Modeling Rumor Propagation and Control Strategy on Social Networks	77
Yuanyuan Bao, Chengqi Yi, Yibo Xue and Yingfei Dong	
Studying Graph Dynamics Through Intrinsic Time Based Diffusion Analysis	103
Alice Albano, Jean-Loup Guillaume, Sébastien Heymann and Bénédicte Le Grand	
A Learning Based Approach for Real-Time Emotion Classification of Tweets	125
Olivier Janssens, Rik Van de Walle and Sofie Van Hoecke	
A New Linguistic Approach to Assess the Opinion of Users in Social Network Environments	143
Luigi Lancieri and Eric Leprêtre	

**Visual Analysis of Topical Evolution in Unstructured Text:
Design and Evaluation of TopicFlow 159**
Alison Smith, Sana Malik and Ben Shneiderman

Explaining Scientific and Technical Emergence Forecasting. 177
James R. Michaelis, Deborah L. McGuinness, Cynthia Chang,
John Erickson, Daniel Hunter and Olga Babko-Malaya

**Combining Social, Audiovisual and Experiment Content
for Enhanced Cultural Experiences 193**
Kleopatra Konstanteli, Athanasios Voulodimos,
Georgios Palaiokrassas, Konstantinos Psychas,
Simon Crowle, David Salama Osborne,
Efstathia Chatzi and Theodora Varvarigou

Social Network Analysis for Biometric Template Protection. 217
Padma Polash Paul, Marina Gavrilova and Reda Alhadj

Glossary 235

Index 239

Contributors

Alice Albano Sorbonne Universités, Paris, France; CNRS, Paris, France; Université Paris 1 Panthéon-Sorbonne, CRI, Paris, France

Reda Alhajj University of Calgary, Calgary, AB, Canada

Mohammad Y. Allaho Department of Computer Science and Engineering, The Pennsylvania State University, University Park, PA, USA

Olga Babko-Malaya BAE Systems, Burlington, MA, USA

Yuanyuan Bao Tsinghua National Laboratory for Information Science and Technology, Tsinghua University, Beijing, China

Cynthia Chang Rensselaer Polytechnic Institute, Troy, NY, USA

Efstathia Chatzi Foundation of the Hellenic World, Athens, Greece

Yew-Yih Cheng Living Analytics Research Centre, Singapore Management University, Singapore, Singapore

Calder Coalson Department of Computer Science, Carleton College, Northfield, MN, USA

Simon Crowle IT Innovation Centre, University of Southampton, Southampton, UK

Yingfei Dong Department of Electrical Engineering, University of Hawaii, Honolulu, USA

John Erickson Rensselaer Polytechnic Institute, Troy, NY, USA

Marina Gavrilova University of Calgary, Calgary, AB, Canada

Bénédicte Le Grand Sorbonne Universités, Paris, France; CNRS, Paris, France; Université Paris 1 Panthéon-Sorbonne, CRI, Paris, France

Jean-Loup Guillaume Sorbonne Universités, Paris, France; CNRS, Paris, France; Université Paris 1 Panthéon-Sorbonne, CRI, Paris, France

Sébastien Heymann Sorbonne Universités, Paris, France; CNRS, Paris, France; Université Paris 1 Panthéon-Sorbonne, CRI, France

Sofie Van Hoecke Department of Electronics and Information Systems, Ghent University, 9050 Ledeborg-Ghent, Belgium

Daniel Hunter BAE Systems, Burlington, MA, USA

Olivier Janssens Department of Electronics and Information Systems, Ghent University, Ledeborg-Ghent, Belgium

Carissa Knipe Department of Computer Science, Carleton College, Northfield, MN, USA

Kleopatra Konstanteli National Technical University of Athens, Athens, Greece

Luigi Lancieri Laboratoire D'Informatique Fondamentale de Lille, Université de Lille1, Villeneuve-d'Ascq, France

JooYoung Lee Department of Electrical Engineering and Computer Science, Syracuse University, Syracuse, New York

Roy Ka-Wei Lee Living Analytics Research Centre, Singapore Management University, Singapore, Singapore

Wang-Chien Lee Department of Computer Science and Engineering, The Pennsylvania State University, University Park, PA, USA

Eric Leprêtre Université de Lille1, Laboratoire D'Informatique Fondamentale de Lille, Villeneuve-d'Ascq, France

David Liben-Nowell Department of Computer Science, Carleton College, Northfield, MN, USA

Ee-Peng Lim Living Analytics Research Centre, Singapore Management University, Singapore, Singapore

Sana Malik University of Maryland, College Park, MD, USA

Deborah L. McGuinness Rensselaer Polytechnic Institute, Troy, NY, USA

James R. Michaelis Rensselaer Polytechnic Institute, Troy, NY, USA

Jae C. Oh Department of Electrical Engineering and Computer Science, Syracuse University, Syracuse, New York

David Salama Osborne Atos, Spain

Georgios Palaiokrassas National Technical University of Athens, Athens, Greece

Padma Polash Paul University of Calgary Calgary, AB, Canada

Konstantinos Psychas National Technical University of Athens, Athens, Greece

Ben Shneiderman University of Maryland, College Park, MD, USA

Alison Smith University of Maryland, College Park, MD, USA; Decisive Analytics Corporation, Arlington, VA, USA

Theodora Varvarigou National Technical University of Athens, Athens, Greece

Athanasios Voulodimos National Technical University of Athens, Athens, Greece

Rik Van de Walle Department of Electronics and Information Systems, Ghent University, 9050 Ledeborg-Ghent, Belgium

Yibo Xue Tsinghua National Laboratory for Information Science and Technology, Tsinghua University, Beijing, China

Chengqi Yi School of Computer Science and Technology, Harbin University of Science and Technology, Harbin, China

Feida Zhu Living Analytics Research Centre, Singapore Management University, Singapore, Singapore

A Node-Centric Reputation Computation Algorithm on Online Social Networks

JooYoung Lee and Jae C. Oh

Abstract In online social networks, reputations of users (nodes) are emerged and propagated through interactions among the users. These interactions include intrinsic and extrinsic consensus (voting) among neighboring users influenced by the network topology. We introduce an algorithm that considers the degree information of nodes (users) to model how reputations spread within the network. In our algorithm, each node updates reputations about its neighbors by considering the history of interactions and the frequency of the interactions in recent history. The algorithm also captures the phenomena of accuracy of reputations deteriorating over time if interactions have not occurred recently. We present the following two contributions through experiments: (1) We show that an agent's reputation value is influenced by the position of the node in the network and the neighboring topology; and (2) We also show that our algorithm can compute more accurate reputations than existing algorithms especially when the topological information matters. The experiments are conducted in random social networks and Autonomous Systems Network of the Internet. In addition, we show the efficacies of each component in our algorithm and present their effects on the algorithm.

Keywords Reputation management · Multi-agent systems · Propagation of information · Trust in autonomous systems

J. Lee (✉) · J.C. Oh
Department of Electrical Engineering and Computer Science Syracuse University,
Syracuse 13244, New York
e-mail: j.lee@innopolis.ru, jlee150@syr.edu

J.C. Oh
e-mail: jcoh@ecs.syr.edu

1 Introduction

Accurate reputation information about nodes in social networks improves services provided by the networks. For example, reputations can be calculated and updated for web sites and servers to identify malicious nodes and connections. The ability to observe and analyze propagations of reputations within a large social network structures is also important.

There are centralized and decentralized approaches in reputation computations. Centralized reputation management systems are often used in commercial applications, such as *eBay* and *Amazon*, where a centralized authority is relatively explicit, but such approaches fail in environments that lack a central authority. In online networks such as the Internet, social networks, and other multi-agent systems environments, a distributed reputation computation algorithm is naturally more suitable.

Several distributed reputation algorithms including *AFRAS* [1], *REGRET* [2] and *HISTOS* [3] exist. However, these algorithms do not consider frequencies and *velocity* of interactions; frequency of interactions is an important measure of reputation of the users involved. Velocity of interactions measures the second order information of frequency, i.e., the rate of changes in frequency. Existing algorithms also lack the consideration of topological information of the networks and the subjectivity of reputations (i.e., two different nodes may perceive the reputation of a node differently).

This article presents a new reputation management model that addresses the above issues. Our algorithm considers frequencies and velocity of interactions online. Because our algorithm is developed by modeling the behavior of social networks, we show the algorithm can be used as an analytical tool for studying social network behaviors as well as a query tool for retrieving reputation values for specific nodes at a given time. Through experiments, we show that how reputations emerge and propagate in random social networks and how the model captures the idea of dynamic reputation propagations from one part of the network to another. We also show experiments on real *Autonomous Systems Networks (ASN)* of the Internet for identifying malicious ASN nodes through our model. We compare our results with an existing reputation values computed by another well accepted ASN reputation management system. The results show that our algorithm computes similar reputation values as the values provided by the existing algorithm. However, we show that our algorithm is better suited to find many malicious ASN nodes while the compared method is good for finding the worst malicious node only. Finally, we extend the previous work presented in [4] to test the effectiveness of each components in computing reputation values.

The paper is organized as follows. Section 2 presents motivations and contributions of the work. Section 3 presents some of the well known reputation computation algorithms. In Sect. 4 we propose our reputation computation algorithm in detail. Then in Sect. 5, we explain experiment settings as well as the results. Following this, in Sect. 6, we discuss the contributions of the algorithm. Finally, in Sect. 7, we conclude and propose future works.

2 Motivations and Contributions

Reputation is an estimate or prediction of an agent's real behavior. Reputation values should dynamically reflect past behaviors of an agent and also tolerate bad behaviors which may have been caused by mistakes. The proposed algorithm has the following uniquely novel features:

- Because reputation is subjective, as it is one's estimate of another, we assume that reputation values of other agents are a private opinion of an agent. Each agent uses a recursive voting algorithm to incorporate opinions of other nodes for a more objective view of a target agent.
- Many social network analysis show that the degree of a node is an important indication of the node's influence as well as the topological information of the networks [5–7]. In the voting process, degrees of neighboring nodes are used to compute the weight of the votes. In this article, we assume that each node knows the degrees of its neighbors and the total number of nodes in the network. In many cases, including *HISTOS* [3], the topology of the network is assumed to be known to enable aggregations of reputation.
- In many times, how frequently interactions occurs within a limited period can be an important evidence about reputations of the nodes. The algorithm incorporates frequencies of interactions as well as velocity. Existing distributed algorithms usually only consider the total number of interactions.
- If two agents have not interacted for a long time, even if the total number of interactions is large, the reputation values for each other may not be up to date. Our algorithm applies a time decaying function to balance the previous reputation value and the evaluation of the most recent interaction.

3 Related Work

In this Section, we introduce some of the most well known reputation management algorithms, *AFRAS* [1], *REGRET* [2] and *HISTOS* [3]. According to the classifications discussed in [8], *AFRAS*, *REGRET* and *HISTOS* are distributed algorithms and combine direct interactions as well as the witness information to determine reputation. *HISTOS* considers only a given number of recent interactions and recursively aggregates reputation values following paths to the target agent. It also weights opinions from others with their reputation values. *AFRAS* uses fuzzy sets to represent reputation with truth levels. It combines the previous reputation and the current satisfaction level to compute a new reputation. It introduces a dynamic memory factor as a weight, updated with the similarity between the previous reputation and the current satisfaction level. *REGRET* uses the three dimensionality of reputation, namely *individual*, *social* and *ontological* dimensions. The *individual* dimension computes reputations based on direct experiences while *social* dimension col-

lects witness information. In addition, the *ontological* dimension adds possibility of combining different aspects of reputation.

The common idea behind the algorithms discussed above is that updating reputation involves weighted aggregation of the previous reputation and current feedbacks. The previous reputation is weighted with a factor of time or memory, and current feedbacks are computed through evaluation methods. The difference lies on how the algorithms interpret current interactions with respect to the given information about the target agent. In addition to that, we incorporate network degree information to measure the weight of the node's own opinion about the target node. For example, an agent gives more weight to its own evaluation than opinions from others if it has a relatively small number of neighbors in a network while it assigns more weight on others' opinions if it has more connections.

The reputation function in *HISTOS* algorithm focuses on estimating reputation values when an agent asks for reputation of another at distant, meaning it recursively aggregates ratings given by neighbors following the shortest paths. If the agent is directly connected to the target agent, it doesn't need to combine ratings from other nodes. On the other hand, our algorithm is more focused on updating reputations after interactions. After having a direct interaction with a target agent, agents still combines their direct experience with indirect experiences (through voting). This enables agents to keep personalized reputations of other agents that are not biased by unfortunate wrong impressions.

Other reputation systems include but not limited to *FIRE* [9] which identifies dishonest and mistaken agents, *TRAVOS* [10] which attempts to determine the credibility of witnesses when combining opinions and *PeerTRUST* [11] which addresses the bootstrapping problem in peer-to-peer systems.

4 ReMSA: Reputation Management for Social Agents

In this Section, we present the proposed algorithm—*ReMSA: Reputation Management for Social Agents* [4]. A reputation value is computed when an interaction between two agents occurs. In an interaction, an agent can be an observer, observee, or both. After an interaction, the observer evaluates the interaction to compute the reputation of the observee. If both are observers, they will compute reputations of each other. The more interactions occur between two agents, the more accurate the reputations are computed for the agents. As interactions occur within the network over time, reputations of agents in one part of the network will propagate to another part of the network, much similar to what is happening in real-world social networks. At any given time, any agent can query about reputation of an arbitrary agent. Note that the returned value to the agent may be different from the result of the query initiated by another agent. When there's enough interactions among overall agents in the network, reputations will propagate to the entire network, and more homogeneous views of reputations of agent will emerge.

Fig. 1 The sequence of processes agents take to update reputation values

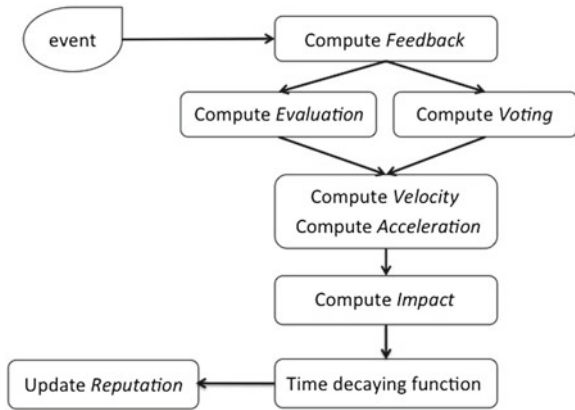


Figure 1 shows the flowchart of reputation computation for an observer node when an event occurs. Following subsections explain each process in Fig. 1.

4.1 Events

We define an event as an interaction between two agents with time information. There are two types of events in terms of who owns the event. When an interaction happens between two agents, if both agents can observe each other’s behavior, then both own the event. If only one of the agents can observe the other’s behavior, only the observer is the owner of the event. All the following computations are based on the observer agent’s point of view.

- The set of agents and events are defined as follows.

$$A = \{a_1, a_2, \dots, a_n\}$$

$$E_i = \{e_1, e_2, \dots, e_m\}$$

$$e_j = (t_j, a_j)$$

where, a_i is an observer agent, E_i is a set of events that a_i as an observer, and e_j is an event which consists of its time, t_j , and the observee agent, a_j .

- Given a network, we define α and β where nodes are agents and edges are relationships.

$$\alpha_i = \frac{d_i}{\max_{m \in A} \{d_m\}}$$

$$\beta_{il} = \frac{d_l}{\sum_{k \in N_i} d_k}$$

where, N_i is a set of i 's neighboring agents, and d_a is the degree of agent a . α_i is a ratio of i 's degree and the maximum degree in the network. α_i is used to weight i 's opinion (i.e., the observer's) since α_i represents i 's position in the network. It implies that we can infer an agent's confidence from its relative degree information. Given i and its neighbor l , β_{il} is a ratio of l 's degree and the sum of i 's neighbors' degrees. β_{il} represents l 's relative credibility from i 's perspective. i will use β_{il} to collect opinions from its neighbors. The neighbors of each l will recursively compute β_{l*} in turn until one of the three terminating conditions is met as explained in Sect. 4.2.3. In the voting algorithm shown in (1), i evaluates voting weights for each of its neighbor l .

4.2 Compute Feedback

Feedback process consists of two subprocesses, *Evaluation* and *Voting*. *Feedback* is a part of reputation updating function in Sect. 4.6.

4.2.1 Compute Evaluation

After each interaction, agents that were involved in the interaction evaluate the interaction according to their own evaluation methods. Therefore, *Evaluation* of interactions is subjective and can be implemented depending on applications. *Evaluation* of an event e is represented as a function, $Eval(e)$.

4.2.2 Compute Voting

While the evaluation function is computed by each agent, agents collect opinions from other agents before updating the reputation of the target agent through a voting process to combine diverse opinions. If a_i had an interaction with a_l , a_i can take a vote about a_l to its neighbors to obtain more objective views. The neighbors of a_i can either return a vote to a_i with their own evaluations (if they don't have neighbors other than a_i or a_l) or they can spread the vote to their neighbors. $V_{a_i a_l}^e$ is the weighted sum of voting results and we define it as follows.

$$V_{il}^e = \sum_{k \in N_i} \beta_{ik} \times F_{kl}^e \quad (1)$$

β_{ik} represents i 's delegation trust towards k and is multiplied by F_{kl}^e which is a weighted sum of i 's evaluation of l on the event e and collected feedbacks from

k 's neighbors about l . *Feedback* process is represented with the function F_{il}^e which recursively calls the voting function, V_{il}^e .

$$F_{il}^e = \alpha_i \times Eval_i(e) + (1 - \alpha_i) \times V_{il}^e \quad (2)$$

α_i implies self-confidence of i and it is multiplied by $Eval_i(e)$, the self evaluation on event e .

As shown in formula (1) and (2), *Feedback* and *Voting* processes are defined recursively.

4.2.3 Stopping Criteria

To avoid infinite loops or circular voting processes, we need to specify a few restrictions. First, when an agent takes a vote to its neighbors, it excludes itself. Since the agent's opinion about the target agent is already included in the evaluation function, it only needs to hear from its neighbors. Second, for the voters to avoid casting duplicate votes, each agent keeps history of votes which it has already participated. This is beneficial to the voters so that they don't waste their own resources on duplicate votes. Third, the base condition of the recursive voting is: (1) when an agent has only one neighbor which originated the voting process, (2) when an agent has two neighbors one being the originator and the other being the target agent and (3) when an agent has already participated in the current vote. In the first two cases, (1) and (2), the voter agent returns its reputation value of the target agent.

4.3 Compute Velocity and Acceleration

We also consider the frequency of events to compute reputation since an event with a dormant period should be treated differently from frequent ones. We define the velocity for each event to compute the acceleration of events. Then the acceleration of an event influences *Feedback* value of the event through the *Impact function*.

Velocity of an event is defined as follows.

$$Vel(e) = \frac{1}{t_e - t_{e'}} \quad (4)$$

where e' is the most recent previous event.

It is obvious that there needs to be at least two events to compute a velocity, otherwise the velocity is zero. Also, since we consider time with increasing positive integers, $t_e - t_{e'} > 0$ and $Vel(e) \in [0, 1]$.

Now, we can compute the acceleration of event e to identify if its velocity is increasing or decreasing through $Acc(e)$.

$$Acc(e) = Vel(e) - Vel(e') \quad (5)$$

4.4 Compute Impact

We introduce *Impact function* to calculate the influence of $Acc(e)$, defined in (5), on the feedback, F^e .

$$I(Acc(e), F^e) = |Acc(e)| \times F^e 3^{\frac{-Acc(e)}{|Acc(e)|}} + (1 - |Acc(e)|) \times F^e \quad (6)$$

The magnitude of $Acc(e)$ determines the curve of the function which decides how much to increase or decrease from F^e . When $Acc(e) > 0$, I increases the original feedback value, $I(Acc(e), F^e) > F^e$, and when $Acc(e) < 0$, I decreases the original feedback value, $I(Acc(e), F^e) < F^e$. If $Acc(e) = 0$ then $I(Acc(e), F^e) = F^e$ which means that when there is no change in the velocity, no impact is made to the feedback value, F^e .

4.5 Time Decaying Function

Time decaying function is an essential part of the reputation computation since it captures the temporal nature of information; old reputation value may not be as accurate as a new one. Intuitively, an interaction shortly after the previous one can make more use of the built-up reputation (current reputation) while an interaction after a long inactive period should rely more on the current feedback values since the built-up reputation is not up to date. As discussed in [12], time decaying function should to be designed carefully, based on the context (e.g. a periodic function) so that it can adjust time sensitivity weights when computing reputations.

We use an exponential decay function to capture the idea. Our time decaying function relies on the elapsed time since the last interaction.

$$D(x) = e^{-x}$$

where x is $t_e - t_{e'}$.

4.6 Update Reputation

Finally, we are now ready to explain the reputation update function that utilizes the functions discussed so far. A new reputation value is computed when a new event occurs. The new reputation is a weighted sum of the current reputation and the feedbacks. The current reputation is weighted by the time decaying function and *Impact function* is applied to the feedbacks. Finally, we formally define the reputation update function as follows.

Reputation update function:

$$R_{il}^{t_e} = d \times R_{il}^{t_{e'}} + (1 - d) \times I(\text{Acc}(e), F_{il}^e)$$

where $d = D(t_e - t_{e'})$.

4.7 Ask Function

In our algorithm, each agent keeps a list of reputation values of the neighbors. However, in some occasions, an agent might wonder about another agent's reputation other than the neighbors. Therefore, we implement *Ask function* to query a target agent's reputation who is not a neighbor. *Ask function* is the same as *Feedback function* except the agent does not have its own evaluation of the target agent. *Ask function*, then, goes through the same processes as in *Voting function* as in (1).

$$Ask_{il} = \begin{cases} R_{kl} & l \in N_k \\ \sum_{k \in N_i} \beta_{ik} \times Ask_{kl} & \text{otherwise} \end{cases}$$

5 Experiments

In this section, we present two sets of experiments. First, we compute reputations of Autonomous Systems (AS) in the Internet using our algorithm. And we compare our results with *AS-CRED* [13], which is a well-known reputation service for AS network. We use subsets of the real AS networks obtained from *RouteViews* [14]. Second, we study the emergence and propagation of reputations within random social networks generated by Graph-Generator [15]. Graph-Generator is a small Java program to create random graphs in which the number of nodes, edges and the maximum degree are specified.

5.1 Experiment 1: Computing Reputations of Autonomous Systems

We apply *ReMSA* algorithm to compute reputation values of Autonomous Systems. An Autonomous System is a collection of connected Internet Protocol (IP) routing prefixes under the control of one or more network operators. Since behaviors of each AS represent human interests, we consider AS network as social network. We analyze Border Gateway Protocol (BGP) updates data, which is the standard communication protocol for interconnecting ASes, to evaluate validity of activities among ASes in the network and compute reputation values based on the evaluations. The reputations of ASes could be incorporated for the routes deciding algorithm for each AS since reputations of ASes directly represent the behaviors of ASes.

5.1.1 Network Sampling Algorithm

As shown in Table 1, the number of nodes in the original AS network is very large because it represents all ASes in the Internet. However, only less than 10% of ASes appear in each day’s BGP update data we use. Therefore, for the tractability of the experiments, we extract two representative, scaled down, sub-networks from the original AS network. If we sample sub-networks from the original AS network using existing algorithms, most of the ASes in sampled sub-networks don’t appear in the BGP data. Instead of using the algorithms discussed in [16], we designed a context-based network extraction algorithm in order to sample meaningful sub-networks which contain most of the autonomous systems appearing in BGP update data. Therefore, we extract ASes that appeared in the BGP data so that we can compute reputations of the ASes. For each randomly chosen *ASPATH* in BGP update data on January 1, 2010, we add ASes which appear in the *ASPATH* to the sub-network and repeat the process until the desired number of nodes for the sub-network is reached (in this case, 5,000).

In order to measure whether the sub-networks represent the original AS network reasonably, we evaluate the sub-networks by the metrics defined in [16].

Table 1 shows the number of nodes and edges of the original network and two sampled sub-networks. The number of nodes of sub-networks (5,000) is approximately 15% of the real network (33,508) which is enough to match the properties shown in Table 2 [16]. Table 2 shows five different distributions of two sample networks measured using Kolmogorov-Smirnov D-statistics. *D*-statistic measures the agreement

Table 1 Nodes and edges information of networks

	Original	Sub-network1	Sub-network2
# Nodes	33,508	5,000	5,000
# Edges	75,001	19,953	22,379

Table 2 Sampling criteria for sub-networks

	In-deg	Hops	Sng-val	Sng-vec	Clust	AVG
Sub-network1	0.2743	0.3500	0.1883	0.1180	0.2346	0.2399
Sub-network2	0.0703	0.3500	0.1234	0.0357	0.1944	0.1547

between the true and the sample network property [16]. In the last column, we show the average of the results. Lower average values means more agreement between original network and the sub-network. Some good average values discussed in [16] were 0.202 and 0.205. Therefore, both of the sub-networks qualify for scaled-down sub-networks well representing the original.

5.1.2 Evaluation of BGP

We use BGP (Border Gateway Protocol) update data from *RouteViews* [14] dated from January, 2010. Also, we use the same analysis of AS-behavior discussed in [13] and [17] to compute feedbacks of BGP activities in order to compare our reputation computation results with *AS-CRED*. In order to use our algorithm, we need to define *events* and associated *observer* and *observee* in the problem domain of AS reputation computation. In BGP update data, for each update message sent from say, *AS0* to *AS1*, *AS1* analyzes the message as an observer and evaluates *AS0*'s behavior. Such a message is an event as we defined in Sect. 4.1. And an event can be a non-empty subset of the set $\{AS\text{-Prefix behavior}, AS\text{-Path behavior}, AS\text{-Link behavior}\}$, which represents the *observee*'s behaviors. Each behavior can be evaluated to be positive or negative and then the result of the evaluation is accumulated for that message. In other words, each message will have an associated score representing the behavior of the *observee*. We describe how each behavior is evaluated below.

- *AS-Prefix behavior*: For the observee AS and its prefix p , we compute two temporal metrics, *persistence* and *prevalence*. These two metrics can represent positive or negative behavior of the observee AS. We will not discuss the details of the metrics because they are beyond the scope of this paper. The value of the persistence and prevalence are compared against a set of thresholds mentioned in [17] and feedback is provided. For good behaviors, evaluation of 1 is provided and otherwise -1 .
- *AS-Path behavior*: We use AS relationship data from [18] to evaluate the valley free property of AS *paths*. None of the ASes in the AS *path* should form a valley. The observee AS provides an AS-Path. If a valley is found in the provided AS-Path and the first AS forming the valley is the observee, it gets evaluated by its observer with -1 .
- *AS-Link behavior*: For each link in the AS-Path provided by the observee, we compute persistence and prevalence values, then these are compared with the threshold

discussed in [17]. If the classification of the behavior is good, an evaluation of 1 is provided, otherwise the link is unstable, therefore, the evaluation is -1 .

5.1.3 Results

We compute reputations for each AS appeared in BGP update data for each day between January 1, 2010 and January 31, 2010. We average the reputations computed by *ReMSA* with two different sub-networks. We exclude ASes with good behaviors (positive reputation values) to compare the result with *AS-CRED* which accumulates reputation values when bad behavior occurs (zero is considered the best reputation in *AS-CRED*).

In Fig. 2, we show the distribution of reputation values of ASes computed by *AS-CRED* for January 1, 2010. In Fig. 3, we show the distribution of reputation values of ASes compute by *ReMSA* for the same day. The ranges of reputation values shown in the figures are different as they represent the raw reputation values computed by *AS-CRED* and *ReMSA*. Since *AS-CRED* computes centralized reputation values, we averaged reputation values computed for each AS by *ReMSA*. The purpose of each algorithm is implied by the distribution of reputation values shown in the figures. *AS-CRED* computes reputation values of ASes to detect globally malicious ASes while *ReMSA* computes distributed reputation values for each AS to measure the trustworthiness of relationships between neighbors. Therefore, *ReMSA* allows each AS to have its private perception of its neighbors based on the history of interactions and the witness information rather than to rely on global computed values.

Figure 4 shows average reputation values of ASes computed by *AS-CRED* and *ReMSA* over one month. The non-zero reputation values of ASes are added and averaged from our sub-networks. Similarly, the reputation values of respective nodes from *AS-CRED* were averaged. We normalized values computed from *AS-CRED*

Fig. 2 Reputations computed by *AS-CRED*

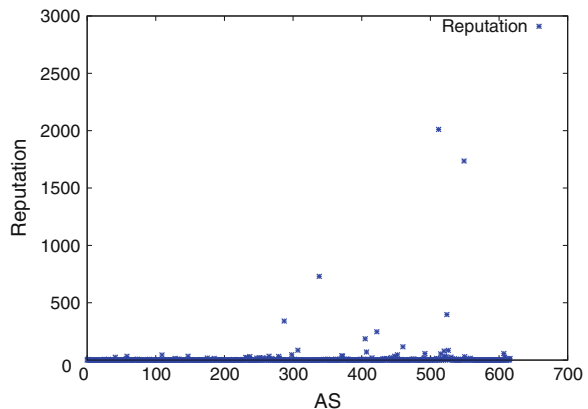


Fig. 3 Reputations computed by *ReMSA*

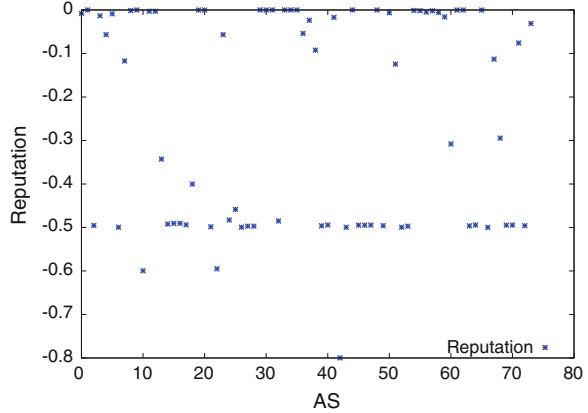
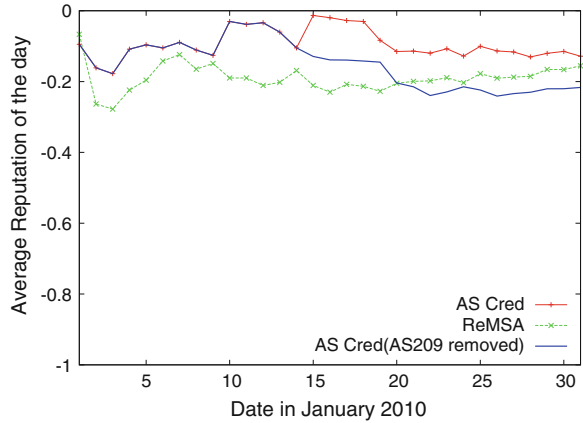


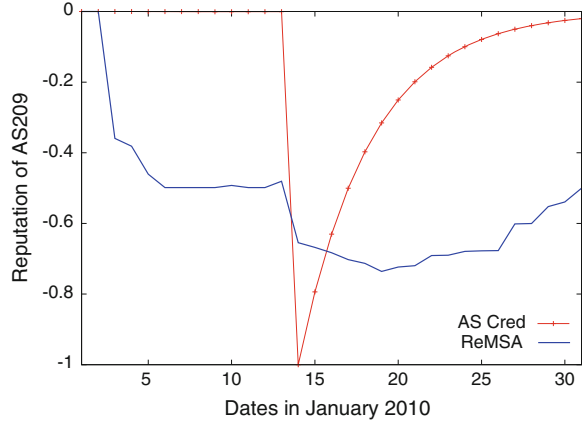
Fig. 4 Reputations computed by *AS-CRED* and *ReMSA*



since zero is the best reputation value and the higher values represent the worse reputation in their algorithm.

The two lines from *AS-CRED* differs in that the one below doesn't include *AS209*, which has an extremely bad reputation with the raw value 2755.52. We investigated the differences shown in Fig. 4 and found out that whenever there are big gaps, e.g., on the 14th day, there was an AS with extremely bad reputation (i.e. *AS209*) in *AS-CRED*. Therefore, when we normalized the reputation values, since the worst reputation value in *AS-CRED* becomes -1 , it makes other reputation values negligibly small. Consequently, the normalized average for *AS-CRED* is smaller than our algorithm's average. For example, on the 14th day, *AS209* received an extremely bad reputation (the raw value 2755.52) when most of other ASes received less than 10. Such a huge difference among reputation values makes other reputation values negligible which enables moderately malicious ASes to get hidden under an extremely malicious AS.

Fig. 5 Reputation values computed by *AS-CRED* and *ReMSA* for *AS209*



Now let's observe *AS209* more closely. Figure 5 shows the reputation value of *AS209* computed by *AS-CRED* and our algorithm. In the figure, we normalized reputation values from *AS-CRED* with the largest value *AS209* had, which was on January 14th, 2010. *AS209* had bad behaviors before the 14th, but because of the huge variances among the reputation values over time in *AS-CRED*, the reputation values of *AS209* on other days except the 14th became almost zero after normalization. *AS-CRED* may be useful to identify the most malicious AS, but it may lose other important information such as how reputation value changes over time.

5.2 Experiment 2: Propagation of Reputation on Random Networks

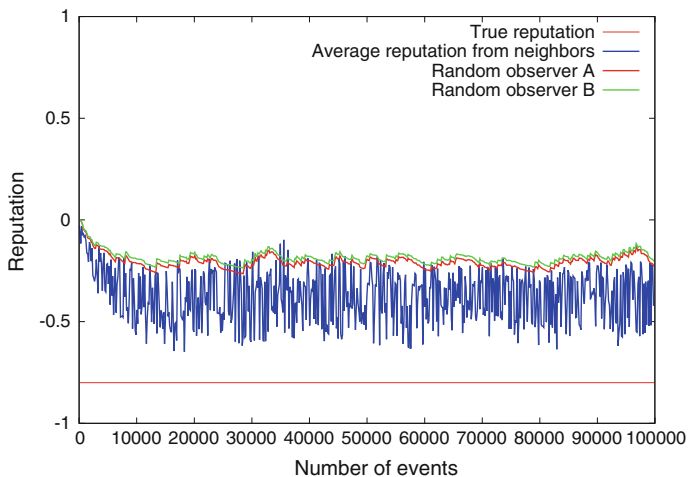
In addition to the performance evaluations on a real social network (ASN) presented in Sect. 5.1, we test the propagation of reputation values in random networks. We study a *sparse* and *adense* network in order to show how topology of the networks (degrees of nodes) influence propagation of information (reputation values). Table 3 shows the statistics of the two random networks.

For each network, we pick an observee node, a_0 , and observe how reputation computed by its neighbors change over time. We also pick two observers distant from a_0 , say A and B , in order to show a_0 's reputation values obtained by each observer. Note that each observer will obtain a subjective reputation value about a_0 . We generated random events that are associated with time information, an observee node and the evaluation of event. Each node has behavioral probability (positive or negative) and the observer evaluates behaviors of observee nodes.

The straight line in Fig. 6 shows the true behavior of a_0 based on its behavioral probability, p_{a_0} . We define p_{a_0} as the probability that a_0 's behavior is evaluated to 1.

Table 3 Statistics of two random networks

	Sparse network	Dense network
# Nodes	5,000	5,000
# Edges	10,000	50,000
Average degree	4	20
Density	0.001	0.004
Diameter	11	4
Average path length	6.624	3.129

**Fig. 6** Propagation of reputation in a sparse network

In this case, the probability was 0.1 and therefore the true reputation (true reputation $= 2 * p_{a_0} - 1$) is -0.8 since the reputation value is between -1 (when $p_{a_0} = 0$) and 1 (when $p_{a_0} = 1$). The neighbors have interactions with a_0 and update reputations based on the probabilistic random behaviors of a_0 . The average reputation values of a_0 computed by the neighbors lie right above the true reputation value in Fig. 6. The two other lines on top represent the reputations values seen by A and B . For each time the neighbors' reputation values of a_0 are computed, A and B query reputation of a_0 using *Ask* function presented in Sect. 4.7. As we can see in Fig. 6, it is not hard to believe that direct interactions with a_0 help compute more accurate reputation values of a_0 compared to the reputation values received only by collecting opinions from other nodes. Also we can see that the changes in reputation values become more stable as nodes learn a_0 's true behaviors and share subjective reputation values of a_0 through voting processes. Since A is 4-hop-away from a_0 and B is 6-hop-away from a_0 , we can see that A has closer values to the true value than B .

We repeat the process on the dense network and the results are shown in Fig. 7. We set the behavioral probability of the observee, say a_1 , the same. A and B both were

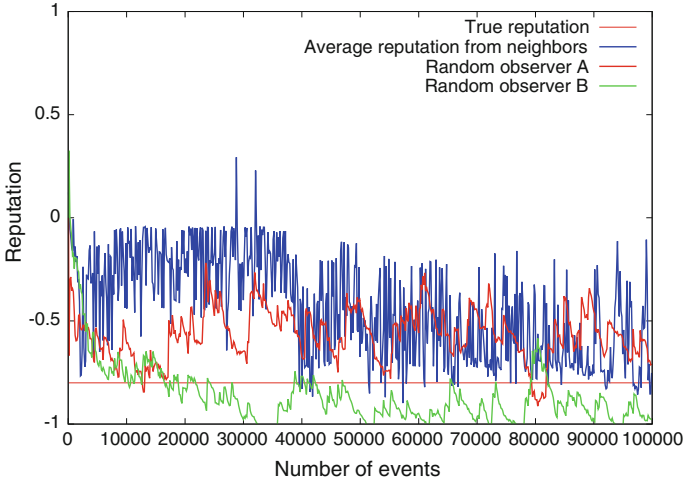
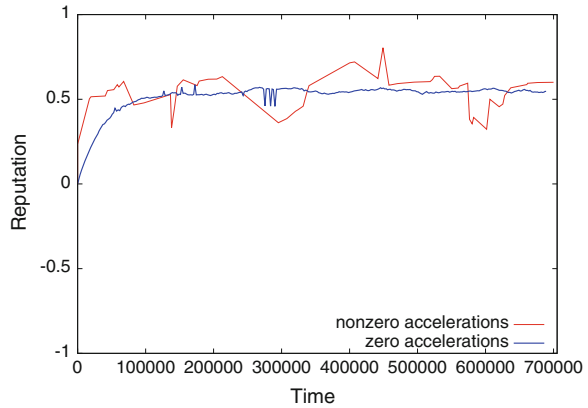


Fig. 7 Propagation of reputation in a dense network

Fig. 8 Reputations computed with constant and increasing velocity



3-hop-away from a_1 . The average reputation values computed by a_1 's neighbors converge closer to the true value. On the dense network, reputation values seen by the two observers fluctuate because, as shown in Table 3, the network is so dense and the reputation of the target agent is influenced by many different agents through *Ask Function*.

In Fig. 8, we show how velocity of events influence reputation values. As discussed in Sect. 4.4, the *Impact* function adjusts the computed feedback values based on the acceleration of the event. On a random network, we pick a node with behavioral probability 0.8 and observe reputation values from a neighbor changing over time when the velocity of interactions is constant and when the velocity of interactions increases. As we can see in Fig. 8, the reputation computed without acceleration becomes stable as it reaches close to the node's true reputation, 0.6, while the reputa-

tion computed with nonzero accelerations fluctuates more. Since the *Impact function* emphasizes the influence of accelerations of events, the reputation values become more sensitive to the current feedbacks when the rate of events is more frequent.

5.3 Effects of Voting, Time Decaying Function and Impact Function

In this Section, we show the effect of each mechanism in the reputation update formula, introduced in Sect. 4.6. We create a scale-free network using *Albert-Barabási* algorithm with 5,000 nodes and 50,000 edges. This network was used in the following experiments. We discuss the results of experiments using a sample agent picked randomly which represents the typical behavior of agents in general. The reputation values are computed from the neighbors of the sample agent.

5.3.1 Effects of Voting

Voting process is introduced in Sect. 4.2.2. Through the voting, one can aggregate others’ opinion so that the computed reputation values are objective. The balance between direct experiences (self opinion) and indirect experiences (others’ opinions) is automatically controlled by the degree of each agent as discussed in Sect. 4.2.2.

Figure 9 shows the reputation computed with and without the voting process which means that the *feedback* function is replaced by *Evaluation* (self opinion) only. The straight line is the true reputation of an agent. The reputation values computed with

Fig. 9 Effects of Voting

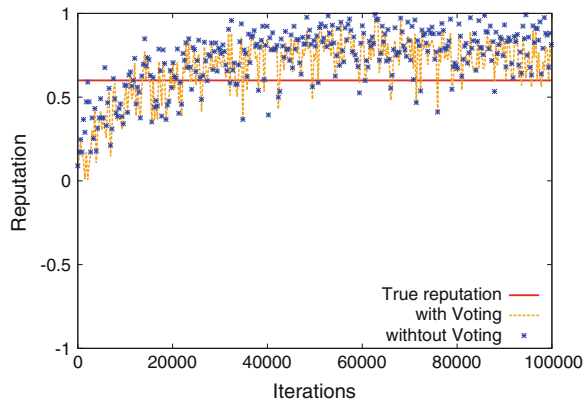


Table 4 Average distance from the true reputation

Node	1453	4999	3387	4102
Degree	10	11	33	57
With voting	0.247	0.187	0.156	0.142
Without voting	0.311	0.291	0.234	0.175

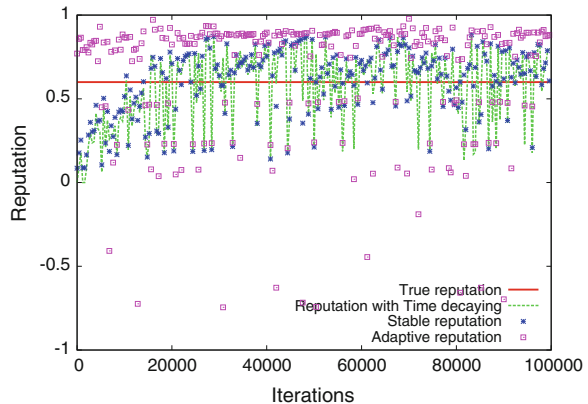
voting, which considers others' opinions, are closer to the true reputation value compared to the reputation values computed without voting. Table 4 shows the average distance from the true reputation value over the iterations for four sample nodes. The average distance from the true reputation is lower when the reputation values are computed with voting. We also observe that as the degrees of node increases, the average distance from the true reputation decreases since having more neighbors lead to getting more opinions.

5.3.2 Effects of Time Decaying Function

The time decaying function, discussed in Sect. 4.5, utilizes the frequencies of interactions to balance the current reputation and the feedbacks. In some applications, the steady-state reputation is more valuable, while in other cases, reputation values need to be adaptive so that they reflect the up-to-date information. In *ReMSA*, this is automatically controlled by the time decaying function.

Figure 10 shows four different reputation values. The straight line shows the true reputation value of the agent under observation. The line shows reputation values computed using standard *ReMSA*. We also show steady-state reputation values as well as adaptive reputation values, plotted with square points and star points respectively.

Fig. 10 Effects of time decaying function



As discussed before, new reputation is computed as a weighted sum of current reputation and new feedback. Steady-state reputation has more weight on current reputation while adaptive reputation has more weight on new feedback. For the comparison, we weight current reputation with 0.9 and new feedback with 0.1 for steady-state reputation and vice versa for adaptive reputation. Intuitively, if the current reputation is weighted more than the new feedback, the reputation value is stable meaning it does not fluctuate much. On the other hand, if the new feedback is weighted more than the current reputation, the reputation value is more adaptive since the updated reputation value reflects more of the current behavior of an agent than the history of the past behaviors. As shown in the Fig. 10, adaptive reputation values are distributed near 1, 0, or -1 , which are the raw feedback values. Steady-state reputation values are distributed near reputation values computed by standard *ReMSA*; this is because the interactions are randomly generated and the frequencies of interactions don't vary too much.

5.3.3 Effects of Impact Function

The purpose of *Impact function* is to reflect accelerations of repeated interactions to the *feedback*. In real social networks, repeated interactions in a short period of time may imply greater closeness of two entities involved. Therefore, we emphasize sudden increases of interactions rate using *Impact function*. In Fig. 11a, reputation values of interaction with increasing accelerations are shown and in (b), reputation values of interactions with decreasing accelerations are shown. Since the time information of the interactions are randomly generated integers, the acceleration values are small. For example, if three interactions with time 10, 20, 25 occurs, the velocity of the second and third interactions are 0.1 and 0.2 and the acceleration of the third interaction is 0.1. Therefore, even the acceleration of the third interaction is positive and emphasized by *Impact function*, the difference is not big. With positive accelerations, positive *feedback* values are rewarded and negative *feedback* values are punished according to the acceleration. In Fig. 11a, reputation values below the red line show that reputation values computed with *Impact function* are lower than the ones computed without *Impact function*. Since the acceleration is positive, negative feedbacks were punished (decreased) by *Impact function*. On the other hand, in Fig. 11b, reputation values of interactions with negative accelerations are shown. Reputation values below the red line shows that reputation values computed with *Impact function* are higher than the ones computed without *Impact function* since negative *feedbacks* with decreasing accelerations are rewarded (increased) according to the acceleration values.

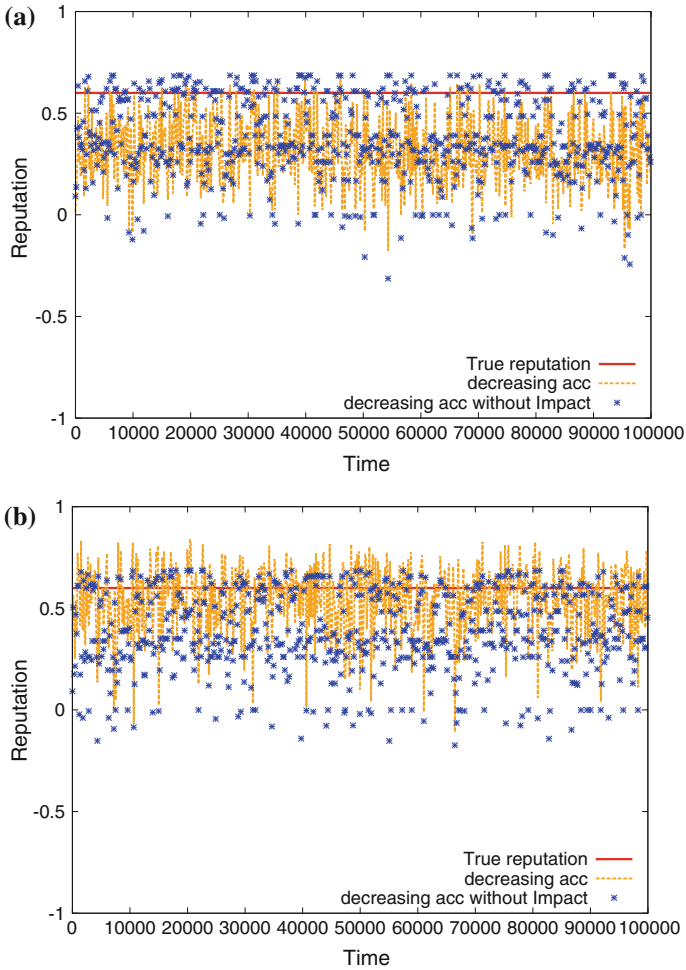


Fig. 11 Effects of Impact function.

6 Discussions

A natural way of measuring an agent’s reputation is to accumulate interaction history in some form of weighted sum. We started with this idea and incorporated frequency and velocity of interactions as well as the topological information of the network. Generally, reputation computation takes an approach that combines the previous reputation and new experience.

Our algorithm assumes that each agent is aware of its neighbors and its position in the network and makes use of the information. Therefore if the topology of network changes, an agent perceives its new sociological information and its reputation

updating function changes accordingly. This is explained by human behavior; people tend to act with more responsibly when there are more observers (i.e., a higher degree). Instead of using neighbor's reputation value as a weight (as in *HISTOS*), we take advantage of neighbor's relative degree as a weight which is more objective. Also, in our algorithm, when an agent takes a vote of its neighbors, the neighbors can recursively take votes of their neighbors.

Through the experiments, we show that our algorithm can compute quality reputation values by comparing the results with an existing reputation computation algorithm (*AS-CRED*). The algorithm can also successfully model propagation of reputations within the network. We show that in a dense network, reputations travel much quicker and diffuse wider given the same number of interactions. In addition, we show how frequencies of interactions can influence the reputation values. Since a higher rate of interactions implies greater significance, we believe that the velocity of interactions is an important parameter in our algorithm.

Since *ReMSA* is designed for distributed environments, the algorithm is employed within each agent. Therefore the time complexity of *ReMSA* for each agent is dependent on the number of events and the number of neighbors each agent has. Then the time complexity of the algorithm is $O(d_i * |E_i|)$ for each agent i .

7 Conclusions and Future Work

In this paper, we developed a new reputation computation algorithm, *ReMSA*, in which reputation values are considered subjective and reputation computation utilizes the topological information of the given network as well as velocity of interactions among agents. *ReMSA* also employs a distributed voting process to overcome biases that an agent might have towards others. To our best knowledge, no reputation computation algorithm considers velocity of events. Instead, most of the algorithms use the total number of interactions occurred. We believe that by taking velocity into consideration, it is possible to detect some abnormal activities since they tend to happen in a short period of time.

We use some of the degree distribution of the given network to take advantage of the topological information of the network. Many studies that involve social networks assume the second degrees (degrees of the neighbors) are known either because the degree distribution of agents contains high information. We are currently relaxing this assumption and trying to estimate the second degree distribution using bayesian probabilistic methods. We assume that each agent start the estimation with the same degree of its own [6]. The degree distribution of a given social network is assumed to follow a power-law distribution and we let each agent apply power-law to the estimated second degree distribution. By estimating the second degrees we do not rely on the information from outside since each agent can estimate all the information needed which will also prevent possible deceptions from outside sources.

Our algorithm is an online algorithm that can be deployed to observe ASes in real-time and can compute reputations of ASes as in *AS-CRED* [13]. According

to [19], centralized methods have limitations in BGP security and no solution has yet provided complete security. Unlike *AS-CRED* [13], *ReMSA* incorporates social dimensions and velocity of events that are important features of the *AS* network.

Also, as *ReMSA* was originally developed for social networks, we plan to apply it to more general social networks where reasoning about private friendship is an important value.

References

1. Carbo J, Molina JM, Davila J (2003) Trust management through fuzzy reputation. *Int J Coop Inf Syst* 12(01):135–155
2. Sabater J, Sierra C (2001) Regret: reputation in gregarious societies. In: Proceedings of the fifth international conference on autonomous agents. *AGENTS '01*. ACM Press, New York, pp 194–195
3. Zacharia G (2000) Trust management through reputation mechanisms. *Appl Artif Intell* 14:881–907
4. Lee JY, Oh JC (2013) A model for recursive propagations of reputations in social networks. In: Proceedings of the 2013 IEEE/ACM international conference on advances in social networks analysis and mining. *ASONAM '13*. ACM Press, New York, pp 666–670
5. Galan J, Latek M, Rizi S (2011) Axelrod's metanorm games on networks. *PLoS One* 6(5):e20474
6. Galeotti A, Goyal S, Jackson MO, Vega-Redondo F, Yariv L (2010) Network games. *Rev Econ Stud* 77(1):218–244
7. Szabo G, Fath G (2007) Evolutionary games on graphs. *Phys Rep* 446(4–6):97–216
8. Pinyol I, Sabater-Mir J (2011) Computational trust and reputation models for open multi-agent systems: a review. *Artif Intell Rev*, 40(1):1–25
9. Huynh TD, Jennings NR, Shadbolt NR (2004) Fire: an integrated trust and reputation model for open multi-agent systems. In: Proceedings of the 16th european conference on artificial intelligence (ECAI), pp 18–22
10. Teacy WTL, Patel J, Jennings NR, Luck M (2006) Travos: trust and reputation in the context of inaccurate information sources. *J Auton Agents Multi-Agent Syst* 12:2006
11. Xiong L, Liu L (2004) Peertrust: supporting reputation-based trust for peer-to-peer electronic communities. *IEEE Trans Knowl Data Eng* 16:843–857
12. Lee J, Duan Y, Oh JC, Du W, Blair H, Wang L, Jin X (2012) Social network based reputation computation and document classification. *j-jucs* 18(4):532–553
13. Chang J, Venkatasubramanian KK, West AG, Kannan S, Lee I, Loo BT, Sokolsky O (2012) *As-cred*: reputation and alert service for interdomain routing. *IEEE J Syst*, 1:99
14. University of Oregon RouteViews project. <http://www.routeviews.org/>
15. Graph-generator. <https://code.google.com/p/graph-generator/>
16. Leskovec J, Faloutsos C (2006) Sampling from large graphs. In: Proceedings of the 12th ACM SIGKDD international conference on knowledge discovery and data mining. *KDD '06*, ACM Press, New York, pp 631–636
17. Chang J, Venkatasubramanian K, West A, Kannan S, Loo B, Sokolsky O, Lee I (2011) *As-trust*: a trust quantification scheme for autonomous systems in bgp. In: McCune J, Balacheff B, Perrig A, Sadeghi AR, Sasse A, Beres Y (eds) *Trust and trustworthy computing*. Lecture notes in computer science, vol 6740. Springer, Heidelberg, pp 262–276
18. The CAIDA AS Relationships dataset, January 1, 2010—January 31, 2010. <http://www.caida.org/data/active/as-relationships/>
19. Butler K, Farley T, McDaniel P, Rexford J (2010) A survey of bgp security issues and solutions. *Proc IEEE* 98(1):100–122

Measuring Centralities for Transportation Networks Beyond Structures

Yew-Yih Cheng, Roy Ka-Wei Lee, Ee-Peng Lim and Feida Zhu

Abstract In an urban city, its transportation network supports efficient flow of people between different parts of the city. Failures in the network can cause major disruptions to commuter and business activities which can result in both significant economic and time losses. In this paper, we investigate the use of centrality measures to determine critical nodes in a transportation network so as to improve the design of the network as well as to devise plans for coping with the network failures. Most centrality measures in social network analysis research unfortunately consider only topological structure of the network and are oblivious of transportation factors. This paper proposes new centrality measures that incorporate travel time delay and commuter flow volume. We apply the proposed measures on the Singapore's subway network involving 89 stations and about 2 million commuter trips per day, and compare them with traditional topology based centrality measures. Several interesting insights about the network are derived from the new measures. We further develop a visual analytics tool to explore the different centrality measures and their changes over time.

Keywords Network centrality · Commuter flow centrality · Delay flow centrality · Visual analytics · Transportation network

Y.-Y. Cheng · R.K.-W. Lee (✉) · E.-P. Lim · F. Zhu
Living Analytics Research Centre, Singapore Management University,
Singapore, Singapore
e-mail: roylee.2013@phdis.smu.edu.sg; roy.lee.2009@sis.smu.edu.sg

Y.-Y. Cheng
e-mail: ycheng.2012@mais.smu.edu.sg

E.-P. Lim
e-mail: eplim@smu.edu.sg

F. Zhu
e-mail: fdzhu@smu.edu.sg

1 Introduction

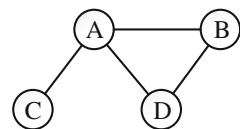
Motivation Transportation network is an important part of a complex urban city system. Each transportation network is expected to support efficient flow of people between different parts of the city. Failures in the network can cause major disruptions to commuter and business activities which result in both significant economic and time losses. As the transportation network continues to grow and interact with other parts of urban city, it is imperative to study how the network can cope with increase in human flow, new network nodes and connections, as well as new city developments.

One way to study the transportation network is to identify the *centralities* in the network which represent the more critical nodes whose degrees of reliability have major impact to the network efficiency. Network centrality is a concept introduced in social science to analyze important nodes in social networks [8, 13]. The key examples of network centrality measures include degree centrality, closeness centrality, betweenness centrality [7], and pagerank [3]. In a recent study by Derrile [5] on metro networks represented by transfer stations, terminal stations and their connections only, it was shown that betweenness is more evenly distributed among stations when the metro network is large. This hopefully will allow the stations to share commuter load more equally.

All the above traditional network centrality measures nevertheless consider network topology only but not factors associated with transportation. A node identified to be important topologically does not need to be important from the commuter flow and delay perspective. Consider the network example in Fig. 1, node *A* has the highest degree, closeness and betweenness values by topology. If we know that the commuter flow between *B* and *D* far exceeds those of other connections, *B* and *D* should be deemed to be more central than the other nodes. If we further know that many commuters will take much longer travel time should *B* fails, one may even consider *B* to be more central than *D*.

Research Objectives and Contributions In this paper, we therefore aim to devise new centrality measures that incorporate *commuter flow* and *travel time delay*, the two transportation related factors. The objective of our research is to determine critical nodes in a transportation network so as to improve the design of the network as well as to devise plans for coping with the network failures. Unlike network topology, commuter flow and travel time delay may change dynamically with time, allowing us to study the evolving node importance. This time-dependent approach to measure node importance permits us to study a transportation network system in a much finer time granularity. As a result, the insights gained can also be readily associated with time in addition to topology.

Fig. 1 Example network



In the following, we summarize the contributions of this paper:

- We propose *Commuter Flow*, *Time Delay* and *DelayFlow* centrality measures that incorporate commuter flow and travel time delay unique to transportation networks. To compute these measures, we develop methods to compute commuter flow and travel time delay from a network with trip transaction data and a public web service that offers travel time information respectively.
- We apply the proposed measures on the Singapore's subway network involving 89 stations and more than 2 million commuter trips per day, and compare them with traditional centrality measures. Both time-independent and time-dependent analyses of the various centrality measures are conducted.
- We derived several interesting insights about the network using the new measures. Our experiments show our proposed centrality measures are different from the traditional centrality measures. The commuter flow and DelayFlow centrality values of stations can vary a lot throughout a day. They can also be very different between weekdays and weekends. This also justifies the usefulness of the new measures.
- Finally, we have developed a visual analytics tool to evaluate the newly proposed centrality measures and other traditional ones. The tool allows users to select centrality measure, time of the day and other factors to be analyzed easily. The visual presentation gives users a very friendly and quick way to examine and compare the different centrality measures.

Paper Outline The rest of the paper is organized as follows. Section 2 describes several works related to our study. Section 3 introduces our proposed centrality measures. We describe the Singapore subway network dataset in Sect. 4. Our experiments on the various measure are covered in Sect. 5. We present our visual analytics tool in Sect. 6 before concluding the paper in Sect. 7.

2 Related Work

Transportation network analysis has been an active area of research. De Montis et al. [4], conducted an analysis of an interurban transportation network involving 375 municipalities and 1.6 million commuters. In their work, a weighted network is used to model the traffic flow between municipalities and several properties, e.g., weighted degree distribution, cluster coefficients, heterogeneity of commuter flows, etc., about the network are derived. Berche et al. also studied the resilience of transportation networks due to attacks on nodes [1]. In the context of air transportation, Guimera et al., found the worldwide air transportation network demonstrating scale-free small world properties [11]. This finding is interesting as this illustrates that transportation networks may evolve in an organic way similar to social networks [12].

For the purpose of city planning and building design, Sevtsuk and Mekonnen proposed a set of network centrality measures including *Reach*, *Gravity Index*, and *Straightness* in addition to the traditional ones to determine how easy one can reach other locations in a city from a given location [15]. In this paper, we however focus

on centrality measures relevant to transportation networks where time delay and commuter flow are the major factors.

There are very few works on applying network centrality measures to analyze transportation networks. Derrible studied the betweenness centrality in several subway networks of different countries [5]. His work however focuses on transfer and interchange stations and does not introduce new centrality measures considering transportation related factors. Our work is quite similar to that of Scheurer et al. [14], who also proposed a set of centrality measures for transportation networks. Unlike ours, their measures do not consider commuter flow and travel time of alternative means between stations.

In the literature, there is a large body of works on the robustness of networks which include transportation networks [2]. Goh et al. [10], found out that the betweenness centrality of scale-free networks follows power law distribution with an exponent that determines the robustness of the network. Gao et al. [9], proposed to use graph energy to measure robustness of very large networks in an efficient way. These works however do not use network centrality to identify important stations when failures occur.

3 Centrality Measures

In this section, we first provide the definitions of the traditional network centrality measures. This is followed by our proposed centrality measures which take two factors into consideration: (i) commuter flow of the node, and (ii) the amount of time delayed that will incur due to failure of the node.

3.1 Overview of Network Centrality

We model a *transportation network* as an undirected graph $\langle V, E \rangle$ with node set V representing stations and edge set E representing the connections between stations. Every node $i \in V$ is associated with two numbers, $in(i)$ and $out(i)$, that refers to the number of commuters entering and exiting the station of node i respectively. Given that the total numbers of commuters entering and exiting the stations of the network are the same, the equality $\sum_{i \in V} in(i) = \sum_{i \in V} out(i)$ holds.

We now review the definitions of some existing network centrality measures used in social network analysis.

Degree Centrality The degree centrality of a node i , $C_{deg}(i)$, is defined as:

$$C_{deg}(i) = |\{(i, j) | (i, j) \in E\}|$$

Closeness Centrality We denote the shortest path distance between two nodes i and j by $d(i, j)$. The closeness centrality of a node i , $C_{cls}(i)$, is defined as:

$$C_{cls}(i) = \frac{1}{d(i)}$$

where $d(i) = \sum_{j \in V, j \neq i} d(i, j)$.

Betweenness Centrality Let g_{jk} denote the number of shortest paths between nodes j and k , $g_{jk}(i)$ denote the number of shortest paths between nodes j and k through node i . The betweenness centrality of a node i , $C_{brw}(i)$, is defined as:

$$C_{brw}(i) = \sum_{j \in V} \sum_{k \in V, k > j} \frac{g_{jk}(i)}{g_{jk}}$$

3.2 DelayFlow Centrality

Commuter Flow Centrality The commuter flow centrality of a node i , $C_f(i)$, is defined as the number of commuters affected per hour when node i is down. We classify the affected commuters into three categories:

- Commuters traveling from node i to other nodes;
- Commuters traveling from other nodes to node i ; and
- Commuters traveling through node i .

Hence, we define the commuter flow centrality of a node i to be:

$$C_f(i) = \sum_{j \in V} h_{ij} + \sum_{j \in V} h_{ji} + \sum_{j \in V, j, k \neq i, j \neq k} h_{jk}(i)$$

where h_{ij} denotes the number of commuters from node i to node j per hour, and $h_{jk}(i)$ denotes the number of commuters from node j to node k through node i per hour. In Sect. 4, we will describe how the two sets of commuter flow data can be derived from trip transactions.

Time Delay Centrality Time delay is incurred for commuters to find alternative means to reach destinations when a node is down. To determine the extent of delay caused to the people affected, we consider the following:

- $t_{exp}(i, j)$: The expected time taken to travel from node i to node j ;
- $t_{alt}(i, j)$: The time taken to travel from node i to node j using an alternative means of transportation (e.g. bus) which varies by the hour.

Assuming that commuters do not receive advance notice about node failures in the transportation, they have to find alternative routes from the failed node to their

final destinations, or from the origins to the failed nodes. Let n be the number of nodes in the network. The time delay centrality for a node i $C_{del}(i)$ is thus defined as:

$$C_{del}(i) = \frac{\sum_{j \in V, j \neq i} l_{ij} + \sum_{j \in V, j \neq i} l_{ji}}{2(n-1)}$$

where

$$l_{ij} = \frac{t_{alt}(i, j)}{t_{exp}(i, j)}$$

In the above definition, we assume that t_{exp} is static and t_{alt} varies throughout the day. The time delay factor l_{ij} is asymmetric as both $t_{exp}(i, j)$ and $t_{alt}(i, j)$ are asymmetric. When the time delay factor l_{ij} equals to 1, it signifies no delay. When l_{ij} is greater than 1, commuters will take a longer than expected time to reach their desired destinations. The larger the l_{ij} value (> 1), the greater the time delay. $l_{ij} < 1$ means an improvement of the alternative means of transportation, which is an unlikely scenario when disruptions occur to the network.

Notice that time delay centrality $C_{del}(i)$ does not take commuter flow into consideration in determining time delay. For example, a commuter may incur a long travel time from node i to node j when using an alternative means of transport over the network (i.e., l_{ij} is large), but only a small number of commuters are actually affected by it. Thus, there is a need to combine time delay factor with the underlying commuter flow volume. We therefore propose the DelayFlow centrality by summing the product of commuter flow and average time delay factor of all paths traveling to, from and through a node.

DelayFlow Centrality The DelayFlow centrality of node i , $C_{dflow}(i)$, is defined as:

$$C_{dflow}(i) = \frac{\sum_{j \in V, j \neq i} h_{ij} l_{ij} + h_{ji} l_{ji} + \sum_{j \in V, j, k \neq i, j \neq k} h_{jk}(i) l_{jk}(i)}{\sum_{j \in V} in(j)}$$

where $\sum_{j \in V} in(j)$ is the total commuter flow of the transportation network. l_{ij} and l_{ji} have been earlier defined and

$$l_{jk}(i) = \frac{t_{alt}(j, k)}{t_{exp}(j, i) + t_{exp}(i, k)}$$

4 Datasets

The Singapore's mass rapid train (MRT) network consists of 89 stations in four train lines. Figure 2 depicts these train lines (in different colors) and stations. The node color represents the different train lines: red for North-South line (NSL), green for East-West line (EWL), purple for North-East line (NEL), orange for Circle line

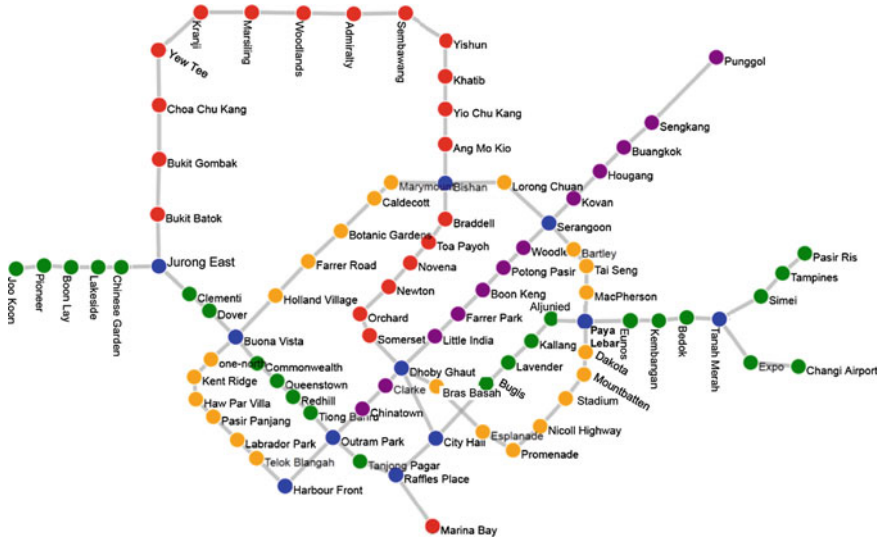


Fig. 2 Singapore MRT Network

(CL), and blue for the interchange stations. We use two datasets about MRT network. The first dataset MRTDB consists of nearly 2 millions commuter trip transactions per day. In our experiment, we used three days worth of trip transaction data from November 26 (Saturday) to November 28, 2011 (Monday) to derive the commuter flow information. The second dataset GOTHEREDB provides data about the travel time information.

MRTDB Dataset Each trip transaction consists of the origin station, destination station and the timestamps at the two stations. We use the trip transactions to derive commuter flow h_{ij} 's. We compute the overall h_{ij} by dividing the total number of trips between stations i and j (not necessarily the origin and destination stations) by the number of MRT operating hours, i.e., 19h (from 0500 to 0000h). In a similar way, we compute $h_{jk}(i)$ from trips between j and k through i .

To study commuter flow centrality and DelayFlow centrality at different time of the day, we divide the trip transactions into 19, one-hour long disjoint segments from 0500 to 0000h. A trip is assigned to a time segment if its origin station's timestamp falls within the time segment. We can then compute the *time specific* commuter flow numbers h_{ij} 's and $h_{jk}(i)$'s for each time segment.

To determine the path a commuter takes to travel from one station to another for determining h_{ij} 's and $h_{jk}(i)$'s, we first assume that all commuters with the same origin and destination stations take the same path and the path should be the shortest among all other path options. We use Dijkstra's algorithm to compute these shortest paths [6]. To represent delays at interchange stations, create k separate proxy stations for an interchange station for k train lines and $\binom{k}{2}$ edges to connect these pairs of proxy stations. Each edge is assigned a 2 min transfer time at the interchange station.

Fig. 3 Commuter flow (weekday)

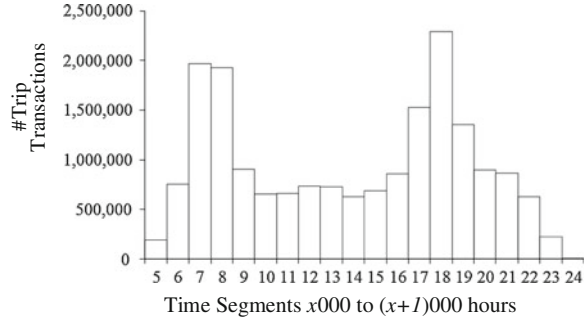
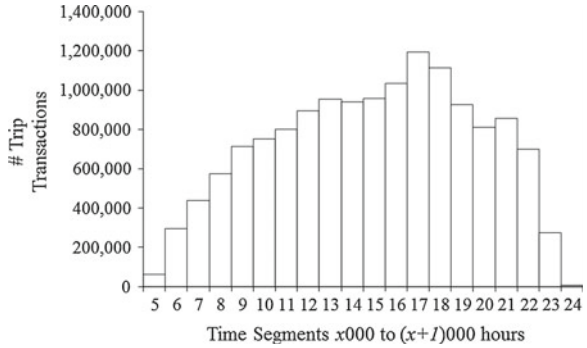


Fig. 4 Commuter flow (weekend)



Figures 3 and 4 show the number of commuters at different 1 h time segments on a weekday and weekend day respectively. They show that the number of commuters peaks on both the morning and evening rush hours on weekdays, but peaks only in the evening on a weekend. The weekday trend indicates that many commuters use MRT trains to get to their work places in the morning, and return home from work in the evening. At the peaks, we have more than two millions commuters using the train network, representing about $\frac{2}{3}$ of the Singapore’s population. On weekends, commuters clearly show a late start of their travel which peaks at around 5pm. For the rest of this paper, we shall just focus on the weekday data.

GOTHEREDB Dataset To determine the expected travel time and travel time using alternative routes, we made use of a third-party route suggestion service known as “*gothere.sg*”.¹ The *gothere.sg* API’s allow us to determine the travel time by MRT train (expected) or bus (alternative) for each origin and destination station pair. Bus is chosen as it is the next most commonly used mode of public transportation. Given that we have altogether 89 stations in the MRT network, we use the APIs to derive $t_{exp}(i, j)$ and $t_{alt}(i, j)$ time for all $89 \cdot 88 = 7832$ station pairs. To keep things simple, we assume that $t_{exp}(i, j)$ ’s and $t_{alt}(i, j)$ ’s are independent of the time of the day.

¹<http://gothere.sg>.

5 Comparison of Centrality Measures

Distribution of Degree, Closeness and Betweenness Centralities Figures 5, 6 and 7 show the distribution of the degree, closeness and betweenness centrality values of the stations in MRT network. As shown in Fig. 5, most nodes have degree = 2 or 1 as they are non-interchange stations. There are only 10 interchange stations with degrees ranging from 3 to 5. The closeness centrality values, as shown in Fig. 6, concentrate within the small range between 0.06 and 0.14. This suggests that the shortest paths between stations have relatively short length. Figure 7 shows the betweenness centrality is well spreaded between 0 to 1300. Only very few stations have betweenness greater than 1000 serving in the middle of many shortest paths between other stations.

Fig. 5 Degree centrality C_{deg} distribution

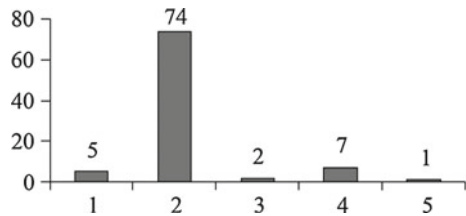


Fig. 6 Closeness centrality C_{cls} distribution

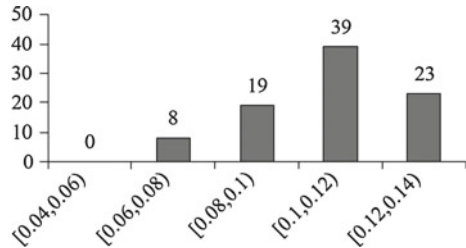
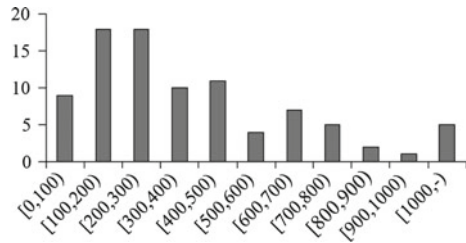


Fig. 7 Betweenness centrality C_{btw} distribution



Distribution of Commuter Flow Centrality Figure 8 shows the commuter flow centrality distribution on a weekday. The centrality values can be as large as 29,362. Most stations (90%) have relatively small commuter flow centrality values with values in [0,20000). Only 9 stations (10%) have larger commuter flow centrality values with values in [20000,30000). Not surprisingly, these are mainly the interchange stations with high volume of commuters. The only exceptions are two stations which are located in densely populated township and highly popular shopping area.

Figure 9 shows the commuter flow centrality distribution at different time durations of a weekday as the centrality. The figure shows that there are more stations having higher commuter flow centrality values in the [0800,0900) and [1800,1900) time segments due to larger concentration of commuters using these stations during that time. Otherwise, most stations have commuter flow centrality values in the [0,20000) range, consistent with what we observed in Fig. 8.

Fig. 8 Commuter flow centrality C_f distribution (weekday)

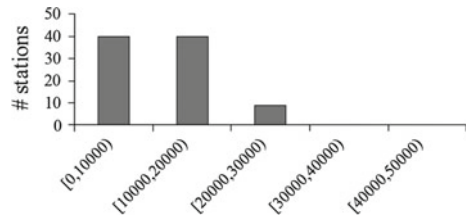
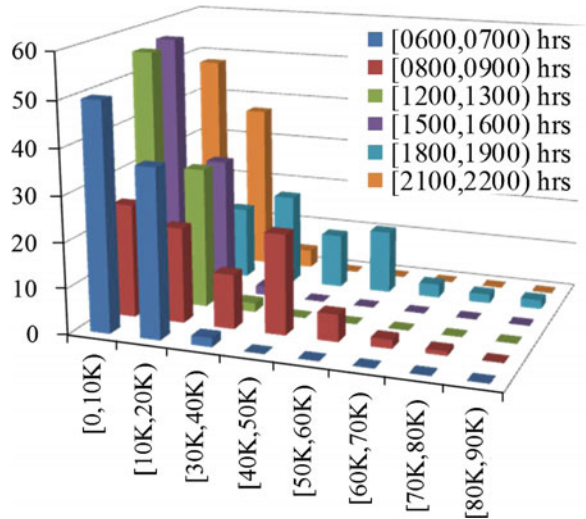


Fig. 9 Commuter flow centrality distribution at selected time segments (weekday)



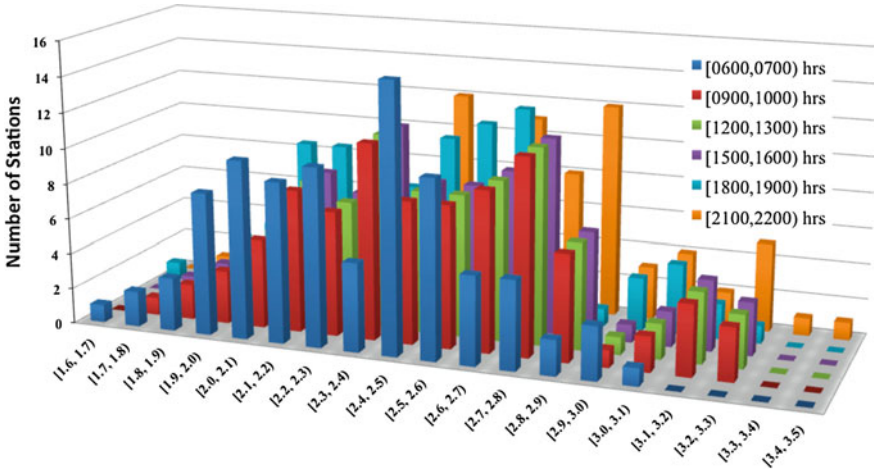


Fig. 10 Time delay centrality C_{del} distributions at selected time segments (weekday)

Distribution of Time Delay Centrality Figure 10 depicts the time delay centrality distribution at different time durations of a weekday as the centrality. The centrality ranges between 1.5 and 3.4 showing that the alternative means of transportation can take between 50 to 240% more than the expected time required. This result shows that the alternative means of transportation for the stations with high time delay centrality can be further improved. The average time delay centrality is 2.395.

Distribution of DelayFlow Centrality Figure 11 depicts the distribution of DelayFlow centrality on a weekday. There are a cluster of stations with small DelayFlow centrality values between 0.0 and 0.1. Another major cluster of stations have DelayFlow centrality values between 0.2 and 0.4. There are very few stations with high centrality values. Figure 12 shows the distribution of DelayFlow centrality at different selected time segments of a weekday. We observe that the two peaks of the overall centrality distribution are contributed largely by more busy stations during the busy morning and evening hours (i.e., [0600,0700), [0900,1000), [1800,1900), and [2100,2200) h). During the non-peak hours, there are more stations with smaller DelayFlow centrality values making the two peaks less obvious.

Correlation Between Centrality Measures Table 1 shows the Pearson Correlation scores between the different centrality measures. The table shows that among the traditional centrality measures based on network topology, degree centrality and betweenness centrality are more similar than with closeness centrality. The nodes with high closeness are more likely be near the center of the network while high degree and betweenness nodes may be located away from the center.

Among the new centrality measures based on travel time and commuter flow, the commuter flow and DelayFlow centrality are more similar with degree centrality and

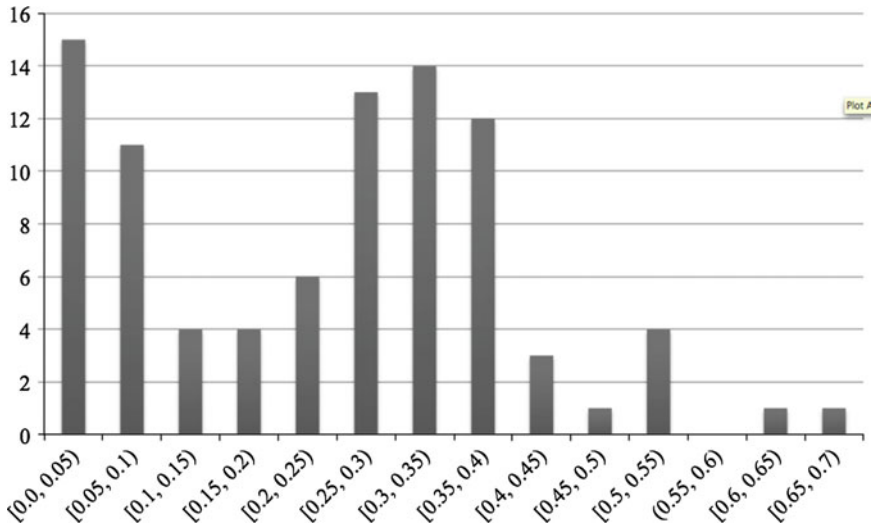


Fig. 11 DelayFlow centrality C_{dflow} distribution (weekday)

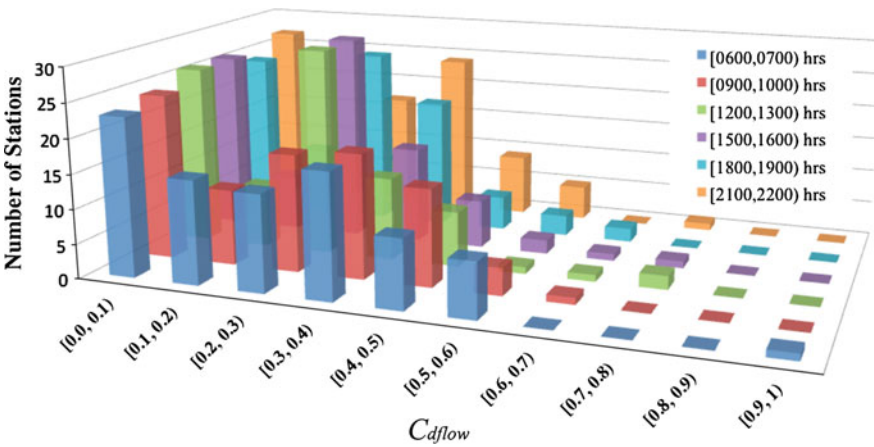


Fig. 12 DelayFlow centrality distributions at selected time segments (weekday)

betweenness centrality with correlation scores above 0.5. The time delay centrality is quite different from the rest except closeness centrality. This suggests that those station that have large time delay to other stations are likely to be those near the center of the network. This is later verified by the top 10 stations of each centrality measure in Table 2. We also observe that DelayFlow centrality is highly similar to commuter flow centrality measure with a correlation score of 0.97. Again, we can also see this in Table 2.

Table 1 Pearson correlation between centrality measures

	C_{deg}	C_{cls}	C_{btw}	C_f	C_{del}	C_{dflow}
C_{deg}	1.0	0.42	0.67	0.63	0.41	0.64
C_{cls}	–	1.0	0.62	0.40	0.57	0.39
C_{btw}	–	–	1.0	0.57	0.45	0.52
C_f	–	–	–	1.0	0.28	0.97
C_{del}	–	–	–	–	1.0	0.38
C_{dflow}	–	–	–	–	–	1.0

Table 2 Highest centrality stations

Rank	C_{deg}	C_{cls}	C_{btw}	C_f	C_{del}	C_{dflow}
1	S10	S3	S3	S11	S110*	S10
2	S3	S106	S106	S10	S106	S11
3	S11	S203*	S20	S15	S203*	S24
4	S12	S10	S35	S3	S10	S3
5	S15	S205*	S203*	S12	S201*	S15
6	S20	S11	S10	S24	S4*	S12
7	S35	S4*	S15	S20	S3	S106
8	S106	S107*	S64*	S21*	S107*	S2*
9	S24	S113*	S201*	S31*	S111*	S20
10	S34	S201*	S21*	S35	S103*	S5*

In Table 2, the top stations ranked by most centrality measures are usually the interchange stations. The non-interchange stations are annotated with “*”. Only the time delay centrality gives highest score to a non-interchange station, S110. It also ranks several other non-interchange stations highly. These stations are expected to have large delay factor values compared with other stations.

6 Visual Analytics of Centrality Measures

In this research, we also develop a visual analytics tool to feature the stations with different centrality measures, and to explore the changes to centrality value over time. Figure 13 shows the main web interface of the visual analytics tool. The figure shows the layout of station nodes of the Singapore’s MRT network. We show the nodes with size proportional to the selected centrality values of the nodes—the bigger the node, the higher the centrality. User can select one of the six centrality measures to be analyzed, namely, Degree Centrality, Closeness Centrality, Betweenness Centrality, Commuter Flow Centrality, Time Delay Centrality and DelayFlow Centrality. For

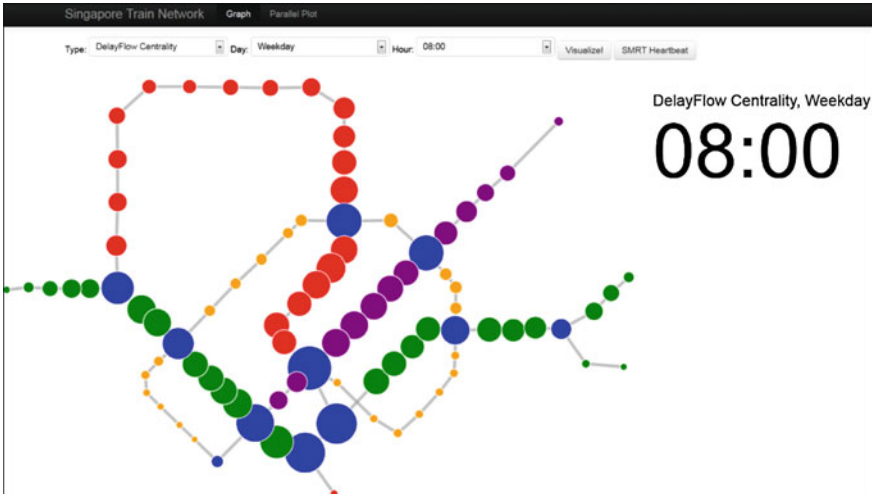


Fig. 13 Visualization of centralities

the three new centrality measures, we can also visualize them for a selected day of the week and for specific hour of the day.

The visual analytics tool can also show a parallel plot to depicts different centrality values of every station represented by a polyline as shown in Fig. 14. One can explore and compare the DelayFlow centrality against the other centrality measures. In addition, information pertaining to the respective train stations will also be displayed on the top right-hand corner when the end-user performs a mouse-over action on the corresponding line in the parallel plot. Lastly, the table on the bottom right-hand corner shows the top 20 train stations with the highest DelayFlow centrality values.

To visualize the dynamic changes of commuter flow and DelayFlow centrality, the tool provides a continuous rendering of the centrality values for the network from 0500 to 0000 h. In this way, a quick overview of the importance of stations over time can be obtained. As shown in Fig. 15, in a selected weekday, stations in the west have higher DelayFlow values (or more importance) as commuters begin to travel to the central and east areas for work around 0600h. Failures at these stations can also cause major delays. At around 0900h, stations in the central area gain higher DelayFlow values due to increased activities in the area, while stations in the west see their DelayFlow values reduced. At around 1200h, all stations appear to have smaller DelayFlow values. At around 1800h, commuters begin to travel to the west causing some increase to the DelayFlow of relevant stations. These stations see their DelayFlow values return to normal at around 2100h. The above dynamic changes of centrality are very useful when developing response plan for network failures as the stations give different impact to the transportation system at different time of the day.

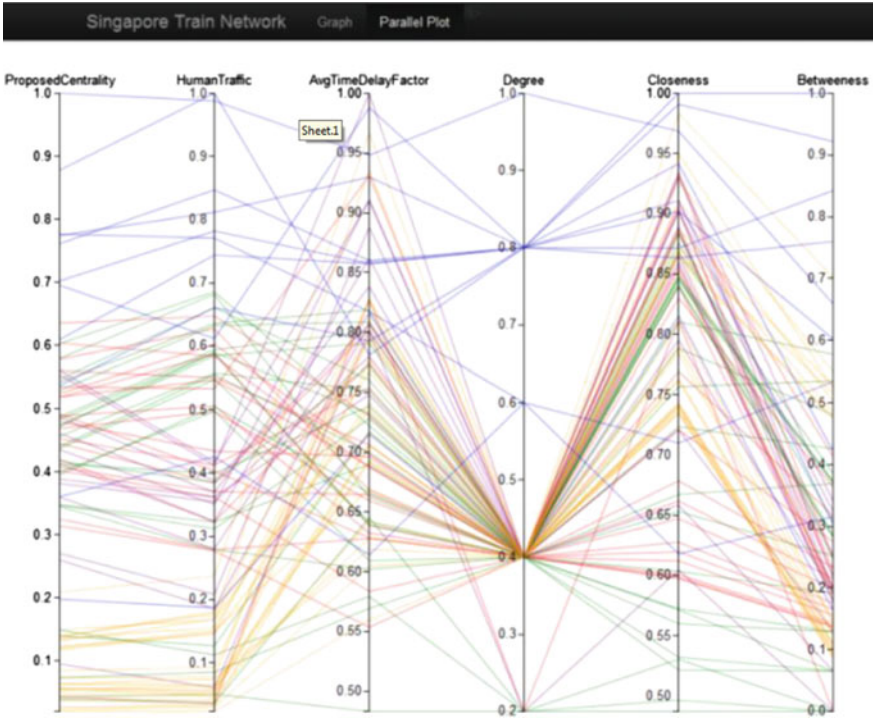


Fig. 14 Parallel plots of centrality

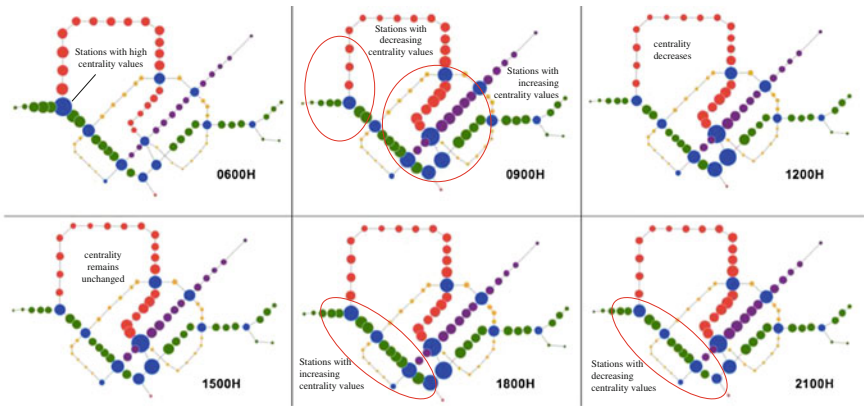


Fig. 15 Dynamic change of DelayFlow centrality

7 Conclusion

In our paper, we have demonstrated the importance of considering transportation factors such as commuter flow and time delay in measuring network centrality of transportation network. When incorporating trip data and time information, we derive three new centrality measures which can dynamically vary with time. These dynamic centrality measures allow us to generate unique insights giving better guidance to the design and improvement of the transportation network. They can also be extremely useful for optimizing travel schedules for commuters and increasing the standard of transportation services. Compared with the network topology based centrality measures, our new centrality measures are more relevant to the transportation domain in identifying critical nodes. A visual analytics tool has also been developed to illustrate the efficacy of the new centrality measures.

With this foundation research, we plan to apply the measures on other transportation networks to explore some common properties among the network centralities. We have so far assumed that the expected and alternative travel times are independent of the time of the day in our centrality definitions. This assumption will be relaxed in the future by computing the travel times at different time of the day. Relationship between the new centralities and the underlying population distribution, as well as extensions to consider other modes of transportation are also among the interesting topics for future research.

Acknowledgments We would like to thank the Land Transport Authority (LTA) of Singapore for sharing with us the MRT dataset. This work is supported by the National Research Foundation under its International Research Centre@Singapore Funding Initiative and administered by the IDM Programme Office, and National Research Foundation (NRF).

References

1. Berche B, von Ferber C, Holovatch T, Holovatch Y (2009) Resilience of public transport networks against attacks. *Eur Phys J B* 71(1):125–137
2. Boccaletti S, Latora V, Moreno Y, Chavez M, Hwang D (2006) Complex networks: structure and dynamics. *Phys Rep* 424(4–5):175–308
3. Brin S, Page L (1998) The anatomy of a large-scale hypertextual web search engine. In: *The seven international conference on world wide web*, pp 107–117
4. De Montis A, Barthelemy M, Chessa A, Vespignani A (2005) The structure of inter-urban traffic: a weighted network analysis
5. Derrible S (2012) Network centrality of metro systems. *PLoS One* 7(7): e40575
6. Dijkstra E (1959) Dijkstra's algorithm. A note on two problems in connexion with graphs. *Numerische Mathematik* 1:269–271
7. Freeman L (1977) A set of measures of centrality based on betweenness. *Sociometry* 40:35–41
8. Freeman L (1978/79) Centrality in social networks: conceptual clarification. *Soc Netw* 1:215–239
9. Gao M, Lim E-P, Lo D (2013) R-energy for evaluating robustness of dynamic networks. In: *ACM conference on web science*
10. Goh K-I, Oh E, Jeong H, Kahng B, Kim D (2002) Classification of scale-free networks. *PNAS* 99(20):12583–12588

11. Guimer R, Mossa S, Turtschi A, Amaral LAN (2005) The worldwide air transportation network: anomalous centrality, community structure, and cities' global roles. *Proc Natl Acad Sci USA* 102(22):7794–7799
12. Newman M (2008) The mathematics of networks. In: Blume L, Durlauf S (eds) *The new Palgrave encyclopedia of economics*, 2nd edn. Palgrave Macmillan, Canada
13. Opsahl T, Agneessens F, Skvoretz J (2010) Node centrality in weighted networks: generalizing degree and shortest paths. *Soc Netw* 32(3):245–251
14. Scheurer J, Curtis C, Porta S (2008) Spatial network analysis of multimodal transport systems: developing a strategic planning tool to assess the congruence of movement and urban structure. In: Technical report GAMUT2008/JUN/02, GAMUT Australasian centre for the governance and management of urban transport, The University of Melbourne
15. Sevtsuk A, Mekonnen M (2012) Urban network analysis toolbox. *Int J Geomat Spat Anal* 22(2):287–305

Indifferent Attachment: The Role of Degree in Ranking Friends

David Liben-Nowell, Carissa Knipe and Calder Coalson

Abstract The MySpace social networking site allows each user to designate a small subset of her friends as “Top Friends,” and place them in a rank-ordered list that is displayed prominently on her profile. By examining a large set of ≈ 11 M MySpace users’ choices of their #1 (best) and #2 (second-best) friends from historical crawl data from when MySpace was more popular than it now is, we discover that MySpace users were nearly indifferent to the popularity of these two friends when choosing which to designate as their best friend. Depending on the precise metric of popularity we choose, the fraction of users who select the more popular of these two individuals as their best friend wavers above and below 50%, and is always between 49.3 and 51.4%: that is, the popularity of the two candidates is essentially uninformative about which will be chosen as the best friend. Comparisons of other pairs of ranks within the Top Friends (e.g., #1-versus-#3, #2-versus-#3, ...) also reveal no marked preference for a popular friend over a less popular one; in fact, there is some evidence that individuals tend to prefer less popular friends over more popular ones. To the extent that ranking decisions form a window into broader decisions about whom to befriend at all, these observations suggest that positing individuals’ tendency to attach to popular people—as in network-growth models like preferential attachment—may not suffice to explain the heavy-tailed degree distributions seen in real networks.

Keywords Preferential attachment · Best friends · Friendship ranking

D. Liben-Nowell (✉) · C. Knipe · C. Coalson
Department of Computer Science, Carleton College, 1 N. College St.,
Northfield, MN 55057, USA
e-mail: dlibenno@carleton.edu

C. Knipe
e-mail: knipec@carleton.edu

C. Coalson
e-mail: coalsonc@carleton.edu

1 Introduction

Different social relationships have different priorities. Implicitly or explicitly, we all make daily decisions in which we choose one friend over another: we answer *that* email first; we find time for coffee with *this* person after telling another that we were too busy; we mention the job opportunity or free tickets to *that* friend instead of the other one. The way in which we prioritize one friend over another is an interesting, and important, question about our social relationships; it reflects the way in which a social network is used and constructed. Our prioritization decisions may also be a window into friend-making in general: the mechanisms by which Alice prioritizes her friend Bob over her friend Charlie may be very similar to the mechanisms by which Alice chooses to befriend Bob instead of befriending Charlie. (The intimacy of any social relationship falls on a continuum ranging from “best friend” via “distant friend” and “acquaintance” to “stranger,” and both of Alice’s decisions are comparisons of Bob to a reference point—Charlie or “friend”—on that continuum.)

As our daily lives have moved increasingly online over the last decade or so—and started to leave behind mineable data on social interactions—a voluminous body of computational research exploring the structural and behavioral properties of individuals embedded in social networks has blossomed. The preponderance of this research has treated ties in social networks as binary—we are friends; we are not friends—but a growing thread of this research has begun to consider the comparative strength of relationships.

The importance of relationship strength, and indeed the importance of *weak* relationships, has been studied in the social sciences for decades—most prominently in Mark Granovetter’s “The Strength of Weak Ties” [12]. An expanding body of recent computational research has explored relationship strength, too. A nonexhaustive sampling of these papers follows. Onnela et al. [22] studied a massive dataset of weighted ties, constructed via the rate of interactions among mobile phone users. Adamic, Lauterbach, and various coauthors [1, 16, 24] have studied the role of friendship strength in determining levels of trust among members of the Couch-Surfing community; for example, these authors showed that there is an inflationary effect in users’ ratings of others if those ratings are made publicly and nonanonymously. Wuchty and Uzzi [25] compared self-reported “close relationships” with those inferred based on email-response times. Backstrom et al. [2] have studied how Facebook users distribute their “attention” (fraction of wall posts, photo comments, etc.) across their friends. Gilbert and Karaholios [10] and Xiang et al. [26] constructed predictive models of users’ perceptions of relationship strength, based on a collection of measures of profile similarity and interaction between users. Perhaps the work most similar to our own is by Kahanda and Neville [14], who attempt to classify edges in Facebook as “strong” or “weak” links (as denoted by the presence or absence of a friend in a Facebook application in which a user could list their “top” friends) using the same type of structural and transactional properties.

In this chapter, we address a fine-grained question of prioritization among friends: rather than considering the *rating* of friendships on an absolute scale (from “distant”

to “close”), we will consider the *ranking* of friendships on a relative scale. Specifically, we examine a feature of the MySpace online social networking site, called *Top Friends*. Each MySpace user may choose to select a subset of his or her friends to designate as Top Friends. The user puts these Top Friends into a totally ordered list, from #1 down through # k , where the cardinality k of the Top Friends list is chosen by the user. (Nearly all users choose $k \leq 40$; for some period of time predating the data acquired for the present work, MySpace fixed the cardinality $k = 8$, so disproportionately many users have 8 Top Friends. As a result, the feature is also sometimes called the “Top 8.”) These Top Friends are displayed prominently on the user’s profile; the remaining (non-Top) friends are accessible by clicking through to the list of “All Friends.” We are most interested in a user’s choice as to which of two individuals will be her #1-ranked friend and which will be her #2-ranked friend [8], though we also consider # i -versus-# j decisions for all other $i, j \in \{1, \dots, 8\}$.

There is some evidence that suggests that people work hard to avoid public declarations of the ranking of their friendships—choosing weekly themes for their Top Friends or fake profiles with profile photos of Scrabble tiles that spell out, in order, a profane anti-“Top Friends” message; ranking top friends appears to be an angst-generating activity for many MySpace users [4]. This phenomenon is related to the fact that users give more generous (higher) ratings when their ratings are public and signed than when their ratings are private and anonymous [24]; MySpace users may work to avoid making their true rankings of friends known. But enough MySpace users (millions of users in our dataset) do provide rankings of real profiles that we can begin to pose, and answer, some interesting structural questions.

Indeed, the question of friendship ranking—as opposed to the question of friendship rating or of friendship existence—is a setting of scarce resources; after all, Alice can only have one #1-ranked friend. There is a minimal cost of accepting a distant acquaintance’s friend request in an online social network—perhaps just a slight risk to one’s reputation if the “friend” does something embarrassing, and the mild mental taxation of having one more relationship to track [9, 11]. Similarly, there is little cost in the type of “grade inflation” in rating one’s friends observed in CouchSurfing [24]. But the scarcity of highly ranked friendship slots means that the ranking environment may shed a different light on potentially awkward social decisions.

The Present Work This chapter addresses the role of the *popularity* of Bob and Charlie when Alice chooses which of the two to prioritize over the other. Suppose that Bob is more popular than Charlie, as measured by degree in the social network. One can formulate intuitive arguments on both sides as to which of Bob or Charlie would be a better friend: Alice should tend to prefer Charlie (he’s a more “committed” friend because he has fewer distractions) or Bob (he’s a more “valuable” friend because he knows more people). Indeed, a number of “rich get richer” network-growth models designed to account for the empirically observed degree distribution of real social networks take this second view: most prominently, preferential attachment [3] posits that the probability of u being involved in a new friendship is linearly increasing in u ’s current popularity.

Here we consider a large (≈ 11 M-profile) sample of MySpace users, each of whom has selected a #1- and #2-ranked friend: a best friend and a second-best friend. Using several distinct but straightforward measures of degree, we compute the relative popularity of these two friends. What we observe, essentially, is that the popularity of these two candidates has nearly negligible predictive power in separating the #1- and #2-ranked friend. Depending on precisely which measure of popularity we use, we observe that the probability that the more popular candidate is chosen as the best friend wavers between being above 50% (as high as 51.4%) and below 50% (as low as 49.3%).

We also perform the analogous comparisons for individuals' choice of # i -versus-# j -ranked friends for all other pairs of ranks $i, j \in \{1, \dots, 8\}$. As in the #1-versus-#2 decision, the fraction of individuals who prefer the more popular candidate as the better-ranked (closer to #1) friend varies from slightly above 50% (as high as 51.8%) to a bit further below 50% (as low as 46.1%).

At best, individuals exhibit a very mild preference for popularity in their choice of which of two friends to rank better; at worst, they are indifferent or even prefer an unpopular friend over one who is more popular. This lack of empirical support for individuals preferring the more popular candidate as a closer friend suggests one of two things. Either the reason for the heavy-tailed degree distribution seen in real social networks is subtler than the reinforcement-type mechanisms suggested by models based on a preference for the popular, or something about the way that we decide on the relative closeness of two close friends is fundamentally different from the way that we decide friend-versus-nonfriend.

2 The Data

We make use of a sample of the MySpace social network acquired using a cluster of desktop machines executing a parallelized BFS-style crawl over about 5 months in 2007–2008 [8]. We excluded profiles that were private, syntactically anomalous, had more than 20K total friends reported on their main profile page, or failed to list both an age and a sex. (Requiring both an age and a sex is a crude way of filtering out profiles of bands and other nonpersonal users from our data.) The resulting dataset contained the profiles of approximately 11 million MySpace users—10,989,190 users, to be precise. A partial BFS-style crawl is biased towards more “central” nodes; nodes in the sample may, e.g., have higher PageRank and higher degree than typical MySpace profiles [13, 21]. (Of course, there are also important differences between a typical MySpace user and a typical person in the real world, and—in this work as in all social-behavior research based on data from social networking sites, or indeed based on data from any kind of restricted population—we must be judicious in the generality of our conclusions.)

Because we focus here on the popularity of individuals, we consider several relevant measures of degree for a user u :

- The *listed degree* of u is the number of friends that are declared in u 's profile, as in "Alice has 150 friends." We have found that this number was occasionally unreliably reported; MySpace profiles in our crawl occasionally seemed to declare fewer friends than were actually linked from the user's profile.
- The *ranked outdegree* of u is the number of Top Friends chosen by u . (This quantity is most frequently 8, and ranges up to 40 except in rare cases.)

All of the other quantities are based on the number of people in our sample who claim u as a friend:

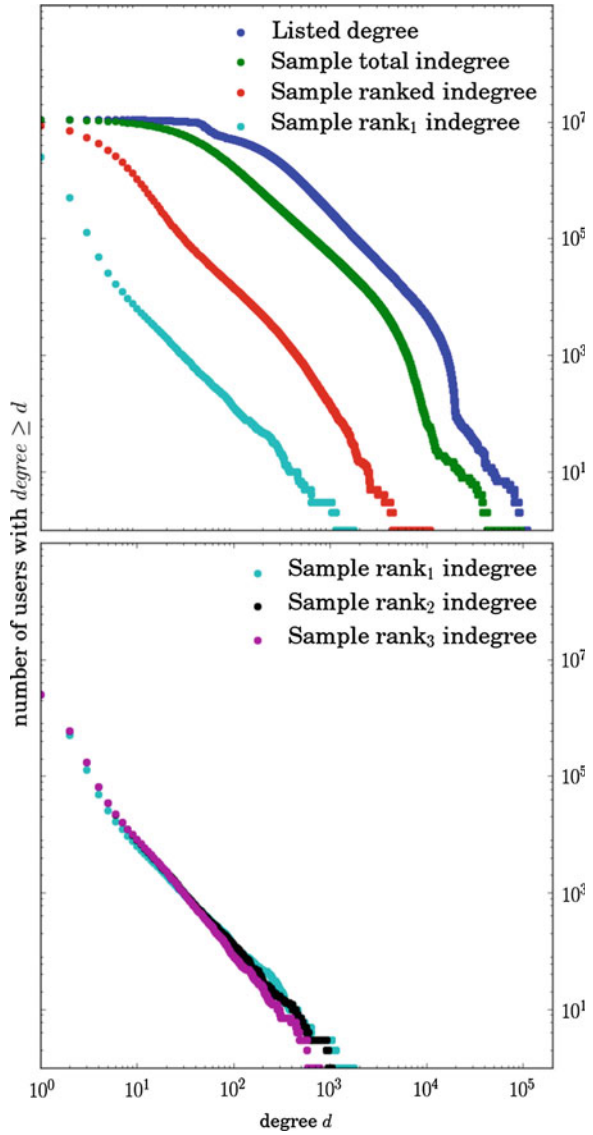
- The (*sample*) *total indegree* of u is the number of the ≈ 11 M MySpace users who list u as a friend.
- The (*sample*) *rank $_k$ indegree* of u is the number of in-sample users who list u as their k th-best friend.
- The (*sample*) *ranked indegree* of u is the number of users who include u anywhere in their Top Friends list, given by \sum_k (rank $_k$ indegree of u).

Figure 1 shows the cumulative degree distributions of the ≈ 11 M MySpace users under six of these measures. We see that the listed degree and each of the sample indegrees (total, ranked, rank $_1$, rank $_2$, and rank $_3$) show a heavy-tailed degree distribution. (For our purposes here, we remain agnostic about the particular shape of these distributions; we make no claim about the quality of fit of a power-law model for these distributions [6].)

As one would expect, the rank $_1$, rank $_2$, and rank $_3$ indegrees are smaller in an absolute sense. For any fixed k , rank $_k$ links are a scarce resource; only one such outgoing link is possible per user, so there are only ≈ 11 M such possible links in total. The remaining degree measures are effectively unlimited in the sense that each user can generate arbitrarily many outgoing links and nearly arbitrarily many outgoing ranked links (by lengthening her Top Friends list).

Aside from ranked outdegree, which seems qualitatively different from the other type of degree, all of these quantities seem to describe intuitively similar measures of popularity: each quantity is some version of how well-liked u is. (In contrast, a user u 's ranked outdegree is a decision about how many Top Friends to rank—something about how expansive u 's view of "closest friends" is.) Figure 2 shows the correlations across users among these various measures of degree. Indeed, the ranked outdegree is positively but weakly correlated with the other measures of popularity. The other totaled measures of degree—listed degree, sample ranked indegree, and sample total indegree—are all well-correlated with each other (all > 0.5). In the second part of Fig. 2, we see that rank $_j$ indegree and rank $_k$ indegree are strongly positively correlated for every j, k . In fact, with one minor exception, for every $k \in \{1, \dots, 8\}$ the rank $_k$ indegree is more strongly correlated with rank $_j$ indegree as $j < k$ gets closer and closer to k . (The lone exception is that rank $_4$ is slightly better correlated with rank $_8$ than rank $_5$ is.)

Fig. 1 Cumulative degree distributions in the ≈ 11 M-person dataset, for various measures of degree



3 The Central Prediction Task

Let G_{MS} denote the MySpace social network, represented as a directed graph. Annotate each edge $u \rightarrow v$ with v 's rank in u 's Top Friends list (or "unranked" if v is a friend of u but not a Top Friend).

We focus on the following prediction task. Consider a focal individual u in G_{MS} . The user u names a best friend v_1 and a second-best friend v_2 —the #1- and

	ranked outdegree	in-sample indegrees			
		ranked indegree	total indegree	rank ₁	rank ₂
listed degree	0.1653	0.5075	0.8586	0.2312	0.2934
ranked outdegree		0.1390	0.1183	0.0513	0.0620
ranked indegree			0.6695	0.6124	0.7118
total indegree				0.3186	0.4128
rank ₁					0.6747

	rank ₂	rank ₃	rank ₄	rank ₅	rank ₆	rank ₇	rank ₈	correlation with rank ₁₋₈
rank ₁	0.6747	0.5685	0.5253	0.4882	0.4478	0.4251	0.3918	
rank ₂		0.6818	0.6319	0.5970	0.5723	0.5234	0.4697	
rank ₃			0.6581	0.6324	0.6030	0.5788	0.5067	
rank ₄				0.6429	0.6211	0.5891	0.5523	
rank ₅					0.6268	0.5960	0.5429	
rank ₆						0.6076	0.5625	
rank ₇							0.5762	
rank ₈								

Fig. 2 Correlation of various pairs of degree measures across users. Cells are shaded in proportion to their correlation; for each k , the plot of correlations of rank _{k} with all other ranks is shown in the k th row

#2-ranked friends in u 's Top Friends list, respectively. We erase the ranking labels from the edges from u to v_1 and v_2 . The task is to predict which of $\{v_1, v_2\}$ is u 's #1 friend. A predictor p may access all of the graph G_{MS} , aside from the two erased labels, in making its prediction. We say that p is *correct* on u if it identifies v_1 as u 's #1 friend, *incorrect* on u if it identifies v_2 as u 's #1 friend, and *nondispositive* if p cannot distinguish between v_1 and v_2 . (A predictor may be nondispositive in the case of missing data or in the case that v_1 and v_2 have the same value under the predictor in question.)

Previous work on this prediction task, performed in collaboration with Peter DeScioli, Robert Kurzban, and Elizabeth Koch [8], has shown that MySpace users have a statistically significant preference for *homophily* [18] in choosing their best friends—that is, individuals tend to be more demographically similar to their #1 friend than to their #2 friend. In particular, over 56% of individuals have selected a best friend who is geographically closer than their second-best friend. (Note that this work ignored individuals on which geographic distance was nondispositive because of missing/malformed geographic data or ties in geographic distance.) A similar but substantially weaker homophilic effect holds for age: individuals tend to have best friends who are closer to their own age than their second-best friends are. In this previous work, we also identified another structural predictor that performed extremely well. Define the *relative rank* of u 's friend v as the rank that v assigns to u in v 's own

Top Friends list. We showed that 68.8% of MySpace users selected a best friend who ranks u better than their second-best friend does. (Note again that users for which the predictor was nondispositive were ignored.) DeScioli and Kurzban [7] take an evolutionary psychological perspective on friendship, and argue for an alliance-based view of the function of friendship that anticipates the success of the relative rank predictor. Other recent work has built a prediction system for *when* a link will be reciprocated in Twitter [5] or BuzzNet [15].

4 Using Popularity to Predict Preferences Among Friends

One can imagine many ways of attacking the central prediction task described in the previous section. In this chapter, we concentrate on purely *popularity-based* predictors. That is, to what extent does user u 's decision about which of $\{v_1, v_2\}$ to choose as u 's #1 friend correlate with the relative popularities of v_1 and v_2 ? To address this question, we filter the ≈ 11 M MySpace profiles to identify all those focal individuals u for which u 's #1- and #2-ranked friends also appear in the sample. We culled a population of ≈ 1.36 M (precisely: 1,360,879) profiles using this filtering process.

For the purposes of our main prediction task, we must ensure that we do not “cheat” by baking the correct answer into the measure of popularity. In particular, when we refer to the rank_1 and rank_2 indegrees for the purposes of predicting a user u 's #1 friend, we mean the rank_1 and rank_2 indegrees *excluding the ranked edge from u* . (Our other measures of popularity are affected equally by the edges from u to v and w , so excluding this edge makes no difference.)

A scatterplot displaying the popularities of each user's Top 2 friends, under four of the popularity measures discussed previously, is shown in Fig. 3. To the extent that these measures of popularity are helpful in differentiating #1- and #2-ranked friends, we would see an asymmetry in the distributions. Specifically, if individuals tend to prefer friends who are more popular, then we would see more points below the diagonal line. What is perhaps visually striking about Fig. 3 is that all four panels appear highly symmetric across the diagonal; not only is there no obvious preponderance of points on one side of the diagonal, but the points appear to be distributed close to symmetrically across that diagonal line.

To make this observation more precise, for each measure μ of popularity described previously, we compute the number of users u whose #1-ranked friend is more popular under μ than u 's #2-ranked friend is (a *win* for μ); the number of users u whose #1- and #2-ranked friends are equally popular under μ (a *tie*); and the number of users u whose #2-ranked friend is more popular under μ (a *loss*). Figure 4 shows the results.

Because a random guess will be correct on 50% of its predictions, absolute deviation from 50% is the most interesting measure of success. Every popularity-based predictor has a success rate within 0.014 of 0.5; measured by deviation from 50%, the two most successful are rank_1 (51.4%) and rank_2 (49.3%) indegrees: there is a

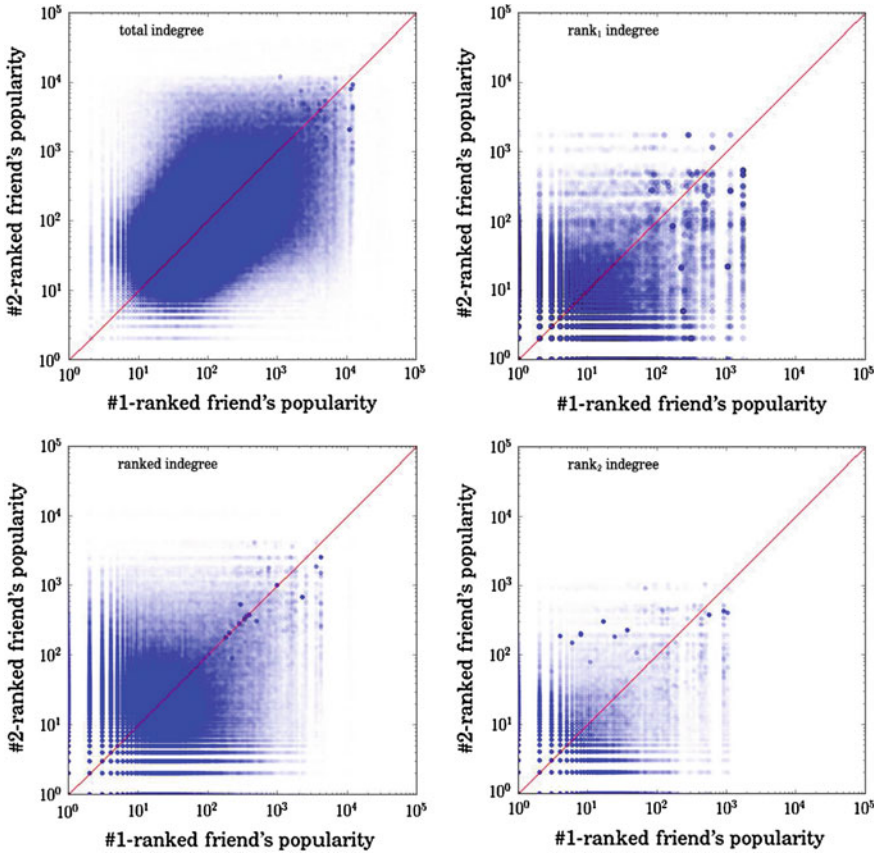


Fig. 3 The popularities of u 's #1- and #2-ranked friends under four measures of popularity: total indegree; ranked indegree; rank_1 indegree; and rank_2 indegree. In each *panel*, one point is shown for each of the ≈ 1.36 M users in the culled dataset

mild tendency for the #1-ranked friend to be ranked #1 more often by others, and for the #2-ranked friend to be ranked #2 more often by others. (See below for some discussion of the phenomenon that rank_1 indegree better predicts #1-rank and rank_2 indegree better predicts #2-rank.) All other measures of popularity perform between 49.5 and 50.2 %.

Even the most informative measures give only weak information, and indeed the various measures of popularity even differ in directionality: four of the predictors (listed degree, total sample indegree, rank_1 indegree, and rank_8 indegree) say that individuals (weakly) prefer others who are *more* popular; the remaining eight predictors say that individuals (weakly) prefer others who are *less* popular. For the sake of comparison, two other predictors are displayed in Fig. 4: the geographic distance predictor (“ u prefers the friend who lives geographically closer to u ”) and the relative rank predictor (“ u prefers the friend who ranks u better”).

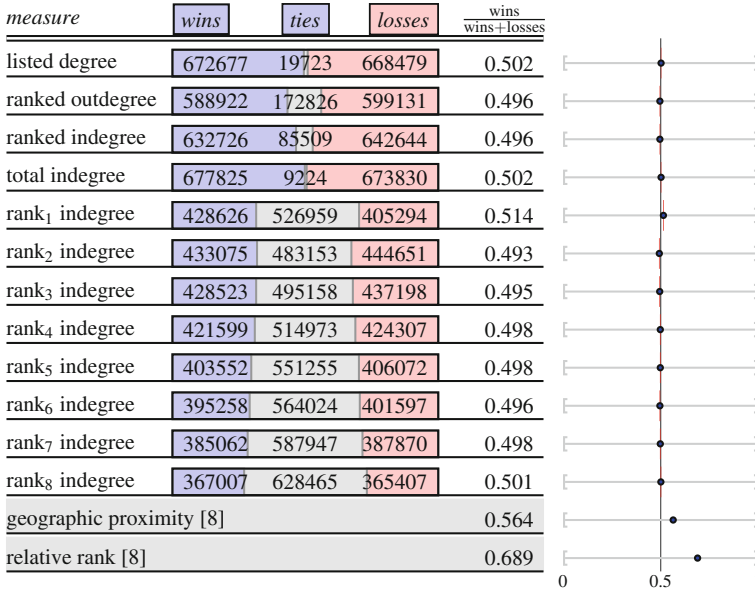


Fig. 4 The performance of each popularity-based predictor on the ≈ 1.35 M culled MySpace users. A *win* for a predictor is an individual for whom $\text{degree}(\#1) > \text{degree}(\#2)$, where $\#1$ and $\#2$ are the best- and second-best friends, respectively. A *tie* is an individual for whom $\text{degree}(\#1) = \text{degree}(\#2)$. A *loss* is an individual for whom $\text{degree}(\#1) < \text{degree}(\#2)$. The smallest number of non-tied data points is $n = 732,414$; viewing each of these predictors as a n -trial binomial distribution, the standard error for each of these measures is ≤ 0.0012 , and 99.9% confidence intervals are shown as the *small vertical red bars*

4.1 Beyond the “Top 2”

We have focused our discussion thus far on distinguishing #1-versus-#2-ranked friends, but the same calculations can be performed for any pair of ranks. For any two ranks i and $j > i$, we compute the fraction of focal individuals for whom friend $\#i$ is more popular than friend $\#j$. (Our previous discussion was for $i = 1$ and $j = 2$.) Figure 5 displays the tables of results for eight predictors, omitting only the rank₅₋₈ indegree predictors, which are qualitatively similar to the rank₄ table. (As before, we must avoid baking the correct answer into the predictor: when predicting a user u ’s $\#k$ -versus- $\#j$ friends, we exclude the two edges from u when computing the rank _{k} and rank _{j} predictors.)

Figure 5 reveals that the qualitative pattern of the #1-versus-#2 comparison remains true for other pairs of ranks. For the broader degree measures (listed degree, ranked outdegree, total indegree, ranked indegree), the fraction of individuals who prefer the more popular candidate friend is generally close to or below 0.50, and only rarely greater than half. In each of these cases, the fraction of individuals whose best friend is more popular than their $\#j$ -friend decreases with j ; for example, fewer

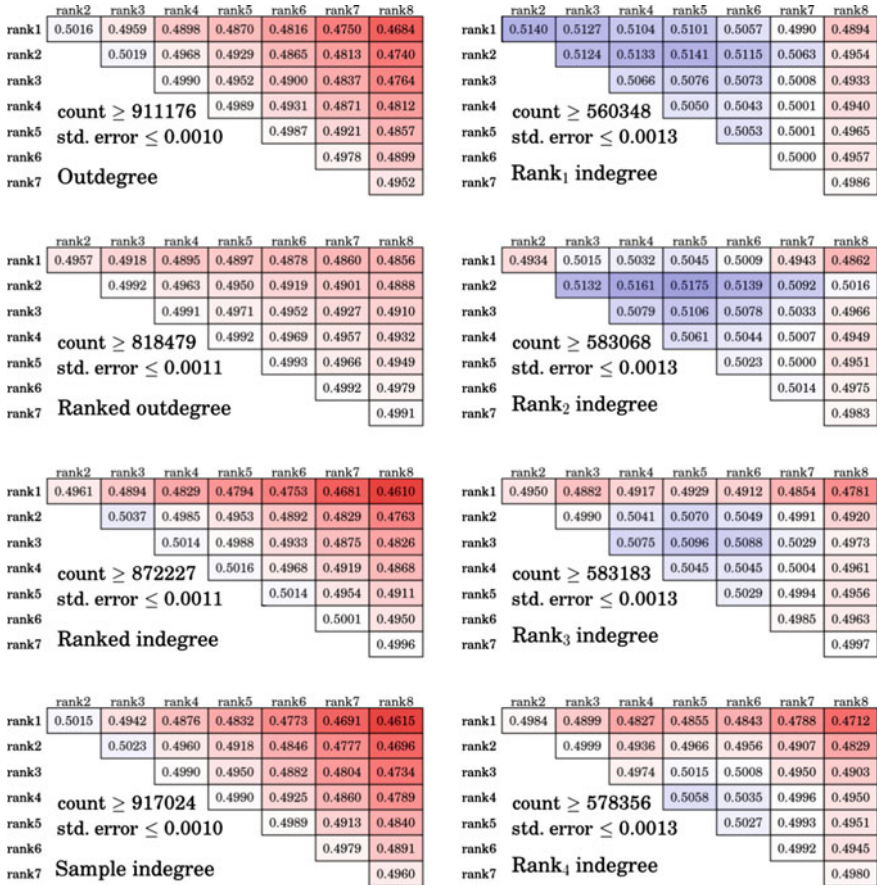


Fig. 5 Results for other rank comparisons. In each panel, the i -versus- j cell displays the fraction of individuals u whose $\#i$ th and $\#j$ th friends are ordered so that u prefers the one who is more popular. (The 1-versus-2 cells correspond to Fig. 4.) Red-shaded cells indicate that more individuals prefer the less popular friend, and blue-shaded cells indicate that more individuals prefer the more popular friend; the darker the shading, the further from 0.50. The displayed count is $\min_{i,j}(\text{wins} + \text{losses})$ for $\#i$ -versus- $\#j$ friends; an upper bound on the standard error, viewing each of these predictions as a binomial distribution, is shown as well

than 47% of individuals have a best friend who is more popular than their #8 friend under listed degree, total indegree, and ranked indegree.

We do see the hints of one exception to this trend: out of two candidate friends of an individual u , the one with a higher rank $_i$ indegree is generally more likely to be u 's $\#i$ friend. (Recall that when predicting u 's $\#i$ friend, "rank $_i$ indegree" means "rank $_i$ indegree *aside from the edge from u herself.*") This phenomenon is visible in the rank- i row of the rank $_i$ panels in Fig. 5: for example, having a higher rank $_2$ indegree corresponds to a 51.3–51.8% chance of being ranked #2 instead of being ranked

{#3, #4, #5, #6}. One partial explanation for this observation is the “Top Friends”—avoiding tactics employed by some users that survived in the data set: choosing a “Top 8” whose profile pictures, letter-by-letter and in order, spelled out a particular four-letter word plus T+O+P+8. The profile with the “U” Scrabble tile—the second letter of this four-letter word—as its profile photo would have a high rank_2 indegree (and a low $\text{rank}_{i \neq 2}$ indegree); this profile would often be correctly predicted to be a #2 friend using the rank_2 indegree predictor. Still, this avoidance behavior appears to be relatively rare, and even among the $\text{rank}_{i \in \{1,2,3,4\}}$ indegree predictors, there are slightly more pairs of ranks in which individuals prefer the less popular friend.

5 Discussion: Indifferent Attachment?

In the earliest days of the present era of computational social network research, Barabási and Albert [3] published an influential paper on degree distributions of social networks. Barabási and Albert made two key observations. First, they showed empirically that the degree distributions of real social networks are heavy tailed. (And, they argue, the form of the degree distribution is specifically well modeled by a power law, though Clauset et al. [6] raise some serious concerns about the quality of the power-law fit for this type of network data.) Second, Barabási and Albert proposed *preferential attachment (PA)* as a generative model of social network growth. (Both observations were presaged in the literature of other disciplines earlier; see the early work of Yule [27] and Simon [23], and the more recent survey by Mitzenmacher [19].) As other structural properties of social networks have been discovered, alternative generative models have been proposed. These models—e.g., community-guided attachment and forest-fire models [17]—do not seem to make as-obvious predictions about how preferences among individuals will be expressed; thus, we focus our discussion here on network-formation models, like PA, with some form of popularity reinforcement—nodes with higher degree gain edges at a higher rate than nodes with lower degree.

Here is the preferential attachment model, in its basic form. We start with a small network, which grows by one node and one edge at each time step. At time step t , a new node u_t appears in the system, and it forms one edge from u_t to an existing node. More popular existing nodes are more likely to be chosen as u_t 's neighbor; specifically, the probability that u_t chooses v is directly proportional to the current value of $\text{degree}(v)$. PA is a particular instantiation of what we might call the *preference for popularity*—that, given a choice between two candidate friends, the more popular candidate is the one more likely to be preferred. (Other nonlinear instantiations of this preference occur in other models.)

While the basic form of PA does not speak directly to rankings of friends, the underlying preference for popularity does make particular predictions about ranking. PA can be most straightforwardly adapted to the ranked setting by modeling a node as ranking its neighbors in the order in which edges formed. (So the #1-ranked friend for u is the friend u chose when joining the network; u 's #2-ranked friend is the first

node $v \neq u$ that chose u when v joined the network, u 's #3-ranked friend is the second node that chose u upon joining, etc.) We simulated this Ranked-PA (RPA) network growth model for a 100,000-node network, and observed that friend ranking is much better predicted by popularity in RPA than in MySpace: over 95 % of RPA nodes had a best friend that was more popular than their second-best friend, and between 59 and 61 % of nodes had a # i -ranked friend more popular than their # $(i + 1)$ -ranked friend for $i \in \{2, 3, 4, 5\}$. (In PA/RPA, the age of a node and the node's degree are positively correlated; thus the edge formed earlier is more likely to have been formed from a neighbor that would eventually become more popular. The #1-ranked friend is special—it was chosen with explicit preference towards high degree, instead of by age—and so its popularity advantage over the #2-ranked friend is much higher than the advantage of #2 over #3 and the other pairs.)

The empirical and modeling observations of Barabási and Albert sparked a large body of literature, empirical and theoretical, that has made a great deal of progress in modeling and analyzing the structural properties of real-world social networks—particularly regarding a hypothesis of the origin of the apparently ubiquitous heavy-tailed degree distributions. But the results shown in Figs. 4 and 5, coupled with the simulations of RPA, suggest that a preference for popularity may not provide a full explanation for empirically observed heavy-tailed degree distributions: when a user is choosing which of two friends she prefers, the popularity of the two candidates is at best essentially uninformative about which will be chosen by that user, and at worst she actually prefers the less popular friend. While the twelve measures of popularity that we consider here are strongly positively correlated, they are not perfectly aligned (Fig. 2). But they are all aligned with respect to our central prediction task: each is at best only marginally informative about the preferences of individuals among their closest friends.

Perhaps the analogy between choosing whether to befriend an individual and choosing whether to rank an individual highly is a weak one; those decisions may be made differently, and thus the results of this chapter may not speak to the underlying “to friend or not to friend” decision. (It is an interesting direction for future research to assess to what extent deciding whether to befriend an individual and whether to rank an individual highly *are* similar or different. An understanding of why, and how, a pair of individuals decide to assign a “friend” label to their relationship is generally missing from the current computational literature on social networks—and this understanding is obviously crucial to properly interpreting what the edges in the network actually mean. Interactions in online social networking sites have some key differences from real-world interactions: the question “does our relationship rise to the level that deserves the ‘friends’ label?” is rarely explicitly called in the offline world, and the fact that it is continually raised by a social networking site may impose different or stronger social pressures in how we react online. Still, massive digital data presents a promising opportunity to better understand the friendship-or-no decisions.)

But to the extent that ranking decisions and befriending decisions are analogous, our observations suggest that network-growth models based on a preference for popularity miss some important behavioral properties; we will need a different explanation to account for empirically observed heavy-tailed degree distributions.

And even if ranking and befriending decisions are fundamentally different, the heavy-tailed degree distribution for rank₁ indegree (Fig. 1) seems to require an explanation fundamentally different from preferential attachment.

Mitzenmacher [20] argues compellingly that, as a research community, we must move toward the necessary future direction of *validation* (or, at least, *invalidation*) in research on heavy-tailed degree distributions. We hope that the present work can serve as a small step forward in that enterprise.

Acknowledgments An abbreviated preliminary version of this work appears in the *Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM'13)*. Thanks to Peter DeScioli, Elizabeth Koch, Robert Kurzban, Dave Musicant, Jeff Ondich, and Aaron Swoboda for helpful discussions and comments. This work was supported in part by NSF grant CCF-0728779 and by Carleton College.

References

1. Adamic LA, Lauterbach D, Teng C-Y, Ackerman MS (2011) Rating friends without making enemies. In: Proceedings of the international conference on weblogs and social media (ICWSM'11)
2. Backstrom L, Bakshy E, Kleinberg J, Lento T, Rosen I (2011) Center of attention: how Facebook users allocate attention across friends. In: Proceedings of the international conference on weblogs and social media (ICWSM'11)
3. Barabási A-L, Albert R (1999) Emergence of scaling in random networks. *Science* 286:509–512
4. Boyd D (2006) Friends, friendsters, and MySpace top 8: writing community into being on social network sites. *First Monday* 11(12)
5. Cheng J, Romero D, Meeder B, Kleinberg J (2011) Predicting reciprocity in social networks. In: Proceedings of the 3rd IEEE conference on social computing (SocialCom'11)
6. Clauset A, Shalizi CR, Newman MEJ (2009) Power-law distributions in empirical data. *SIAM Rev* 51(4):661–703
7. DeScioli P, Kurzban R (2009) The alliance hypothesis for human friendship. *PLoS One* 4:e5802
8. DeScioli P, Kurzban R, Koch EN, Liben-Nowell D (2011) Best friends: alliances, friend ranking, and the MySpace social network. *Perspect Psychol Sci* 6(1):6–8
9. Dunbar RIM (1992) Neocortex size as a constraint on group size in primates. *J Hum Evol* 22:469
10. Gilbert E, Karahalios K (2009) Predicting tie strength with social media. In: Proceedings of the 27th international conference on human factors in computing systems (CHI'09), pp 211–220
11. Gonçalves B, Perra N, Vespignani A (2011) Modeling users' activity on Twitter networks: validation of Dunbar's number. *PLoS One* 6(8):e22656
12. Granovetter MS (1973) The strength of weak ties. *Am J Sociol* 78(6):1360–1380
13. Henzinger MR, Heydon A, Mitzenmacher M, Najork M (2000) On near-uniform URL sampling. *Comput Netw* 33:295–308
14. Kahanda I, Neville J (2009) Using transactional information to predict link strength in online social networks. In: Proceedings of the international conference on weblogs and social media (ICWSM'09)
15. Lang J, Wu SF (2011) Anti-preferential attachment: if I follow you, will you follow me? In: Proceedings of the 3rd IEEE conference on social computing (SocialCom'11)
16. Lauterbach D, Truong H, Shah T, Adamic L (2009) Surfing a web of trust: reputation and reciprocity on CouchSurfing.com. In: Proceedings of the 2009 international conference on computational science and engineering, pp 346–353

17. Leskovec J, Kleinberg J, Faloutsos C (2007) Graph evolution: densification and shrinking diameters. *ACM Trans Knowl Discov Data* 1(1):2
18. McPherson M, Smith-Lovin L, Cook J (2001) Birds of a feather: homophily in social networks. *Annu Rev Sociol* 27:415–444
19. Mitzenmacher M (2004) A brief history of generative models for power law and lognormal distributions. *Internet Math* 1(2):226–251
20. Mitzenmacher M (2006) Editorial: the future of power law research. *Internet Math* 2(4):525–534
21. Najork M, Weiner JL (2001) Breadth-first crawling yields high-quality pages. In: *Proceedings of the 10th international conference on world wide web (WWW'01)*, pp 114–118
22. Onnela J, Saramaki J, Hyvonen J, Szabo G, Lazer D, Kaski K, Kertész J, Barabási A-L (2007) Structure and tie strengths in mobile communication networks. *Proc Natl Acad Sci* 104(18):7332–7336
23. Simon HA (1955) On a class of skew distribution functions. *Biometrika* 42(3/4):425–440
24. Teng C-Y, Lauterbach D, Adamic LA (2010) I rate you. You rate me. Should we do so publicly? In: *3rd workshop on online social networks (WOSN'10)*
25. Wuchty S, Uzzi B (2011) Human communication dynamics in digital footsteps: a study of the agreement between self-reported ties and email networks. *PLoS One* 6(11):e26972
26. Xiang R, Neville J, Rogati M (2010) Modeling relationship strength in online social networks. In: *19th international world wide web conference (WWW'10)*
27. Yule G (1925) A mathematical theory of evolution based on the conclusions of Dr. J.C. Willis. *Philos Trans Royal Soc Lond* 213:21–87

Analyzing the Social Networks of Contributors in Open Source Software Community

Mohammad Y. Allaho and Wang-Chien Lee

Abstract We conduct an extensive statistical analysis on the social networks of contributors in Open Source Software (OSS) communities using datasets collected from two most fast-growing OSS social interaction sites, Github.com and Ohloh.net. Our goal is to analyze the connectivity structure of the social networks of contributors and to investigate the effect of the different social ties structures on developers' overall productivity to OSS projects. We, first, analyze the general structure of the social networks, e.g., graph distances and the degree distribution of the social networks. Our social network structure analysis confirms a power-law degree distribution and small-world characteristics. However, the degree mixing pattern shows that high degree nodes tend to connect more with low degree nodes suggesting a collaboration between experts and newbie developers. We further conduct the same analysis on affiliation networks and find that contributors tend to participate in projects of similar team sizes. Second, we study the correlation between various social factors (e.g., closeness and betweenness centrality, clustering coefficient and tie strength) and the productivity of the contributors in terms of the amount of contribution and commitment to OSS projects. The analysis is conducted under the contexts of global and local networks, where a global network analysis considers a developer's connectivity in the whole OSS community network, whereas a local network analysis considers a developer's connectivity within a team network that is affiliated to a project. The analysis demonstrates evident influence of the social factors on the developers' overall productivity.

M.Y. Allaho (✉) · W.-C. Lee
Department of Computer Science and Engineering,
The Pennsylvania State University, University Park, PA, USA
e-mail: mya111@cse.psu.edu

W.-C. Lee
e-mail: wlee@cse.psu.edu

Keywords Open source software (OSS) · Social network · Affiliation network · Global network analysis · Local network analysis · Degree of connectivity · Ties strength

1 Introduction

Large collaborative online communities have appeared and grown rapidly in the presence of Web 2.0 technology, witnessed by the massive success of Open Source Software (OSS) projects such as the Apache projects and GNU/Linux. As members of OSS projects are usually volunteers [12], their motivations usually involve personal goals/interests, e.g., sharpening their programming skills and gaining experience, following fellow peers, networking with the OSS community members, or simply supporting free open software projects [8]. In contrast to commercial software with dedicated responsibility to software engineers, the amount of participation and commitment by the volunteering developers are crucial factors for the success of projects [16].

To investigate and better understand the success factors of OSS, several research studies have been reported in the literature [4, 6, 9, 17, 18, 20]. However, these studies are mainly focusing on success factors related to the user-developer interaction (e.g., user ratings and bug reporting) and the project related factors (e.g., programming languages, operating systems and the open source license used). Recently, the importance of social factors in OSS community has drawn a growing amount of attention. A study by Antwerp et al. [1] argues the importance of studying OSS social structure as an indicator of OSS projects success. Moreover, studies in [2, 7] suggest that the rapport created from collaboration among individuals in open collaborative communities indeed plays a crucial role leading to the success of OSS projects. However, the potential correlation between the productivity of OSS developers and the social ties among those developers in the community has not been quantitatively studied before. In this paper we aim to answer these research questions: What are the general structure and features of the OSS social network? What are the social ties patterns in OSS community network that affect the developers' amount of contribution and commitment? Moreover, does the connectivity structure and strength of social ties in a team affect the amount of contribution and commitment in OSS projects?

In order to answer these critical research questions, we first analyze the general structure of the OSS social network and then investigate the correlation between various social ties structures and the amount of contribution and commitment of volunteering developers to OSS projects. In particular, we conduct an extensive social link analysis on the Social Network (SN) and Affiliation Network (AN) of OSS developers. We define the OSS SN as a directed graph where nodes represent developers and links represent a following/follower relationship just like the Twitter network, while defining the OSS AN as an undirected graph where nodes represent developers and links represent co-authorship in OSS projects. The SN and AN represent general but different social ties among the contributors of OSS communities,

i.e., different kinds of link representation may draw different conclusions about the studied community.

We collected our data from two most fast-growing OSS collaboration sites, *Github.com* and *Ohloh.net*. These two sites are unique since they provide many social interaction tools for developers, including the ability to form a network of followings/followers relationship which gives explicit information of acquaintances. We emphasize here that most previous studies on the SNs of OSS typically construct the SN based on communication links which results in an implicit SN (i.e., the links may not represent true acquaintances for various reasons). Aiming to better understand the characteristics of OSS SNs, we perform a series of analysis on the social network structure, including measuring SNs distances (i.e., diameter and average path length), analyzing degree distributions, testing power-law model and examining degree mixing patterns between nodes (developers) in the OSS SN and AN.

Our social network structure analysis confirms a small-world characteristics for the OSS SNs. Also, the analysis shows that the degree distribution of the SNs follows a power law distribution that fits the scale-free network model. However, the degree distribution of the ANs does not fit the scale-free model which implies different characteristics of the two networks. Moreover, the degree mixing pattern analysis for the SNs shows that high degree nodes tend to connect more with low degree nodes suggesting collaboration between experts and newbie developers. On the other hand, the degree mixing pattern for the ANs shows an opposite trend.

In addition to the analysis of network structures, we also investigate the factors of social ties under the contexts of global and local network graphs to explore the social factors that affect the developers' amount of contribution and commitment. A Global network analysis studies the connectivity among individuals in the whole OSS community, whereas a Local network analysis studies the connectivity among members in a team working on a particular project P , where a team is a subset of the community as illustrated in Fig. 1. For the global network analysis, we consider the *betweenness* and *closeness* centrality and for the local network analysis we consider the *clustering* patterns and *ties strength*. Note that nodes' degree analysis is considered a global analysis if it considers all nodes in the community and a local analysis if it considers only team nodes. Both global and local network analysis demonstrate evident influence of the social factors on the developers' overall productivity.

This work constitutes the first large-scale study that analyzes the social network of volunteering developers that is based on explicit acquaintances. Moreover, our work is the first to focus on investigating the correlations between social factors and developers' productivity. We, further, quantify the developers' productivity from an abstract meaning into measurable metrics in order to comprehend the amount of contribution and commitment each developer exerts in a project. In addition, we study the social factors in a community context and in a team level context to capture any social factor that might be related to either context. Our findings may then pave the road for future recommendation systems related to online open collaborative projects.

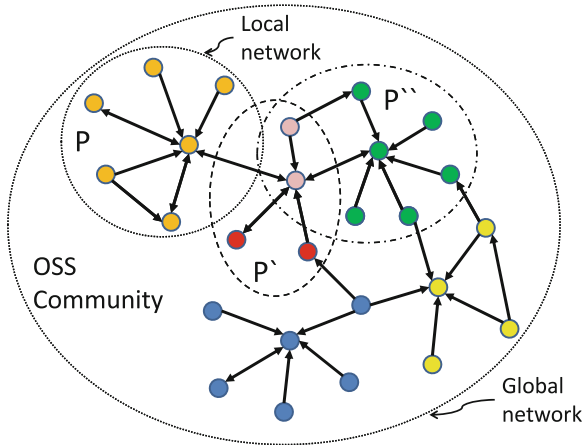


Fig. 1 A community network example illustrating the difference between global and local network graphs. A node represents an individual/developer in the community and a link represents a social relationship. The OSS community network represents a global network graph, whereas a team network of project P represents a local network graph. Here nodes of the same color belong to one project. Note that it is possible for a node to belong to multiple projects and, therefore, can belong to multiple local networks as illustrated in the graph by the two tainted-color nodes belonging to P' and P'' .

The rest of the paper is organized as follows. Section 2 introduces the dataset collected and used in this study. Section 3 shows the analysis of network structure. The analysis includes network structure measures, degree distribution and degree mixing patterns analysis. Section 4 investigates the correlation between the various social ties' structures and developers' overall productivity globally, in the whole OSS community, and locally, in a team network. Section 5 introduces several related works and, finally, Sect. 6 concludes the paper.

2 Dataset

We collect our data from two fast-growing OSS social interaction sites, *Github.com* and *Ohloh.net*. These two sites provide a variety of information regarding the social relationship and contributions of their volunteering developers. Unlike other OSS collaboration sites (e.g., *sourceforge.net*, *freshmeat.net*) Github and Ohloh provide facilities for contributors to explicitly recognize each other through a *follow* link in Github and *Give kudo* in Ohloh which creates a directed social network of contributors/developers. We collected data of over a million contributor and over two millions OSS project. The data were collected via APIs provided by Github and Ohloh.

Table 1 summarizes some general statistics of the collected data from Github and Ohloh. In Table 1, *Participations* is the summation of each developer's number of

Table 1 Collected data statistics

	<i>Ohloh</i>	<i>Github</i>
Participations	671,835	9,268,644
Contributors	297,046	1,034,996
Contributors with account	14,958	652,040
SN edges	36,205	1,788,803
AN edges	1,096,208	14,474,929
Projects	456,811	2,332,749

affiliations in OSS projects, whereas *Contributors*, in the second row, is the distinct number of developers in the community. For example, in Fig. 1, the *Contributors* is the total number of nodes (developers) in the global network graph. To compute *Participations*, we first count each individual's number of affiliations, e.g., each node in Fig. 1 has one affiliation (participates in one project) except the two tainted-color nodes belonging to P' and P'' where each has two affiliations since they participate in both project P' and project P'' . Then the summation of each developer's affiliations is the *Participations* value. Furthermore, the contributors with account are the contributors that are registered in the site with a profile information. In our analysis, we only consider contributors with account because it is possible to link between their contribution information and social attributes which is not possible with anonymous contributors. This may cause a threat to validity since anonymous contributors could add weight to any connectivity pattern. Another threat is that we only capture explicit acquaintances among the developers, where there might be acquaintances that are not declared. However, since we conduct the analysis on large number of OSS developers and projects without selection criteria the threats to validity is minimized. Moreover, the results in the next sections show consistency on both Ohloh and Github data.

3 Network Structure Analysis

In this section we study the general graph distance measures, the node degree distribution and the mixing pattern of degrees in both SN and AN.

3.1 Basic Measures of the Network Structure

We first measure the SNs graph distances to comprehend the general structure of the networks. Table 2 shows the diameter and average path length of Github and Ohloh SNs. Recall that the graph diameter is the longest distance of the shortest distances

Table 2 OSS social network structure measurements

	Diameter	Avg. path length	Avg. degree	Avg. CC	Avg. close.
Github	22	6.25	4.31	0.112	0.19
Ohloh	23	7.11	3.21	0.25	0.15

between every pair of nodes in the network, while the average path length is the average length of those shortest paths. The average path length found in both Github and Ohloh confirms the result by Milgram [13] which states that any two people are on average separated by six intermediate connections, known as the “six degrees of separation” in the “small world” phenomenon. Also, the diameter of the OSS SN is small compared to non-social networks (e.g., the Web network has a diameter of 905 [14]), which is a feature of most social graphs. Additionally, Table 2 includes the average degree, clustering coefficient (CC) and closeness centrality of the OSS SNs. Besides, the average betweenness centrality for Github and Ohloh are 3.44×10^{-6} and 1.58×10^{-4} , respectively (refer to Sect. 4 for measurement definitions). The structure measures of these two SNs show great similarity. In the next section, we test the OSS SN and AN against the scale-free network model.

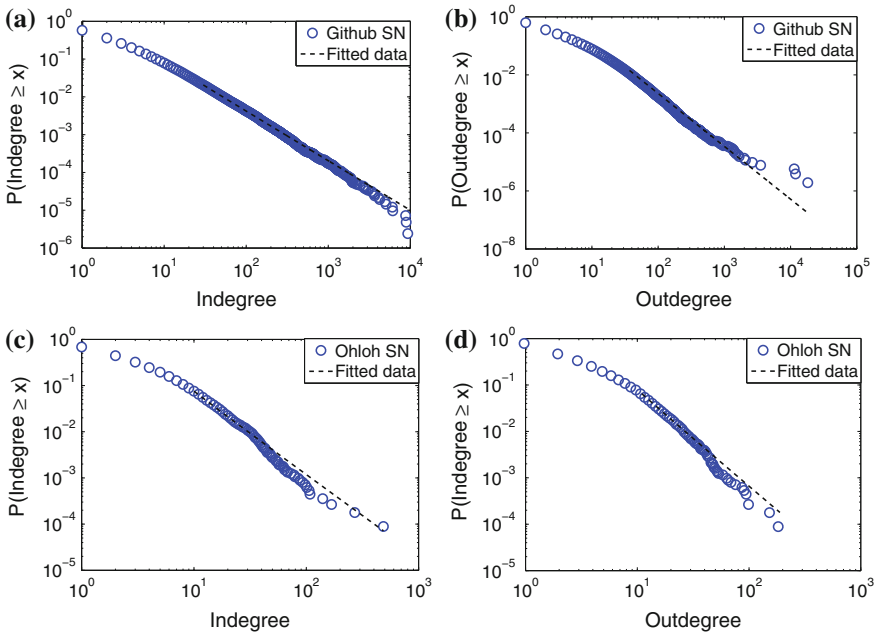


Fig. 2 Github and Ohloh Out/Indegree Complementary Cumulative Distribution Functions (CCDF) for the OSS SN ($x = \text{Out/Indegree}$). All social networks show properties consistent with power-law networks

3.2 Power-Law Degree Distribution

Node degree analysis is essential to reveal the connectivity trends between nodes (developers) in a graph. Thus, we first investigate whether the OSS SN and AN follow a scale-free network model or not. A scale-free network has a degree distribution following a power-law model where the exponent α is typically $2 < \alpha < 3$ in most cases. Figure 2 shows the outdegree and indegree Complementary Cumulative Distribution Function (CCDF) of the OSS SN for Github and Ohloh, respectively. Basically, the plots show the probability of any node having a degree greater than or equals the given degree x . Both outdegree and indegree distributions are consistent with the fitted power-low distribution model. To confirm this result, Table 3 shows the maximum likelihood estimate of the scaling exponent α and the Kolmogorov-Smirnov goodness-of-fit statistic D , where a value close to zero indicates that the data is closely estimated by the given model. Both networks follow a power-law distribution and confirm properties of scale-free networks where few nodes have high degrees and the majority have low degrees.

On the other hand, the OSS AN does not show a good fit to the scale-free model. Figure 3 shows a sharp drop on the CCDF on high degrees showing that at some point of high degree the probability of a node having higher degree drops dramatically indicating big gaps of degrees among large degree nodes. This is because the degree of a node in affiliation networks may increase in high order. For example, once a node x joins a project all the nodes participating in that project become connected to

Table 3 Power-law coefficient estimates (α) and corresponding Kolmogorov-Smirnov goodness-of-fit metric (D)

	Indegree		Outdegree	
	α	D	α	D
Github	2.31	0.0258	2.82	0.0433
Ohloh	2.77	0.0281	3.04	0.0231

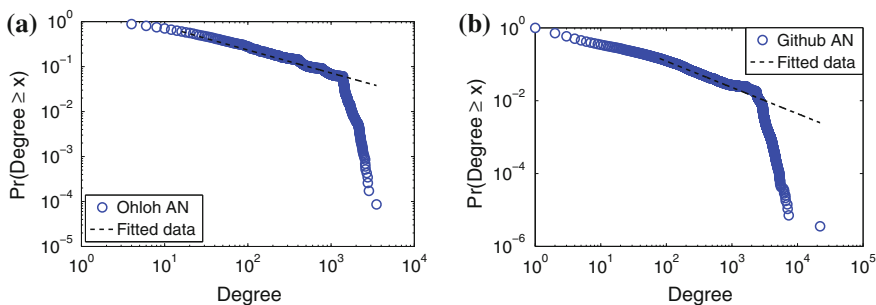


Fig. 3 Ohloh (left) and Github (right) degree complementary cumulative distribution functions (CCDF) for the OSS AN ($x = \text{degree}$). The networks does not show a good fit to power-low model

that node causing a large increase in the node degree of x . The maximum likelihood estimate of the scaling exponent α for Ohloh AN is 1.51 and 1.72 for Github AN.

3.3 Node Degree Mixing Pattern

In this section, we investigate the mixing pattern of degrees between connected nodes. In other words, we would like to know whether similar degree nodes tend to connect to each other or is it a mixture and how much is that mixture? This study is important since it reveals important insights to the structure of the network. Connectivity between similar degree nodes indicates a centralized network where high-degree nodes are connected together. This kind of network is more susceptible to diffuse information fast. On the other hand, networks where high-degree nodes tend to connect to low-degree nodes show the opposite behavior but it is more robust than centralized networks since removing a high degree node will not dramatically affect the network connectivity.

3.3.1 Degree Correlation

First we study the *Joint Degree Distribution* (JDD), also referred to as the *mixing pattern* [15] and the *2K distribution* [11]. JDD gives the probability of a node of degree k_1 to be connected with a node of degree k_2 . JDD is approximated by the *degree correlation function* k_{nn} . For an undirected graph, k_{nn} is a mapping between every degree and the average degree of all neighboring nodes (connected) to nodes of that degree. On the other hand, k_{nn} for a directed graph is a mapping between every outdegree and the average indegree of all neighboring nodes (connected) to nodes of that outdegree. Figure 4 shows the OSS SN k_{nn} plots. Both Ohloh (Fig. 4a) and Github (Fig. 4b) show that high-degree nodes tend to connect to lower-degree nodes (note that the slope of the k_{nn} distribution is negative), which indicates that OSS SN have a mixture of degree connectivity. This mixing between different degree nodes shows that there are interactions between experts and newbie developers which is very positive in an open collaborative environment.

On the other hand, the OSS AN k_{nn} distribution shows an opposite trend where similar degree nodes tend to connect with each other as shown in Fig. 5 for both Ohloh and Github. We note that if every node had the same number of project affiliations as its neighbors, then Fig. 5 would be symmetric on the diagonal. Since it is not, then this result shows that developers who participate in large team size projects tend to participate in other large team size projects. Also, the same trend is true for developers who participate in small or medium team size projects. In the next section, we further investigate the mixing pattern by introducing the assortativity coefficient measure to quantify the degree of mixing.

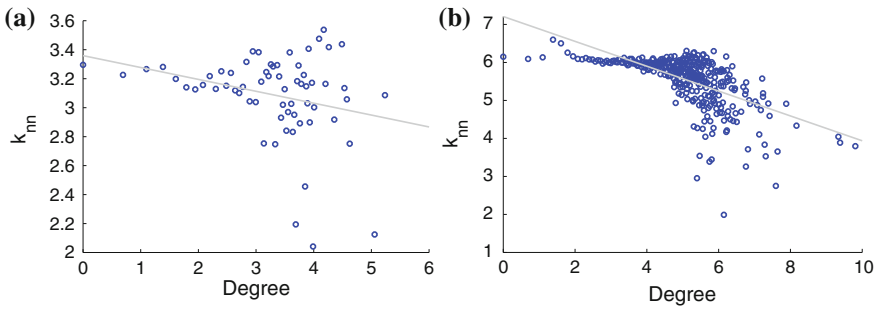


Fig. 4 Ohloh (*left*) and Github (*right*) k_{nn} functions show that, in OSS SN, high degrees tend to connect to lower degrees which indicate a mixture of degree connectivity.

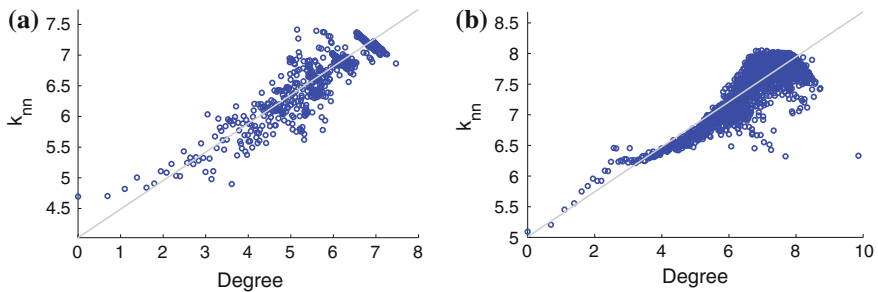


Fig. 5 Ohloh (*left*) and Github (*right*) k_{nn} functions show that, in OSS AN, nodes tend to connect with other nodes of similar degree.

3.3.2 Assortativity

Assortativity is a measure of the likelihood for nodes to connect to other nodes with similar degrees. The assortativity coefficient r ranges between -1 and 1 . Positive assortativity coefficient means that nodes tend to connect to other nodes with similar degree, while negative assortativity coefficient means that nodes tend to connect to other nodes with different degrees. Table 4 shows the assortativity coefficient for OSS SN and AN for both Ohloh and Github. The OSS SN negative assortativity coefficients indicate that high-degree nodes tend to connect to low-degree nodes, while the OSS AN positive assortativity coefficients indicate that similar degree nodes tend to connect together, which confirms the observation of degree correlation

Table 4 Assortativity coefficients

	<i>Ohloh</i>	<i>Github</i>
Social Net.	-0.0177	-0.0465
Affil. Net.	0.5811	0.4817

discussed earlier. The small negative value of the assortativity coefficient for the OSS SN indicates that the degree mixture is not high between nodes.

4 Social Networking Factors and Developer Contribution Analysis

To discover the social factors that affect the contributors' participation and contribution in OSS projects, we conduct an extensive analysis to investigate the correlations between various social connectivity structures and the amount of contribution exerted by the contributors. We first conduct a global graph analysis, where we analyze the developers' connectivity in the whole community graph. In particular, we study the effect of node (developer) indegree, outdegree, betweenness and closeness centrality on the developer's contribution and commitment. Further, we conduct a local graph analysis, where we analyze the developers' connectivity in a team level graph. In particular, we study the effect of nodes' different tie patterns and tie strength on developers' amount of contribution and commitment. In the next two sections, we elaborate in more details every network analysis and every social factor and contribution measure.

4.1 Global Graph Analysis

We analyze the correlation between node indegree/outdegree, betweenness and closeness centrality against developers' contribution and commitment. The amount of contribution for each developer is estimated by two metrics: (i) the number of participations in distinct OSS projects, and (ii) the total number of commits to all OSS projects, where a commit is an update submitted to a project by a developer. Furthermore, a developer commitment is estimated by the number of months where a developer submitted at least one commit denoted by *months-work*. This metric is collected from Ohloh only since Github does not provide the number of active months for developers. We note that months-work is not simply the number of months between first and last commit for a developer. In particular, the number of idle months are not counted. Note that this information is hard to monitor and collect for a large number of developers, unless it is provided by the social platform provider.

4.1.1 Indegree and Outdegree Effect

Figure 6 shows the in/outdegree correlation with the number of participations for Ohloh and Github data. In particular, for each degree we take the average participation of nodes with that degree. Both indegree and outdegree show positive correlation

with the number of participations indicating that developers with high indegree or outdegree have more tendency to participate in more OSS projects.

Figure 7 shows the in/outdegree correlation with the total number of commits in OSS projects for Ohloh and Github datasets. In particular, for each degree we take the average commits of nodes with that degree. Similar to participations, both indegree and outdegree show positive correlation with the total commits indicating that developers with high indegree or outdegree have more tendency to commit more work in OSS projects.

In Figs. 6 and 7, we notice a bulky head for the correlation points distribution, which suggests a wider range of participation and total commits values for nodes with high in/outdegree. This is more obvious in Github dataset than Ohloh dataset since the amount of data in Github is much more than the amount of data in Ohloh. Since we are aggregating nodes based on in/outdegree and the node degree follows a power-law distribution, where low degree nodes are much more than high degree nodes, then the data points aggregated for low-degree nodes are much more than high-degree nodes which makes the results for low degree nodes more consistent than high degree nodes.

Finally, Fig. 8 shows the in/outdegree correlation with the total number of months-work in OSS projects for Ohloh data. For each degree we take the average months-work of nodes with that degree. Similar to participations and total commits, both

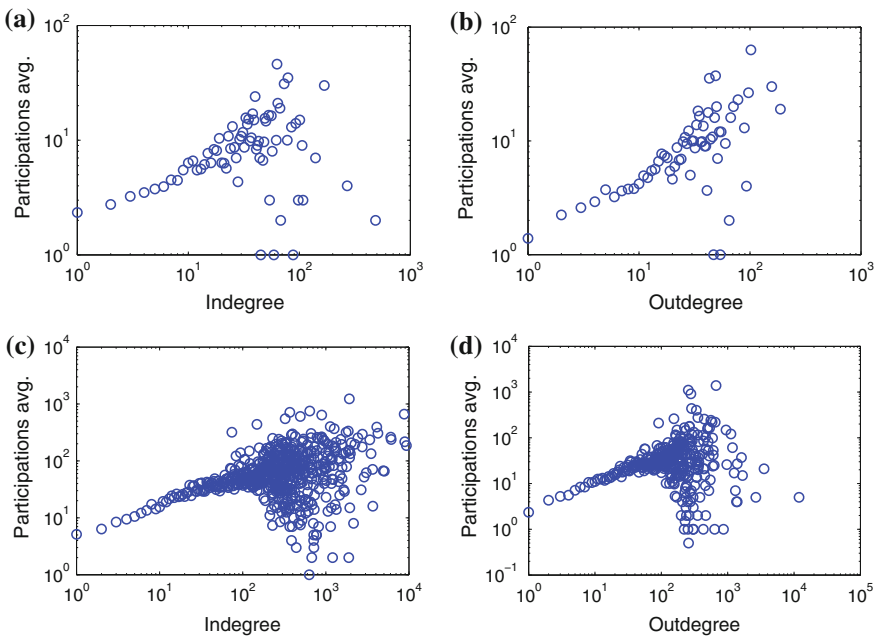


Fig. 6 Indegree and outdegree versus number of participation for Ohloh **a** and **b** and Github **c** and **d**

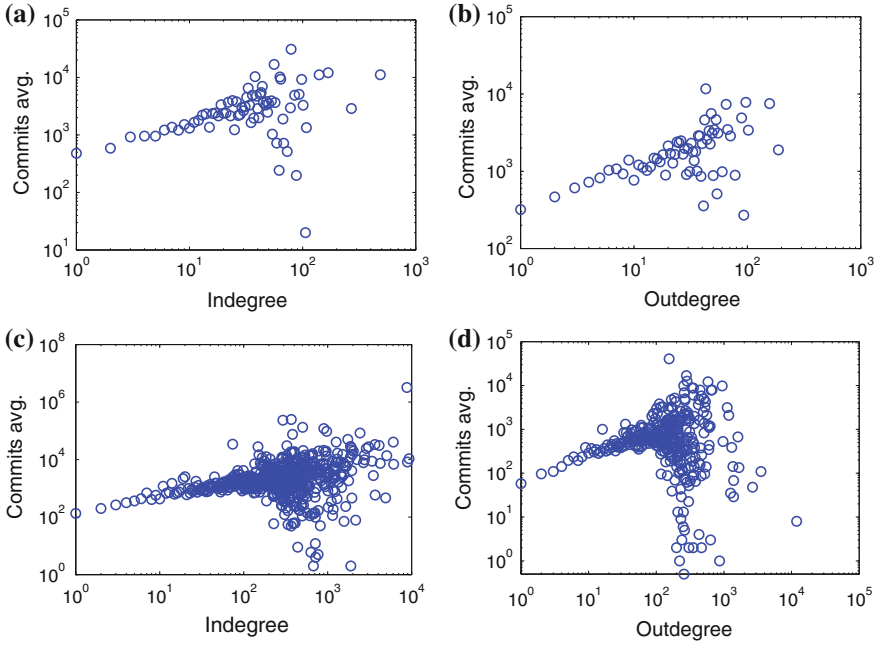


Fig. 7 Indegree and outdegree versus total number of commits for Ohloh **a** and **b** and Github **c** and **d**

indegree and outdegree show positive correlation with the total months-work indicating that developers with high indegree or outdegree have more tendency to stay longer time committing to OSS projects.

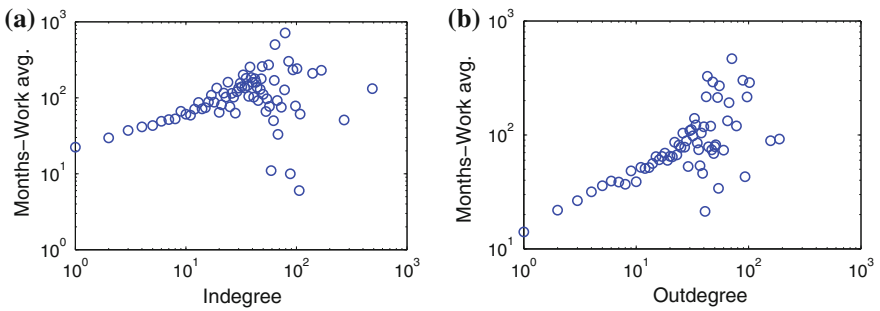


Fig. 8 Indegree (*left*) and outdegree (*right*) versus months-work for Ohloh.

4.1.2 Betweenness and Closeness Centrality Effect

In this section we analyze the effect of betweenness and closeness centrality on a developer amount of contribution and commitment. Recall that betweenness centrality for a node in a graph measures how many times this node acts as a bridge along the shortest path between two other nodes, and closeness centrality is the inverse of the sum of all distances between that node and all reachable nodes in the graph. In other words, a node with high closeness centrality has short path length, on average, to every other node and therefore can spread information fast. Therefore, a node with high betweenness centrality acts as a gate keeper, whereas a node with high closeness centrality acts as a hub in a social network.

Figure 9 shows how the number of participations correlating to the betweenness and closeness centrality of nodes in the Ohloh and Github datasets. In particular, for each participation amount we take the average betweenness and closeness of nodes with the same number of total participations. Both the betweenness and closeness show positive correlation with the number of participations indicating that developers with high betweenness or closeness have more tendency to participate in more OSS projects.

In Fig. 9, we also notice a bulky head for the correlation points distribution which suggests a wider range of betweenness and closeness centrality values for nodes

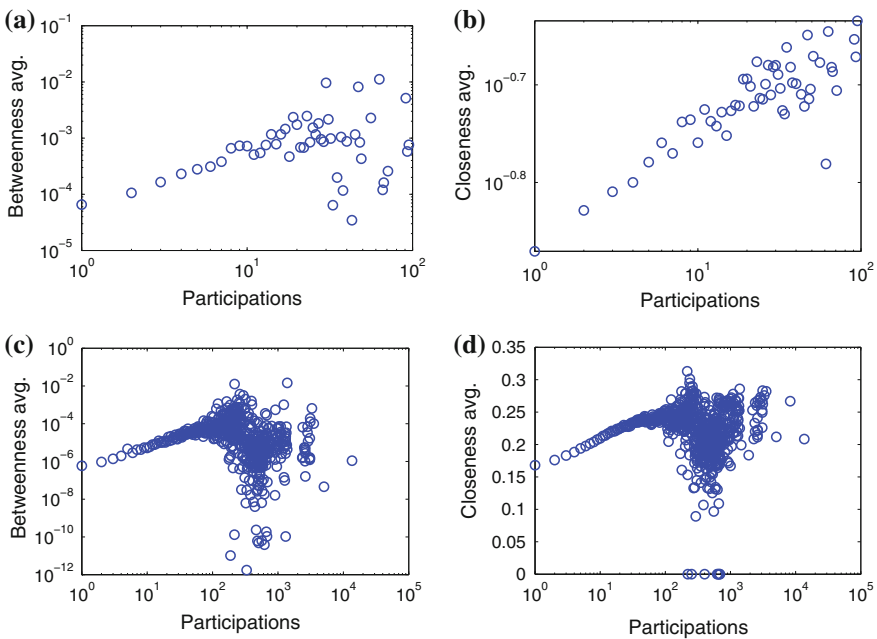


Fig. 9 Betweenness and closeness centrality versus participations for Ohloh **a** and **b** and Github **c** and **d**

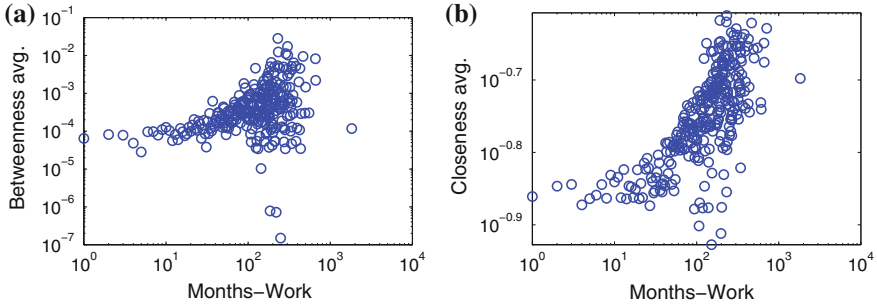


Fig. 10 Betweenness (*left*) and closeness (*right*) centrality versus months-work for Ohloh

of high participation. Similar to the observation in previous section, since we are aggregating nodes based on the total number of participations and the participation distribution follows a power-law distribution where few developers have too many participations while the majority have few participations [10], then the data points aggregated for the low participation nodes are much more than the high participation nodes. This makes the results for the low participation nodes more consistent than the high participation nodes. Analysis on the Ohloh dataset (Figs. 9a, b) shows more clear evidence of the positive correlation between participations and the betweenness and closeness centrality.

Finally, Fig. 10 shows the number of months-work correlation with nodes' betweenness and closeness centrality for Ohloh data. In particular, for each months-work we take the average betweenness and closeness of nodes having the same number of months-work. Both betweenness and closeness show positive correlation with the number of months-work indicating that developers with high betweenness or closeness have more tendency to stay longer time committing to OSS projects. However, the node closeness effect seems more than the node betweenness effect.

Interestingly, we did not find a clear evidence of a correlation between either betweenness or closeness centrality and the total number of commits. We argue that developers with high betweenness or closeness, usually, have administrative roles in OSS projects where it imposes high number of social ties. Following this argument, administrative contributors have a higher number of participations and a smaller number of commits than non-administrative contributors. Also, based on Fig. 10, we argue that administrative contributors stay longer time committed to their projects than non-administrative contributors.

4.2 Local Graph Analysis

In this section, we investigate the impact of social connectivity and the ties strength between contributors in a team on the amount of contribution and commitment

observed on those team members. The connectivity of contributors in a team is measured using the Local *Clustering Coefficient* (CC), whereas the ties strength is represented by the frequency of co-participation between the contributors. Moreover, The amount of contribution from a developer in a project is measured by the average number of commits made per month (denoted by commits/month), and the commitment by a developer to a project is measured by the number of active months where a developer submitted at least one commit to a particular project (denoted by months-work). The amount of contribution shows the average amount of work produced by a developer per month in a project, whereas the commitment shows the period of time that a developer stayed committed and active in a project. Note that these metrics are similar to the metrics used in the global network analysis. However, the metrics in the local network analysis estimates the contribution and commitment of developers corresponding to a particular project instead of all participated-in projects as in the global network analysis.

In details, the social factors are measured as follows: For each OSS project team, we extract the team subgraph from the OSS community social graph and, first, compute the connectivity for each developer i in a team T with the local *Clustering Coefficient* (CC), which represents the connectivity ratio between the acquaintances of developer i as in Eq. (1).

$$CC_T(i) = \frac{2(k_{N_i})}{|N_i|(|N_i| - 1)} \quad (1)$$

where N_i is the set of i 's neighbors (acquaintances) in T , and k_{N_i} is the number of links connecting these neighbors. Second, we compute a link weight between developers i and j (denoted by $w_{i,j}$) by calculating the frequency of co-participation between i and j and normalize it by dividing by the maximum link weight in the overall social graph, such that $w_{i,j} = [0, 1]$.

We observe the effects of social factors in various project team sizes. In particular, we group projects into the following team size ranges: [5,10), [10,25), [25,50), [50,75), [75,100) and [100 and above]. In this study, we analyze over 1300 OSS projects of different team sizes and topics. There are many more projects in the dataset, but we only consider the projects with team size of more than four developers to exhibit the different connectivity patters.

4.2.1 Degree of Connectivity Effect: Highly Versus Lowly Connected

This study analyzes the connected contributors and divides them into two sub-categories, Highly Connected (HC) and Lowly Connected (LC) besides the Not-Connected (NC) contributors. The HC contributors are those with $CC > 0$, i.e., some or all of a contributor's acquaintances are connected, forming one or several cliques pattern. The LC contributors are those with $CC = 0$, i.e., a contributor's acquaintances are not connected, forming a star, line or circle shape pattern. Finally, the NC contributors are those with no acquaintances. To proceed, we select the

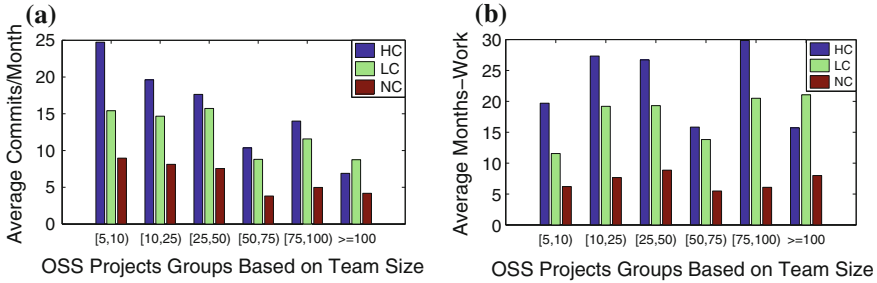


Fig. 11 Effect of degree of connectivity on **a** average Commits/Month and **b** average Months-Work for projects of various ranges of team sizes.

projects that show all the three types of connectivity patterns (HC, LC and NC) in order to conduct the comparison.

Figure 11 analyzes all projects with team sizes of more than four contributors according to the team size sets mentioned above. Figure 11a shows that the average commits per month is larger for HC contributors than those LC ones in most team sizes. Similarly, Fig. 11b shows that the average months-work is larger for HC contributors than those LC ones in most team sizes. Moreover, both figures show that the NC contributors make the lowest contribution and commitment. These results show a strong statistical evidence that, in general, HC contributors contribute more and are more committed to open projects than LC contributors in various team size ranges.

4.2.2 Ties Strength Effect: Strong Ties Versus Weak Ties

Next, we study the effect of tie strength on the amount of contribution and commitment, under different team size ranges. Since we measure the strength of a tie between two contributors by the frequency of their co-participations in projects, then one co-participation is counted as one unit of link weight between two developers. Accordingly, we consider a contributor belonging to the *Weak Tie set* if she has no more than two units of link weights. Contributors with more than two units of link weights are in the *Strong Tie set*.¹ Finally, contributors with no acquaintances are in the *No Tie set*. To proceed, we select the projects that show all the three types of tie weights (Strong, Weak and No Ties) in order to conduct the comparison.

Figure 12 analyzes all projects with team sizes of more than four contributors according to the team size sets mentioned above. Figure 12a shows that the average commits per month is larger for the Strong Ties contributors than those Weak Ties ones in most team sizes. Also, Fig. 12b shows that the average months-work is larger

¹We decided to have a threshold of two units of link weights to distinguish the weak ties from the strong ties because participants follow power law distribution [10] and most OSS projects have one to two participants.

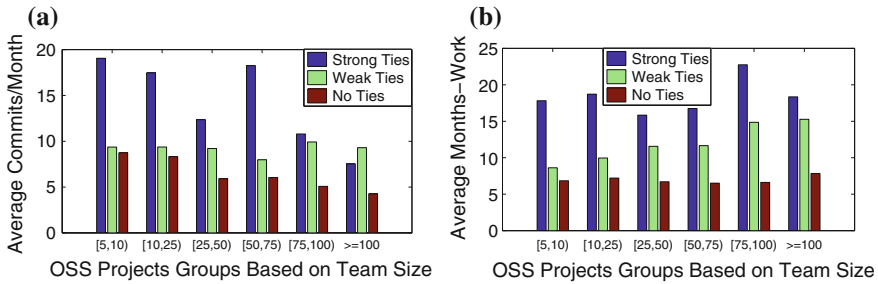


Fig. 12 Effect of tie strength on **a** average Commits/Month and **b** average Months-Work for projects of various ranges of team sizes.

for the Strong Ties contributors than those Weak Ties ones in all team sizes. These results show a strong statistical evidence that, in general, contributors connected with Strong Ties contribute more and are more committed to open projects than the Weak Ties and No Ties contributors.

In Figs. 11 and 12, most very large teams (those with one hundred and above members) demonstrate different trends, where LC and Weak Tie contributors show more contribution and commitment than HC and Strong Tie contributors. We argue that, in these large teams, contributors with high connectivity and strong social ties may have more administrative and controlling roles. On the other hand, contributors with low connectivity and weaker social ties may commit more time to contribute to the project since they may join big projects to gain experience and form connections with reputable developers. Moreover, Fig. 12 shows that the difference between Weak Ties and No Ties increases as the team size increases. This suggests that Weak Ties become important as the team size increases. We intend to investigate these different trends more in the future.

5 Related Works

There exist several works that analyze the OSS social network. Both [3] and [5] study the structure of a collaborator communication network extracted from the e-mail network and find that small teams have a centralized structure, whereas larger teams have a modular structure. Also, Surian et al. [19] study the structure of collaboration pattern among developers. However, they do not study the correlation between network structure and developers productivity, also, their network is not based on explicit social ties but rather an affiliation network. In contrast to the above works, our datasets include social networks based on explicit acquaintances, which facilitate a more clear study of the effect of direct social ties among developers on the success of OSS projects. In another work, Subramaniam et al. [18] study other factors that may affect the development activity, such as developer interest, user

interest, project activity (i.e., bug reports and user comments), project status (i.e., new project, development phase, maintenance phase) and project attributes (i.e., OSS license, operating system, programming language). On the contrary, we focus on the social tie factors of the contributors in the OSS projects. Moreover, Casaló et al. [4] collect information via questionnaires and study the correlation between developers commitment to their OSS project and the reputation of those projects. They conclude that OSS reputation have an indirect effect on the commitment of collaborators. In contrast to questionnaires, the contribution and commitment information in our study are collected from control version repositories that monitor the commits log for each developer.²

6 Conclusion

We conducted an extensive statistical social network analysis on more than a million of online volunteering developers. Our study included a global network analysis (i.e., OSS community level) and a local network analysis (i.e., team network level). The analysis shows that the OSS SNs and ANs follow a power-law distribution and the SNs exhibit small world properties. Also, the OSS SN degree mixing pattern shows that nodes with different degrees are connected together, suggesting interactions between experts and newbie developers. On the other hand, the OSS AN degree mixing pattern shows that developers who participate in projects of certain team size tend to participate to other projects of similar team size. Second, we show positive correlations between indegree, outdegree, betweenness and closeness centrality and the developers' contribution and commitment in OSS projects. Finally, the local network analysis revealed a strong statistical evidence that, in general, highly connected and strongly tied contributors are more productive than the low connected, weakly tied and not connected contributors. These results lead into better understanding of the OSS social factors that influence team productivity and commitment essential to the success of OSS projects. Moreover, this study paves the road for future recommendation systems related to online open collaborative projects to possibly recommend experts and teams of high productivity and expertise, which is our future work.

References

1. Antwerp MV, Madey GR (2010) The importance of social network structure in the open source software developer community. In: HICSS, IEEE Comput Soc, pp 1–10
2. Backstrom L, Huttenlocher D, Kleinberg J, Lan X (2006) Group formation in large social networks: membership, growth, and evolution. In: Proceedings of the 12th ACM SIGKDD international conference on knowledge discovery and data mining, KDD '06, ACM, New York, pp 44–54

²The commits log is collected by *Ohloh.net* and *Github.com*.

3. Bird C, Pattison D, D'Souza R, Filkov V, Devanbu P (2008) Latent social structure in open source projects. In: Proceedings of the 16th ACM SIGSOFT international symposium on foundations of software engineering. SIGSOFT '08/FSE-16, ACM, New York , pp 24–35
4. Casaló LV, Cisneros J, Flavián C, Guinaliu M (2009) Determinants of success in open source software networks. *Ind Manage Data Syst* 109(4):532–549
5. Crowston K, Howison J (2003) The social structure of open source software development teams. *First monday*, 10(2)
6. Garousi V (2009) Investigating the success factors of open-source software projects across their lifetime. *J Software Eng Stud* 4:115
7. Hahn J, Moon JY, Zhang C (2008) Emergence of new project teams from open source software developer networks: impact of prior collaboration ties. *Informa Syst Res* 19(3):369–391
8. Hertel G, Niedner S, Herrmann S (2003) Motivation of software developers in open source projects: an internet-based survey of contributors to the Linux Kernel. *Res Policy* 32(7):1159–1177
9. Lee S-YT, Kim H-W, Gupta S (2009) Measuring open source software success. *Omega* 37(2):426–438
10. Madey G, Freeh V, Tynan R (2002) The open source software development phenomenon: an analysis based on social network theory. In: Proceedings of the Americas conference on information systems (AMCIS 2002), Dallas, Texas, pp 1806–1813
11. Mahadevan P, Krioukov D, Fall K, Vahdat A (2006) Systematic topology analysis and generation using degree correlations. In: Proceedings of the 2006 conference on applications, technologies, architectures, and protocols for computer communications, SIGCOMM '06, ACM, New York, pp 135–146
12. Michlmayr M, Hill BM (2003) Quality and the reliance on individuals in free software projects. In: Proceedings of the 3rd workshop on open source software engineering, Portland, Oregon, pp 105–109
13. Milgram S (1967) The small world problem. *Psychol Today* 61:60–67
14. Mislove A, Marcon M, Gummadi KP, Druschel P, Bhattacharjee B (2007) Measurement and analysis of online social networks. In: Proceedings of the 7th ACM SIGCOMM conference on Internet measurement. IMC '07, ACM, New York, pp 29–42
15. Newman MEJ (2003) Mixing patterns in networks. *PRE* 67(2):026126
16. Robles G, Gonzalez-Barahona JM, Michlmayr M (2005) Evolution of volunteer participation in libre software projects: evidence from Debian. In: Proceedings of the first international conference on open source systems, Genova, Italy, pp 100–107
17. Samoladas I, Gousios G, Spinellis D, Stamelos I (2008) The sqo-oss quality model: measurement based open source software evaluation. In: OSS, pp 237–248
18. Subramaniam C, Sen R, Nelson ML (2009) Determinants of open source software project success: a longitudinal study. *Decis Support Syst* 46(2):576–585
19. Surian D, Lo D, Lim E-P (2010) Mining collaboration patterns from a large developer network. In: Antoniol G, Pinzger M, Chikofsky EJ (eds) WCRE, IEEE Comput Soc, pp 269–273
20. Wu J, Goh K-Y (2009) Evaluating longitudinal success of open source software projects: a social network perspective. In: HICSS, pp 1–10

Precise Modeling Rumor Propagation and Control Strategy on Social Networks

Yuanyuan Bao, Chengqi Yi, Yibo Xue and Yingfei Dong

Abstract In this paper, we first point out the limitations of the classic epidemic spreading model SIR in representing rumor spreading processes, and then identify the effect of public opinions in dividing Infected states (I) into Positive infected (P) states and Negative infected (N) states. Based on this observation, we propose a new SPNR model. To evaluate the proposed model, we compare the simulation results with the real-world data obtained on Sina Weibo, the largest micro-blogging tool in China. The results show that the new model effectively captures the rumor spreading process. Furthermore, to develop an effective rumor control strategy, we propose an opinion guidance rumor control strategy based on SPNR model. We compare different control strategies using real-world data sets and a synthetic network generated according to a scale-free model. The results show that the proposed strategy is effective in fighting rumor spreading.

Keywords Rumor propagation · Social networks · Information diffusion · Rumor control · Rumor dynamic

This paper is an extended version of [1].

Y. Bao · Y. Xue (✉)
Tsinghua National Lab for Information Science and Technology,
Tsinghua University, Beijing, China
e-mail: yiboxue@tsinghua.edu.cn

Y. Bao
e-mail: by51800@163.com

C. Yi
School of Computer Science and Technology,
Harbin University of Science and Technology, Harbin, China
e-mail: garnettyige@163.com

Y. Dong
Department of Electrical Engineering, University of Hawaii, Honolulu, USA
e-mail: yingfei@hawaii.edu

1 Introduction

Today's constant Internet connectivity and mobile devices dramatically changed our lifestyles. More and more people are using Blogs, Micro-blogs, RSSs, and social networks for communications. Based on recent data, Facebook has over 1.2 billion users and generates more than 70 billion pieces of contents per month. Twitter has over 0.5 billion users and generates 190 billion tweets per day. China has about 0.7 billion social network users. Sina Weibo has more than 0.5 billion users and generates more micro blogs than Twitter. As the rapid development and widely adoption of social networks have dramatically changed the way that people communicate with each other, a huge amount of real-time contents are generated in this new way of information diffusion. However, because of the flexibility of social networks and the lack of proper control, the quality and authenticity of information on social networks cannot be guaranteed. In particular, many rumors or inaccurate news disseminate quickly in social networks, and cause serious consequences to individuals and the society [2–4].

A rumor is unverified information [5–10], which is of local or current interest or importance and intended primarily for establishing belief/disbelief [11]. Social psychologists argue that rumors arise in contexts of ambiguity, potential threat, entertainment, alliance making, and wish fulfillment. The spreading of rumors has bad influences on the society and may cause unnecessary panics. Especially, malicious users actively spread false information on social networks. For example, during the super storm Sandy, many storm-related rumors were spread on Twitter and resulted in various issues. Therefore, how to identify, verify and control rumors is an urgent challenge.

Although current rumor control strategies have achieved some successes, more accurate rumor propagation models and more effective control strategies are still highly desired, due to the dynamics and complexity of fast-growing social networks. Although the classic epidemic spreading models are effective in addressing the virus spreading problem, we cannot simply apply these models (e.g., the SIR model) to represent rumor propagation, because rumor propagation on social networks is largely different from the spreading of common disease or viruses. Note that different people usually have different opinions about the same event. So the infected states of rumor should be further divided into two categories: *positive infected state* and *negative infected state*, different from common epidemic spreading. Furthermore, current rumor control strategies can also be improved. For example, the most frequently used strategy is user immunization, such as random immunization or targeted immunization. Both methods are less effective for rumor control, because the random immunization requires immunizing a large portion of network and the targeted immunization requires the complete knowledge of a network. Due to the lack of detailed analysis of control mechanisms, it is hard to identify effective methods for rumor control.

To address these issues, we propose a novel rumor spreading model, called the *SPNR model*, by splitting the infected states with two types of infected states: *positive*

infected (P) and *negative infected* (N). By analyzing spreading processes, we can determine the concrete relationships in model and obtain the outbreak threshold of rumors based on these relationships. We further analyze the effects of the parameters of the proposed model on the maximum value of steady state, the point of decline, and the life cycle of a rumor via simulations. We evaluate the proposed model with both simulations and real-world data obtained on Sina Weibo. Furthermore, we propose an *opinion guidance rumor control strategy* based on the proposed model. We then compare the proposed strategy with other existing strategies to show that the proposed strategy is effective in fighting the spread of rumor.

The highlights of this paper are:

- Based on practical observations, we divide the rumor infected state into two opposite infected states—positive infected and negative infected, and extend the SIR model into the SPNR model.
- We further obtain the rumor outbreak threshold based on the proposed model, and analyze how the parameters of the model affect the rumor propagation process.
- We also evaluate the proposed model on a scale-free network with MATLAB; we validate the model by comparing the simulation results with empirical results obtained on real-world data on Sina Weibo.
- We propose an opinion guidance rumor control strategy and evaluate its effectiveness by simulating rumor spread on real social networks and a synthetic network.

The remainder of this paper is organized as follows. In Sect. 2, we present the related work on rumor diffusion and rumor control on social networks. In Sect. 3, we first point out the shortcomings of existing rumor spreading models and rumor control strategies in theory and practice, and then present a method to address these problems via analysis and simulations. In Sect. 4, we evaluate the proposed model in rumor diffusion via simulations with real social networks data from Sina Weibo. In Sect. 5, we propose a rumor control strategy and show its effectiveness with simulations on real social networks and a synthetic network. In Sect. 6, we conclude this paper and discuss our further work.

2 Related Work

The growing impacts of social networks attract a lot of attentions in recent years. Many valuable conclusions have been reached by considering various factors such as network structure, community detection, information diffusion, and synchronization process. Meanwhile, rumor control remains a challenging issue.

Due to widely existence of rumors on social networks, researchers focus on the information diffusion process and try to identify the characteristics of information propagation and efficient control strategy. To improve the resistance of the community against rumors, it is essential to understand of the basic mechanism of rumor propagation and establish an appropriate strategy to achieve social stability.

The study of rumors has been strongly influenced by the study of epidemics, which has been extensively investigated since 1940s. An important work in this area is the *SIR* (Susceptible—Infected—Removed) *compartmental model* [12–14], by Kermack and McKendrick. In *SIR* model, there are three states including *susceptible state* representing individuals not yet infected with epidemics, *infected state* representing individuals who have been infected with epidemics and are capable of spreading the epidemics to those in the susceptible state, and *removed state* representing individuals who have been infected and then removed from the epidemics. Those in this state are not able to be infected again or to transmit the infection to others.

Based on the *SIR* model, Goffman and Newill proposed an analogy between spreading an infectious disease and the dissemination of information [15]. This analogy was formalized as the Daley-Kendall (DK) model [16]. The MT model is a variant of the DK model introduced by Maki-Thomsan [17]. In both models, the homogeneous people are divided into three groups: *susceptible*, *infected* and *recovered*. The difference between them is mostly in infected mechanism. However, these models are less effective in explaining rumor spreading in real world, because they do not consider the concrete relationships among entities, i.e., the topology of networks. In fact, the structure of social networks heavily influences rumor spreading and scale. Nekovee studied the rumor model on small world networks [18]; Pastor-Satorras examined the same problem on scale free networks. They obtained the critical point of rumor transmission intensity and spreading scale [19, 20]. Isham investigated the stochastic spread of epidemics and rumors on networks taking into account the structure of the underlying network at the level of the degree correlation function [21]. Many other investigations have been devoted to studying the so-called epidemic threshold, to determine the condition under which an epidemic will break out. Anurag modified the *SIR* model considering the rumor spread rate on the degree of the spreader and the informed nodes and obtained the rumor threshold and scale [22]. Wang et al. and its follow-up paper by Ganesh et al. found that, for the flu-like *SIS* model, the epidemic threshold for any arbitrary, real graph is determined by the leading eigenvalue of the adjacency matrix of the graph [23, 24]. Prakash et al. further discovered that the leading eigenvalue is the only parameter that determines the epidemic threshold for all virus propagation models in the standard literature [25]. Besides, there are several other well studied models for rumor spread in social networks found in the literature: Voter model [26, 27], linear threshold model, independent cascade model [28, 29] etc.

For rumor control three existing immunization schemes are random immunization, targeted immunization, and acquaintance immunization. Random immunization strategy works well in homogeneous networks while requiring immunizing a very large fraction of networks [30]. In contrast, targeted immunization of the most highly connected individuals is more effective for preventing rumor in scale free networks (similar with real-world social networks) [30]. However, it requires global information about the social network. The acquaintance immunization strategy is effective for both scale free networks and networks with bimodal degree distribution [31]. Several improved strategies were proposed based on these basic methods [32, 33]. Recently, Rudra proposed two methods combating the spread of rumors such as

delayed start model and beacon model. He found that the beacon model coupling the detection and an anti-rumor strategy by embedding agents was effective in fighting rumor spreading [29] Hayashi et al. derived the extinction conditions under random and targeted immunization methods for the SHIR model (Susceptible, Hidden, Infectious, and Recovered) [34]. Tong et al. proposed an effective node immunization strategy for the SIS model by approximately minimizing the leading eigenvalue [35] Briesemeister et al. studied the defending policy in power-law graphs [36]. Prakash et al. proposed effective algorithms to perform node immunization on time-varying graphs [37, 38] Gu et al. proposed a new rumor spreading SEIR with considering both the spreading characteristics of rumors in the real online social network and the epidemic dynamics models with latency. Based on the proposed SEIR model, an important acquaintance immunization strategy is obtained and simulation results show that this strategy is an optimal scheme to solve the inhibition of rumor spreading in the online social network [39].

Through the above analysis of immunization strategies, we notice that all these works focus on operating on the node level to affect the outcome of the dissemination. However, these strategies are mostly lack of specificity and with low efficiency. In order to make the rumor control strategy more accurate and reduce the pressure of the public opinion, some strategies focusing on edge level have been proposed [40]. These strategies modify the edges of the graph for slightly different purposes, such as, slowing down the influenza spreading [41], minimizing the average infection probability [42], evaluating and comparing the attack vulnerability [43], etc.

3 SPNR: The Proposed Model

3.1 Issues in Current Models and Control Strategies

Recent research on rumor spreading is mainly based on common epidemic spreading models such as SI, SIR and SIS. All these models only consider one infected state. Although these models are effective in modeling virus spreading, they are less effective in representing the rumor spreading, because the rumor propagation on social networks is very different from disease or virus spreading. In an actual rumor spreading process, people may have two opposite opinions to the same rumor: some people believe in the rumor, while some other people do not believe (called *anti-rumor*). To analyze the public opinions on a rumor and the transition between a positive infected state (believe) and a negative infected state (disbelieve) widely exists in a rumor spreading process, it is necessary to have such detailed classification. Therefore, we divide the infected states into two categories: a *positive* state and a *negative* state, different from the single infected state used in the classic epidemic spreading models. This concrete observation helps us improve the tradition SIR model and identify the rumor outbreak threshold.

Existing rumor control strategies have several limitations. The most frequently used control strategy is immunization by limiting the rumor infection rate to control rumor spreading. However, both of the random immunization and targeted immunization have difficulties in practice. Random immunization requires immunizing a large portion of a network, while targeted immunization requires the complete knowledge of a network. On the other hand, broadcasting information to the entire population by authorities is mainly used to debunk rumor in reality. However, its main weakness is the authority may have its own interest, and it is hard to convince common people. In summary, we do not have effective rumor control strategies that have good performance in theory and also simple to implement.

To address these issues, we propose the *SPNR* model (Susceptible—Positive Infected—Negative Infected—Recovered). We further analyze the spreading process of rumor under this new model, and obtain the rumor outbreak threshold. To better understand the fluctuation of steady states, we simulate the *SPNR* model on scale-free networks, and examine how the infection rate, the transition rate, and the recovery rate influence the steady states, the point of decline, and the life cycle of a rumor. To evaluate the proposed model, we analyze the actual rumor spreading data and compare empirical results with experimental results. Lastly, we propose an opinion guidance rumor control strategy and verify its effectiveness.

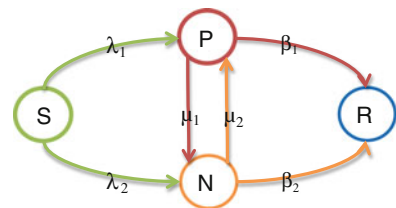
3.2 The Proposed *SPNR* Model

Consider the differences between virus spreading and rumor propagation, we should develop a specific propagation model for rumor spreading. As different people have different opinions for the same event, we should split the infected states into different levels, or simply two separate states according to whether one person support or oppose the information: positively infected and negatively infected.

In the proposed model, we have four states including *susceptible state* $S(t)$, *positive infected state* $I_p(t)$, *negative infected state* $I_n(t)$ and *recovered state* $R(t)$, as shown in Fig. 1. The rumor propagation model can be described as follows.

- i. When a susceptible user meets a positive spreader, it turns into a positive spreader with probability λ_1 . When a susceptible user meets a negative spreader, it turns into a negative spreader with probability λ_2 .

Fig. 1 States of *SPNR* model



- ii. When a positive spreader meets a negative spreader, it turns into a negative one with probability μ_1 . If the positive one meets a recovered user, it turns into a recovered user with probability β_1 .
- iii. When a negative spreader meets a positive spreader, it turns into a positive one with probability μ_2 . If the negative spreader meets a recovered user, it turns into a recovered user with probability β_2 .

Based on the above assumptions, we develop the formal relationship of rumor spreading on scale-free networks. We consider the SIR model defined on a network with general *connectivity distribution* $P(k)$ and a finite average connectivity $\langle k \rangle = \sum_k kP(k)$. To take into account the heterogeneity induced by the presence of nodes with different connectivity, we consider the time evolution of magnitudes $S_k(t)$, $I_{p,k}(t)$, $I_{n,k}(t)$ and $R_k(t)$, which are the density of susceptible, positive infected, negative infected, and recovered nodes of connectivity k at time t , respectively. These variables are connected by means of the normalization condition.

$$S_k(t) + I_{p,k}(t) + I_{n,k}(t) + R_k(t) = 1 \quad (1)$$

At the mean-field level, these densities satisfy the following set of coupled conditions:

$$\frac{dS_k(t)}{dt} = -\lambda_1 k \Omega_p(t) S_k(t) - \lambda_2 k \Omega_n(t) S_k(t) \quad (2)$$

$$\frac{dI_{p,k}(t)}{dt} = \lambda_1 k \Omega_p(t) S_k(t) - \beta_1 I_{p,k}(t) - (\mu_1 - \mu_2) I_{p,k}(t) I_{n,k}(t) \quad (3)$$

$$\frac{dI_{n,k}(t)}{dt} = \lambda_2 k \Omega_n(t) S_k(t) - \beta_2 I_{n,k}(t) + (\mu_1 - \mu_2) I_{p,k}(t) I_{n,k}(t) \quad (4)$$

$$\frac{dR_k(t)}{dt} = \beta_1 I_{p,k}(t) + \beta_2 I_{n,k}(t) \quad (5)$$

The factor Ω_p represents the probability that any given link points to a positive infected site. The factor Ω_n represents the probability that any given link points to a negative infected site. The probabilities of those two factors are given by

$$\Omega_p(t) = \frac{\sum_{k=1}^n k P(k) I_{p,k}(t)}{\langle k \rangle}, \quad \Omega_n(t) = \frac{\sum_{k=1}^n k P(k) I_{n,k}(t)}{\langle k \rangle} \quad (6)$$

Combined with the initial conditions $S_k(0) \approx 1$, $I_{n,k}(0) \approx 0$, $I_{p,k}(0) \approx 0$, $R_k(0) = 0$, we define the SPNR model with Eqs. (2)–(5) on any complex network with connectivity distribution $P(k)$.

By analyzing the outbreak condition, we can get the rumor outbreak threshold, such as which parameters must satisfy, and the rumor propagation scale to better evaluate the influence of a rumor. $R(t)$ is the density of recovered nodes at time t . Then, from the definition of $R_k(t)$, we can get that

$$R(t) = \frac{\sum_{k=1}^n kP(k)R_k(t)}{\langle k \rangle} \quad (7)$$

In order to simplify this model, we assume that λ_1 equals to λ_2 and β_1 equals to β_2 . Then, from Eqs. (5)–(7), yielding

$$\frac{dR(t)}{dt} = \beta(\Omega_p(t) + \Omega_n(t)) \quad (8)$$

Combined with Eq. (2), we can find that

$$\frac{dS_k(t)}{dt} = -\frac{\lambda}{\beta} S_k(t) \frac{dR(t)}{dt} \quad (9)$$

After computing the above Eq. (8), we can get

$$S_k(t) = (1 - I_{n,k}(0) - I_{p,k}(0))e^{-\frac{\lambda}{\beta}kR(t)} \quad (10)$$

To get a closed relation for the total density of infected individuals, it is more convenient to focus on the time evolution of the average magnitude $R(t)$. For this purpose, let us compute its derivative $dR(t)/dt$. Combining Eq. (1) with Eq. (8), we can find that

$$\begin{aligned} \frac{dR(t)}{dt} &= \beta \left(\frac{\sum_{k=1}^n kP(k)I_{p,k}(t)}{\langle k \rangle} + \frac{\sum_{k=1}^n kP(k)I_{n,k}(t)}{\langle k \rangle} \right) \\ &= \beta \frac{\sum_{k=1}^n kP(k)(1 - R_k(t) - S_k(t))}{\langle k \rangle} \\ &= \beta(1 - R(t) - \frac{\sum_k kP(k)}{\langle k \rangle} S_k(t)) \end{aligned} \quad (11)$$

Combined the above Eq. (11) with Eq. (10), we can find that

$$\frac{dR(t)}{dt} = \beta(1 - R(t) - \frac{\sum_k kP(k)}{\langle k \rangle} (1 - I_{n,k}(0) - I_{p,k}(0))e^{-\frac{\lambda}{\beta}kR(t)}) \quad (12)$$

We should analyze the equation:

$$f(x_0, i_n, i_p, \lambda, \beta) = \beta(1 - x_0 - \frac{\sum_k kP(k)(1 - i_{n,k} - i_{p,k})}{\langle k \rangle} e^{-\frac{\lambda}{\beta}kx_0}) = 0 \quad (13)$$

According to theory [39], it exists only one $x_0 > 0$ that makes $f(x_0, i_n, i_p, \lambda, \beta) = 0$. Let $R_0(i_n, i_p, \lambda, \beta)$ to be x_0 , then we get the equation:

$$1 - R_0(i_n, i_p, \lambda, \beta) - \frac{\sum_k k P(k)}{\langle k \rangle} (1 - I_{n,k}(0) - I_{p,k}(0)) e^{-\frac{\lambda}{\beta} k R_0(i_n, i_p, \lambda, \beta)} = 0 \quad (14)$$

After computing the derivative of Eq. (14) with respect to $R_0(i_n, i_p, \lambda, \beta)$, we can get

$$-1 + \frac{\lambda \sum_k k^2 P(k)}{\beta \langle k \rangle} (1 - I_{n,k}(0) - I_{p,k}(0)) e^{-\frac{\lambda}{\beta} k R_0(i_n, i_p, \lambda, \beta)} < 0 \quad (15)$$

If the assumption shown as the following is valid:

$$\text{if } \frac{\lambda}{\beta} > \frac{\langle k \rangle}{\langle k^2 \rangle}, \text{ then } \inf R_0(i_n, i_p, \lambda, \beta) = 0 \quad (16)$$

That is when λ/β is much bigger than $\langle k \rangle/\langle k^2 \rangle$, the infimum of $R_0(i_n, i_p, \lambda, \beta)$ equals to 0. It exists $i_{n,h}$, $i_{p,h}$ and $R_0(i_{n,h}, i_{p,h}, \lambda, \beta)$ and satisfies:

$$i_{n,h} \rightarrow 0, i_{p,h} \rightarrow 0, R_0(i_{n,h}, i_{p,h}, \lambda, \beta) \rightarrow 0 \quad (17)$$

Combining Eqs. (15) and (17), we have

$$\frac{\lambda}{\beta} < \frac{\langle k \rangle}{\sum_k k^2 P(k)} = \frac{\langle k \rangle}{\langle k^2 \rangle} \quad (18)$$

The conclusion in Eq. (18) is opposite to the assumption in Eq. (16). So we conclude that if $\lambda/\beta > (\langle k \rangle/\langle k^2 \rangle)$, the infimum of $R_0(i_n, i_p, \lambda, \beta)$ will be greater than 0. That is to say, the spreading threshold is $\langle k \rangle/\langle k^2 \rangle$.

3.3 Simulation of SPNR Model

We simulate this rumor propagation process based on the definition of the SPNR model with MATLAB and observe how the steady states fluctuate.

To make the simulation more accurate, we first generate a scale-free network as close as possible to actual social networks. We collect 0.15 billion users of Sina Weibo and calculate the degree distribution as shown in Fig. 2. The degree distribution follows a power law distribution, and the index of power law is 1.36.

We generate a scale-free network following the scale free algorithm and obtain a network with the index of power law closely about 1.36 as shown in Fig. 3. Based on this network, we simulate the rumor spreading process following Algorithm 1 as shown in Fig. 4.

In situation 1, we assume $\lambda_1 = 0.2$, $\lambda_2 = 0.2$, $\beta_1 = 0.2$, $\beta_2 = 0.1$, $\mu_1 = 0.2$, $\mu_2 = 0.1$, we can simulate the rumor spreading process as shown in Fig. 5, and

Fig. 2 Degree distribution of Sina Weibo

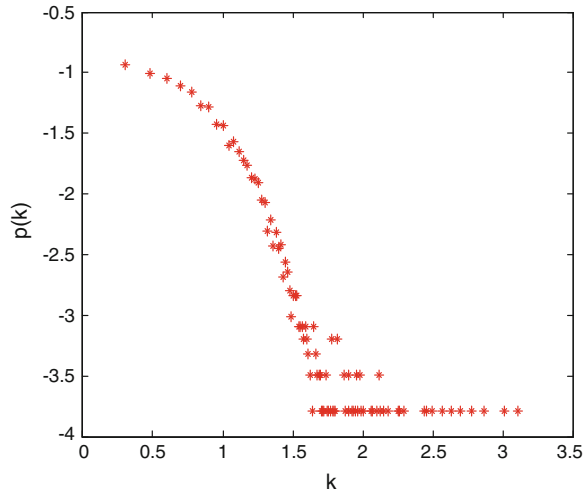
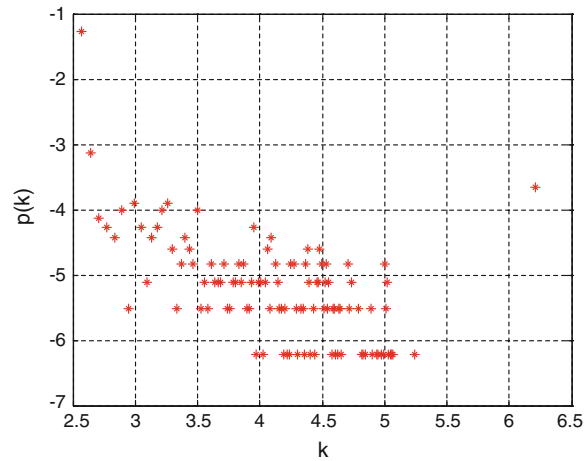


Fig. 3 Degree distribution of synthetic network



obtain the changes of steady states. In this case, the negative infected state increases more quickly than the positive infected state. The maximum number of negatively infected users is also higher than that of the positively infected users because the positive transition rate μ_1 is higher than the negative transition rate μ_2 , and the negative immunization rate β_1 is lower than the positive immunization rate β_2 . In situation 2, we assume $\lambda_1 = 0.5, \lambda_2 = 0.5, \beta_1 = 0.02, \beta_2 = 0.02, \mu_1 = 0.2, \mu_2 = 0.3$, the simulation results are shown in Fig. 6. In this case, the positive infection rate always much higher than the negative infection rate because of the negative transition rate μ_2 is much higher than the positive transition rate μ_1 . Through these simulations, we can conclude that the parameters of the SPNR model, such as the infection rate, the

Algorithm 1 Algorithm of the SPNR Model

Input:
Positive infection rate λ_1 , Negative infection rate λ_2
Positive immunization rate β_1 , Negative immunization rate β_2
Positive transition rate μ_1 , Negative transition rate μ_2
State of each node: B
Node of susceptible state: $B(1,i)=0$
Node of positive infected state: $B(1,i)=1$
Node of negative infected state: $B(1,i)=-1$
Node of recovered state: $B(1,i)=2$

Output:
State of all nodes after time interval t: B

1. **Initialization of the adjacent matrix and original state**
 - 1) Generating Scale-free Network: $S=\{V, E\}$, adjacent Matrix: A
 - 2) Assuming the original state: $B=[1, -1, 0, \dots, 0]$
2. **while (interval < t) do**
3. **State how the node accomplish the transitions when affected by positive infected state nodes**
 - 1) positive_infected= find($B(1,:) == 1$)
 - 2) **for** i=1:length(positive_infected)
 - 3) positive_nonzero = find(A(positive_infected (1,i),:))
 - 4) **for** j=1:length(positive_nonzero)
 - 5) **switch** B(1, positive_nonzero (1,j))
 - 6) case 0:
 - 7) transfer to 1 with λ_1
 - 8) case -1:
 - 9) transfer to 1 with μ_2
 - 10) **end switch**
 - 11) **end for**
 - 12) **end for**
4. **State how the node accomplish the transitions when affected by negative infected state nodes**
 - 1) negative_infected= find($B(1,:) == -1$)
 - 2) **for** i=1:length(negative_infected)
 - 3) negative_nonzero= find(A(negative_infected (1,i),:))
 - 4) **for** j=1:length(negative_nonzero)
 - 5) **switch** B(1, negative_nonzero (1,j))
 - 6) case 0:
 - 7) transfer to -1 with λ_2
 - 8) case 1:
 - 9) transfer to -1 with μ_1
 - 10) **end switch**
 - 11) **end for**
 - 12) **end for**
4. **State how the node of positive infected state accomplish the transitions to recovered state**
 - 1) positive_infected= find($B(1,:) == 1$)
 - 2) **for** i=1:length(positive_infected)
 - 3) transfer to 2 with β_1
 - 4) **end for**
5. **State how the node of negative infected state accomplish the transitions to recovered state**
 - 1) negative_infected= find($B(1,:) == -1$)
 - 2) **for** i=1:length(negative_infected)
 - 3) transfer to 2 with β_2
 - 4) **end for**
6. **end while**
7. **return B**

Fig. 4 Algorithm of the SPNR model

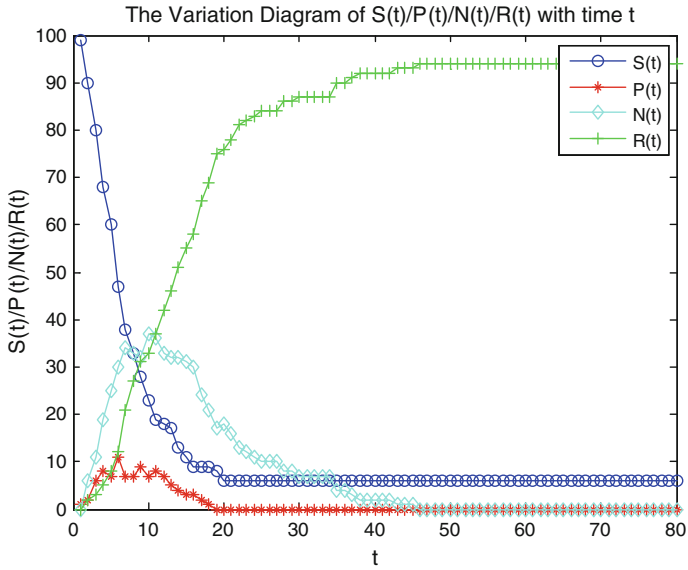


Fig. 5 Rumor spreading in SPNR model ($\lambda_1 = 0.2, \lambda_2 = 0.2, \beta_1 = 0.2, \beta_2 = 0.1, \mu_1 = 0.2, \mu_2 = 0.1$)

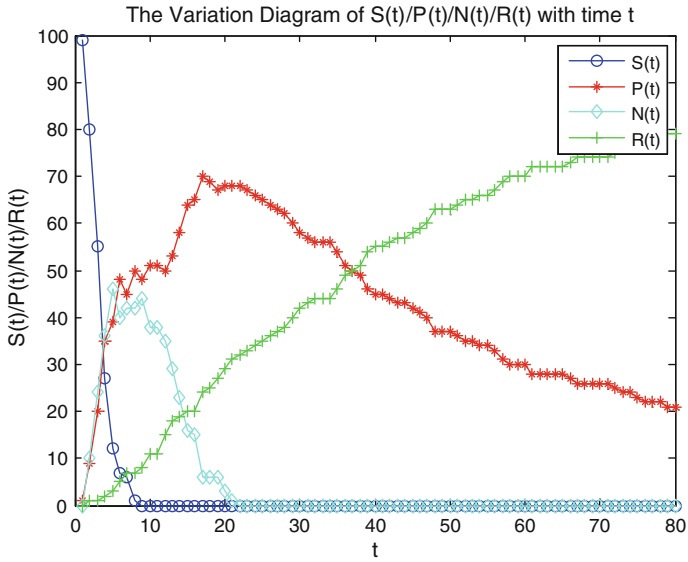


Fig. 6 Rumor spreading in SPNR model ($\lambda_1 = 0.5, \lambda_2 = 0.5, \beta_1 = 0.02, \beta_2 = 0.02, \mu_1 = 0.2, \mu_2 = 0.3$)

transition rate, and the immunization rate play a crucial role in rumor spreading. So, it is essential to figure out how the parameters affect the rumor spreading process.

3.4 Parameters of SPNR Model

Three main parameters of the SPNR model are the infection rate, the transition rate, and the immunization rate. These parameters have huge impact on the rumor spreading process. The most important properties of spreading are:

- The maximum value of steady state $I_p(t)$: M_p
- The point of decline $I_p(t)$: T_p
- The life cycle of a rumor $I_p(t)$: L_p

We examine how the infection rate, the transition rate, and the immunization rate affect the above properties of spreading by simulations.

- Reducing the positive infection rate results in the quick drop of M_p , as shown in Fig. 7. However, this method does not help for quickly reaching the point of decline and reducing the life span of a rumor.
- When we reduce the negative transition rate, M_p decreases sharply as shown in Fig. 8. At the same time, the point of decline and the life span are also heavily affected;

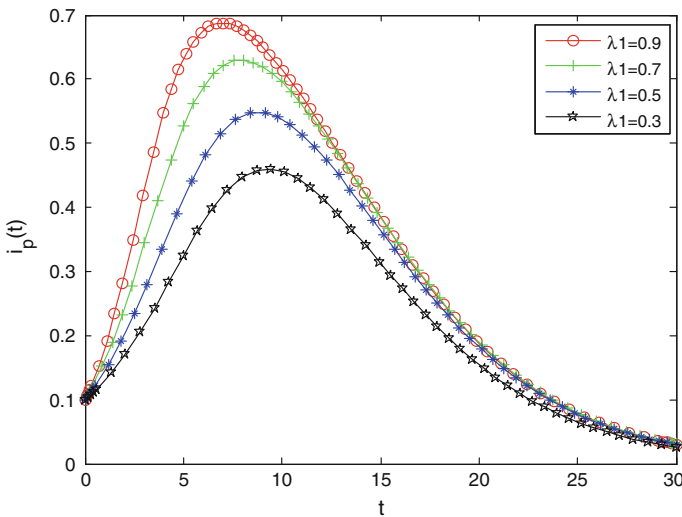


Fig. 7 The relations between $I_p(t)$ and λ_1

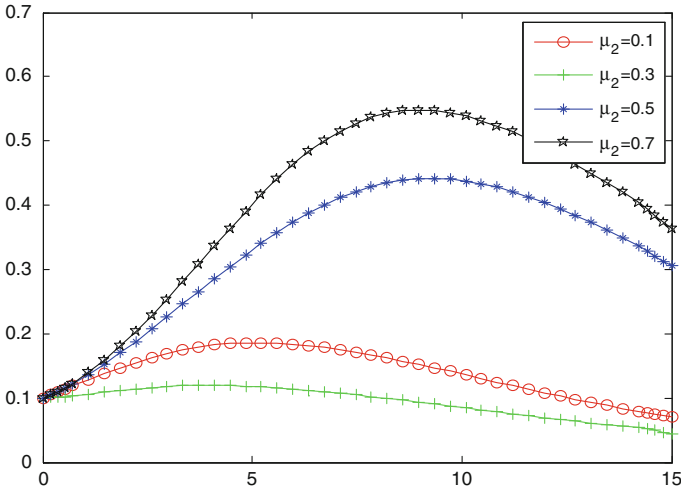


Fig. 8 The relations between $I_p(t)$ and μ_2

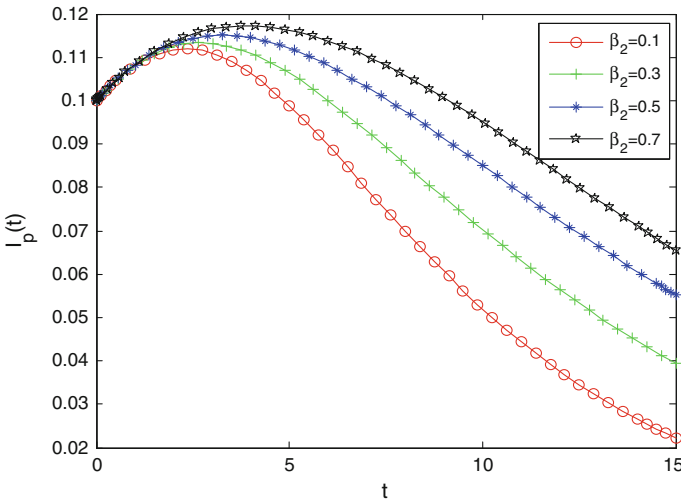


Fig. 9 The relations between $I_p(t)$ and β_2

- When we reduce the negative immunization rate, the life span of a rumor L_p decreases sharply as shown in Fig. 9. However, this method does not help in quickly reaching the point of decline and reducing M_p

4 Empirical Investigation

4.1 Data Description and Visualization

We collected a large set of data from Sina Weibo. The total quantity of micro blog topics is about 5,000,000 and the total retweets of these topics are more than 0.9 billion. Using sentiment analysis and manual selection to classify rumors in micro blogs, we found ten typically rumors in 2013. The basic parameters of each rumor are shown in Table 1. Each rumor is marked by the unique *statusid*, with a *post-time*. The column of *repost* represents the total amount of retweets caused by a rumor. The column of *anti-believe* represents the time when the first retweet of anti-believe appears. For example, the rumor posted at 08:06, July 25, 2013 with *statusid* 3603860545131642 has 71,040 reposts, and the first repost of anti-believe is at 20:47, July 27, 2013.

We use Gephi to visualize the rumor propagation process and mark spreading nodes with different colors according to the amounts of repost. After applying for the layout of Yifan Hu, which is a graph drawing algorithm based on a force-directed algorithm and can unfold a complete hierarchical community structure [44], the rumor propagation situation can be visualized clearly. Only considering the amounts of retweets, the spreading process of a rumor with *statusid* 3547977337564354 is shown in Fig. 10.

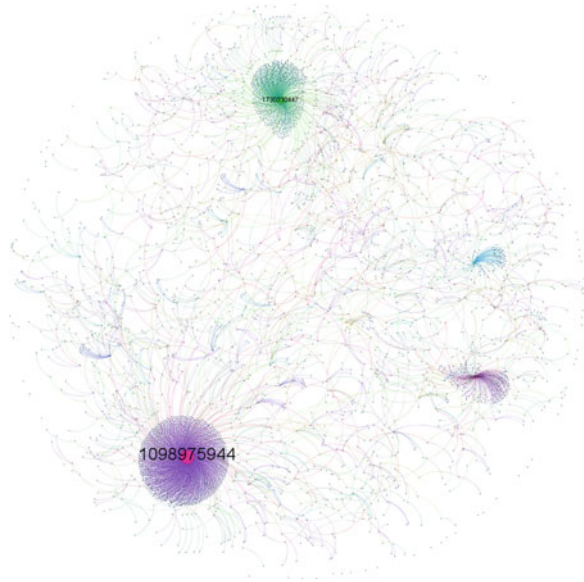
After analyzing the spreading process, we can see:

- In the rumor spreading, only a small number of users have huge reposts, in contrast with a large amount of users having no repost.
- If the influential users initially believe in rumor and later realize that it is a rumor, they will post another anti-rumor repost. Both types of micro-blog will always have a large number of reposts.

Table 1 Basic parameters of rumor

Statusid	Repost	Post-time	Anti-believe
3603860545131642	71,040	2013-07-25 08:06	2013-07-27 20:47
3615973581983273	38,320	2013-08-27 18:19	2013-08-27 20:57
3604623678104048	11,537	2013-07-27 10:38	2013-07-27 19:56
3625703373470078	11,297	2013-09-23 14:41	2013-09-24 09:43
3604581294714902	10,009	2013-07-27 07:50	2013-07-29 13:24
3553697869041863	9581	2013-03-08 21:57	2013-03-09 08:32
3606421822581559	8504	2013-08-01 09:43	2013-08-03 09:55
3547977337564354	6699	2013-02-21 03:06	2013-02-21 06:43
3535580233029694	5698	2013-02-21 01:52	2013-02-21 21:07
3553516544823295	5575	2013-03-08 09:57	2013-03-08 23:20

Fig. 10 Rumor spreading trend (Statusid = 3547977337564354)



In data processing, we delete the retweets concerning to advertisements, marketing and some other meaningless contents. Furthermore, in order to demonstrate the proposed model, we restrict the retweets into positive and negative attitudes according to the contents. We distinguish whether one user believes in rumor (positive infected) or anti-rumor (negative infected) via sentiment analysis. In order to visualize rumor propagation with two different infected states, we mark different nodes with different color, red representing positive infected while blue representing negative infected. The detailed situation is shown in Fig. 11.

After analyzing the spreading process, we can see:

- The more influence the user have, their followers will believe in what they repost, the public opinion is consist with their followings.
- The same user can make two exploring points. If we want to control the rumor, we can make another famous user an anti-believer of the rumor. It will make the overall public opinion much healthier.
- From this figure, we can find which one make the positive infected increase quickly and which one make the negative infected increase quickly.

4.2 Comparisons Between Experimental Results and Empirical Results

We analyze the rumor spreading in real social networks, particularly the ratio of susceptible persons, positive infected persons, negative infected persons, and recovered

Fig. 11 Rumor spreading trend with public opinion (Statusid = 3547977337564354)

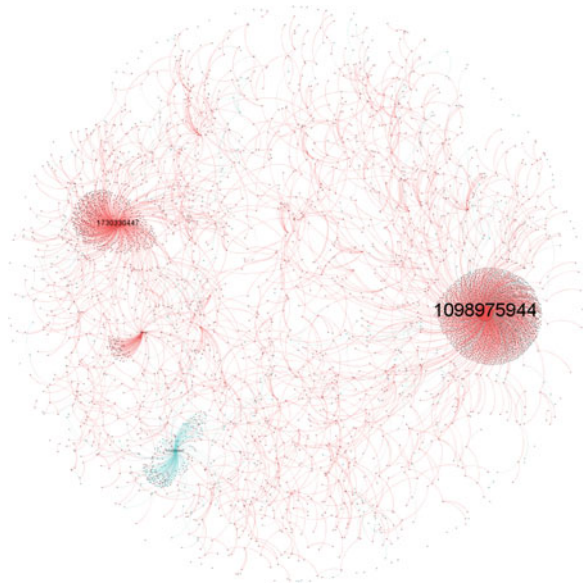
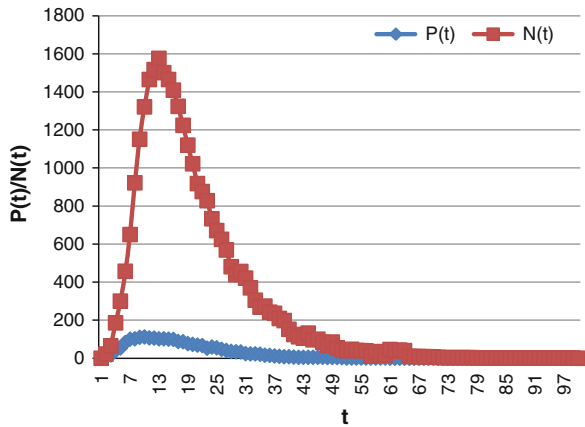


Fig. 12 Trends of positive infected and negative infected (Statusid = 3547977337564354)



persons. Considering the rumor with *statusid* 3547977337564354 as example, the ratios of the positive infected state and the negative infected state fluctuate with time, as shown in Fig. 12.

Through the realistic rumor propagation, we can get the spreading parameters. With these parameters, we simulate the rumor in a scale-free network and obtain the rumor propagation situation as shown in Fig. 13. In order to compare the experimental results with the empirical results above, we firstly unify the horizontal and vertical coordinates, and then we calculate the mean square errors between the exper-

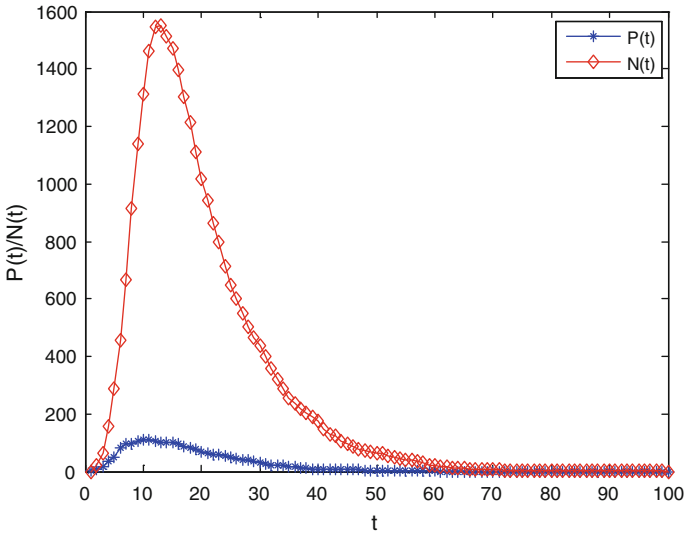


Fig. 13 Rumor spreading in modified SIR model ($\lambda_1 = 0.1, \lambda_2 = 0.1, \beta_1 = 0.1, \beta_2 = 0.1, \mu_1 = 0.1, \mu_2 = 0.1$)

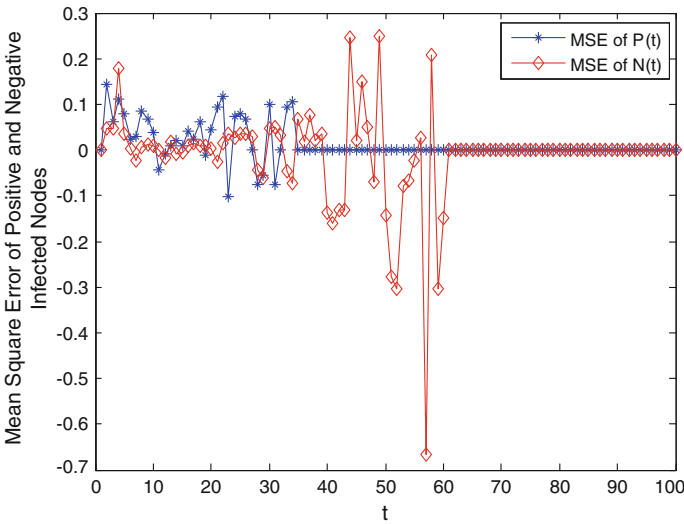


Fig. 14 Mean square errors between experimental and empirical results

imental results and empirical results. The mean square errors of both $P(t)$ and $N(t)$ are shown in Fig. 14. The mean square errors are all below 1%, which confirms the efficiency of the proposed model in representing the rumor propagation process.

5 Rumor Control Strategy

Two main rumor control schemes are random control and targeted control. They have serious limitations as discussed in the above. Random immunization requires immunizing a large portion of a network. Targeted immunization requires knowing the most highly connected individuals for effectively preventing rumor in a scale free network (similar with a real-world social network). Such precise knowledge about the social network is not easy to obtain. As a result, we do not have effective rumor control strategies with good performance and simple in implementation. Recently, Gao et al. demonstrate that the public opinion guide can significantly affect the concern about the event, and lead to rapid increase of propagation velocity. Furthermore, the correlation analysis shows that the more developed areas tend to be more synchronized, which suggests the responses of these areas to the public opinion guide may be faster. And conclude that enhancing the guide of public opinion in developed areas can help our government to control the spread of online public opinion [45].

In the above discussion, we conclude that public opinion has a significant impact on rumor spreading. If we can make as many as people not believing in a rumor to affect the positive infected people automatically, we will debunk the rumor more effectively. Based on the SPNR model, we find that we can guide the public opinion by inserting some opinion guidance nodes. When a rumor is spreading in a social network, we use the opinion guidance nodes to post anti-rumor tweets to debunk the rumor. In this paper, we propose an opinion guidance control strategy, and the detailed algorithm is presented in Fig. 15. The main steps of opinion guidance control strategy are:

- (1) Insert a number of opinion guidance nodes and update the state of each node. Make the opinion guidance nodes connect to all the other nodes of the network and update the adjacent matrix of the network.
- (2) Transition process: if the susceptible node connects with the opinion guidance nodes, the susceptible node transfers into a negative infected node.
- (3) Recovery process: if the positive node connects with the opinion guidance nodes, it will be turned into a recovered node.

To compare different control strategies, we conduct experiments with the data sets obtained from Twitter and Sina Weibo and a synthetic scale-free network. In Twitter and Sina Weibo, we choose about 2,000 users as the research subjects. The number of relationships in the Twitter data set is 182,561, and the number of relationships in the Sina data set is 11,072. We generated a scale free network following the Barabasi Model, which also has 2,000 users. The degrees of each node in the above three networks are shown in the Figs. 16a, 17a and 18a. The degree distributions of the three networks are shown in Figs. 16b, 17b and 18b.

To compare different control strategies, we conduct experiments with the data sets obtained from Twitter and Sina Weibo and also a synthetic scale-free network. In Twitter and Sina Weibo, we choose about 2,000 users as the research subjects. We also collect all the relationships between the users. The number of relationships in

Fig. 15 Algorithm of opinion guidance control strategy

Algorithm 2 Opinion Guidance Control Strategy

Input:
The number of opinion guidance nodes: ogn ; State of Nodes: B
Positive infection rate λ_1 ; Negative infection rate λ_2
Positive immunization rate β_1 ; Negative immunization rate β_2
Positive transition rate μ_1 ; Negative transition rate μ_2
Negative guidance rate α_1 ; Immunized guidance rate α_2
Node of susceptible state: $B(1,i)=0$; Node of recovered state: $B(1,i)=2$
Node of positive infected state: $B(1,i)=1$
Node of negative infected state: $B(1,i)=-1$
Node of opinion guidance state: $B(1,i)=-2$

Output:
State of all nodes after time interval t: B

1. **Initialization of the adjacent matrix and original state**
 - 1) Generating Scale-free Network: $S=\{V, E\}$, adjacent Matrix: A
 - 2) Update A to a new A with inserting opinion guidance node
 - 3) Assuming the original state: $B=[-2, -2, \dots, 1, -1, 0, \dots, 0]$
2. **while (interval < t) do**
3. **State how the node accomplish the transitions when affected by positive infected state nodes**
 - 1) positive_infected= find($B(1,:) == 1$)
 - 2) **for** i=1:length(positive_infected)
 - 3) positive_nonzero = find(A(positive_infected (1,i),:))
 - 4) **for** j=1:length(positive_nonzero)
 - 5) **switch** B(1, positive_nonzero (1,j))
 - 6) case 0:
 - 7) transfer to 1 with λ_1
 - 8) case -1:
 - 9) transfer to 1 with μ_2
 - 10) **end switch**
 - 11) **end for**
 - 12) **end for**
4. **State how the node accomplish the transitions when affected by negative infected state nodes**
 - 1) negative_infected= find($B(1,:) == -1$)
 - 2) **for** i=1:length(negative_infected)
 - 3) negative_nonzero= find(A(negative_infected (1,i),:))
 - 4) **for** j=1:length(negative_nonzero)
 - 5) **switch** B(1, negative_nonzero (1,j))
 - 6) case 0:
 - 7) transfer to -1 with λ_2
 - 8) case 1:
 - 9) transfer to -1 with μ_1
 - 10) **end switch**
 - 11) **end for**
 - 12) **end for**
4. **State how the node of positive infected state accomplish the transitions to recovered state**
 - 1) positive_infected= find($B(1,:) == 1$)
 - 2) **for** i=1:length(positive_infected)
 - 3) transfer to 2 with β_1
 - 4) **end for**
5. **State how the node of negative infected state accomplish the transitions to recovered state**
 - 1) negative_infected= find($B(1,:) == -1$)
 - 2) **for** i=1:length(negative_infected)
 - 3) transfer to 2 with β_2
 - 4) **end for**
6. **State how the node accomplish the transitions when affected by opinion guidance nodes**
 - 1) guidance_nodes= find($B(1,:) == -2$)
 - 2) **for** i=1:length(guidance_nodes)
 - 3) guidance_nonzero = find(A(guidance_nodes (1,i),:))
 - 4) **for** j=1:length(guidance_nonzero)
 - 5) **switch** B(1, guidance_nonzero (1,j))
 - 6) case 0:
 - 7) transfer to -1 with α_1
 - 8) case 1:
 - 9) transfer to 2 with α_2
 - 10) **end switch**
 - 11) **end for**
 - 12) **end for**
7. **end while**
8. **return B**

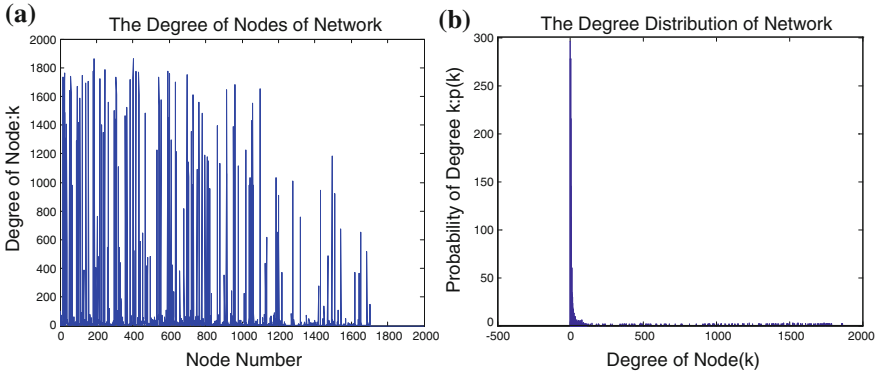


Fig. 16 Node degree and degree distribution of twitter data set

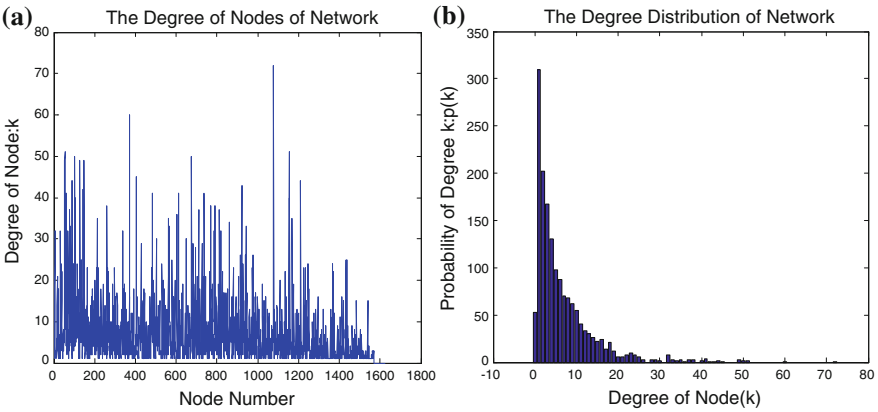


Fig. 17 Node degree and degree distribution of sina weibo data set

the Twitter data set is 182,561, and the number of relationships in the Sina data set is 11,072. We generated a scale free network following the *Barabasi* Model, which also has 2,000 users. The degrees of each node in the above three networks are shown in the Figs. 16a, 17a and 18a. The degree distributions of the three networks are shown in Figs. 16b, 17b and 18b.

From the following figures, we can find that the degree distribution of each network follows a power law distribution. That is, in such network, most of the nodes have a low degree while only a small proportion of nodes have a high degree. For example, most of the nodes in the Twitter data set only have a degree of 2, while only five nodes have a degree larger than 1,800. The heterogeneity of node degree in the Twitter data set is highest among three data sets.

Based on the three networks, we evaluate the effectiveness of the proposed opinion guidance control strategy. We evaluate the efficiency of different control strategies including random control strategy, targeted control strategy and the opinion guidance

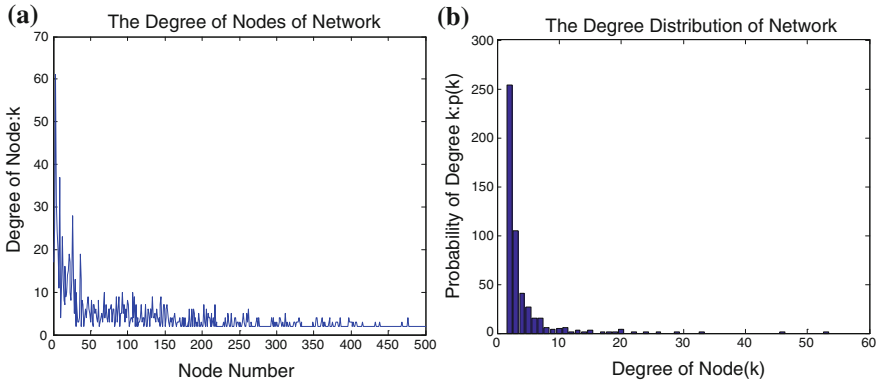


Fig. 18 Node degree and degree distribution of a synthetic scale-free network

control strategy. In the SPNR model, positive infected nodes represent users who believe in rumor, whereas negative infected nodes represent users who disbelieve in rumor. Then we can conclude that the positive infected nodes represent the impact of the rumor. We compare the number of the positive infected nodes under the above three control strategy with the number under no control strategy, in order to verify the effectiveness of our control strategy. We adopt each control strategy on the three networks and we compare the performance of control strategy in term of the number of positive infected users. We consider four situations:

- (1) No control strategy: when we choose no control strategy, the rumor diffusion will follow the SPNR model rigidly.
- (2) Random control strategy: we randomly immunize a small portion of nodes in the network for rumor control, without considering the degree of chose nodes.
- (3) Targeted control strategy: we immunize the nodes with the highest degree for rumor control.
- (4) The proposed strategy: we insert a set of new nodes which are negatively infected by the rumor and make them connected to every other nodes of the network.

The performances under the four situations on the three networks are shown in Figs. 19, 20, and 21. We can see that the number of positive infected nodes is highest when no control strategy, which shows that the three control strategies have certainly helps in rumor control. However, the random control strategy has little effect on rumor control, which only decreases a small proportion of positive infected nodes. The same result is consistent on all three networks. Both the targeted strategy and the proposed strategy can significantly decrease the positive infected nodes. However,, the performance of the proposed strategy are much better than the targeted control strategy on three networks. Because targeted control strategy requires precise

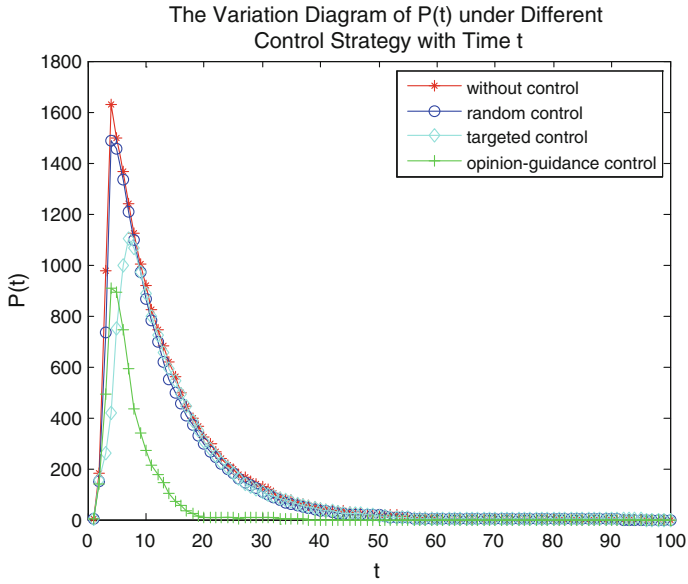


Fig. 19 The variation diagram of $P(t)$ under different control strategies with time t ($\lambda_1 = 0.5, \lambda_2 = 0.5, \beta_1 = 0.02, \beta_2 = 0.02, \mu_1 = 0.2, \mu_2 = 0.3$)

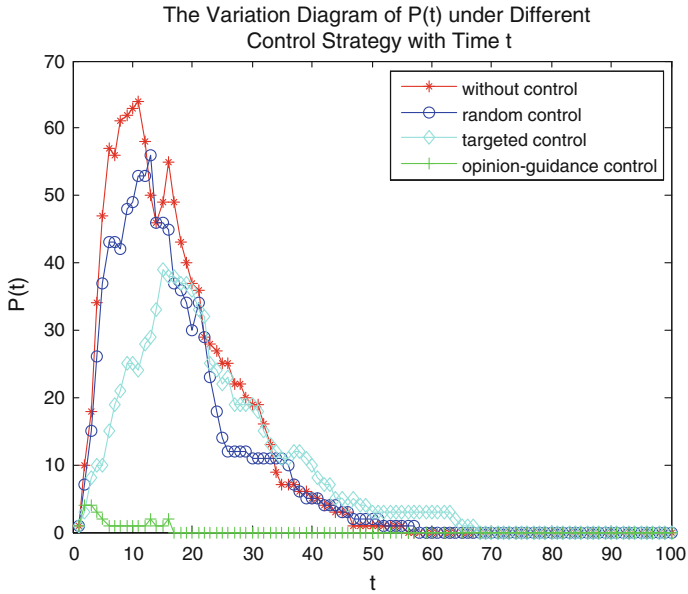


Fig. 20 The variation diagram of $P(t)$ under different control strategies with time t ($\lambda_1 = 0.5, \lambda_2 = 0.5, \beta_1 = 0.02, \beta_2 = 0.02, \mu_1 = 0.2, \mu_2 = 0.3$)

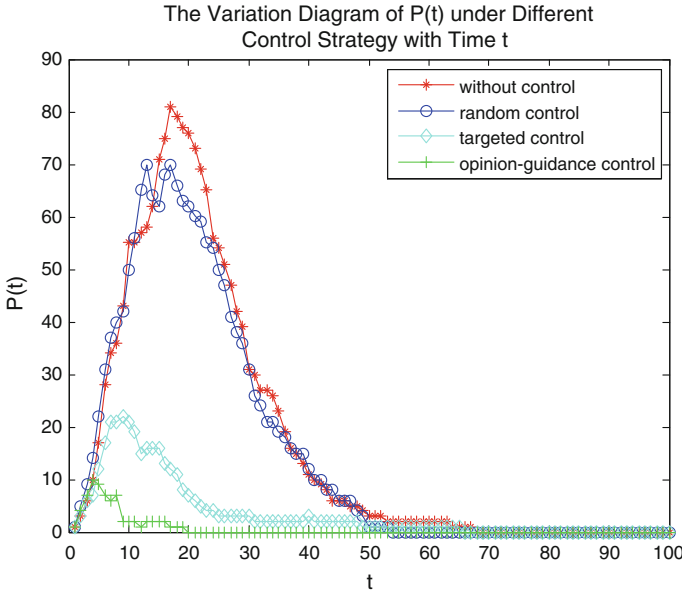


Fig. 21 The variation diagram of $P(t)$ under Different Control Strategies With time t ($\lambda_1 = 0.5, \lambda_2 = 0.5, \beta_1 = 0.02, \beta_2 = 0.02, \mu_1 = 0.2, \mu_2 = 0.3$)

knowledge about the social network, the proposed strategy with inserting opinion guidance nodes is much simpler in implementation. Therefore, we believe that the proposed strategy is highly effective in rumor control with good performance and also simple in implementation.

6 Conclusion

In this paper, we have identified the key difference between rumor spreading and disease spreading, and improved the classic epidemic spreading model considering both positive and negative infected states. We have then proposed a new SPNR model by applying such observation. We have further identified the concrete propagation relationships and obtained the spreading threshold. Moreover, we have analyzed how the parameters of the proposed model affect the maximum value of steady state, the point of decline, and the life cycle of a rumor. We have evaluated the model via simulations and compared the results with real-world data obtained from Sina Weibo. The results show that the proposed model effectively captures the rumor spreading process. Furthermore, we have developed a new rumor control strategy, with the proposed model. Our evaluation shows that the proposed strategy is effective on three types of common social networks in stopping rumor spreading. We will focus

on dynamic behaviors in rumor spreading and develop rumor control based on the proposed model in our future work.

Acknowledgments This work was supported by the National Key Technology R&D Program of China under Grant No.2012BAH46B04.

References

1. Bao Y, Yi C, Xue Y, Dong Y (2013) A new rumor propagation model and control strategy on social networks. In: IEEE/ACM international conference on advances in social networks analysis and mining, pp 1472–1473
2. Knapp RH (1944) A psychology of rumor. *Public Opin Q* 8:22–27
3. Koeing FW (1985) Rumor in the market place: the social psychology of commercial hearsay. Auburn House Pub. Co., Dover
4. Ross L, Lepper MR, Hubbard M (1975) Perseverance in self-perception and social perception: biased attributional processes in the debriefing paradigm. *J Pers Soc Psychol* 32(5):880–892
5. Rosnow RL (1974) On rumor. *J Commun* 24(3):26–38
6. Rosnow RL (1980) Psychology of rumor reconsidered. *Psychol Bull* 87(3):578–591
7. Rosnow RL, Fine GA (1976) Rumour and gossip: the social psychology of hearsay. Elsevier, New York
8. Rosnow RL (1991) Inside rumor: a personal journey. *Am Psychologist* 46(5):484–496
9. Shibutani T (1996) *Improvised news: a sociological study of rumour*. Bobbs-Merill, Indianapolis
10. Difonzo N, Bordia P, Rosnow RL (1994) Reining in rumours. *Organ Dyn* 23(1):47–62
11. Difonzo N, Bordia P (1997) Rumor and prediction: making sense (but losing dollars) in the stock market. *Organ Behav Hum Deci Process* 71(3):329–353
12. Kermack WO, McKendrick AG (1927) Contributions of mathematical theory to epidemics. *Proc R Soc Ser A* 115:700–721
13. Kermack WO, McKendrick AG (1932) Contributions of mathematical theory to epidemics. *Proc R Soc Ser A* 138:55–83
14. Kermack WO, McKendrick AG (1933) Contributions of mathematical theory to epidemics. *Proc R Soc Ser A* 141:94–122
15. Goffman W, Newill VA (1964) Generalization of epidemic theory: an application to the transmission of ideas. *Nature* 204(4955):225–228
16. Daley DJ, Kendall DG (1964) Epidemics and rumours. *Nature* 204(4963):1118
17. Maki D, Thomson M (1973) *Mathematical models and applications*. Prentice-Hall, Englewood Cliff, New Jersey
18. Nekovee M, Moreno Y, Bianconi G, Marsili M (2007) Theory of rumor spreading in complex social networks. *Phys A* 374(1):457–470
19. Pastor-Satorras R, Vespignani A (2001) Epidemic spreading in scale-free networks. *Phys Rev Lett* 86(14):3200–3203
20. Pastor-Satorras R, Vespignani A (2002) Epidemic dynamics in finite size scale-free networks. *Phys Rev E* 65(3):035108
21. Isham V, Harden S, Nekovee M (2010) Stochastic epidemics and rumours on finite random networks. *Phys A* 389:561–576
22. Singh A, Singh YN (2013) Nonlinear spread of rumor and inoculation strategies in the nodes with degree dependent tie strength in complex networks. *Acta Phys Pol B* 44(1):5–28
23. Wang Y, Chakrabarti D, Wang C, Faloutsos C (2003) Epidemic spreading in real networks: an eigenvalue viewpoint. *SRDS*
24. Ganesh A, Massouli E, Towsley D (2005) The effect of network topology on the spread of epidemics. *INFOCOM*

25. Prakash BA, Chakrabarti D, Faloutsos M, Valler N, Faloutsos C (2011) Threshold conditions for arbitrary cascade models on arbitrary networks. *ICDM*
26. Castellano C, Vilone D, Vespignani A (2003) Incomplete ordering of the voter model on small networks. *Europhys Lett* 63(1):153
27. Mizell CM, Sander LM (2009) A generalized voter model on complex networks. *J Statist Phys* 136(1):59–71
28. Kempe D, Kleinberg J, Tardos E (2003) Maximizing the spread of influence through a social network. In: *Proceedings of the 9th ACM SIGKDD international conference on knowledge discovery and data mining*, pp 137–146
29. Tripathy RM, Bagchi A, Mehta S (2010) A study of rumor control strategies on social networks, *CIKM'10*, Toronto, Ontario, Canada, pp 1817–1820 (October 26–30)
30. Pastor-Satorras R, Vespignani A (2002) Immunization of complex networks. *Phys Rev E* 65(3):036104
31. Madar N, Kalisky T, Cohen R, Avraham D, Havlin S (2004) Immunization and epidemic dynamics in complex networks. *Euro Phys J B* 38(2):269–276
32. Gallos LK, Liljeros F, Argyrakis P, Bunde A, Havlin S (2007) *Phys Rev E* 75:045104
33. Yiran Gu, Lingling Xia (2012) The propagation and inhibition of rumors in online social network. *Acta Phys Sin* 61(23):238701
34. Hayashi Y, Minoura M, Matsukubo J (2003) Recoverable prevalence in growing scale-free networks and the effective immunization. [arXiv:cond-mat/0305549](https://arxiv.org/abs/cond-mat/0305549). v2, Aug 6
35. Tong H, Prakash BA, Tsourakakis CE, Eliassi-Rad T, Faloutsos C, Chau DH (2010) On the vulnerability of large graphs. *ICDM*, pp 1091–1096,
36. Briesemeister L, Lincoln P, Porras P (2003) Epidemic profiles and defense of scale-free networks. *WORM 2003*, Oct. 27
37. Prakash BA, Tong H, Valler N, Faloutsos M, Faloutsos C (2010) Virus propagation on time-varying networks: theory and immunization algorithms. *ECML/PKDD* 3:99–114
38. Valler N, Prakash BA, Tong H, Faloutsos M, Faloutsos C (2011) Epidemic spread in mobile ad hoc networks: determining the tipping point. *Networking* 1:266–280
39. Gu Y, Xia L (2012) The propagation and inhibition of rumors in online social network. *Acta Phys Sin* 61:23
40. Tong H, Aditya Prakash B, Eliassi-Rad T, Faloutsos M, Faloutsos C (2012) Gelling, and melting, large graphs by edge manipulation. *CIKM*
41. Marcelino J, Kaiser M (2009) Reducing influenza spreading over the airline network. *PLoS*
42. Schneider CM, Mihaljev T, Havlin S, Herrmann HJ (2011) Restraining epidemics by improving immunization strategies. *CoRR*, abs/1102.1929
43. Holme P, Kim BJ, Yoon CN, Han SK (2002) Attack vulnerability of complex networks. *Phys Rev E*
44. Hu YF (2005) Efficient and high quality force-directed graph drawing. *Math J* 10:37–71
45. Gong K, Tang M, Shang M, Zhou T (2012) Empirical study on spatiotemporal evolution of online public opinion. *Acta Phys Sin* 61

Studying Graph Dynamics Through Intrinsic Time Based Diffusion Analysis

Alice Albano, Jean-Loup Guillaume, Sébastien Heymann
and Bénédicte Le Grand

Abstract Complex networks may be studied in various ways, e.g., by analyzing the evolutions of their topologies over time, and in particular of their community structures. In this paper, we focus on another type of dynamics, related to diffusion processes on these networks. Indeed, our work aims at characterizing network dynamics from the diffusion point of view, and reciprocally, it evaluates the impact of graph dynamics on diffusion. We propose in this paper an innovative approach based on the notion of *intrinsic time*, where the time unit corresponds to the appearance of a new link in the graph. This original notion of time allows us to somehow isolate the diffusion phenomenon from the evolution of the network. The objective is to compare the diffusion features observed with this intrinsic time concept from those obtained with traditional (extrinsic) time, based on seconds. The comparison of these time concepts is easily understandable yet completely new in the study of diffusion phenomena. We experiment our approach on three real datasets and show the promising results of intrinsic time-based diffusion analysis.

Keywords Diffusion · Intrinsic time · Complex network · Temporal networks · Community structure · Dynamic networks

A. Albano · J.-L. Guillaume · S. Heymann · B.L. Grand (✉)
Sorbonne Universités, UPMC Univ Paris 06, UMR 7606, LIP6, F-75005 Paris, France
e-mail: benedicte.le-grand@lip6.fr

A. Albano
e-mail: alice.albano@lip6.fr

J.-L. Guillaume
e-mail: jean-loup.guillaume@lip6.fr

S. Heymann
e-mail: sebastien.heyman@lip6.fr

A. Albano · J.-L. Guillaume · S. Heymann · B.L. Grand
CNRS, UMR 7606, LIP6, F-75005 Paris, France

A. Albano · J.-L. Guillaume · S. Heymann · B.L. Grand
Université Paris 1 Panthéon-Sorbonne, CRI, France

1 Introduction and Context

Diffusion phenomena occur in many kinds of real-world complex networks, such as biological, information or social networks. These various contexts lead to different types of diffusion, like disease spreading, information broadcasting or innovative products adoption. Several classes of diffusion models have therefore been proposed in the literature, among which epidemiological models such as Susceptible-Infected (SI), Susceptible-Infected-Removed (SIR) or Susceptible-Infected-Susceptible (SIS) [1], and adoption models [2].

Many studies aim at investigating diffusion as an evolving phenomenon but mostly occurring on static networks [3], although most real-world networks evolve over time with the creation of new nodes or links, or their disappearance. It is already known that this dynamic behaviour has an impact on the diffusion process [4] and that this impact should not be neglected [5]. Network dynamics may eventually become an asset as it may be used to slow down diffusion speed with regard to static networks [6], e.g., to limit the propagation of an epidemic [7].

One way to understand the impact of network dynamics on diffusion is to consider a given diffusion model, then to fix the values of all its parameters and observe the impact of various types of topology evolutions on the diffusion process. This approach has already been applied in a static context [8]; the authors aimed at evaluating the impact of various random graphs' topologies on a diffusion simulated with an adoption model. Although different topologies have been considered, the authors of [8] have not studied graph evolutions over time. This method, although interesting, is difficult to apply in practice in this context. Indeed, evolutions of the graph topology may lead to simultaneous variations of several features of the network, such as clustering coefficient, node degree, number of nodes and links. It is consequently difficult to actually keep fixed values for all parameters while the topology changes. This approach is therefore limited to very basic dynamics in order to obtain easily interpretable results.

Another way to study diffusion in evolving networks is empirical, consisting in studying real networks, in observing them and finally in proposing a diffusion model consistent with the observations. The authors of [9] have applied this methodology to analyze the spreading of H1N1 virus -modeled with a SEIR model- in a dynamic contact network. Other works have studied diffusion processes in a phone network [10]. However, this approach is global, as it does not distinguish the results which are entirely related to the type of diffusion, from the observations which are actually due to the evolution of the graph structure.

The goal of our work is to understand, while observing a diffusion process, which part is *intrinsically* related to the type of diffusion, and consequently which part is merely due to the evolution of the network. Of course, it is not possible to completely separate diffusion from graph dynamics, as both phenomena are strongly related, but rather to attempt to normalize our observations, e.g., to see if a very significant diffusion at a given moment is due to a sudden growth of the graph. In this paper, we study further the approach we have developed in [11]. This approach is simple yet

innovative to study the impact of graph dynamics on diffusion. This methodology does not require any new computation once the diffusion process has been measured; instead of observing the diffusion phenomenon as a function of usual time -e.g., measured in seconds- which we call here *extrinsic time*, we propose to observe it as a function of what we call *intrinsic time* [12]. Indeed, this definition of time is intrinsically related to graph dynamics as an *intrinsic* time slot is not absolute: it corresponds to a modification of the network's topology (for instance, appearance or disappearance of a link, a node or a triangle, i.e., three nodes with a link between each pair of nodes), as explained in the following of the paper.

This article is organized as follows. In Sect. 2, we describe our methodology for studying diffusion in evolving networks, which relies on the *intrinsic time* concept and we illustrate it on a synthetic graph. In Sect. 3 we describe the real-world networks on which we have applied our approach, extracted respectively from the Github software sharing network, from French blogs and from the Infectious SocioPatterns event. Section 4 is dedicated to the successful experiments we have conducted on these dynamics networks, on the Github and the blog networks. In Sect. 5, we show the importance of an appropriate definition of intrinsic time through diffusion analysis on the Infectious SocioPatterns dataset. We finally conclude this paper and propose perspectives for future work in Sect. 6.

2 Methodology

2.1 *Intrinsic Versus Extrinsic Time*

Time is a controversial concept that one can see as a dimension in which changes occur in a sequence. In this perspective, time is considered as absolute, i.e., changes happen independently from the flow of time [13, 14]. But if we consider time as a relative concept, time then depends on space. Many techniques exist to measure it. The unit adopted by the International System of Units is the second, which is defined as the transition between two states of the caesium-133 atom [15]. This unit is therefore related to movements measured in the physical space.

In this paper we use a concept of relative time from a network perspective, called *intrinsic time* of the network, as opposed to *extrinsic time*, which is the traditional concept of absolute time.¹ Let the *extrinsic time* of the network be the time measured using the second (or its derivative units like days and years). We call it *extrinsic* because its flow is independent from the changes that occur in the network. Let the *intrinsic time* of the network be the time measured by the transition between two states of the network. The unit is thus the (spatial) change of the network, i.e., the addition or removal of one node or one link for instance. We call it *intrinsic* time

¹The choice of the transition considered as an intrinsic time unit will depend on the dataset, as explained in Sects. 4 and 5.

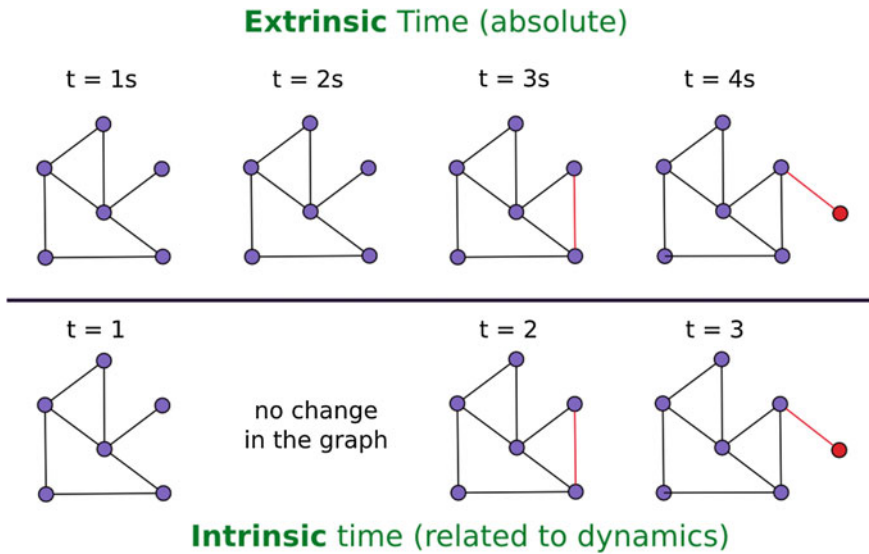


Fig. 1 Comparison between intrinsic and extrinsic time

because it depends on the changes that occur in the network, and changes depend on such time to happen.

Figure 1 explains the relation between extrinsic and intrinsic time when an intrinsic time step is related to the appearance of a new link. In extrinsic time, between the two first time steps, there is no change in the graph. Therefore in intrinsic time, there is no new time step. There is a link creation at times 3 and 4 in extrinsic time, therefore there are also two new intrinsic time steps.

This notion of *intrinsic time* is not a new concept: discrete time Markov chains can be seen in the same manner. When something happens (a random event), the chain changes its state. In our graph, the event is not random, but when it occurs, we increment the intrinsic time. Asynchronous clocks use exactly the same concept as the notion of intrinsic time: when an event occurs, the time is incremented. It does not depend on the classical flow of time.

Whereas *extrinsic time* is broadly used without notice, we have found out in [12] that the choice between extrinsic and intrinsic time has a very significant impact on the measurement of statistical properties of temporal networks. Our previous results on social networks seem to suggest that intrinsic time is better at characterizing the endogenous dynamics of the network, because extrinsic time is more likely to capture exogenous patterns like day-night activity of users.

Our methodology consists in observing a simulated diffusion process on dynamic networks at extrinsic and intrinsic time scales, in order to study the impact of graph dynamics on this propagation.

2.2 Diffusion Model

We have chosen a very well-known diffusion model for our simulations: the Susceptible-Infected (SI) model. This model has been proposed by [1] in 1927, and it has been widely used since then. In an SI model, each node of the graph may be either in susceptible (sane) or infected state. A susceptible node with an infected neighbor has the probability p of becoming infected too.

This model is particularly interesting in this case as it has very few parameters, namely the contamination probability p and the choice of infected nodes at the beginning, i.e., from which the spreading starts. In the following, the values of both parameters are fixed in order to focus on the correlation between graph dynamics and diffusion process. It will be therefore easier later to distinguish observations directly related to the model from those related to the graph topology or the time notion used.

In the SI model, the evolution equation is the following:

$$\frac{di_k(t)}{dt} = p[1 - i_k(t)]k\Theta_k(t), \quad (1)$$

where p is the probability of infection, $i_k(t)$ is the probability that a vertex with degree k is not infected, and Θ_k is the density of infected neighbors of vertices of degree k .

In this paper, we will only simulate the diffusion, therefore we will only need to select a relevant value of p and try, at each time step, to infect neighbors of infected nodes.

2.3 Intrinsic Versus Extrinsic Time-Based Diffusion Analysis

In the following, we will represent (and compare) for each simulated diffusion process the total number of infected nodes as a function of three different time scales:

- **extrinsic time** which is the classical notion of time.
- **converted extrinsic time**, which results from the conversion of extrinsic time into intrinsic time to observe the diffusion, as illustrated on Fig. 1. For instance, if the second link is created at extrinsic time 15, its intrinsic time is 2. The number of infected nodes at extrinsic time 15 will therefore correspond to the same number of infected nodes at intrinsic time 2. Note that the diffusion process observed with converted extrinsic time is the same.
- **intrinsic time**, i.e., we represent the speed of infection versus the number of created links. This curve corresponds to a different diffusion process, simulated with intrinsic time.

2.4 Example on Synthetic Barabási-Albert Graphs

In order to illustrate our methodology, we will now study diffusion on a BA graph [16] using the notions of extrinsic and intrinsic times. In the BA model, nodes are added one by one to the network. Each new node is connected to a fixed number of existing nodes of the graph according to the *preferential attachment* principle: the higher the degree of node x in the graph, the more likely the new node connects to x .

BA graphs are therefore evolving increasing networks, through this step-by-step addition of nodes and edges. This growth is an important feature for the concept of intrinsic time, as discussed in Sect. 2. The BA model [16] is characterized by four parameters: the initial and final numbers of nodes in the network, the time step between the creation of new nodes and finally the preferential attachment parameter, noted m , which corresponds to the number of edges generated by the creation of a new node, i.e., the number of existing nodes to which each new node gets connected. The BA graphs considered in this Section contain 500 nodes initially, and we generate 500 additional nodes using the BA model, leading to a graph with 1000 nodes at the end of the evolution.

For example, Fig. 2 shows the behaviour of the SI model on a Barabási-Albert graph when the preferential attachment parameter m varies and with different values of time steps, and confirms known results. In all cases, the number of infected nodes keeps increasing with the SI model as infected nodes remain infected forever. We may observe for most plots that after an initial rapid growth at the beginning of the diffusion process, the number of infected nodes grows more slowly at some point,

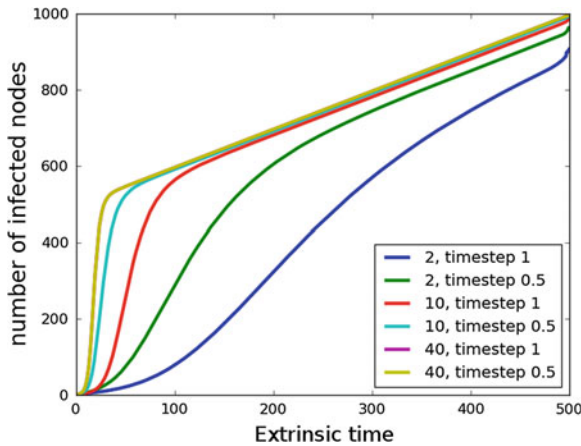


Fig. 2 SI diffusion on an evolving BA graph, from 500 to 1000 nodes. The parameter respectively equal to 2, 10 and 40 corresponds to the number of connections established by each new node of the graph. The simulations are also conducted with 2 distinct values of time step: time step = 0.5 means that there are 2 infection tests per time unit, versus 1 infection per time unit with time step = 1. These plots correspond to the average result obtained from 500 simulations

after which the slope of the growth is only due to the (increasing) size of the graph. As expected, the value of m has an impact on the shape of the plot: diffusion is much faster with $m = 40$ than with $m = 2$. With $m = 2$, this saturation point is not reached during the simulation.

Figure 2 also shows the importance of the time step value. Indeed, dividing the time step by 2 actually increases the diffusion speed. However the diffusion does not happen twice faster, as we could have thought.

In order to investigate the concepts of *intrinsic* and *extrinsic* time scales and their correlation with the diffusion process, we have simulated 3 different types of dynamics. The difference between these dynamics is the inter-arrival delays which are generated using three distinct probability laws (the inter-arrival delay is the time between the creation of two consecutive nodes). We use different laws for the generation of links because there is no extrinsic notion of time in the BA model, only the creation of links (which is for us an intrinsic notion of time). We choose the parameters of these laws so that the average inter-arrival delay is equal for the three laws. More precisely, the three laws we have used are:

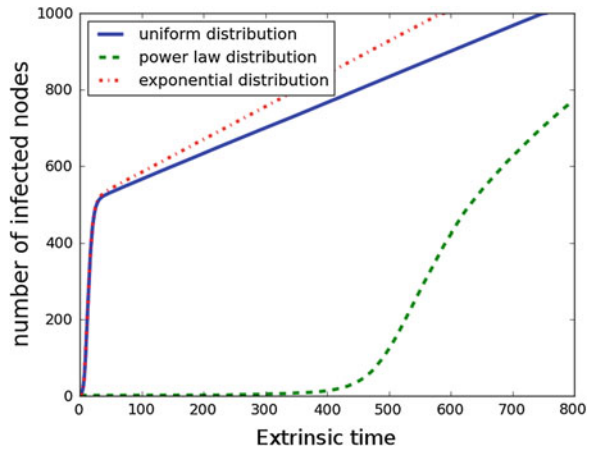
- a discrete uniform distribution where each inter-arrival delay in the interval is equally likely to happen. The density of this law is described by the following equation: $f(x) = \frac{1}{b-a}$, where $b = 11$ and $a = 1$,
- a negative exponential distribution, i.e., the probability of observing long delays decreases exponentially fast. The density of this law is described by the following equation: $f(x) = \lambda e^{-\lambda x}$, where $\lambda = \frac{1}{5}$,
- a power law distribution where the probability of observing long delays decreases polynomially. This law is described by the following equation: $f(x) = ax^k$, where $k = -2.2$ and $a = 100$.

Finally, whatever the time unit considered, we perform one step of the SI diffusion per time unit. We have simulated a diffusion with a SI model for these three types of dynamics. We have fixed the probability of being infected $p = 0.005$. For each type of BA dynamics, we compare the different time scales using the extrinsic, extrinsic converted into intrinsic and intrinsic representations of the total number of infected nodes over time.

2.4.1 Simulation in Extrinsic Time

We first study a simulation of diffusion with the SI model on the three generated BA graphs. Results are presented in Fig. 3. This Figure exhibits very different behaviours and the variation of inter-arrival delays drastically modifies the diffusion behaviour. This result is natural, indeed it is very likely that the random inter-arrival time generated at a given moment is higher than 1. In extrinsic time if we generate, for instance, a delay of 10 between the arrival of two consecutive nodes, we perform ten SI diffusion steps instead of one in intrinsic time.

Fig. 3 SI diffusion on a BA graph, from 500 to 1000 nodes, simulated and observed at extrinsic time scale. The time between the creation of 2 new nodes follows 3 distinct probability laws: uniform, exponential and power-law

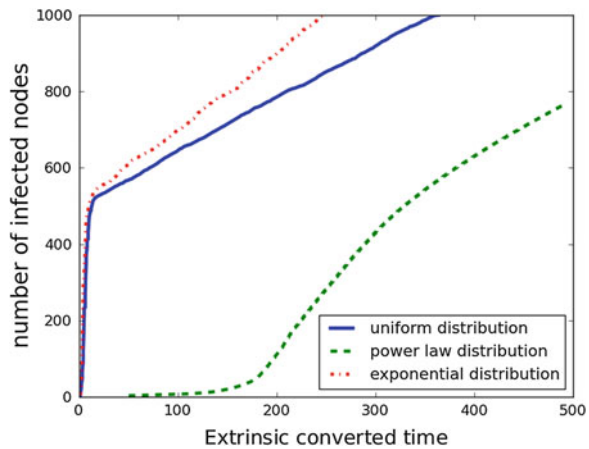


2.4.2 Diffusion Represented with Extrinsic Converted Time

To compare extrinsic and intrinsic time notions, and to better understand the diffusion behaviour in extrinsic time, we convert the previous results obtained with extrinsic time, into intrinsic time. Let us recall that if the second link is created at *extrinsic time* 15, its *intrinsic time* is 2. In this case, we will therefore plot the number of infected nodes at *extrinsic time* 15, and it corresponds to the same number of infected nodes as at *intrinsic time* 2. A non uniform scaling is therefore performed from Fig. 3 to obtain Fig. 4.

We observe that the conversion from *extrinsic time* to *intrinsic time* leads to similar observations from those obtained with *extrinsic time* (see Fig. 3). The infection spreads quickly at the beginning up to a strong inflexion in the speed of diffusion

Fig. 4 SI diffusion on a BA graph, from 500 to 1000 nodes, simulated and observed at extrinsic converted time scale. The time between the creation of 2 new nodes follows 3 distinct probability laws: uniform, exponential and power-law



(except for the power law distribution). Then all curves follow a linear behaviour. This observation is strongly related to the network dynamics and the SI model. Indeed, at the beginning, many nodes are susceptible of being infected, which explains the rapid growth phase. This phase can also be observed in extrinsic time in Fig. 3. After this phase, most nodes are infected and the infection spreading is therefore limited by the speed of the creation of new nodes, which is exactly one per time unit in intrinsic time.

The three converted curves do not change much during the conversion. Indeed, the extrinsic converted time is supposed to stretch bursts of link creations, and to shorten phases with no activity. As we study here a random graph with no bursts or stable phases at all, it is therefore normal to observe a similar behaviour between Figs. 3 and 4.

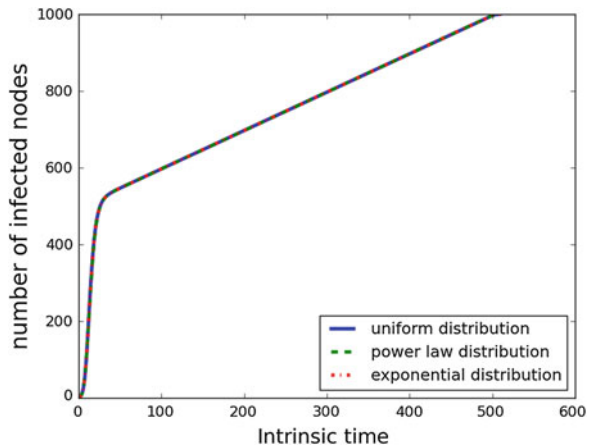
2.4.3 Simulations in Intrinsic Time

The goal here is to show that SI simulated as a function of intrinsic time is somehow immune to dynamics. We generate random BA graphs with an inter-arrival delay following the three laws presented before and we simulate an SI diffusion. Figure 5 shows the results of these simulations.

We observe on this Figure that the three plots have very similar behaviours. Indeed, the notion of *intrinsic time* only reflects the modification of the topology and not the delays between these modifications, therefore the different inter-arrival laws have no impact when observed with *intrinsic time* and we perform one SI diffusion step per link creation.

With this experience, we can clearly see the impact of the time scale on the observation of the diffusion. Indeed, when the diffusion is simulated in extrinsic time, we see that it is sensitive to inter-arrival delays, while in intrinsic time, diffusion is

Fig. 5 SI diffusion on a BA graph, from 500 to 1000 nodes, simulated and observed at intrinsic time scale. The time between the creation of 2 new nodes follows 3 distinct probability laws: uniform, exponential and power-law



completely independent from delays. We observe significant changes in the diffusion behaviour by simulating it in intrinsic time. According to what we want to know about the diffusion process, it is therefore very relevant to change the time scale: in intrinsic time, we can observe a diffusion which is not affected by bursts or by phases with little activity. This time scale can, in a certain way, isolate the graph dynamics from the diffusion.

The extrinsic converted time scale is relevant when we want to observe more precisely events in the graph and their impact on the diffusion. Indeed, as we have said before, it will stretch bursts. In our example on BA graphs, we could not see that because, despite of various inter-arrival delay, our graphs dynamics were mostly homogeneous and there was no event. We will see all the interest of this time scale in Sect. 4.

In the following, we describe datasets we have used, and we will perform the same experiments on these datasets.

3 Datasets Description

In this Section, we describe the three real-world evolving networks to which we have applied our methodology. On these graphs, we have computed the total -cumulated- number of infected nodes as a function of extrinsic, extrinsic converted and intrinsic times.

3.1 *Github Network*

3.1.1 Dataset Description

Github.com is an online platform created in 2008 to help developers share open source code and collaborate. Built on the Git decentralized version control systems, it facilitates the contributions and discussions by providing a Web interface. Github reached 3 million users on January 16, 2013, who collaborate on 5 million source code repositories [17].

The Github dataset we use here describes the complete activity between users and repositories on the platform from March 11, 2012 to July 18, 2012. We have extracted the data from the Github Archive [18], which is a record of every public activity on Github. Then we have built the graph of “who contributes to which repository”, where nodes represent users and repositories, and where links represent any kind of user activity on repositories: commit and push source code, open and close issues for bug reports, comment on issues, commit or pull requests (i.e., asking for a patch to be merged), create or delete branches and tags, and edit the repository wiki. We have ignored other activities: fork (i.e., repository duplication), mark repositories as

favorite, and follow of the timeline of another user or repository. The resulting graph contains a bit more than 336,000 nodes and 2.2 million links.

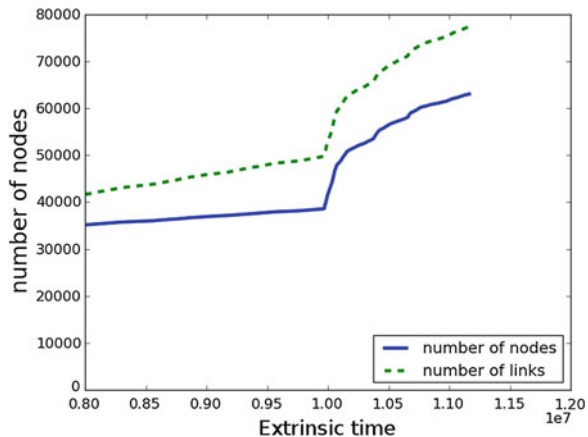
In order to simulate a diffusion on this dataset, we consider only the largest connected component of the graph, which has a little more than 60,000 nodes.

3.1.2 A Few Statistics

Figure 6 shows the total number of nodes and links in the network as a function of (usual) extrinsic time. We see on this figure that the total numbers of nodes and links grow rather slowly, then suddenly increase significantly. This change happens on July, 4th 2012 and it is correlated to a sudden increase in the maximum degree of the sub-graph. We discover that the *Try-Git* project interacts with 506 users, which explains this high degree. This project is a tutorial for Git, one of Github’s underlying tools; the first action required from the user in this tutorial is to create a clone with a new project (by sending this user a *CreateEvent* message). The instant of the event corresponds to the moment when *Try-Git* was made public, on July 4th, 2012 at 5 pm (this information was confirmed by a post on the Github.com blog²). This event radically modifies the structure of the network. This is confirmed by the maximal node degree in the graph, illustrated in Fig. 7, whose order of magnitude changes significantly after the appearance of a node with a very high degree, corresponding to the event described above.

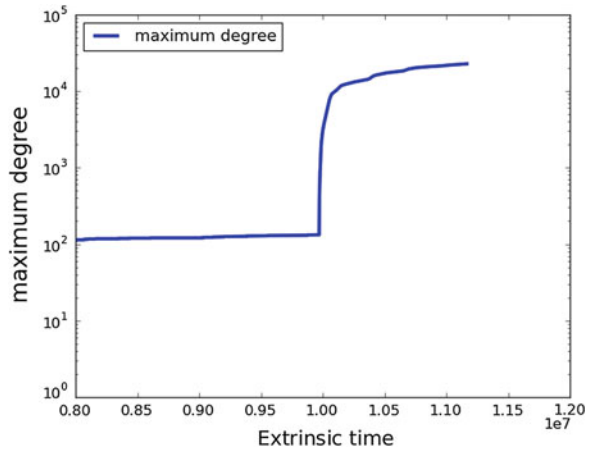
This dataset exhibits regular behaviour, except for the very important event observed: it is therefore appropriate for our study, as it allows us to analyse the impact of this event on the diffusion, and to compare it with the “stable” phase. The burst in link creations will be stretched in extrinsic converted time, as we will see in

Fig. 6 Number of nodes and links in the Github graph, restricted to the largest connected component



²<https://github.com/blog/1183-try-git-in-your-browser>.

Fig. 7 Evolution of the maximal node degree in the Github graph, with y-axis in a logarithmic scale



Sect. 4. Moreover, the important event in the graph will also be relevant to study in intrinsic time, as we have seen that it can isolate diffusion from graph dynamics.

3.2 Webfluence Dataset

3.2.1 Dataset Description

The WebFluence dataset has been collected from French blogs in the context of a national research project, from February 1st to July 1st 2010 (5 months). These blogs have been selected according to their popularity and activity in the French-speaking blogosphere, by a company specialized in blog and opinion analysis (<http://linkfluence.net>). The dataset is composed of 10,309 blogs, 84,026 posts and 1,079,195 citation links.

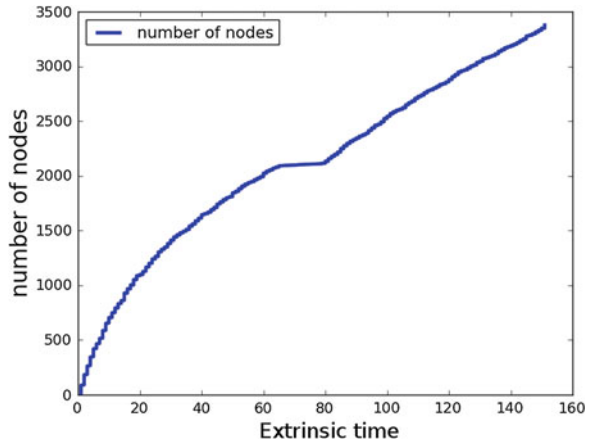
We have extracted a graph from this dataset, which we call the blog network. This blog network is a directed acyclic graph where there is a link from blog A to blog B if A cites B, i.e. there is at least one citation link from a post (publication) from blog A towards an earlier post from blog B. In terms of information spreading, we may say that B content has been spread towards A.

Note that the number of blogs in our network only reflects blogs which are involved in citation links, i.e. blogs which are cited by or cite other blogs.

3.2.2 A Few Statistics

We first observe the number of nodes in the blog network (Fig. 8).

Fig. 8 Evolution of the number of nodes in the blog network



We see on this figure that the number of nodes grows quite regularly, except within 65 and 80 days after the beginning of the measure. Between these two dates, the number of nodes in the graph increases very slightly and is almost constant. The 65th day corresponds to April 5th, and the 80th day to April 20th. This reduced blog activity is due to the period, as these dates correspond to spring holidays in France in 2010.

This dataset is quite different from the Github dataset, since the event linked to spring holidays is not as significant as the major event in the Github dataset. It will be interesting to compare the diffusion behaviour using different time scales for both datasets. The little event observed here will be a good test to see the precision of our intrinsic time definition: can we observe it or not in extrinsic converted time, in spite of its little importance? In intrinsic time, do we observe a regular diffusion, not affected by the event?

3.3 Infectious SocioPatterns Dataset

3.3.1 Dataset Description

This dataset, studied in [19], was collected during the Infectious SocioPatterns event that took place from April to July 2011 (69 days). Visitors were equipped with a RFID captor in order to record interactions between people. The resulting dataset is therefore a contact network. Nodes are visitor, and there is a link between two people if they are face-to-face and close enough. We know the appearance time for each new link.

3.3.2 A Few Statistics

We have selected four days (out of 69) which exhibit different dynamics. Figure 9 represents the cumulated number of nodes and links as a function of time for these four days.

We see on this figure that the number of nodes and links increases quite irregularly, except for the curve at the lower left corner. On May, 2nd at instant 5,000, we observe a slower increase of infected nodes than before. At instant 9,000, the diffusion accelerates again. On May 7th, we observe quite the contrary. At the beginning, growth is slow, then the number of infected nodes suddenly increases much faster until instant 6,000. After that, the diffusion becomes much slower until the end of the simulation. On July 4th, we observe a few events, at instants 7,000 and 10,000 for instance. Finally, on June 16th, diffusion seems more regular than during the three other days.

We can see on these curves that dynamics in this contact network is very different from the previous datasets, and that there are major differences even from one day to another. We observe many little events, and a few short stable phases. We expect the extrinsic converted time to accentuate these events in the observation of the diffusion.

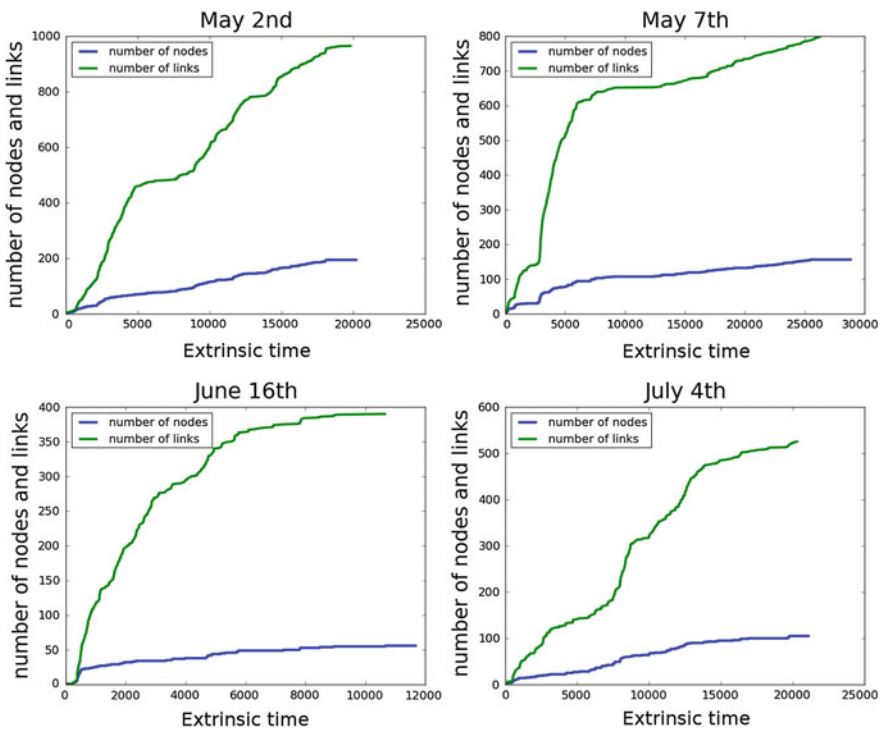


Fig. 9 Evolution of the number of nodes and links in the Infectious SocioPatterns network

4 Experimental Results

This section is dedicated to the results of our intrinsic time-based diffusion analysis on Github and blog networks respectively.

4.1 Results on Github Dataset

4.1.1 Simulation in Extrinsic Time

We start the study of the diffusion using the extrinsic time, i.e., the second. For the simulation we make an infection test every 30 s. On average, there is a new link every 91 s in the Github network, therefore, there are fewer diffusion steps in intrinsic time than in extrinsic time. However, we do not compare the actual speed of diffusion on a quantitative basis.

The results are presented in Fig. 10 (top left). For interpretation purposes, we have added the number of nodes at each time step. We observe that the diffusion process has a slow start and a fast growing phase, as expected with an exponential infection. We can easily observe the impact of the creation of the tutorial *Try-Git* on July 4th, 2012: the number of nodes in the network increases fastly and similarly the diffusion itself undergoes an acceleration. The acceleration is due to the presence of the new high degree node which facilitates the diffusion.

4.1.2 Simulation in Extrinsic to Intrinsic Converted Time

We perform the conversion from extrinsic to intrinsic time as explained in Sect. 2.4. The results of this conversion are presented in Fig. 10 (top right corner). We observe a very different behaviour from the extrinsic time representation from Fig. 10 (top left corner). At intrinsic time 50,000, which corresponds to the event of July 4th, 2012, the growth of the number of infected nodes slows down drastically. This can be explained as follows: after the event, the creation of links becomes much faster, i.e., there are much more links per seconds than before the event. Intrinsic time does not take this into account since it does not consider the speed of link creation. Conversely, in extrinsic time the difference is noticeable, as observed on Fig. 10 (top left).

When the extrinsic time is converted to intrinsic time, we naturally observe a plateau in the diffusion (between time steps 50,000 and 60,000) correlated with the acceleration in intrinsic time. Indeed, in intrinsic time we perform one infection test per link creation, while in extrinsic time there are more than one link created per diffusion time step.

Another way to understand this phenomenon is to consider the number of links creation over time, as shown on Fig. 6. We can observe a fast increase at time 1×10^7 and smaller accelerations at times 1.04×10^7 and 1.07×10^7 . Each increase in the

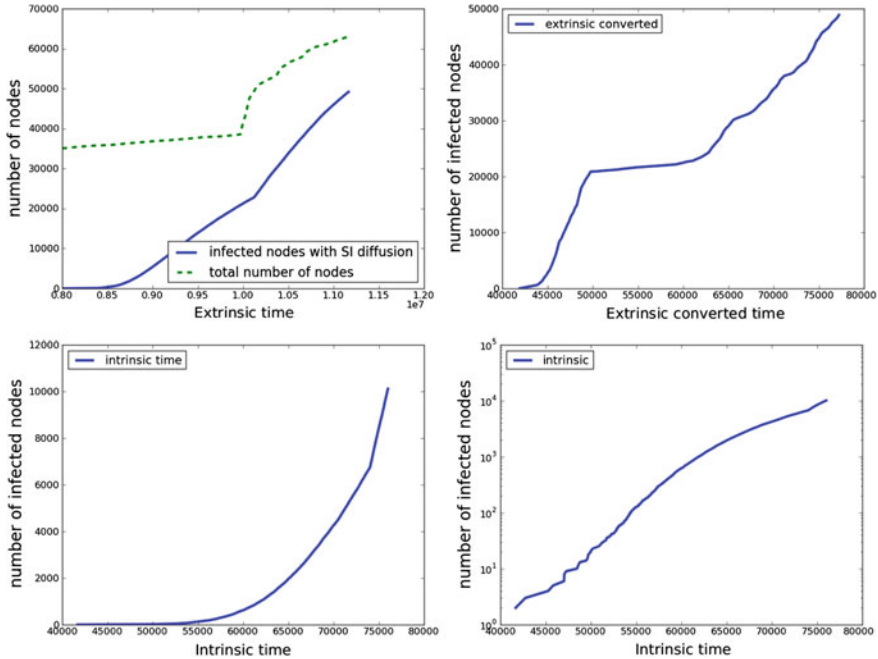


Fig. 10 *Top left* in blue, SI diffusion on the Github graph simulated and observed in extrinsic time. The time between two trials of diffusion is 30 s. In green: number of nodes in the graph at each time step. *Top right* SI diffusion on the Github graph simulated and observed in extrinsic time converted into intrinsic time. *Bottom* SI diffusion on the Github graph simulated and observed in intrinsic time, in linear scale (*left*) and in y-logarithmic scale (*right*)

number of links corresponds to a plateau in the number of infected nodes, at intrinsic times 50,000, 65,000 and 71,000 respectively. The increase slope is correlated with the length of the plateau.

Observing at intrinsic time scale the diffusion simulated at extrinsic time scale helps spotting important moments in the evolution of the graph. We have seen that the definition of intrinsic time (related to link appearance) seemed relevant for the studied network. Simulating diffusion at intrinsic time allows us to confirm that it reflects the dynamics of the graph properly.

4.1.3 Simulation in Intrinsic Time

The diffusion using the intrinsic time on the Github dataset is presented in Fig. 10 (bottom). We do not see a radical change in the diffusion at the time of the creation of the tutorial *Try-Git* (which corresponds to time 50,000 in intrinsic time). Indeed, the curve looks like a standard diffusion with the SI model: we see an exponential growth, with a slow start, then a growing number of nodes that accelerates quickly.

We do not observe the saturation phenomenon at the end of the simulation, because the number of infected nodes in the network is relatively far from the total number of nodes. These results are confirmed by the representation of the same curve on a logarithmic scale, which shows a linear aspect.

Finally, this confirms that diffusion using intrinsic time is rather immune to changes of topology in the network. Intrinsic time using links creation is therefore a good definition: it allows us to isolate the diffusion phenomenon from the graph dynamics.

4.2 Results on Blog (Webfluence) Dataset

4.2.1 Simulation in Extrinsic Time

We have performed the same experiments on the Webfluence dataset, where the *extrinsic time* unit is the day. The *intrinsic time* corresponds to new citations between two blogs, more precisely between two posts from two different blogs.

We start the study of the diffusion using the *extrinsic time*, i.e., the day. For the simulation we have performed ten infection tests per day. There are consequently 1,510 contamination tests in *extrinsic time*, versus 12,000 in *intrinsic time*. Therefore, there are more diffusion steps in *intrinsic time* than in *extrinsic time*. The results are presented in Fig. 11 (top left).

For interpretation purposes, we have added the number of nodes at each time step. We observe that the diffusion process has a slow start and a fast growing phase, as expected with an exponential infection. The diffusion at the time of the French Spring holidays is a little slower, but not radically different.

4.2.2 Simulation in Extrinsic to Extrinsic Converted Time

We perform the same conversion of time from *extrinsic* to *intrinsic* as in Sect. 2.4. The results of this conversion are presented in Fig. 11 (top right corner). We observe a different behaviour from the *extrinsic time* representation from Fig. 11 (top left corner). Around *intrinsic time* 5,400, which corresponds to Spring vacation, the growth of number of infected nodes increases drastically. This can be explained as follows: during Spring holidays, the creation of links becomes much slower, i.e., there are much fewer new links per day than before. *Intrinsic time* does not take this into account since it does not consider the speed of link creation. We may note that it was the opposite situation with the Github network, in which the graph dynamics corresponded to a higher number of new links.

In the blog network, observing the diffusion as a function of *intrinsic time* stresses the change in the network dynamics, whereas the modification of topology in the Github network was already visible in the diffusion observed with *extrinsic time*.

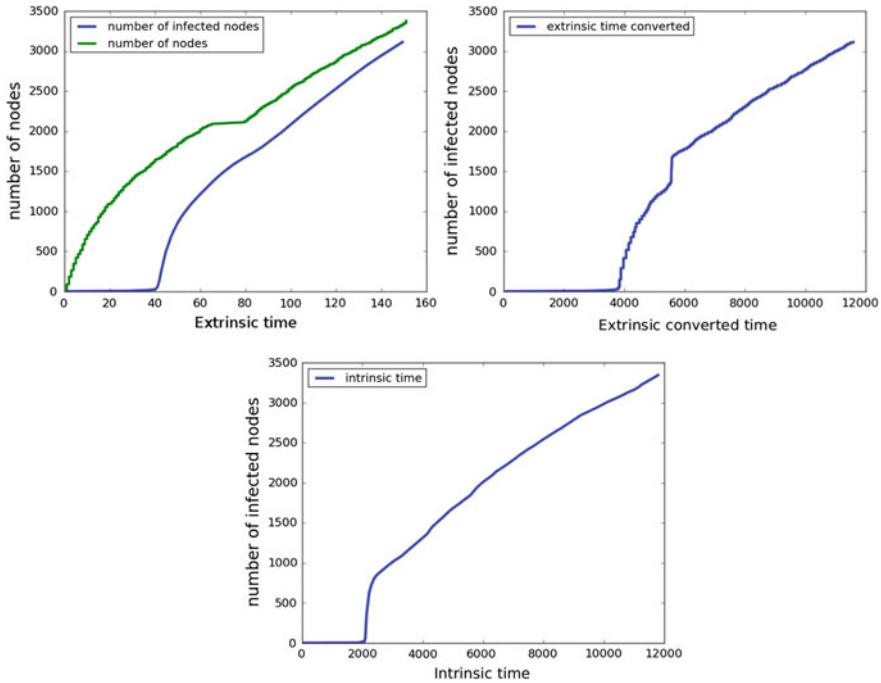


Fig. 11 *Top left* in green, SI diffusion on the blogs graph simulated and observed in extrinsic time. There are ten trials of contamination between two days. In blue: number of nodes in the graph at each time step. *Top right* SI diffusion on the blogs graph simulated in extrinsic time and observed in intrinsic time. *Bottom* SI diffusion on the blogs graph simulated and observed intrinsic time

When the *extrinsic time* is converted to *intrinsic time*, we naturally observe a very high increase in the diffusion around the Spring holidays correlated with the deceleration in *intrinsic time*. Indeed, during this period, there are very few link creations, and it therefore corresponds to a shorter period in intrinsic time. The increasing of the number of infected nodes observed in extrinsic time around this period is therefore the same but on a shorter time scale in extrinsic converted time, which explains the sudden increase observed in this curve.

4.2.3 Simulation in Intrinsic Time

The diffusion simulated with the *intrinsic time* on the Webfluence dataset is presented in Fig. 11 (bottom). The diffusion starts very slowly, and then increases suddenly at time 2,000. When we look at the dataset, we see that this *intrinsic time* corresponds to the twentieth day of the mesure. This increase is correlated to the natural behaviour of the SI model. After a very important increase, the number of infected nodes grows regularly until the end of the dataset. No significant change may be observed during the French Spring holidays (which corresponds to the period between instants

5,100 and 5,600 in *intrinsic time*). After that, the number of infected nodes becomes closer to the total number of nodes, and we observe a more linear growth. Like in the Github network, diffusion using *intrinsic time* is less affected by changes of topology in the network, which means that the definition of intrinsic time used here (new link appearance) is valid.

5 Impact of Intrinsic Time Definition: Results on the Infectious SocioPatterns Network

We have seen with the Github and the Webfluence datasets two cases for which our definition of intrinsic time unit as the appearance of a new link was very relevant. In this section, we study a different type of network, a contact network. As explained in the following, this dataset will allow us to study the limits of the intrinsic time definition.

5.1 Simulation in Extrinsic Time

We first study simulations of diffusion on this contact network over extrinsic time. We make a SI simulation on the four specific days shown in Fig. 9. Results of these simulations are shown on Fig. 12 (left column).

We see on this Figure that diffusions have different behaviours: the first one (top left corner) looks like a regular SI diffusion, with the S-shape. The three others are more irregular, and seem more affected by events in the graph. Moreover, when we look at Fig. 9, it is difficult to find a correlation between irregularities in the diffusion and events in the number of nodes and links: for instance, for the diffusion on July 4th, we can see a correlation with the number of links, but for June 16th, the diffusion behaviour is quite surprising.

In the following, we use the intrinsic time notion in order to see if we can explain this behaviour.

5.2 Simulation in Extrinsic to Intrinsic Converted Time

We now study the same diffusion process as before, but converted into intrinsic time, in the same way as in previous sections. Results are represented in Fig. 12 (middle column). The diffusion behaviour in extrinsic converted time is quite surprising here: on May 2nd, we observe a very irregular growth although in extrinsic time, diffusion is very regular. On May 7th and June 16th, global behaviours are close, but we observe a few more events in extrinsic converted time. On July 4th, the two curves are very similar. For each of these days, events observed in extrinsic converted time

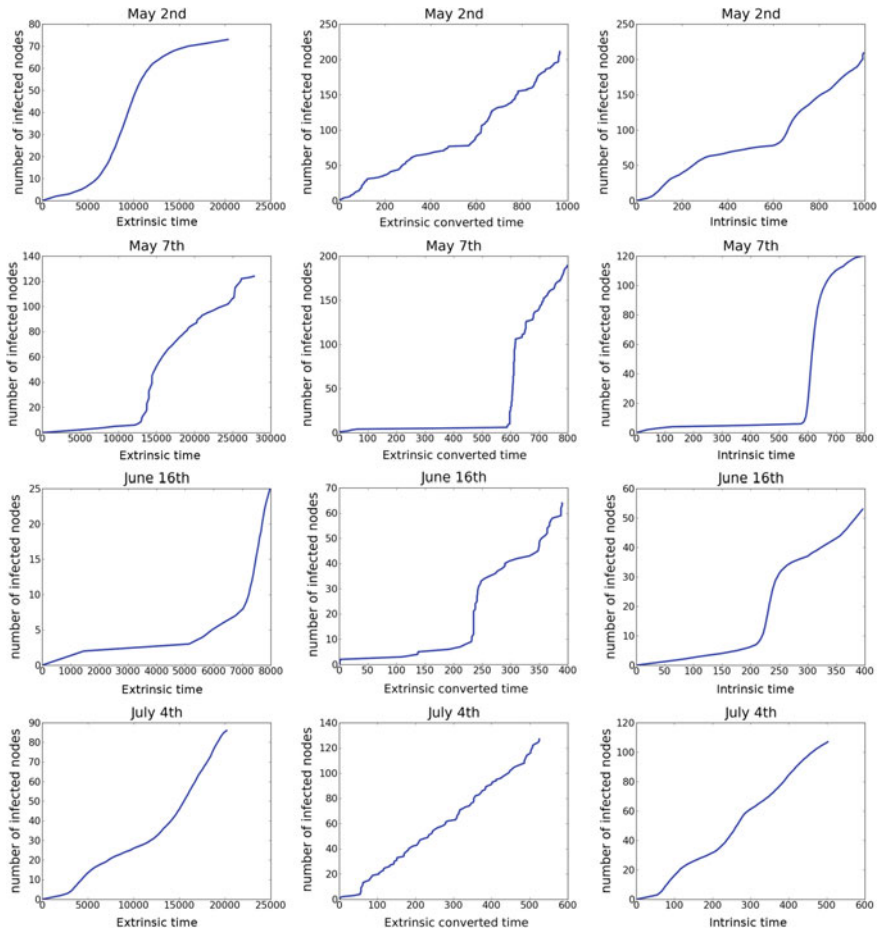


Fig. 12 *Left column* SI diffusion simulated an observed in extrinsic time on the Infectious SocioPatterns network. *Middle column* SI diffusion simulated in extrinsic time and observed in intrinsic time on the Infectious SocioPatterns network. *Right column* SI diffusion simulated and observed in intrinsic time (defined with number of links) on the Infectious SocioPatterns network

are difficult to explain using the number of nodes and links in the network (see Fig. 9). We then simulate another diffusion in intrinsic time in order to see if we can explain further these results.

5.3 Simulation in Intrinsic Time

We simulate now a diffusion with the SI model on the same four days. Results of these simulations are shown on Fig. 12 (right column). First, we can see that the four

diffusion curves have quite different behaviours: for instance, on the curve in the upper right corner, we see a very slow beginning of diffusion, and a sudden increase in the number of infected nodes. In contrast, for the curve of June 16th, growth is slow at first, then suddenly accelerates and slows down again afterwards. When we compare these four diffusion simulations with the number of nodes and links in the network (Fig. 9), we find that the spreading behaviour is quite surprising since on the four observed days diffusions in intrinsic time are not regular at all. Moreover, the diffusion behaviour in intrinsic time is quite similar to behaviour in extrinsic time.

These observations show us that the definition of intrinsic time based on the appearance of links over time does not seem the most appropriate here: indeed, we see clearly that the diffusion using intrinsic time is very sensitive to events in the graph, and therefore we cannot properly isolate diffusion from the graph dynamics in this case.

In summary, this experiment on Infectious SocioPatterns shows that the use of intrinsic time may lead to very complementary and valuable observations, providing that the definition of intrinsic time is appropriate.

6 Conclusion and Future Work

In this article, we have used the concepts of intrinsic and extrinsic times to study diffusion phenomena in evolving networks. We have first illustrated the impact of these two concepts of time on diffusion in synthetic BA graphs. This study has shown that intrinsic time allows us to somehow isolate the network dynamics from the diffusion phenomenon.

Subsequently, we have used these concepts to observe simulated diffusion on real datasets. Our results on these datasets have shown significant differences in the analysis of the diffusion in the extrinsic, extrinsic converted into intrinsic and intrinsic times, provided that the definition of intrinsic time is adapted. Indeed, when this is the case, the diffusion observed is not really impacted by the evolution of the network topology. In extrinsic time on the contrary, this network topology plays a major role. We have shown that it is very interesting to study the same diffusion with extrinsic time and extrinsic time converted into intrinsic time, as converted time provides additional information for the interpretation of diffusion in extrinsic time. However, we have also seen with the Infectious SocioPatterns case that intrinsic time does not always gives us more information than extrinsic time when it is not adapted to network dynamics.

In our future work, we will first focus on the generalization of intrinsic time notion. Indeed, for the moment, this concept is only defined for networks which grow over time. We will therefore study how we can extend this notion for other types of dynamics. For instance, for contact networks, we have seen that a definition of intrinsic time based on link appearance is not appropriate. We will consider other definitions for intrinsic time, based on triangles creation for instance. Thereafter, we will use these extended definitions to study diffusion on other datasets, with different

topologies and dynamics, where links and nodes can appear and disappear. We will finally use other types of diffusion models for our simulations: other epidemiological models like SIS or SIR, and other classes of diffusion models like threshold models.

Acknowledgments This work is supported in part by the French National Research Agency contract DynGraph ANR-10-JCJC-0202, and by the CODDDE project ANR-13-CORD-0017-01.

References

1. Kermack M, McKendrick A (1927) Contributions to the mathematical theory of epidemics. part i. *Proc R Soc A* 115(5):700–721
2. Jensen R (1982) Adoption and diffusion of an innovation of uncertain profitability. *J Econ Theor* 27(1):182–193
3. Barthelemy M, Barrat A, Pastor-Satorras R, Vespignani A (2005) Dynamical patterns of epidemic outbreaks in complex heterogeneous networks. *J Theor Biol* 235(2):275–288
4. Albano A, Guillaume J-L, Le Grand B (2012) File diffusion in a dynamic peer-to-peer network, in mining social network dynamics, in conjunction with the world wide web conference. ACM, pp 1169–1172
5. Fefferman N, Ng K (2007) How disease models in static networks can fail to approximate disease in dynamic networks. *Phys Rev E* 76(3):031919
6. Liu S-Y, Baronchelli A, Perra N (2013) Contagion dynamics in time-varying metapopulation networks. *Phys Rev E* 87(3):032805
7. Lee S, Rocha LE, Liljeros F, Holme P (2012) Exploiting temporal network structures of human interaction to effectively immunize populations. *PLoS ONE* 7(5):e36439
8. Delre SA, Jager W, Bijmolt TH, Janssen MA (2010) Will it spread or not? the effects of social influences and network topology on innovation diffusion. *J Prod Innovation Manage* 27(2):267–282
9. Eames KT, Tilston NL, Brooks-Pollock E, Edmunds WJ (2012) Measured dynamic social contact patterns explain the spread of h1n1v influenza. *PLoS Comput Biol* 8(3):e1002425
10. Miritello G, Moro E, Lara R (2011) Dynamical strength of social ties in information spreading. *Phys Rev E* 83(4):045102
11. Albano A, Guillaume J-L, Heymann S, Le Grand B (2013) A matter of time—intrinsic or extrinsic—for diffusion in evolving complex networks. In: Proceedings of the international conference on advances in social networks analysis and mining (ASONAM)
12. Heymann S, Grand BL (2013) Monitoring user-system interactions through graph-based intrinsic dynamics analysis. In: Proceedings of the 7th IEEE international conference on research challenges in information science
13. Newton I (1687) *Philosophiæ Naturalis Principia Mathematica*
14. Kant I (1781) *Kritik der reinen Vernunft*
15. de la Convention du Mètre OI (2006) The international system of units (SI), Bureau international des Poids et Mesures. Tech Rep 8
16. Barabási A-L, Albert R (1999) Emergence of scaling in random networks. *Science* 286(5439):509–512
17. Github press page [Online]. <https://github.com/about/press>
18. Github archive [Online]. <http://www.githubarchive.org>
19. Isella L, Stehlé J, Barrat A, Cattuto C, Pinton J-F, Van den Broeck W (2011) What’s in a crowd? analysis of face-to-face behavioral networks. *J Theor Biol* 271(1):166–180

A Learning Based Approach for Real-Time Emotion Classification of Tweets

Olivier Janssens, Rik Van de Walle and Sofie Van Hoecke

Abstract Emotion recognition can be done in a wide range of applications to enhance the user experience. Because of these many types of applications there are a large range of different data types that can be processed, such as text, video, speech, sound, accelerometer data and various bio-sensor data types. In order to bring emotion recognition into everyday use, it is important to work with data types and sources that are available to everyone. Therefore in this chapter twitter data is used for emotion recognition. Since emotion recognition applications need to uncover the user's emotion fast, the focus lies on real-time emotion classification. Sentiment analysis or emotion recognition research often uses a lexicon based approach, though in this chapter a learning based approach is used. Nine emotion classification algorithms are compared with focus on precision and timing. This chapter shows that accuracy can be enhanced by 5.83 % compared to the current state-of-the-art by improving the features and that the presented method work in real-time.

Keywords Social network analysis · Emotion recognition · Machine learning

1 Introduction

As emotions influence everything in our daily life, e.g. relationships and decision-making, a lot of research tackles the identification and analysis of emotions. Research shows that the automatic detection of emotions by computers allows for a broad range of applications, such as stress detection in a call center [17], general mood

O. Janssens (✉) · R.V. de Walle · S.V. Hoecke
Department of Electronics and Information Systems, Ghent University,
Gaston Crommenlaan 8, 9050 Ledeborg-Ghent, Belgium
e-mail: odjansse.janssens@ugent.be

R.V. de Walle
e-mail: rik.vandewalle@ugent.be

S.V. Hoecke
e-mail: sofie.vanhoecke@ugent.be

detection of a crowd [18], emotion detection for enhanced interaction with virtual agents [26, 35], detection of bullying on social networks [7], and detection of focus for an automatic tutoring systems [8].

By introducing emotion recognition in a system, direct unspoken feedback of the user can be obtained, allowing a system to adapt its content in real-time in order to enhance the user experience. This automatic adaptation of content can result in a prolonged interaction with the system and better performance of the user in an application.

In order to capture the emotions of a user, different sensor data has been researched. Popular sensors and media are electromyography [26], which uses an electromyograph to detect the electrical potential generated by muscle cells; skin conductance [18], which measures the electrical conductance of the skin and varies with moisture level; body posture measurement system [8], which can be used to monitor the gross body language of a person; and of course video [44, 45, 47], speech [35] and text [1, 3, 4, 21, 25, 31, 37, 43].

Because these sensors are expensive or not operable by a non-trained person, they can only be used in the controlled environment of a lab. In order to bring emotion recognition to every person and every application, emotion recognition needs to be enabled on devices which are available to the general public. The perfect example of such devices are pcs, smartphones or tablets. These devices allow a user to send chat messages, visit social networking sites and send e-mails.

Because platforms which use text are widely used, e.g. Twitter, Facebook, sms, the goal of this paper is to improve emotion recognition on text. Whereas current state-of-the-art techniques use lexicon based approaches [25, 43], this paper uses a learning approach to improve emotion recognition on text. Lexicon based techniques use dictionaries of words with pre-computed scores expressing the emotional value of the word. The advantage of this technique is that a dataset with ground truth is not required. However, the disadvantage of this method is that no new features can be extracted from the relevant dataset. In this chapter, a learning based approach is used which requires the creation of a model by training a classifier with labelled examples. This means that the vocabulary for the feature vectors are built based on the dataset.

It is safe to state that identifying the expressed emotions in text is very challenging for two reasons: (1) emotions can be hidden in the text without explicit reference to words such as “scare” and “panic”; (2) identifying emotions based on keywords can be very subtle as for example “hate” can belong to different emotions (i.e. anger and embarrassment) [43]. So, important aspects are correct classification but also processing time. In order to add emotion recognition to real-time services like Twitter, the emotion recognition has to work in real-time. These two goals (performance and precision) are set so that the following use cases, amongst others, become possible for mobile users:

Use case 1

When a person wants to tweet his opinion on a certain event, the underlying emotion of the text can be detected via a Twitter client which supports emotion recognition. As a result, this Twitter client can suggest a hashtag to the person, corresponding with his feelings at that moment.

Use case 2

When a user is playing a game and the user gets frustrated, he might tweet while being angry. If this emotion can be detected in his tweets, the gameplay can be automatically adjusted to make the game easier so that the user becomes happier.

Use case 3

When a user is using an application on his mobile device and is happy with the application, he can tweet about the application and his appreciation for it. It is possible to detect this emotion in the tweet and send automatic feedback to the developers that people are enjoying their application.

One of the use cases which has already been researched in the state-of-the-art is to detect a user's interest/anticipation based on a tweet. This interest/anticipation is important to detect "intention tweets". Intention tweets are tweets through which people explicitly or implicitly express their intentions to consume certain goods or services that can be used to predict future sales. Rui and Whinston [33] recently proposed a framework to forecast movie box office revenues during the opening weekend and forecast daily revenue after 4 weeks based on intention tweets.

To conclude, emotion recognition can be used to give feedback on the content being used and help forecast certain events. The focus of this chapter is on adding emotion recognition to a real-time service like Twitter. The remainder of this chapter is as follows. Section 2 gives an overview of different emotion models for emotion classification. Next, Sect. 3 discusses the designed architecture for emotion recognition. Subsequently, Sect. 4 describes the dataset and feature selection in order to build the training and testing set that is used in Sect. 5 to compare the different classification algorithms. Section 5 show the evaluation results. Finally, Sect. 6 lists the general conclusions.

2 Emotion Classification

The definition of emotions is stated to be: "Episodes of coordinated changes in several components (including at least neurophysiological activation, motor expression, and subjective feeling but possibly also tendencies and cognitive processes) in response to external or internal major significance to the organism" [34]. Emotion classification starts with defining these emotions into an emotion model.

2.1 Emotion Models

One way to categorize emotions is to categorize them using the “wheel of emotions” [30] (see Fig. 1), designed by Plutchik. The wheel of emotions consists of 8 primary bipolar emotions, like trust versus disgust, joy versus sadness, and anger versus fear, which all can be represented in a circle. The wheel of emotions can be seen as a color wheel, where colors are represented as a combination of primary colors. This color wheel concept is transferred to the wheel of emotions so primary emotions can mix to form new complex emotions. This emotion model is used by Rui and Whinston to predict movie box office revenues as discussed earlier. The interest/anticipation can be seen at the top left-hand side of the wheel.

Another frequently used model is the “arousal and valence model” [11], representing emotions in a 2D space. In this space, valence represents the way one



Fig. 1 Wheel of emotion [30]

judges a situation, from unpleasant to pleasant, and arousal expresses the degree of excitement, from calm to exciting.

The main difference between these two models is that the arousal and valence model is continuous, representing every emotion on a two dimensional axis and thus using coordinates to classify emotions, while the wheel of emotions is discrete, classifying emotions to a discrete state. The coordinates in the arousal and valence model define how much arousal and how much valence an emotion has.

Finally, another widely used emotion model is given by Ekman [9] who defined six basic emotions, i.e. surprise, fear, happiness, sadness, anger and disgust, based on universal facial reactions. The emotions are considered universal, although other theories list slightly different basic emotions [5, 12–14, 22, 24, 27, 28, 39, 40]. A model, related to the model of Ekman, is the model of Parrott [13], consisting of a tree structure where the first nodes are the six basic emotions of Ekman. When going from top to bottom in the tree, secondary and tertiary emotions are added underneath the six basic emotions.

In this chapter an adaptation of the six basic emotions of Ekman is used. As this emotion model was also used in the acquired dataset, the improvements of this chapter compared to the state-of-the-art can easily be quantified. Details are described in Sect. 4. First, the next section gives an overview of the designed architecture for emotion classification of tweets.

3 Architecture for Emotion Recognition for Real-Time Emotion Recognition

When using Twitter, the focus lies on speed. It is important for the user to tweet fast. This implies that there should be almost no delay when sending a tweet into the world. The efficiency of processing a tweet is especially important when using mobile devices since they have less computing power than a computer. In order not to restrict emotion recognition to high-end, multi-core smartphones, desktop computers and tablets, a client-server architecture is designed so the emotion recognition can be done on a dedicated server. Figure 2 illustrates the general concept. As can be seen in this figure, the Twitter client provides tweet data to the platform. This data is processed by the server and the resulting classified emotion is sent back to the client. The application can then, as illustrated in the use cases above, react or behave differently depending on the detected emotion.

The architecture is designed based on the principles of service-oriented architectures, wherein all components are implemented as REST services [32]. In view of the broad support for REST services, they are ideal for the integration of heterogeneous software components since applications can easily be distributed and expose well-defined functionality as a service. Important to notice is that no processing is done on the client itself, assuring that for example smartphones are able to use emotion recognition, independent of their processing power or storage capabilities. By

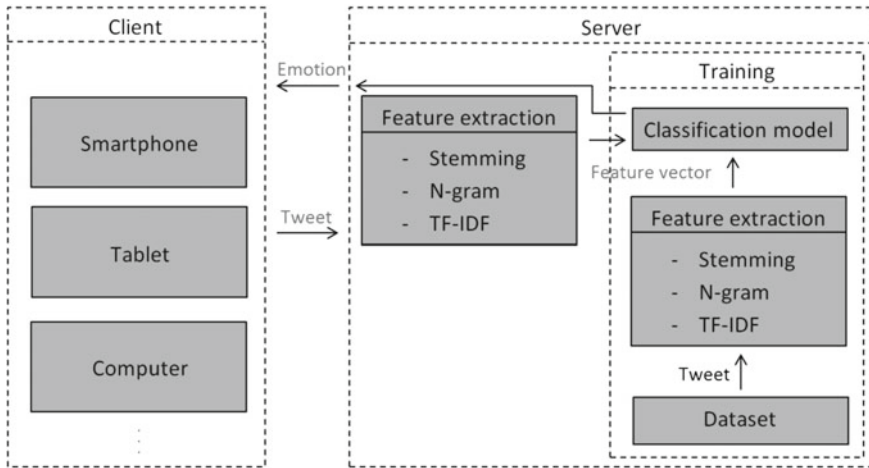


Fig. 2 A high-level architecture for emotion classification of tweets

using the generic concepts of service-oriented computing, the architecture is also not restricted to emotion recognition in tweets. Due to its generic and scalable design, it can be extended to detect emotions in different sorts of data, using other sensors as well, or it can even be extended to provide multi-modal emotion recognition where the precision of the classification of emotions is improved by combining different media [6, 36, 42].

Since the emotion classification is done server-side, the performance and processing time of the server has to be optimized on one hand, and on the other hand, the communication overhead has to be kept to a minimum to optimize performance and restrict bandwidth usage. As the real-time aspect is important, REST services are a lightweight alternative to the proven Web service and RPC technologies [41] and more convenient for mobile devices with acceptable battery consumption [15]. As a result, the REST communication overhead is negligible compared to the classification time, which includes time for the feature extraction and actual classification of the tweet. Therefore it is important to minimize this classification time.

Apart from the time aspect, the classification accuracy is also very important. It is required that the tweets are correctly classified. Therefore several state-of-the-art classification algorithms are compared to see which one has the best classification rate. The next section discusses the emotion classification based on tweets in more detail.

4 Emotion Recognition on Twitter

Twitter was founded in 2006 and its popularity keeps rising every day. Currently there are over 500 million registered users who generate over 340 million tweets a day. One of the reasons Twitter is so popular is because it is very accessible. Tweeting can be

done from various platforms, ranging from websites, mobile apps to even toasters.¹ As it is possible to tweet from a broad range of devices, Twitter is a good medium to gather data for text classification.

Although a lot of data is available, automatic text classification is hard for a computer due to the innumerable amount of features (words). The classification of tweets is even harder due to the limited word count in a tweet. Automatic emotion classification on Twitter has recently been published by Wang et al. [43] where they use a lexicon based approach.

In order to easily allow comparison with the state-of-the-art, the same dataset as Wang et al. [43] is used in this chapter. This dataset consists of seven emotion classes, one more than the six basic emotions of Ekman, and is discussed in more depth in the next subsection. As already stated, the goal of this chapter is to improve the classification rate and additionally minimize the processing time to support real-time emotion recognition of Tweets. The state-of-the-art, nor Wang's work contains any indication of the processing time.

4.1 Dataset

The dataset used in this chapter is part of the large dataset from Wang et al. This original dataset of 2,500,000 tweets was split into ten representative parts: 1 training dataset, 8 development dataset and 1 testing dataset. For this chapter the training and testing dataset were obtained consisting of 248,892 and 249,993 tweets respectively. In the research of Wang, at first, the same training set was used to train the algorithm and the testing set was used to test the algorithm. By using these two sets an accuracy of 61.63 % was achieved by Wang et al. They were able to improve their results by training the classifier with a 10 times larger dataset, containing 2,500,000 tweets, which resulted in a 65.57 % accuracy. However as stated before the reduced set of approximately 250,000 tweets is used in this chapter.

In Table 1, the emotion occurrences for both the training and the test set are presented. As can be seen, the test set is very representative for the training set in number of occurrences for each emotion. However, a significant imbalance between the emotion categories can be seen. This imbalance needs to be taken into account when choosing the parameters for the various classification algorithms. It is also self-evident that the testing and training sets are disjoint.

All tweets were automatically annotated with hashtags such as #irritating, #nervous, #love. This was done by the Twitter users and was a fast way to create a ground truth for all these tweets. However, this method does not take into account the fact that the tag does not always correspond well with the content of the tweet. For example the tweet can be meant sarcastically and therefore having the emotion angry while being annotated with #love. It is also possible that the tweet is not in the same language as the model is trained on, making it almost impossible to classify the tweet. In order

¹www.instructables.com/id/social-networking-for-my-toaster/.

Table 1 Frequency of emotion occurrence in the dataset

Emotion	Training count	Training (%)	Test count	Test (%)
Anger	57,500	23.10	57,530	23.01
Fear	13,434	5.40	13,877	5.55
Joy	70,805	28.45	71,194	28.48
Love	30,269	12.16	30,223	12.09
Sadness	61,317	24.64	61,591	24.64
Surprise	2,421	0.97	2,391	0.96
Thankfulness	13,146	5.28	13,187	5.28
Total	248,892		249,993	

to prevent these problems, Wang et al. randomly sampled 400 tweets which were inspected by two annotators who indicated if the tweets are relevant or irrelevant. A tweet was deemed relevant if the hashtag corresponded to the writer's emotion. Wang et al. initially started of with 5 million tweets, which were slimmed down to 2.5 million tweets by using three filtering heuristics. These filtering heuristics were developed on the 400 randomly sampled tweets. The filters are as follows: (i) only tweets with hashtags at the end were kept, since those are more likely to correspond to the writer's emotion; (ii) tweets with less than 5 words were discarded; (iii) tweets with URLs or quotations were also removed from the dataset.

The first step in order to be able to classify tweets is to preprocess them and standardize the text.

4.2 Preprocessing

First, the emotion hashtags are removed from the dataset in order to construct the ground-truth data. If the hashtags are kept in the dataset, data leakage [20] would be created which results in an unrealistically good classifier. The usernames which are lead by the *at sign* are kept in the dataset. The next step consist of stemming, which reduces words to their stem. An example: kneeling, kneeled, kneels are all reduced to kneel. As the feature space, when using learning based methods, can grow large very quickly, reducing conjugations of the same word to their stem reduces this feature space.

The second step in classifying the short Twitter messages is to convert them to a feature space which can be used as input for various machine learning algorithms. As it is important to choose both good and sufficient features, the next section discusses the chosen feature extraction methods in more detail.

4.3 Feature Extraction Methods

The feature extraction methods transform words and sentences into a numerical representation (feature vectors) which can be used by the classification algorithms. In this research, the combination of N-grams and TF-IDF feature extraction methods is chosen in order to preserve syntactic patterns in text and solve class imbalance respectively.

4.3.1 N-Gram Features

In the field of computational linguistics and probability, an N-gram is a contiguous sequence of N items from a given sequence of text or speech. An N-gram of size 1 is referred to as a “unigram”, size 2 is a “bigram”, size 3 is a “trigram”. An N-gram can be any combination of letters, syllables, words or base pairs according to the application. The N-grams typically are collected from a text or speech corpus. In this chapter, N-grams are used to represent a sequence of N words, including emoticons, in the training set and used as features to convert the tweets to the feature space. Table 2 shows an example for converting a tweet into the feature space using unigrams, bigrams and trigrams.

As can be seen in Table 2, there are 27 possible features for the given example. In this research a combination of 1, 2 and 3 g is used which are then passed to the TF-IDF algorithm.

4.3.2 Term Frequency-Inverse Document Frequency (TF-IDF)

One of the broadly used features is the bag of words representation together with word occurrence [3], equalizing a feature value to the number of times it occurs. This method is fairly straightforward and easy, though it has a major downside as longer documents have higher average count values.

Table 2 An example of converting a tweet into feature space using unigrams, bigrams and trigrams

Original tweet	My	alarm	just	never	works	on	a	saturday,	#Annoying	!
Converted tweet	my	alarm	just	never	works	on	a	saturday,	#annoying	!
Unigram	u1	u2	u3	u4	u5	u6	u7	u8	u9	u10
Bigram	b1		b2		b3		b4		b5	
		b6		b7		b8		b9		
Trigram	t1			t2			t3			
		t4			t5			t6		
		t7				t8				

Table 3 Results of the stochastic gradient descent classifier with occurrence count

	Precision	Recall	f1-score	Number of tweets
Joy	0.6421	0.6978	0.6688	71194
Fear	0.4678	0.4023	0.4326	13877
Sadness	0.6031	0.5260	0.5619	61591
Thankfulness	0.6155	0.4396	0.5129	13187
Anger	0.6116	0.7101	0.6572	57530
Surprise	0.3570	0.1702	0.2305	2391
Love	0.4561	0.4469	0.4514	30223
Avg/total	0.5892	0.5929	0.5880	249993

As can be seen in Table 1, the dataset suffers a major imbalance, for example there are significantly less tweets belonging to the class “fear” and “surprise” than to the other classes. In order to see if this imbalance also reflects in the classification results when using occurrence to construct the feature vectors, a test is done. In this test the tweets are transformed to feature vectors by using the bag of word representation together with word occurrence. When these feature vectors are constructed, they are used by a stochastic gradient decent (SGD) classifier with modified huber loss [29]. When looking at the results presented in Table 3, it is safe to state that the imbalance can be detected. As can be seen fear, surprise and love which are less represented in the dataset, have lower precision.

To avoid these potential discrepancies, the number of occurrences of each word in a document is divided by the total number of words in the document, called normalization. These new features are called Term Frequencies (TF). Additionally, weights for words that occur in many documents can be downscaled making them less dominant than those that occur only in a small portion of the corpus. The combination of term frequencies and downscaling weights of dominant words is called Term Frequency Times Inverse Document Frequency (TF-IDF).

By composing the vocabulary for the feature vectors based on the dataset, the feature vectors will be very high dimensional. But by combining TF-IDF and the N-grams method, a feature vector will be very sparse. This can be beneficial if the used algorithms can work with sparse data.

4.4 Comparison of Classification Algorithms

Now that the tweets can be converted to feature vectors, they are ready to be used as input for various machine learning algorithms. The features of the training set are given to nine classification algorithms for training. Four algorithms are linear classifiers that use stochastic gradient descent with different loss functions. Stochastic gradient descent is a method where the gradient of the loss is estimated each sample

at a time. This means that not all the samples have to be loaded into the memory at once. This results in an algorithm that can be used on large datasets that normally do not fit in the memory. The stochastic gradient descent algorithms respectively use the *modified huber loss*, *hinge loss*, *log loss* and the *perceptron loss*. Because of these different loss functions, the data will be handled differently. For example the modified huber loss reduces the effect of the outliers much more than the log loss function. For more information regarding stochastic gradient descent we refer to the work of Zhang et al. [46]. Apart from the stochastic gradient descent algorithm, the following algorithms are also used: LIBLINEAR, a linear SVM optimized for large datasets [10]; multinomial naive bayes (MNB) [23], which is the naive bayes algorithm implemented for multinomially distributed data; Bernoulli naive bayes (BNB) [23], which is the naive bayes algorithm for data that is distributed according to multivariate Bernoulli distribution; a ridge classifier [2], which is a linear classifier that uses regularized least-squares; and a nearest centroid (NC) classifier [38], which is known as the Rocchio classifier when using TF-IDF feature vectors for text classification. Since linear classifiers are designed to work solely for binary classification problems, we use the One versus all combination method to combine them for the multi-class problem presented here.

The classification results can be found in the results section below.

5 Results

As already stated, both the accuracy and time aspect are important. Both will be discussed here. In order to compare the algorithms, the same dataset and the same features are used for each algorithm. Therefore, the training and testing set of both approximately 250,000 tweets are transformed to the feature space by using a combination of uni-, bi- and tri-grams together with TF-IDF.

5.1 Classification Metrics

In order to be able to compare the performance of the nine algorithms, four error measurements are compared: accuracy, precision, recall and f1-score for which the formulas can be seen in Eqs. 1–4, with $|TP|$ being the amount of true positive classifications; $|TN|$, the amount of true negative classifications; $|FP|$, the amount of false positive classifications and $|FN|$, the amount of false negative classifications.

For comparison reasons the results for a random classifier and also classification according to the majority class are added. Figures 3, 4, 5, 6 show comparisons of the different metrics. The Wang et al. normal bar indicate the results of Wang et al. on the same dataset which is used here. Wang et al. large indicate the results of Wang et al. on the extended dataset of 2,500,000 tweets.

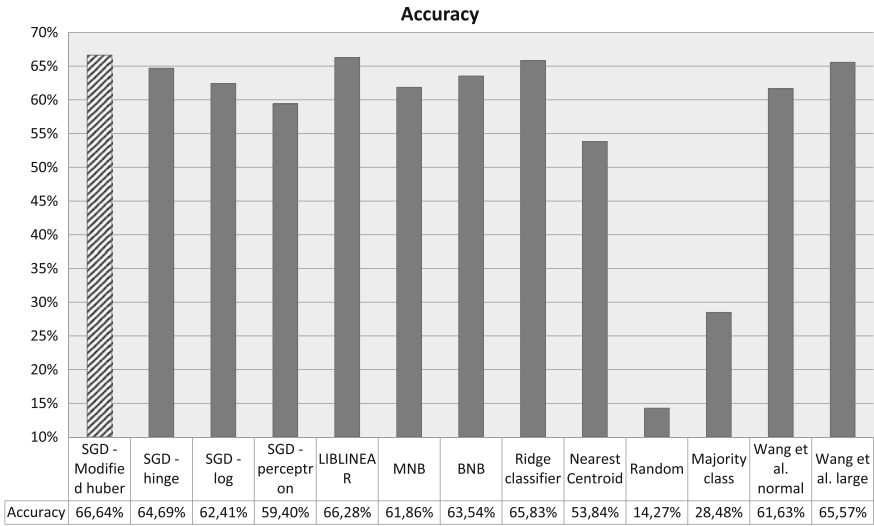


Fig. 3 Accuracy comparison of the different classifiers. The best one is shaded

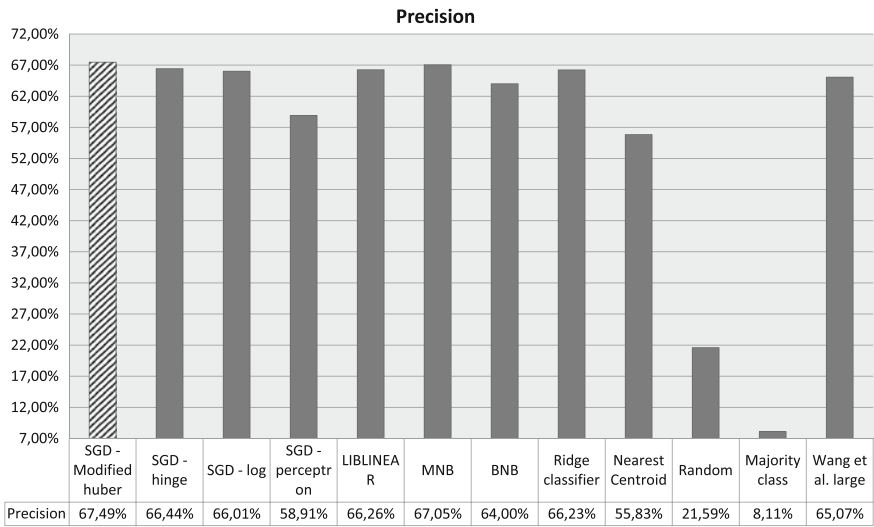


Fig. 4 Precision comparison of the different classifiers. The best one is shaded

$$accuracy = \frac{|TP| + |TN|}{|TP| + |FP| + |FN| + |TN|} \tag{1}$$

$$precision = \frac{|TP|}{|TP| + |FP|} \tag{2}$$

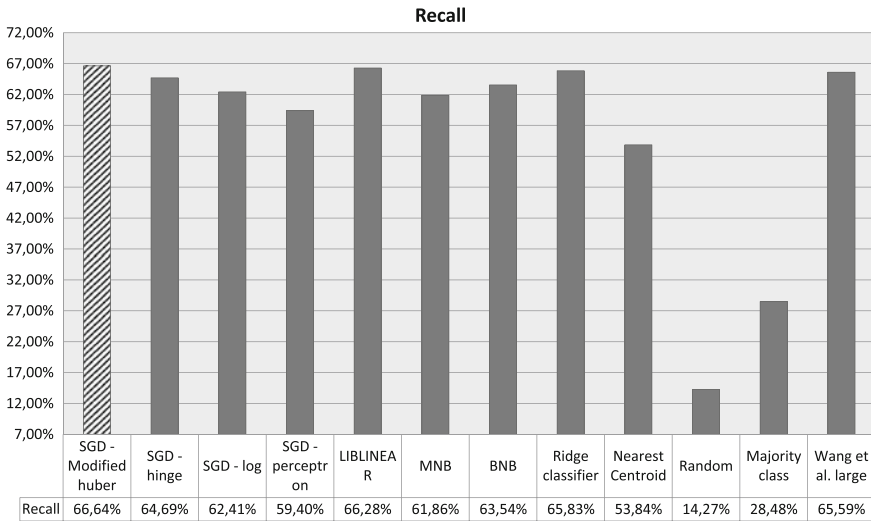


Fig. 5 Recall comparison of the different classifiers. The best one is shaded

$$recall = \frac{|TP|}{|TP| + |FN|} \tag{3}$$

$$f1 - score = \frac{2|TP|}{2|TP| + |PN| + |FN|} \tag{4}$$

As can be seen in Fig. 3, almost all results give a comparable result especially the linear classifiers. However, as can be seen, SGD with a modified huber loss is the best classifier. Note that, although in this chapter only 250,000 tweets are used for all the tests, the accuracy improves by **5.01 %** compared to Wang et al. when the same dataset size is used, and even slightly improves by 1.07 % compared to when the ten times bigger dataset is used. Comparison with Wang’s results based on their precision, recall and f1-score is not possible as these are not provided for the normal dataset. However, when comparing the results of Wang et al. where the large dataset is used, as already stated, an improvement in accuracy (1.07 %) is achieved, but also in precision (2.42 %), recall (1.05 %) and f1-score (0.75 %).

In order to make sure these results are not solely achieved by the division of the data in training and test set, the data is divided again for a 10-fold cross validation test. The classification is done with SGD with modified huber loss. In Table 4. The results can be seen, as well as the improvements compared to the previous results and the results of Wang et al. As can be seen by using 10-fold cross validation the accuracy improvement rises to **5.83 %**

In order to illustrate the confusion of the classifiers, normalized confusion matrices are also constructed for the nine algorithms, see Fig. 7. As can be seen, BNB and

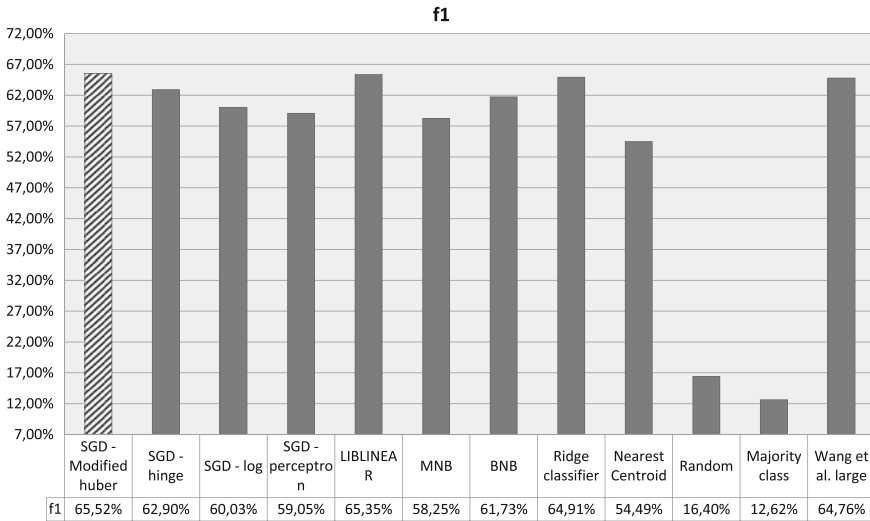


Fig. 6 f1-score comparison of the different classifiers. The best one is shaded

Table 4 Results of 10-fold cross validation on the normal dataset with SGD modified huber loss

	Accuracy (%)	Precision (%)	Recall (%)	f1-score (%)
10-fold cross validation	67.46 (σ 0.26)	68.52 (σ 0.22)	67.41 (σ 0.21)	66.21 (σ 0.23)
Δ SGD modified huber on normal dataset	0.81	1.03	0.76	0.69
Δ Wang et al. large dataset	1.89	3.45	1.82	1.44
Δ Wang et al. normal dataset	5.83	–	–	–

σ is the standard deviation

MNB are the two classifiers which confuse a lot of emotions. It can also be seen that the linear classifiers give similar results. Though fear sometimes gets confused with joy, and surprise sometimes with joy and sadness. The classifier which generally doesn't confuse specific emotions apart from surprise with joy, is the nearest centroid classifier. For all the classifiers it can be stated that there is a bias toward joy, which can be explained as this is the biggest class in the dataset.

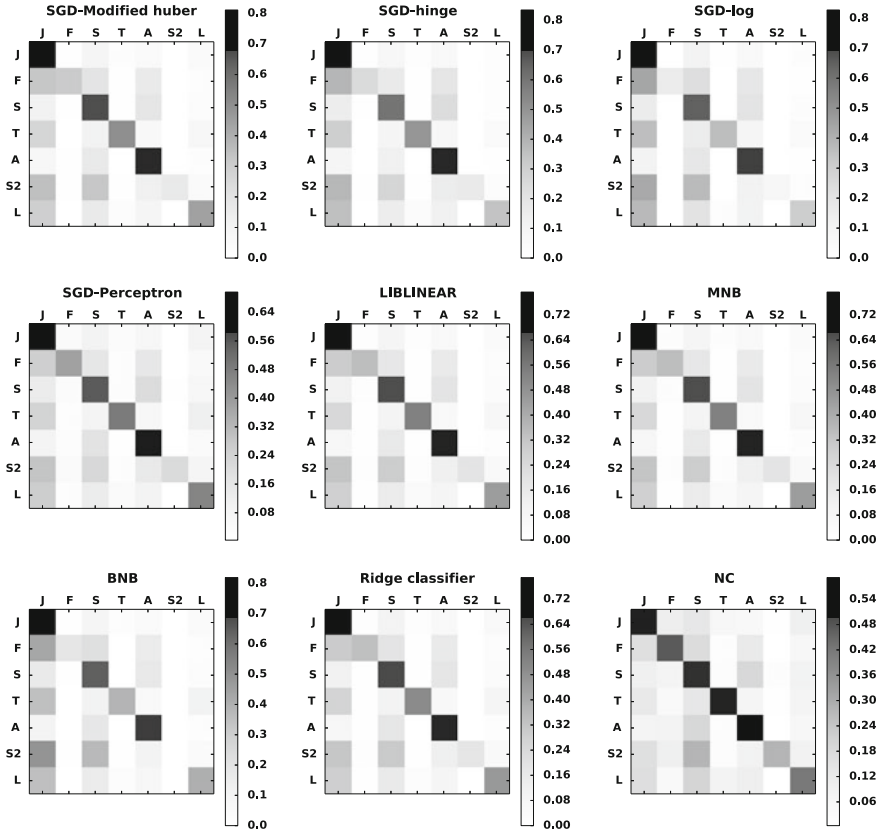


Fig. 7 Confusion matrix of the classification algorithms. J = Joy, F = Fear, S_J = Sadness, T = Thankful, A = Anger, S_2 = Surprise, L = Love

5.2 Time Measurement

Another important aspect in real-time emotion classification is timing. Therefore, the processing time has to be as small as possible. The processing time presented in Table 5 contains the transformation of the test set to the feature space and the emotion classification itself. The server implementation is written in Python with the scikit-learn library [29] and executed on a dual-socket quad-core Intel Xeon Harpertown (L5420) 2.5GHz CPU with 8 gb RAM. As can be seen, all nine algorithms are suitable to work in real-time.

Table 5 Time measurements for classifying the test set which contains 249,993 tweets by the nine algorithms

	Average entire test set (s)	Standard deviation (s)	Average per tweet (μ s)	Standard deviation (μ s)
SGD modified hubber	26.85	1.43	107.41	5.73
SGD hinge	26.88	1.46	107.54	5.84
SGD log	26.89	1.46	107.57	5.85
SGD log	26.88	1.46	107.55	5.85
LIBLINEAR	26.80	1.35	107.23	5.43
MNB	26.80	1.34	107.23	5.38
BNB	27.22	1.40	108.89	5.63
Ridge classifier	26.79	1.33	107.19	5.34
Nearest centroid	27.49	1.44	109.96	5.79

6 Conclusions

In this chapter, nine state-of-the-art classification algorithms are compared with focus on (1) improving the results compared to the results from Wang et al. and (2) achieving real-time classification of tweets. The classification algorithms all use the same features. These features consist of a combination of uni-, bi- and trigrams with TF-IDF. TF-IDF is used because the dataset suffers from imbalance, while a combination of uni-, bi-, and trigrams is used to capture the syntactic patterns in the tweets which would be lost if only unigrams were used.

The best classification results are achieved by stochastic gradient descent with a modified huber loss. This best accuracy score is 5.83 % better than the state-of-the-art provided by Wang et al. It is shown that especially the chosen features are the reason for the improvement of the classification metrics. These features are extracted from the tweets themselves instead of using dictionaries as in a lexicon based approach.

When looking at the time aspect of the classification, it can be seen that the classification is done in real-time as the processing time is between 107.23 μ s and 109.69 μ s for transforming one tweet to a feature vector and classifying its underlying emotion. Compared to a moment in history where the amount of tweets peaked, namely the Super Bowl of 2013 which generated 268,000 tweets per minute,² it is possible to conclude there is even room left for other preprocessing techniques to improve the precision of the classification results while classifying the tweets in real-time.

To conclude, it can be said that a learning based approach for emotion recognition is a good alternative for a lexicon based approach. This chapter also proves that

²<http://cnet.co/14LcGx9>.

emotion recognition can be made very lightweight and fast, which is for example needed to support emotion recognition on mobile devices, allowing a wide spread use in every mobile app.

Acknowledgments This work was partly done within the Friendly Attac project (<http://www.friendlyattac.be/>), funded by the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT).

References

1. Burget R, Karasek J, Smekal Z (2011) Recognition of emotions in czech newspaper headlines. *Radioengineering* 20:39–47
2. Cesa-Bianchi N (2007) Applications of regularized least squares to pattern classification. *Theor Comput Sci* 382:221–231
3. Chaffar S, Inkpen D (2011) Using a heterogeneous dataset for emotion analysis in text. In: Butz C, Lingras P (ed) *Advances in artificial intelligence*, Springer, Berlin, pp 62–67
4. Calix RA, Mallepudi SA, Chen B, Knapp GM (2010) Emotion recognition in text for 3-D facial expression rendering. *IEEE Trans Multimed* 12:544–551
5. Carroll E (1977) *Izard: human emotions*. Springer, New York
6. Datcu D, Rothkrantz LJM (2008) Semantic audio-visual data fusion for automatic emotion recognition. In: *Ghent euromedia, Eurosis*, p 16
7. Dinakar K, Jones B, Havasi C, Lieberman H, Picard R (2012) Common sense reasoning for detection, prevention, and mitigation of cyberbullying. *ACM Trans Interact Intell Syst* 2:1–30
8. D’Mello S, Picard RW, Graesser A (2007) Toward an affect-sensitive AutoTutor. *IEEE Intell Syst* 22:53–61
9. Ekman P (1999) *Basic emotions*. Wiley, New York
10. Fan R, Wang X, Lin C (2012) LIBLINEAR: a library for large linear classification. *J Mach Learn Res* 9:1871–1874
11. Feldman LA (1995) Valence focus and arousal focus: individual differences in the structure of affective experience. *J Pers Soc Psychol* 69:153–166
12. Frijda NH (1986) *The emotions*. Cambridge University Press, New York
13. Gerrod Parrott W (2001) *Emotions in social psychology*, Psychology press, Philadelphia
14. Gray JA (1985) The whole and its parts: behaviour, the brain, cognition and emotion. *Bull Brit Psychol Soc* 38:99–112
15. Hamad H, Saad M, Abed R (2010) Performance evaluation of RESTful web services. *Int Arab J e-Technol* 1:72–78
16. Haq S, Jackson P (2010) Multimodal emotion recognition. In: Wang W (ed) *Machine audition: principles, algorithms and systems*, IGI Global, Hershey, pp 398–423
17. Hernandez J, Morris R, Picard, R (2011) Call center stress recognition with person-specific models. In: DMello S, Graesser A, Schuller B, Martin J-C (eds) *Affective computing and intelligent interaction*. Springer, Berlin, pp 125–134
18. Hernandez J, Hoque ME, Drevo W, Picard RW (2012) Mood meter: counting smiles in the wild. In: *Proceedings of the 2012 ACM conference on ubiquitous computing—ubiComp 12*. ACM Press, New York, pp 301–310
19. James W (1884) What is an emotion? *Mind* 9:188–205
20. Kaufman S, Rosset S (2012) Leakage in data mining: formulation, detection, and avoidance. In: *17th ACM SIGKDD international conference on knowledge discovery and data mining*. pp 556–563
21. Liu C-L, Hsaio W-H, Lee C-H, Lu G-C, Jou E (2012) Movie rating and review summarization in mobile environment. *IEEE Trans Syst Man Cybern Part C Appl Rev* 42:397–407

22. McDougall W (1929) An introduction to social psychology, John W. Luce & Co., Boston
23. Metsis V, Androutsopoulos I, Paliouras G (2006) Spam filtering with naive bayes—which naive bayes? In: Third conference on email and anti-spam
24. Mowrer OH (1960) Learning theory and behavior. Wiley, New York
25. Mohammad S (2012) # Emotional tweets. In: Proceedings of the first joint conference on lexical and computational semantics. Association for Computational Linguistics, Montréal, Canada, pp 246–255
26. Nakasone A, Prendinger H, Ishizuka M (2005) Emotion recognition from electromyography and skin conductance. In: The fifth international workshop on biosignal, Interpretation, pp 219–222
27. Oatley K, Johnson-Laird PN (1987) Towards a cognitive theory of emotions. *Cogn Emot* 1:29–50
28. Panksepp J (1982) Toward a general psychobiological theory of emotions. *Behav Brain Sci* 5:407–467
29. 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 (2011) Scikit-learn: machine learning in python. *J Mach Learn Res* 12:2825–2830
30. Plutchik R (2001) The nature of emotions. *Am Sci* 89:344
31. Quan C, Ren F (2010) Sentence emotion analysis and recognition based on emotion words using Ren-CECps. *Int J Adv Intell* 2:105–117
32. Richardson L, Ruby S (2007) RESTful web services, O'Reilly Mediam, California
33. Rui H, Whinston A (2011) Designing a social-broadcasting-based business intelligence system. *ACM Trans Manag Inf Syst* 2:119
34. Scherer K, (2000) Psychological models of emotion, In: Borod J (ed) The neuropsychology of emotion. Oxford University Press, New York, pp 137–162
35. Sebe N, Cohen I, Huang T (2005) Multimodal emotion recognition. *Handb Pattern Recognit Comput Vis* 4:387–419
36. Sebe N, Cohen I, Gevers T, Huang TS (2006) Emotion recognition based on joint visual and audio cues. In: IEEE 18th international conference on pattern recognition (ICPR06). pp 1136–1139
37. Seol Y, Kim D, Kim H, (2008) Emotion recognition from text using knowledge-based ANN, In: The 23rd international conference on circuits/systems, Computer and Communications, pp 1569–1572
38. Tibshirani R, Hastie T, Narasimhan B, Chu G (2002) Diagnosis of multiple cancer types by shrunken centroids of gene expression. *Proc Natl Acad Sci USA* 99:656–772
39. Tomkins SS (1984) Affect theory, Approaches to emotion, pp 163–195
40. Watson JB (1930) Behaviorism. University of Chicago Press, Chicago
41. Hoecke S Van, Verdickt T, Dhoedt B, Gielen F, Demeester P (2005) Modelling the performance of the web service platform using layered queueing networks. In: Arabnia H, Reza H (eds) International conference on software engineering research and practice (SERP05). CSREA Press, Athens, pp 627–633
42. Verstockt S (2012) Multi-modal video analysis for early fire detection. Ghent University, Ghent
43. Wang W, Chen L, Thirunarayan K, Sheth AP (2012) Harnessing twitter big data for automatic emotion identification. In: IEEE 2012 International conference on privacy, security, risk and trust and 2012 international conference on social computing. pp 587–592
44. Wu T, Bartlett MS, Movellan JR (2010) Facial expression recognition using gabor motion energy filters. 2010 IEEE computer society conference on computer vision and pattern recognition—workshops. pp 42–47
45. Yang S, Bhanu B (2011) Facial expression recognition using emotion avatar image. In: IEEE face and gesture 2011. pp 866–871
46. Zhang T (2004) Solving large scale linear prediction problems using stochastic gradient descent algorithms. Twenty-first international conference on Machine learning—ICML 04. ACM Press, New York, p 116
47. Zhang L, Tjondronegoro D (2011) Facial expression recognition using facial movement features. *IEEE Trans Affect Comput* 2:219–229

A New Linguistic Approach to Assess the Opinion of Users in Social Network Environments

Luigi Lancieri and Eric Leprêtre

Abstract This article describes an automated technique that allows to differentiate texts expressing a positive or a negative opinion. The basic principle is based on the observation that positive texts are statistically shorter than negative ones. From this observation of the psycholinguistic human behavior, we derive a heuristic that is employed to generate connoted lexicons with a low level of prior knowledge. The lexicon is then used to compute the level of opinion of an unknown text. Our primary motivation is to reduce the need of the human implication (domain and language) in the generation of the lexicon in order to have a process with the highest possible autonomy. The resulting adaptability would represent an advantage with free or approximate expression commonly found in social networks environment.

Keywords Sentiment analysis · Human expression · Text-mining · Adaptive classification · Length of text

1 Introduction

In the last decade there has been an increasing effort in the linguistic and data-mining community to address the question of the computation of the opinion from a textual content. Opinion mining is often viewed as a sub-field of sentiment analysis that, as the discourse analysis or the linguistic psychology, seek to evaluate affective state through the analyze of natural language. Nevertheless, many researchers define sentiment, loosely, as a negative or a positive opinion [18, 20]. This relatively new field is promising from a scientific point of view because of its large possible applications. The challenges are linked to the huge quantity of data available. Thus, applications

L. Lancieri (✉) · E. Leprêtre
Université de Lille1, Laboratoire D’Informatique Fondamentale de Lille,
Villeneuve-d’Ascq, France
e-mail: luigi.lancieri@univ-lille1.fr

E. Leprêtre
e-mail: eric.lepretre@univ-lille1.fr

such as business intelligence, trend forecasting or recommendation systems would take benefits from opinion mining.

The basic principle generally starts with the necessity to use or to generate a lexicon that makes a link between words and their opinion value. Then, this lexicon will be used to rate a text by combining the opinion value of each of its words. Unfortunately, the generation of this lexicon is not obvious because of the complexity of the language rules or of the diversity of the modes of expression. Indeed, the language is alive, evolves and is subject to all kinds of exceptions or common mistakes. It is not rare to find sentences with mixed opinions, with several sources (e.g. quotation of other persons) or several targets of the opinion (e.g. comparison of products or features). The literature is full of examples of expressions where the same word can be interpreted differently depending on its use in the sentence. All these cases introduce biases in the opinion interpretation and reduce the performance of a computational evaluation. Denecke shows, for example, that Senti-WordNet (lexicon with opinion labels, see below) has difficulties to evaluate news articles that are generally wrote in central style. In such a case, the accuracy of the classification stood to 40 % [8]. The task is even more difficult in social networks environments that are heterogeneous and noisy by nature. The freedom of expression that tends to lead to abbreviations or malformed sentences does not fit to standard lexicons. It is important to point out that not only the generation of the lexicon is tough from a linguistic point of view but also, a lexicon is highly context dependent. In other words, each context in a specific language needs a dedicated effort to solve these unobvious problems. In the same way, existing lexicons cannot be easily transposed in other languages or other contexts.

In consequence, facilitate the production of the lexicon seems to be a major research issue. Apart the manually generated lexicons, there are several, more or less, automated solutions. Often using machine learning techniques, researchers attempt to make easy the lexicon generation and try to offer a better adaptivity to these multiple variations. But, how far can we go into this simplification?

It is difficult to have a clear answer to this question but the state of the art shows that there is a huge potential of progress. Indeed there are very few works focusing on the adaptable generation of the lexicon. In contrast, obtaining a better performance in the polarity computation has been extensively discussed and seems to have a limited margin of progress. Indeed, we can notice that in the 29 studies from 1997 to 2009, reported by Mejova, 19 reveal more than 80 % of accuracy and 6 of them more than 90 % [20].

In this paper, we present our contribution that takes profit of the natural asymmetry of the expression of opinions. In short, this psycholinguistic feature of the human behavior makes that negative narrations are longer than positive ones. This is observable in situations of “free expression” such as when users give, on-line, their feeling on their buy or on the merchants quality. Probably because there is less to say when all is going well than when it is going bad, negative posts are statistically longer than positive ones. We find that, with enough of such independent narrations, the length of the text can be used to automatically differentiate positive from negative vocabulary and generate a list of words with polarity tags. Such lexicons can then be

used to evaluate the opinion of an unknown text. Since the need of prior knowledge is limited, the human involvement in the generation process is very low and do not need to be technically or linguistically specialized. This allows to create a lexicon as frequently as needed, for a specific domain or language. In order to validate our theory, we collected consumers textual feedbacks and their associated (stars) ratings. First, we generate the lexicon with a first subset of the users' comments. The ratings are not used in the lexicon generation phase. Then, we compute the opinions values of the second subset and we compare the result with the users' ratings.

The rest of this paper is organized in 5 other sections where we first develop a state of the art on opinion-mining. In the section three, we develop the basis of our main hypothesis regarding the relation between the opinion polarity and the length of the expression. Then we present our proposal and the associated results and finally we discuss its perspectives and limitations.

2 State of the Art on Opinion-Mining

Opinion mining techniques can be roughly segmented in two categories as they are bottom-up or top-down. Even if a lexicon is always needed, the way to create it can widely differ. The first category needs the most prior knowledge and starts from existing lexicons often manually adapted to include sentiment knowledge tags. The second approach uses a series of textual documents that have been globally annotated. An example is a 5 lines long text of a customer comment provided with a 4 stars rating. These couples of comments/ratings are used to infer the polarity of words and to generate a reference lexicon containing words and their polarity. These two opposed approaches have also been combined.

2.1 *Lexicons Generation*

In this study, we define a lexicon as a list of words with one or several language attributes. This can include a basic list of words with polarity tags or a more complex dictionary with grammatical and sentiment class attributes. The polarity or the affect annotations can be added in several manners but in all cases it needs prior knowledges.

Most of the time, sentiment based lexicons have been manually constructed by extending general purpose lexicons associating words with affects categories and degree of relatedness with these categories [7, 24]. The probably well-known example is the public domain Princeton WordNet lexicon [19] that has lead to several other versions. The original WordNet is now available in around 80 languages but it is rather static and difficultly open to new languages, to emerging words or to multiple domains. As examples of sensitive lexicons extended from WordNet, we can mention WordNet-Affect that contains labels linking words with several affective categories [23] or Senti-WordNet that adds polarity and subjectivity labels [12]. WordNet has

also been used to automatically generate basic lists of positive and negative terms [14, 15]. Let us also mention, among other examples, the Harvard General Inquirer lexicon from both the “Harvard” and “Lasswell” general-purpose dictionaries [22].

The Scientific communities also provide manually annotated databases¹ that can be used as language resources or for studies validation. Other initiatives as MIT media Lab Open Mind Common Sense focus on the build of a large knowledge from the contributions of thousands of people across the Web. Basic facts including emotional ones are provided by users (e.g. The sun is very hot). Such sentences are analyzed in order to extract concepts, ontologies or basic positive-negative lists of words (see also Cyc.com) [17, 25]. In other cases, resources manually rated such as movies, products or merchant rating available on customer opinion web sites are also often used (CNET, eBay, TripAdvisor, IMDB). The idea is here to use a machine learning algorithm in order to extract a link between words and the rated comment and predict the opinion of an unrated text [6, 13].

2.2 *Identifying the Polarity of Words*

The automation of the lexicon generation involves the use of heuristics that, in short, provide to the algorithm a part of the human expertise. Thus, the identification of the polarity of words can be done using more or less complex methods. In bag-of-words, terms are considered as independent parts of the speech. Elementary grammatical features of words that are known to have polarity values (adjectives, adverb, negation, intensifier, diminisher) are used to separate them. Adjectives or adverbs are then organized in binary classes or associated to a position in a continuum between the two extreme polarities [21]. But, adjectives are not always the best opinion descriptors. For example, in the sentence, there was a lot of noise in this hotel, all the negative weight resides in the noun noise. If we replace it by the noun facilities, the sentence become positive. This shows that, beyond adjectives or adverbs, the polarity depends on a larger range of terms individually or in association. Unfortunately, bags of words neglect the position of words in the sentence and only take into account their presence or their frequency. Alternatively, as suggested by common sense and experiments, the n-gram technique that uses the parts-of-speech patterns has a better efficiency. A basic example is the following where a negation (no, not) involves a very different polarity depending on its position in the sentence (this book was not good—no wonder, every one love this book). The co-occurrence of words is also a key criterion. In short, it is assumed that words that have the same polarity are often found together or more or less close in a sentence. This relationship between words can be computed following different techniques as LSA (Latent Semantic Analysis), PMI (Pointwise Mutual information), Chi-Squared test of independence [6].

¹TREC (Text Retrieval Conference), NTCIR (NII Text Collection for IR), CLEF (Cross Language Evaluation Forum).

Recent works exploit the research and the analysis of seeds words in an unknown text. Seeds [25] have an evident polarity (well, good, bad, ...) and are used to collect other connoted words. The criterion used to extend these lists can be the level of proximity with the seeds. [11]. A statistic analysis of the collected words helps to refine the lists. This technique needs less expert implication but it requires a good knowledge of the target language and domain. Not only the seeds have to be chosen carefully but in case of domain oriented vocabulary some words may be connoted differently (cold is negative for the evaluation of a restaurant but can be positive in other domains as for describing the quality of a fridge). Actually, the influence of the domain is known as a key issue not only for the opinion mining but also for the knowledge management in general. Consequently, several researches have tackled the sensitivity to the domains or in other words to see how to use a lexicon from one topic to another [4, 11].

3 Opinion and Length of Expression

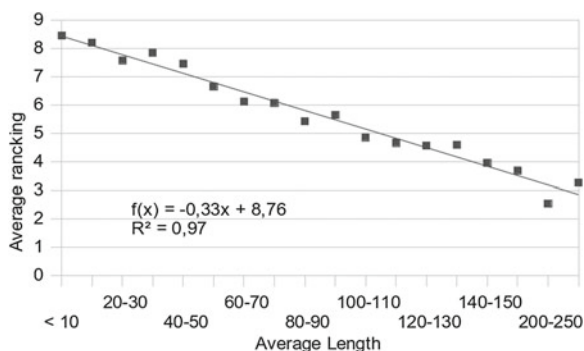
Even if it is clear that taking into account associations of words (n-gram, seeds, ...) provides better performances than a simple bag-of-words method, it is still a question to identify the optimal length of this association [1, 21, 24]. This issue has raised the attention of the community with the spread of micro-blogging platforms, as Twitter, where the size of the message is strongly limited [3, 5, 10].

Nevertheless, from our knowledge, the literature does not provide example of study that takes into account the difference of the expression length in order to statistically separate positive from negative vocabulary and generate lexicons. Though, this psycholinguistic feature of the human expression has already been observed by researchers.

Anderson, for example, has stated that unsatisfied customers engage themselves in greater word-of-mouth than satisfied ones [2]. In the same domain, another study based on reviews of customers shows that positive opinions contain 4 times less words in average than negative ones [16]. In another context, observing the expression of emotions in computer mediated communication as e-mails, Derks sees that in case of negative situations as conflicts, Emoticons are not enough. This leads to more communication between individuals to solve the problem, whereas in positive situations, a simple smiley can be sufficient [9]. It is also surprising to observe that the performance of automated sentiment miners tends to be better within positive texts than within negative ones. This confirms the observation of Gindl [13] and leads to the assumption that positive words are more used in negative opinions than negative words in positive opinions. This would imply that positive opinions are less ambiguous probably due to its conciseness.

In order to confirm the strong link between the polarity of an opinion and the length of its expression, we present a first statistical view involving three different languages. The Figure 1, corresponds to a data set that contains 5014 users' opinions in French collected on the mareduc.com website. This website is convenient because

Fig. 1 Relation between text length and ranking of appreciation



the rating is given on a wide range (from 0 to 10), which gives a better precision. This is pretty rare because, comparatively, most of the ranks range from 1 to 5. Each point in this plot represents the average rating for a class of opinions corresponding to a range of length. For example, the first dot means that the opinions having less than 10 words have an average rank of 8.5 (i.e. very good). The second dot involve that all opinions between 10 and 20 words have an average rank of 8.1, etc.

In order to have an alternative view of the polarity-length relation, we made another measure with 2 other languages having a different graphic representation form (English and Chinese). This measure is not completely comparable with the previous one in French, for several reasons. First, the ranking system is not the same, from 1 to 5 (stars) instead of 0 to 10. Secondly the context is different (hotel instead of High Tech). This is important to consider since it seems reasonable to think that some contexts induce more verbalization than others. We do not test this hypothesis here, we only focus on the polarity-length relation. The set in English is the same as that used for the main experiment (see detailed statistics in the methodology section), the Chinese set is composed of 808 opinions. Table 1 shows the words distribution according to the 2 main polarities (Positive stand for rank 4 and 5, Negative for rank 1 and 2).

Even if the difference between the average length may differ from one language (or one context) to another, we see that the polarity-length relation is consistent. Messages with a positive opinion tend to be shorter than the negative ones. Of course, a larger statistical experiment is needed with more languages, more contexts and a more significant number of opinions. Nevertheless, we have reasonable clues showing that this relation is true whatever the language and whatever the context. This is a psycholinguistic invariant of the human behavior.

Table 1 Average number of words (or ideograms) per opinion

Languages	Positive polarity	Negative polarity
English (words)	98	169
Chinese (ideograms)	177	200

4 Methodology

The core of our proposal consists in to take advantage of this natural relation in order to build a contextualized lexicon that will be used to compute the opinion of an unknown text. In order to validate our approach, we propose two experiments allowing to compare two methods of lexicons generation. The first one uses the polarity-length relation. The second uses the seeds method. Both experiments use the same set of data.

We collected 20,400 users' reviews in English that include comments (68 words average length) and the rating (0–5 stars) from the well known opinion.com web site. This dataset was divided in two parts. The validation set (V) was randomly composed of 1381 texts (96,488 words) from the initial set. The rest of the initial set was used to compose several learning sets (L) in order to evaluate the influence of the size of the learning set. The L subset contains only the comments (i.e. without rating) and is used for the generation of the lexicon. Then we use this lexicon to compute the opinion value of the comments of the V subset (each text independently) and we compare the results with the users rating. In order to estimate the quality of the lexicon generation process we use the recall, precision, and f-index ratios for the positive (Rp,Pp,Fp) and the negative class of opinions (Rn,Pn,Fn).

$$R = \frac{RI \cap Rt}{RI} \quad P = \frac{RI \cap Rt}{Rt} \quad F = 2 \cdot \frac{P \cdot R}{P + R} \quad (1)$$

In these formula, a relevant document (RI) is an opinion text that corresponds correctly to its rating (positive or negative class). A retrieved document (Rt) is a text that has been affected by the process to a specific rating class. Thus, the recall is the percentage of all relevant items identified by the process or in other word, the average probability of complete identification. Symmetrically, the precision is the number of correctly affected items divided by the number of all affected items or in other words, the average probability of relevant affectations. The F-measure as an harmonic mean, balances the precision and recall into an unique indicator. More details and useful links can be found on wikipedia if an introduction is needed on these indicators.

4.1 Polarity-Length Method

At the beginning, the comments of the L set were randomly dispatched into two subsets P0 and N0 that can be viewed as two kinds of bag-of-words with a loss of organization between words. These two sets will be progressively refined through several iterations where new sets (P1, N1 to Pn, Nn) will be generated from the previous ones. At each iteration i, Pi and Ni will be used to generate 2 steps lexicons Lpi, Lni. Let us remark that whereas Pi, Ni aggregate the same number of comments and thus can contain several times the same words, the union of Lpi and Lni sets

contains only one occurrence of a word. At the last iteration, L_{pn} and L_{ln} are expected to contain words with respectively a positive and a negative polarity.

At the second step, the frequency of P_0 and N_0 words are computed in order to generate the L_{p0} and L_{n0} lexicons. Thus, a specific word will be stored in L_{p0} , in L_{n0} or discarded depending on the difference of frequency it has on the two subsets P_0 , N_0 . This operation applied at each iteration allows to eliminate articles or others neutral words that have a similar frequency in all kind of texts. Since connoted words are less frequent, they will be kept in the L_{p0} , L_{n0} lexicons with a higher probability than neutral words (even with a random classification).

Differential frequency algorithm

```

For each unique word : $W_i \in (P_i \cup N_i)$ 

  if ( $\text{Freq}(W_i) \text{ in } P_i$ ) > ( $2 * \text{freq}(W_i) \text{ in } N_i$ )
    then  $W_i$  is stored as unique in  $L_{pi}$ 
      else if ( $\text{Freq}(W_i) \text{ in } N_i$ ) > ( $2 * \text{freq}(W_i) \text{ in } P_i$ )
        then  $W_i$  is stored as unique in  $L_{ni}$ 
          else  $W_i$  is discarded

End for

```

In the third step, the goal is to start the agglomeration of words having the same polarity in the same sets (L_{pi} or L_{ni}). We take again the L subset and, for each comment, we compute its level of polarity (P_{ij}) with the following formula. For example, if a comment is composed of 13 words where 10 appear in L_{p0} and 3 in L_{n0} . The polarity of this comment would be equal to 0.53 (i.e. $10 - 3/10 + 3$).

$$P_{ij} = \frac{\text{Card}(T_j \text{Words} \in L_{pi}) - \text{Card}(T_j \text{Words} \in L_{ni})}{\text{Card}(T_j \text{Words} \in L_{pi}) + \text{Card}(T_j \text{Words} \in L_{ni})} \quad (2)$$

If P_{ij} is ranging between +1 and K (see below), it is assumed to be of positive polarity. Negative, if it is between $-K$ and -1 and neutral if its between $-K$ and K . Then, if P_{ij} is positive, all words of the text T_j , recognized either in L_{p0} or L_{n0} , are stored in P_1 . If P_{ij} is negative, the text words recognized are stored in N_1 . Then the frequency algorithm is processed again but now on P_1 and N_1 in order to generate L_{p1} and L_{n1} . Statistically speaking, each set (L_{p1} , L_{n1}) should be a bit more consistent in term of polarity than L_{p0} and L_{n0} , even if this polarity (positive or negative) is not yet known.

In the fourth step, the number of words in P_1 and N_1 will decide of this polarity. If N_1 has more words than P_1 then L_{n1} is considered as the negative step sub-lexicon and L_{p1} the positive one, else L_{p1} become the negative one (and is renamed L_{n1}) and L_{n1} the positive one.

All the process from the third stage is iterated until L_{pn} , L_{ln} is considered as stable. (experimentally, $N = 20$ iterations was found as enough, see Fig. 4). Then we built the final consensus lexicons on the basis of the words present in the $Z = N/2$ (i.e.

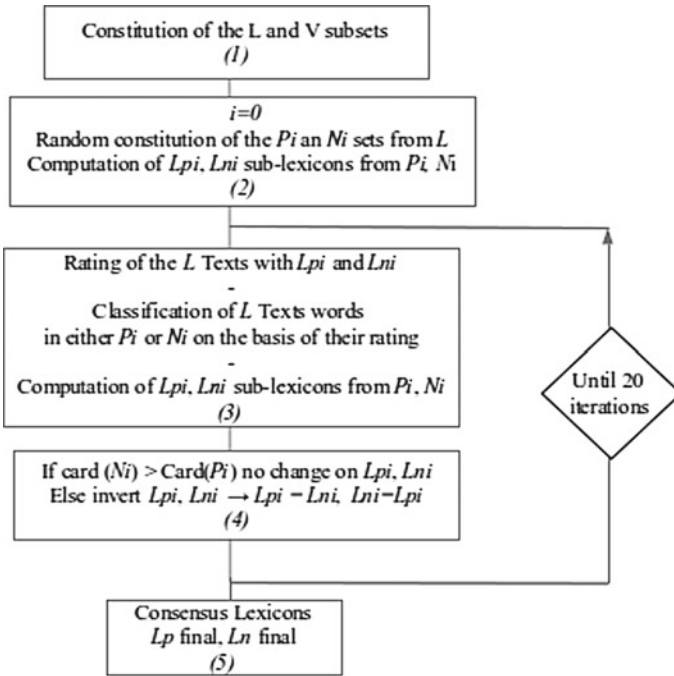


Fig. 2 Polarity-length lexicon processing

10) last step learning lexicons of the same polarity ($lp_{11}, lp_{12}, \dots, lp_{20} \rightarrow \text{final } L_p$; $ln_{11}, ln_{12}, \dots, ln_{20} \rightarrow \text{final } L_n$). Words are kept in the final consensus lexicons if they appear in more than $C = Z/2$ (i.e. 5) of the Z lexicons. The organogram of the Fig. 2 synthesizes the whole processing operations.

The value of the K , Z and C parameters is important. The distance $[-k, +k]$ correspond to the neutral polarity proportion. In order to simplify, we considered that each category (positive, negative and neutral) are proportionally equivalent (i.e. $K = 0.3$). That means that the probability that a text fall in one of these categories is estimated as identical (33%). Actually, this depends on the context and it even seems that, most of the time, negative messages are over represented [16]. K should, also, be different from zero in order to avoid oscillations in the learning process. The N parameter, linked to the need of having a complete learning process, results mainly from the experimental observation. As that can be seen in the Fig. 4, the iterated process stabilizes itself pretty rapidly. Furthermore, it is important to have in mind that since we choose to limit prior knowledge, we have no means, except the use of a heuristic to know when the learning process would be at its optimum. The Z and C parameters as in a vote process define the level of consensus to build the final lexicon.

4.2 Seeds Method

As explained in the state of the art, the seeds method starts from few key words which polarity is known. These words will be used to look for similarities into unknown texts in order to collect new words which polarity will be induced from the nearness with seeds words. All these words (including seeds) will compose the generated lexicon.

In the organogram of the Fig. 3, W_{ij} represents the seeds with $i[1, n]$ the index number and $j[p, n]$ the polarity negative or positive. L_j represents the positive or negative sub lexicon with $j[p, n]$. T_k represents the unknown texts with $k[1, m]$ the index number among texts. P_u with $u[1, q]$ is the phrase that is associated to a text. For example, P4T2 would represent the fourth phrase of the second text.

In the first step (1), the user defines several initial seeds words. Then (2), we collect the texts needed for this learning step. In the iteration loop (3)(6), for all phrases P_u of the texts T_k , if a seed word (of p or n set) appears in the phrase $P_u T_k$, then all the

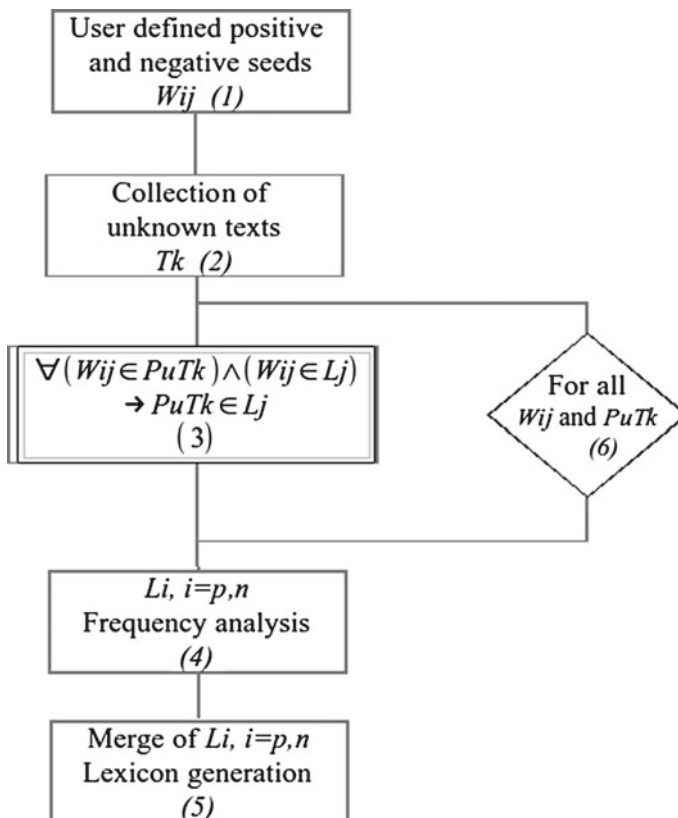


Fig. 3 Seeds method lexicon processing

words of the phrase are stored in the appropriate sub-lexicon (p or n). The frequency of occurrence of each of these words is also stored in the p or n sub-lexicon. At this stage, the L_p and L_n sub-lexicons may have several words in common such as articles or neutral words that can be encountered as well in phrases of positive or negative connotation. The frequency analysis step (4) consists in comparing the frequency of a word appearing in both L_p and L_n in order to decide if it will be discarded or kept in one of the sub-lexicons. This algorithm is the same as that used in the previous section (see differential frequency algorithm). Finally (5), the two sub-lexicons will be merged and ready to use for the opinions computation.

5 Results

In this section, we compare the results of the polarity-length method with those of the seeds approach.

5.1 Experimental Validation of the Polarity-Length Method

In order to have a synthesis of the performances, we define 3 classes of opinions with their rating (negative: 1–2 stars, neutral: 3, positive: 4–5 stars). The estimated opinion was computed from the textual comment in order to fit the same scale (i.e. adapted from the Pij formula). In the ideal situation the user stars rating should correspond to the computed one. The sensitivity of the L size on the performances was evaluated with 8 tested sizes (from 364 to 7053 kB). The performances are reported in the two following tables. The Table 2 presents the ratios with the final consensus lexicons. These results of the full automated process can be compared with that of the Table 3 with the best values during the 20 steps. This comparison shows that most of the time the consensus lexicons give the best results.

We can see that the size of the learning set is not a clear criterion to have good performances (see in Table 2, F-index for 5014, 6124 and 7053 kB). It is important to remind that each L set was composed randomly from the original set. This involves that words are not always the same and can cause different lexicons even if the process was run several times with the same set size. This is the main reason that causes important changes in performances even in the final lexicons. This means that the aggregation process that generate these lexicons does not completely catch the optimum performances but succeed to avoid the lower ones. Nevertheless, as shows the Fig. 4 (with 3 examples of L size), the learning process converge pretty rapidly with stable results in the last steps. Also, we see that a size of learning set L from 4 to 7 MB provides reasonable results. In this figure, we can also observe the oscillations of the F-index near the inversion of the lexicons polarity (fourth learning step, see Fig. 2) generally observed at the third iteration. At this point, the lexicons start to become consistent from the polarity point of view. Before this point of equilibrium,

Table 2 Results with the consensus lexicon

Size (kB)	Pp	Pn	P	Rp	Rn	R	Fp	Fn	F
364	65.1	65.2	65.1	62.3	60.7	61.5	63.7	62.9	63.3
1111	70.3	68.9	69.6	65.9	68.7	67.3	68.0	68.8	68.4
2045	64.6	72.2	67.5	76.2	51.4	63.8	69.9	60.0	65.6
2971	67.8	66.6	67.2	63.5	66.8	65.2	65.6	66.7	66.2
3901	93.2	74.9	81.4	64.3	94.8	79.5	76.1	83.7	80.4
5014	86.8	80.7	83.4	74.9	86.7	80.8	80.4	83.6	82.1
6124	74.7	74.8	74.8	73.5	72.3	72.9	74.1	73.5	73.8
7053	85.8	86.0	85.9	84.8	82.3	83.6	85.3	84.1	84.7

Table 3 The best F-index results

Size (kB)	Pp	Pn	P	Rp	Rn	R	Fp	Fn	F
364	65.1	65.2	65.1	62.3	60.7	61.5	63.7	62.9	63.3
1111	77.6	86.4	81.4	83.9	68.7	76.3	80.7	76.5	78.7
2045	64.6	72.2	67.5	76.2	51.4	63.8	69.9	60.0	65.6
2971	65.9	73.3	68.9	75.8	57.1	66.5	70.5	64.2	67.7
3901	89.1	85.4	87.2	81.4	86.6	84.0	85.1	86.0	85.6
5014	85.7	82.6	84.1	79.2	85.0	82.1	82.4	83.8	83.1
6124	74.3	78.6	76.3	77.1	69.1	73.1	75.6	73.6	74.7
7053	84.9	86.7	85.8	85.0	82.5	83.8	84.9	84.6	84.8

the content of the lexicons is still too random to allow to identify the appropriate polarity.

It is also interesting to analyze the influence of the learning set (L) size on the features of the final lexicon. The Table 4 provides this information for the 8 possible sizes of L. We can observe that the number of words in the final lexicon is not proportional to that of L. The ratio of the number of words of L over that of the final lexicon vary from 0.24 to 1.25 % depending on the size of the learning set.

Finally, in order to have a general view of the performances, we computed the average result with 4 learning sets (average of 7172 reviews) an 4 validation sets (average 716 reviews). We applied the main algorithm (Fig. 2) to each learning set and we measured the results on each validation set. The statistical data corresponding to these 16 tests are reported in the Table 5.

Fig. 4 Convergence of the Learning process

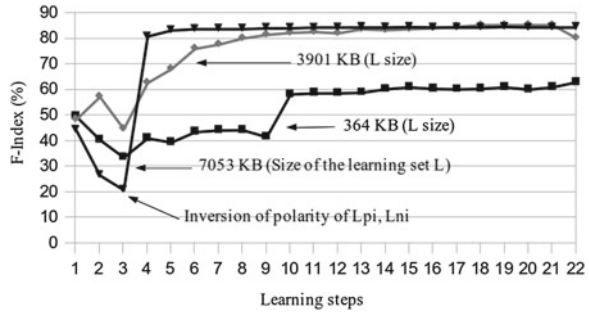


Table 4 Relations between the learning set and the final lexicons features

Size L (kB)	Nb texts in L	Nb words in L	Nb words in Lpn + Lnn
364	980	66,654	773
1111	2980	202,981	1214
2045	5480	373,352	1624
2971	7980	541,472	2067
3901	10,480	710,905	1767
5014	13,480	913,963	2908
6124	16,480	1,116,462	3079
7053	18,980	1,286,107	3224

Table 5 Average results

	F index for 16 tests
Average	81.6
STD	3.8
Max	86.9
Min	73.4

5.2 Seeds Based Method

The use of seeds seems to be the most powerful actual method but the choice of the initial key words (the seeds) may have a strong influence on the final results. In this part of the experiment, we take again the learning set that provide the better results (7053kB) and the validation set and we build the final lexicon on the basis of the seeds method. In order to show the sensitivity to the initial seeds, we use four examples of 6 seeds (3 positives and 3 negatives) and we compute the precision, recall and F ratio. The first set provides equivalent performance compared with our method. In the second and the third set we changed only one world respectively with negative (set 2) and positive polarity (set 3). In the last set, we change several words for each polarity.

Table 6 Recall, precision and F-index using the seeds method

Seed set	Pp	Pn	P	Rp	Rn	R	Fp	Fn	F
1	90	83	86	75	89	82	82	86	84
2	86	73	78	64	85	74	73	79	76
3	83	74	78	63	83	73	72	78	75
4	39	44	42	23	52	38	29	48	40

- Seeds set 1: super, excellent, perfectly, bad, poor, expensive;
- Seeds set 2: super, excellent, perfectly, bad, noisy, expensive;
- Seeds set 3: super, excellent, recommend, bad, poor, expensive;
- Seeds set 4: good, excellent, friendly, poor, noise, bad;

The results show clearly that the seeds method is very sensitive to the choice of the worlds even with a careful attention to the context (here, users' comments on hotels). The case of the last set is very interesting. We can observe that even with words that are evidently consistent in polarity and in context, the performances are very bad. The reason is probably due to the representativity level of the seeds words in the learning set. It is important to say that each of these words is present in the learning set but with a different frequency of occurrence (Table 6).

5.3 Comparison Between Methods

As landmarks, let us remind (see introduction section) that most of the studies obtain more than 80% of accuracy [20]. Moreover, we see that the seeds and polarity-length method have at best, similar performances but with less stable results for seeds method that in addition is more knowledge greedy. It needs more prior knowledge and the process has to be controlled precisely (lexicon generation). Indeed, the seeds method needs either a linguistic and a domain expertise in order to chose the most accurate seeds words or a good learning set in order to statistically compensate the lack of representativity of a specific seed. The lesser stability of this method could be explained by the difficulty to evaluate the added value of this expertise and the effort necessary to gather the proper learning context (seeds, learning set, ...)

6 Discussion

In this paper, we developed several hypothesis related to the psycholinguistic feature of the free individual human expression. First, we present the polarity-length relation that states that the length of the free expression is as long as the opinion polarity is negative. Second, that this feature is true whatever the language and whatever the topic

domain of this expression. Finally, we show that this feature can be used, practically, to enhance an automated process designed to compute the opinion. Even if these hypothesis need more in-depth evidences, we provide steady clues presenting the polarity-length as an invariance of the human behavior. Not only our approach allows a better adaptivity to multiples languages and domains but also a better tolerance to errors, misspellings or approximate expressions (e.g. linguistics shortcuts as in twitter).

Anyway, our methodology has some limitations. Even if we do not need to have a strong knowledge about the collected texts, we need to know that they contain opinions for a majority of them (customer or blog feedbacks, ...). The other limit is that the inconsistency of the sources, in terms of domains, is difficult to be controlled if we want to completely avoid the human interventions. Thus, a complete blind approach could reduces the performances but this can be enough if the goal is a rough classification. Furthermore, as our first goal was to validate the interest of the polarity-length heuristic, we spent low efforts on the question of the syntactic analysis which could be improved. Indeed, our basic bag-of-words strategy can takes benefits from the lot of studies done on this field (n-gram).

In terms of perspectives, outside the improvements that we have just evoked, we wish to evaluate the potential of this approach in several practical applications where the opinion is a key added value.

References

1. Agrawal R, Rajagopalan S, Srikant R, Xu Y (2003) Mining newsgroups using network arising from social behavior. In: 12th international w.w.w conference
2. Anderson EW (1998) Customer satisfaction and word of mouth. *J Serv Res* (1998)
3. Bermingham A, Smeaton AF (2010) Classifying sentiment in microblogs: is brevity an advantage? In: *CIKM '10 proceedings of the 19th ACM international conference on Information and knowledge management*, pp 1833–1836
4. Blitzer J, Dredze M, Pereira F (2007) Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In: *Proceedings of the 45th, annual meeting of the association of computational linguistics*, pp 440–447
5. Bollen J, Mao H, Pepe A (2011) Modeling public mood and emotion: twitter sentiment and socio-economic phenomena. In: *Proceedings of the 5th international AAAI conference weblogs and social media*
6. Cogley J (2010) Sensing sentiment in on-line recommendation texts and ratings. B.A dissertation
7. Dave K, Lawrence S, Pennock DM (2003) Mining the peanut gallery: opinion extraction and semantic classification of product reviews. In: *Proceedings of the w.w.w conference*
8. Denecke K (2009) Are SentiWordNet scores suited for multi-domain sentiment classification? In: *Fourth international conference on digital information management*
9. Derks D, Fischer AH, Bos AE (2008) The role of emotion in computer-mediated communication: a review. *Comput Hum Behav* 24(3):766–785
10. Diakopoulos NA, Shamma DA (2010) Characterizing debate performance via aggregated twitter sentiment. In: *CHI '10 proceedings of the SIGCHI conference on human factors in computing systems*, pp 1195–1198

11. Duthil B, Troussset F, Dray G, Montmain J, Poncelet P (2012) Opinion extraction applied to criteria. *Database and expert systems applications. Lect Notes Comput Sci* 7447(2012):489–496
12. Esuli A, Sebastiani F (2006) Sentiwordnet: a publicly available lexical resource for opinion mining. In: *Proceedings of the 5th conference on language resources and evaluation (LREC)*
13. Gindl S, Liegl J (2008) Evaluation of different sentiment detection methods for polarity classification on web-based reviews. In: *18th European conference on artificial intelligence (ECAI-2008), workshop on computational aspects of affectual and emotional interaction*
14. Hu M, Liu B (2005) Mining and summarizing customer reviews. In: *Proceedings of the conference on human language technology and empirical methods in natural language processing*
15. Kim S-M, Hovy E (2004) Determining the sentiment of opinions. in: *Proceedings of the 20th international conference on computational linguistics*
16. Lancieri L, Lepretre L (2011) Sentiment analysis in social web environments oriented to e-commerce. In: *IADIS web based communities and social media conference (WBC 2011)*
17. Liu H, Lieberman H, Selker T (2003) A model of textual affect sensing using realworld knowledge. In: *Proceedings of the seventh international conference on intelligent user interfaces*, pp 125–132
18. Melville P, Gryc W, Lawrence RD (2009) Sentiment analysis of blogs by combining lexical knowledge with text classification. In: *Proceedings of the conference on knowledge discovery and data mining*
19. Miller GA (1995) WordNet. A lexical database for English. *Commun ACM* 38(11):39–41
20. Mejova Y (2011) Sentiment analysis: an overview, Technical report. University of Iowa, Computer Science Department
21. Pang B, Lee L (2008) Opinion mining and sentiment analysis. *Found Trends Inf Retrieval* 2(1–2):1135
22. Stone PJ, Dunphy DC, Smith MS (1966) *The general inquirer: a computer approach to content analysis*. M.I.T. Press, Oxford, 651 pp
23. Strapparava C, Vlitutti A (2004) Wordnet-affect: and affective extension of wordnet. In: *Proceedings of the 4th international conference on language resources and evaluation*
24. Turney PD, Littman ML (2003) Measuring praise and criticism: inference of semantic orientation from association. *ACM Trans Inf Syst (TOIS)* 21(4):315–346
25. Zhou L, Chaovalit P (2008) Ontology-supported polarity mining. *J Am Soc Inf Sci Technol* 69:98110

Visual Analysis of Topical Evolution in Unstructured Text: Design and Evaluation of TopicFlow

Alison Smith, Sana Malik and Ben Shneiderman

Abstract Topic models are regularly used to provide directed exploration and a high-level overview of a corpus of unstructured text. In many cases, it is important to analyze the evolution of topics over a time range. In this work, we present an application of statistical topic modeling and alignment (binned topic models) to group related documents into automatically generated topics and align the topics across a time range. Additionally, we present TopicFlow, an interactive tool to visualize the evolution of these topics. The tool was developed using an iterative design process based on feedback from expert reviewers. We demonstrate the utility of the tool with a detailed analysis of a corpus of data collected over the period of an academic conference, and demonstrate the effectiveness of this visualization for reasoning about large data by a usability study with 18 participants.

Keywords Statistical topic modeling · Data visualization · Natural language processing, NLP · Topic evolution · TopicFlow

1 Introduction

Statistical topic modeling is a well known technique for discovering the “topics” that occur in a collection of documents. Topic modeling has been used to provide a high-level overview of a corpus as well as directed exploration [14]. Typically, topic modeling is applied as a “batch” process and leads to topics that cover the

A. Smith (✉) · S. Malik · B. Shneiderman
University of Maryland, College Park, MD, USA
e-mail: amsmit@umd.edu

S. Malik
e-mail: maliks@cs.umd.edu

B. Shneiderman
e-mail: ben@cs.umd.edu

A. Smith
Decisive Analytics Corporation, Arlington, VA, USA

© Springer International Publishing Switzerland 2015
P. Kazienko and N. Chawla (eds.), *Applications of Social Media
and Social Network Analysis*, Lecture Notes in Social Networks,
DOI 10.1007/978-3-319-19003-7_9

entire corpus but don't take into account the fact that topics may change over time. Although this is sufficient in some cases, the increasing availability of streaming data has given rise to a number of Use Cases that require an understanding of the evolution of topics. For example, a public relations team would like to monitor how discussion about the company they are representing changes over time; campaign managers may want to understand how their candidate's public perception has changed over time with respect to how they have been represented in the media; a news team would want to identify emerging topics representative of "breaking news"; and researchers may want to track how published work has shifted over time within their respective fields.

Techniques exist for modeling the evolution of topics, but this work does not typically lend itself to identifying emerging topics or modeling the flow of topics as they converge, diverge, and end. Additionally, because these algorithms explicitly incorporate time into the underlying topic modeling algorithm, they are not generalizable to alternative topic modeling implementations (entity topic modeling, hierarchical topic modeling, etc.) and incorporating additional features.

We present binned topic models, an application of statistical topic modeling that is well-suited for studying topic evolution on streaming text data. This technique models complex trends over time, and is particularly suited for discovering emerging topics and following topic divergence and completion. Binned topic models are topic models generated independently for adjacent time slices of text data, so topics generated at one slice do not directly correspond to the topics of another. To align the topics we use the cosine similarity metric. Displaying the results of topic modeling, and in particular topic evolution, is a difficult problem. In this paper, we provide a solution to this problem with our visualization tool, TopicFlow, which visualizes the emergence, convergence, and divergence of complex topics in a data stream.¹

In this paper, we:

1. Describe an analysis technique for text data over adjacent time slices, *binned topic models and alignment*, which is an application of Latent Dirichlet Allocation (LDA) [2] to time-stamped documents at independent time intervals and alignment of the resulting topics,
2. Introduce TopicFlow, an *interactive visualization tool* that aligns similar topics between time slices and displays topics as they emerge, converge, and diverge over a given time period, thereby identifying and providing insights that would otherwise go unnoticed, and
3. Present a multi-part evaluation of TopicFlow that shows its usefulness for following the flow of topics in text.

¹This work is an extension of our prior work [13], in which we originally introduced TopicFlow as a Twitter analysis tool. A video demonstrating this work can be found here: <https://www.youtube.com/watch?v=qql1vMOQaOE\&feature=youtu.be>

2 Related Work

TopicFlow covers two main areas: automatic topic detection by generating topics from a high volume of unstructured text and trend visualization over time.

2.1 Topic Detection

Existing tools follow trends in user-generated web content, however, these either only deal with short phrases [10] or are primarily concerned with locating spikes in activity rather than analyzing the trend throughout the full time range [8].

Latent Dirichlet Allocation (LDA) is an unsupervised algorithm for performing statistical topic modeling that uses a “bag of words” approach, treating each document as a vector of words where order is ignored. Each document is represented as a probability distribution over some topics where each topic is a probability distribution of words. The traditional LDA model does not take into account how topics may change over time.

A few variants of statistical topic modeling exist for incorporating time into the topic model. Topics over Time [22] is a topic model that captures time jointly with word co-occurrence patterns, such that each topic is associated with a continuous distribution of timestamps. In this case, the meaning of a topic remains constant over the time range. Topics over Time performs batch processing, meaning that as new data comes in, the method must re-model the entire data set. Blei and Lafferty [1] presents continuous time dynamic topic models, a dynamic topic model that uses Brownian motion to model latent topics through a sequential collection of documents. Topics are not held constant, and the words that make up the topic may change over time. This technique facilitates evolution analysis of a particular topic over the time range; however, the model fails to represent the emergence of a unique topic within the time range or the convergence or divergence of existing topics.

2.2 Trend Visualization

The primary motivation for TopicFlow is to analyze the evolution of discovered topics for any unstructured text source. In this initial work, we develop a visualization that is representative of topic evolution over a time range.

Two trend visualizations that are closely related to TopicFlow are [4, 6]. ThemeRiver uses a stream graph to visualize thematic variations over time from a large collection of documents. ThemeRiver defines themes as single words, and the strength of a theme is determined by the number of documents containing the word. This definition does not support complex themes that must be defined by more than a single word. TextFlow, shows the evolution of topics over time as well as merging and

splitting. TextFlow uses a semi-supervised clustering technique for topic creation and represents topic convergence and divergence using a flowing “river” metaphor. The river metaphor is visually appealing for a small number of topics, however it quickly becomes cluttered; even at 15. Also TextFlow inhibits access to the underlying data, which limits analysis. The TopicFlow approach involves a general solution which is not limited to analysis of trends over time; unlike these existing trend visualizations, the TopicFlow approach can be adapted to any grouping, for example geographic location, publication, or author.

The TopicFlow visualization was directly inspired by a Sankey diagram [16], which is a type of network graph typically used to represent directional flows of data through a system where the width of the paths are proportional to the flow quantity. TopicFlow uses a generalized version of a Sankey diagram implemented in the Data-Driven Documents library [3], which is a library specifically designed for creating visualizations for large datasets.

3 Binned Topic Models

In this section, we present the application of LDA to a corpus of unstructured text documents *binned* into time slices followed by the alignment of the topics produced for each bin. To begin, the corpus is divided into bins; the number of bins to be used is specified as an input parameter. Each bin represents a time slice of equal length with no restriction on the number of documents it may contain. In future versions a non-parametric modeling approach or an approach based on expected document rate may be more appropriate to determine the bin size and number of appropriate bins. For the underlying Statistical Topic Modeling algorithm, TopicFlow uses an open-source LDA implementation [18]. Standard LDA requires as input the documents and the number of topics² to discover, although algorithms exist to automatically determine an appropriate number of topics based on the data [21]. To produce a binned topic model, LDA is applied independently for the documents of each bin followed by an alignment step, which aligns similar topics between bins.³ The algorithm employs a stop words list to remove common words that do not contribute significant meaning to topic modeling. The TopicFlow stop words list contains standard English, and, additionally, is modified for a given dataset by including query terms used in data collection and stop words specific to the domain.⁴ In later work, we intend to support a dynamic stop words list which will incorporate stop words specified by users, as well as, stop words discovered from the domain [23].

²For TopicFlow, the number of topics is adjustable with a default of 15 to balance granularity and comprehensibility of the resulting topics.

³For this implementation the LDA algorithm runs for 100 iterations with $\alpha = 0.5$ and $\beta = 0.5$.

⁴For example, Twitter-specific stop words include {rt, retweet, etc.} and Spanish stop words include {el, la, tu, etc.}.

The granularity of this modeling approach can be adjusted by varying both the number of topics modeled as well as the size of the bins. Bin size selection depends on the event timescale a user is interested in (e.g. for breaking news, bins on the order of minutes or hours would be preferred; for consumer trends, bins on the order of days or weeks may be more applicable). The number of topics depends both on bin size—larger bins will typically contain more topics—and the level of topical detail users or their analysis requires.

The result of topic modeling is a distribution of words for each topic in the topic model, $P(\text{word}|\text{topic})$, and a distribution of topics for each input document, $P(\text{topic}|\text{doc})$. For our Use Cases, we provide users with the ability to select a topic of interest and see all corresponding documents. To enable this, each document was assigned to the topic resulting in the highest $P(\text{topic}|\text{doc})$. Additionally, in presenting this information to users, we rank the documents by probability, such that documents with higher $P(\text{topic}|\text{doc})$ for the topic are ranked above those with a lower probability. We chose this method because it is a simple and effective way to distribute documents across topics.

3.1 Topic Alignment

The binned topic modeling approach generates an independent topic model for the documents in each bin. Because the topics generated at individual bins do not directly correspond to each other, an alignment step is necessary to associate similar topics in adjacent bins and visualize the topic evolution. Binned topic models result from using cosine similarity to compare each pair of topics from adjacent time slices. Cosine similarity measures the cosine of the angle between two vectors.⁵ This metric is regularly used for the comparison of documents or the cohesion of clusters in text analytics and data mining, respectively [20], and has also been previously used for the comparison of topics produced by LDA [12, 17]. While many metrics exist specifically for measuring the similarity or divergence between two probability distributions [9, 11, 15], small differences in low-probability outcomes may have a relatively large impact on the overall metric. For binned topic models, this is undesirable because the topics produced by LDA are primarily characterized by high probability words and variations in low-probability words may be noisy. By using cosine similarity, the influence of any two corresponding probabilities on the similarity calculation is proportional to the product of those probabilities relative to the products of other pairs, limiting the impact of lower probabilities compared to higher probabilities.

Cosine similarity returns a value between -1 and 1 , where 1 would represent the exact same word distribution for each topic. Although cosine similarity ranges between -1 and 1 , when dealing with probability distributions it must be between 0 and 1 , because there are no negative probabilities. Instead of assigning the one

⁵ $\cos(A, B) = \frac{A \cdot B}{\|A\| \|B\|}$.

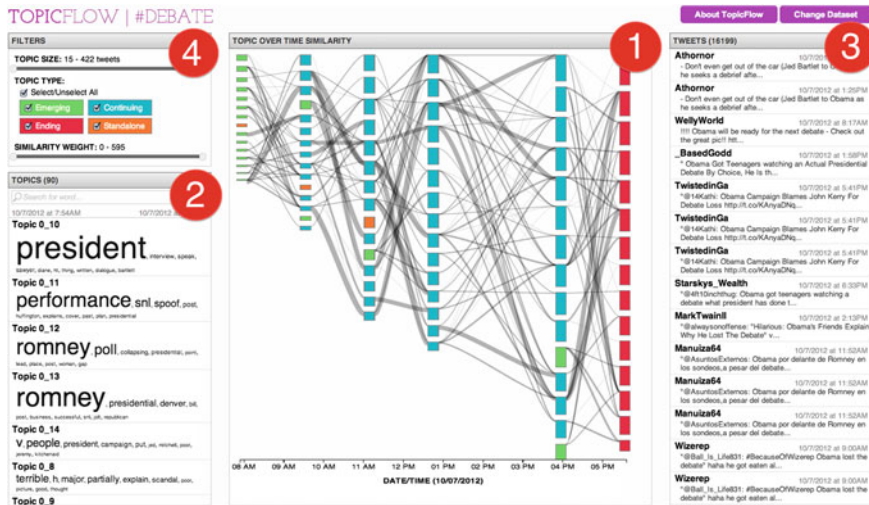


Fig. 1 TopicFlow consists of four coordinated windows: 1 the TopicFlow diagram, 2 a list of topics with their word summaries, 3 a list of the documents (in this case, tweets) in the dataset, and 4 a filter pane

most similar topic at time $n + 1$ for each topic at time n , we present links for any topic pairs with similarity above a certain threshold to enable the visualization of topic convergence and divergence (Fig. 1). The threshold varies with the data set and should be set to balance the discovery of useful topic links with the total number of links displayed.⁶

4 TopicFlow

The purpose of TopicFlow is to allow interactive exploration and analysis of the evolution of topics generated from a corpus of text documents. Figure 2 provides an overview of the TopicFlow system. The system begins by ingesting a corpus for a given time range and splitting the documents into “bins” based on time slices within the range. LDA is then applied independently at each of the bins, producing the corresponding topics. These topics are aligned at neighboring time slices using the cosine similarity metric. The resulting binned topic model is presented to users through an interactive visualization that provides methods for filtering, searching, and performing detailed exploration of the underlying data.

⁶For prototyping and evaluation purposes, the threshold was set between 0.15 and 0.25 depending on the dataset.

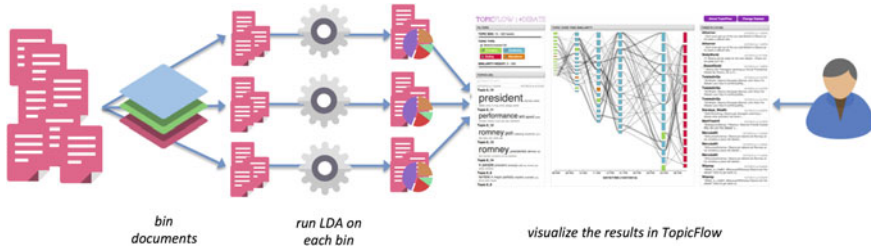


Fig. 2 System overview of the TopicFlow System. The system ingests a corpus for a given time range and splits the documents into time slices, applies LDA at each time slice, and aligns the resulting topics from neighboring time slices. The results are then presented to users through an interactive visualization that includes tools for filtering, searching, and performing detailed exploration of the underlying data through coordinated views

4.1 Design Methodology

TopicFlow⁷ visualizes the evolution of topics of discussion within text corpora, and was designed to support six primary Use Cases:

1. Easily identify the most prominent topics within each time slice. A topic is considered more prominent if there are more documents associated with it.
2. Easily identify which topics are emerging, ending, continuing, or standalone. Here we introduce four new terms:
 - *Emerging*: A topic that was not discussed in the previous time slice. (i.e., there is no topic similar to it in the previous time slice).
 - *Ending*: A topic whose discussion does not continue into the next time slice (i.e., there is no topic similar to it in the next time slice).
 - *Continuing*: A topic that has been discussed before and after its time slice.
 - *Standalone*: A topic which is not related to any topics in either the previous or next time slice.
3. Explore details about a selected topic. These details include its most probable words, assigned documents, and the flow of a topic over time. The *flow* of a topic is defined as the path between a topic and its related topics across all time slices.
4. Identify topics by the words that describe them. A user may be interested in how one or more words appear throughout the dataset. By identifying the topics that are related to these words, a user can understand how the context of a word changes throughout the dataset, as well as discover other words related to it.
5. Compare the top words in two topics that are related. By comparing two topics, a user can identify which words contributed to the topics having a high or low similarity score.

⁷A prototype of the TopicFlow tool is available for demo here: <http://www.cs.umd.edu/~maliks/topicflow/TopicFlow.html>.

6. Filter topics by size, type or similarity weight. Users may want to view only highly similar or highly popular topics, and filtering will allow them to hide the topics in which they are not interested.

The resulting TopicFlow visualization is composed of four coordinated windows (Fig. 1): the flow diagram, topic list, document list, and filter panel, that support detailed analysis of the topic trends and underlying data.

4.2 Flow Diagram

The TopicFlow visualization employs a Sankey diagram [16] implemented in the Data-Driven Documents library [3] for displaying the topic evolution where nodes in the graph represent the topics and the paths between nodes at neighboring time slices represent topic similarity. The paths are weighted by the similarity of the topics as calculated by the cosine similarity metric. This graph is ideal for visualizing convergence and divergence of topics, represented by more than one path entering or exiting a topic, respectively. The color of the nodes is used to distinguish topics by their evolution state: emerging, ending, continuing, or standalone. The nodes are sized by the number of documents attributed to the topic, and they are ordered horizontally from the top by decreasing size. In future work, we intend to provide a number of ordering criteria for users to choose from, such as evolution state or user-specified importance. By ordering based on node size, the most prevalent topics are at the top of the graph, and users can quickly see how the frequency of a topic evolves over time. The design of this diagram was motivated by Use Cases 1 and 2 and is successful in providing insights about the prevalence and life-cycle of the topic.

4.3 Coordinated Panels

In addition to the main data visualization, the TopicFlow tool includes three coordinated panels, a topic panel, document panel, and filter panel that support deeper exploration and filtering of the underlying data.

The topic panel contains a visual representation of the topics discovered for the data. The topics are grouped by their corresponding time slice. The topic in the topic panel can be expanded to gain additional information. The expanded view includes a histogram that represents the distribution of words in the topic.

The document panel contains the underlying documents of the corpus. The individual documents can be expanded to provide users with the full text of the document and a histogram of the five topics with the highest probability for the document, $P(\text{topic}|\text{document})$. In the case of Twitter, users can also follow a link to the author's Twitter page or to view the original Tweet.

Finally, the filter panel, which was designed in support of Use Case 6, includes the following data for filtering the visualization: by node size (number of documents attributed to the topic), node type (emerging, ending, continuing, or alone), and path weights (based on cosine similarity values).

4.4 Interaction

Interaction with a visualization is essential to analysis, because a user must drive the visualization to highlight areas of interest and generate insight with respect to the underlying data. TopicFlow supports the following set of interactive elements.

Topic Search The Topic Panel includes a search functionality for locating topics containing particular keywords. A user can search for a word of interest, and the topic list is filtered to show only topics containing the search term. Additionally, the remaining topics are highlighted in the flow diagram. This functionality supports Use Case 4, by allowing a user to focus analysis on topics containing specific words.

Graph Highlighting When a topic is selected in either the topic panel or the flow diagram, the corresponding node and topic are highlighted. Also, in the flow diagram, the nodes that have a path to the selected topic are highlighted while unconnected topics are greyed out, in order to display the selected node's subgraph (Fig. 3). The selected topic is also expanded in the topic panel. Finally, to support Use Case 3, the document panel is filtered to show the ranked list of documents for the topic.

Topic Comparison When an edge is selected within the flow diagram, a topic comparison box is displayed that uses mirrored histograms to highlight the words the two topics have in common and provide information about the probability of the words for the topics. This supports Use Case 5 (Fig. 4).

Tooltips TopicFlow offers tooltips when hovering over nodes and edges. On nodes, the tooltip displays a “word fade” of the related words, sized and ordered by the words' probabilities. When hovering over edges, the names of the two nodes the edge connects are shown.

Node Filtering The filter pane includes two double-ended range sliders, where users can limit the range of values for the topic sizes (by number of documents) and edge weights (topic similarities). Users can also limit topics by their type—emerging, ending, standalone, or continuing—with checkbox selectors. As nodes and edges are filtered from the graph, the visualization hides topics that become unconnected from the rest of the graph.

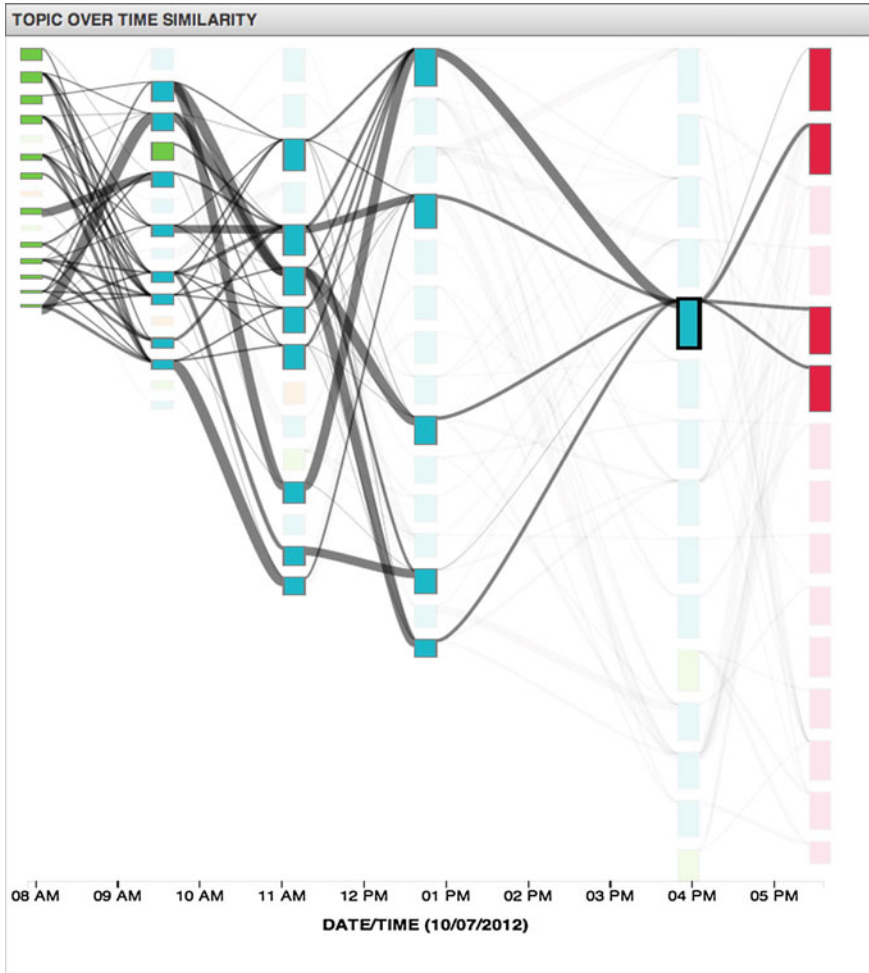


Fig. 3 When a topic is selected, the diagram is highlighted to show the flow of that topic over time

5 Evaluation

TopicFlow was evaluated in three stages: expert reviews with five participants during the design process, case studies performed during development, and one usability study with 18 participants at the end of development. These evaluations primarily used data collected from Twitter⁸ to demonstrate the functionality on streaming unstructured text data.

⁸Twitter's open API and the fact that tweets are rich with metadata, specifically time stamps, makes it an appropriate data source for prototyping and testing.

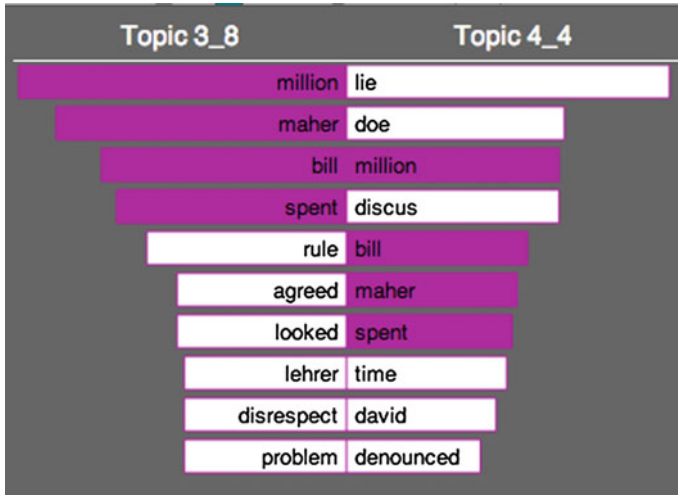


Fig. 4 The topic comparison box shows *bar charts* representing the two topics connected by the edge. The *top words* that the topics have in common are highlighted in *purple*

5.1 Expert Reviews

To drive the design process, expert reviews were conducted over two sessions with five different participants, all of whom had previously performed analysis of text data and had some graduate education or higher. The participants were recruited by email and by word-of-mouth.

The first sessions were conducted with three participants. After a brief introduction and explanation of the tool, we allowed the participants to have a freeform exploration of a data set (approximately 1,500 tweets resulting from a search for the word “earthquake” over 2 days). They were instructed to describe everything that they were doing and why, as well as express any other comments that they might have (think-aloud method). Their comments and our observations (mistakes they made, unreasonable learning curves, bugs, confusing interface actions, missing items, etc.) were documented in handwritten and typed notes, taken by the researchers present during the session. The feedback from these sessions were incorporated into the design of the final tool.

6 Case Studies

During development, TopicFlow was used to analyze a variety of streaming text datasets, including real-time current events (Presidential debates and Hurricane Sandy), communities (University of Maryland), common interests (Modern Family

and Big Data) and other historical data sets (CHI Conference). Each of the datasets contained between 1,500 and 16,000 tweets. We used 7 time bins and 15 topics for each dataset. These values were chosen to balance granularity and accuracy of the topics for the number of tweets and timespan of the datasets. The tweets were collected over varying time spans. A more detailed study was performed for the data gathered about the 2012 CHI (Computer Human Interaction) Conference [19].

2012 CHI Conference TopicFlow was used to analyze 12,000 tweets gathered during the week of the 2012 CHI Conference, which contained the keyword “CHI”. The visualization shows the evolution of topics between 4/26/2012 and 5/12/2012. Figure 5 shows the TopicFlow tool on the CHI Conference data set. A few observations stand out as particularly interesting:

1. There is an emerging topic, { airline, doodle, poll }, prior to the conference, which is associated with a number of tweets discussing a doodle poll that the conference organizers sent out to determine how attendees were planning to commute to the conference.

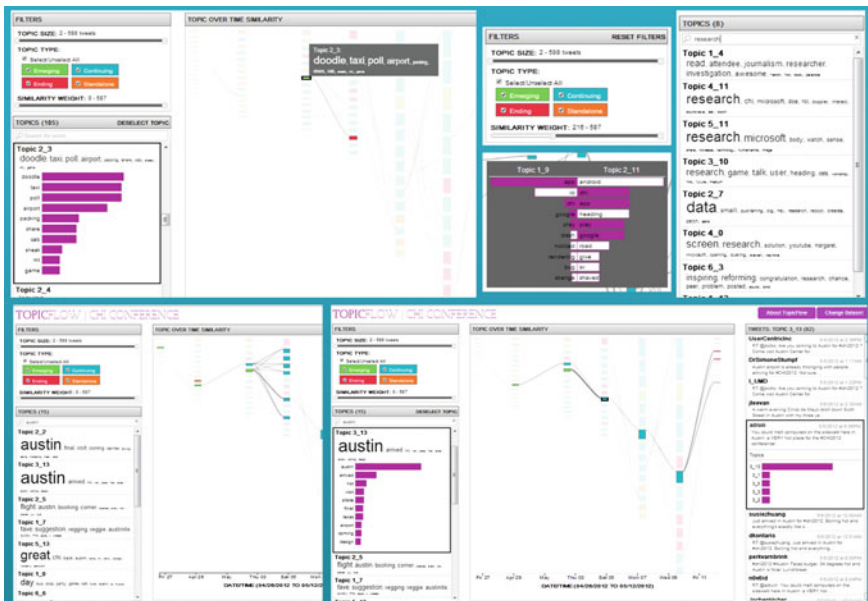


Fig. 5 Results of various interactions with the TopicFlow tool on data gathered about the 2012 CHI Conference. *Top left* Highlighting an { airline, doodle, poll } topic. *Top middle* The filter panel with the minimum similarity weight increased, and a comparison between two topics at consecutive time slices which are about a CHI-related app available on the Google Play store. *Top right* The list of topics resulting from a search for the keyword “research”. *Bottom left* The topics resulting after a search for “Austin”. *Bottom right* Highlighting the { Austin, arriving, heat } topic and a specific tweet associated with the topic

2. After modifying the similarity weight to show only topics that are highly related, a link is prevalent between two topics at consecutive time slices where both are discussing a CHI-related app that at the time was available on the Google Play store.
3. A topic search for “research” shows that topics containing this term are consistent throughout the data set.
4. A topic search for “Austin”, the location of the conference, shows a strong trend of related topics from right before up until the end of the conference. In particular, a topic of people who are discussing “coming to Austin” exists prior to the conference and then shifts to a similar topic, {Austin, arrived, heat} where people are discussing that they have “arrived in Austin” and, in particular, are pointing out the heat.

For the case studies we performed, we found that the binned topic models were most accurate and concise for real-time events which occurred over short time spans. For example, TopicFlow on the corpus of documents related to Hurricane Sandy showed the discussion evolve from emergency preparation to the event itself, then to the aftermath and related recovery efforts. Alternatively, more general data sets, such as University of Maryland, did not have clearly defined or correlated topics due to the high number of diverse, unrelated events that occur on the campus.

6.1 Usability Study

To assess the usability of TopicFlow for exploring text corpora, we conducted a preliminary usability study with 18 participants (8 female), aged 21–49 ($M = 26.5$, $SD = 6.41$). Five of the participants had six to ten years of experience using a computer, and the rest had 11 or more years of experience. Participants were recruited through on campus mailing lists and were compensated \$10 for their time.

The study was performed on a dataset of 16,199 tweets that were collected on October 7, 2012 (4 days after the first 2012 presidential debate) between 8:00 am and 7:30 pm and which contain both the hashtag “#debate” and the word “Obama.” As there is no widely used tool for visualizing and interacting with topics over time, there is no baseline to which to compare TopicFlow. Instead, after a brief introduction to the tool and five training tasks, participants were asked to complete seven tasks that are based on the developed Use Cases.

1. Which topic appears most frequently in the second timeslice and how many tweets are associated with it?
2. What are the top two words for the least frequent topic in the third timeslice?
3. What topic emerges in timeslice 3?
4. Which two topics have the highest similarity index?
5. What is the longest chain of topics connected with weights of 400 or more?
6. Which topic is the word “Romney” *most* relevant to?

7. What is the text of the tweet responsible for the standalone topic in timeslice 3?

The participants then rated each task on a 20-point Likert scale (where a higher score is better) on four metrics based on the NASA Task Load Index [5]: performance, effort, frustration, and effectiveness of the tool. A score over 18.0 was considered to be excellent performance, 15.0–17.9 was considered above average, 12.0–14.9 was average, and a score below 12.0 was considered poor. Each session lasted approximately 30 min. At the end of the session, participants completed a feedback questionnaire and provided comments about the efficacy of TopicFlow’s features.

6.2 Results

The means and standard deviations of 18 participants on time, performance, effort, frustration, and effectiveness (Table 1) vary widely across tasks. Time is measured in seconds, and performance, effort, frustration, and effectiveness were measured on a 20-point Likert scale (higher numbers indicate a more favorable rating).

The results show that the TopicFlow interface allows users to quickly and easily perform tasks which support the initially defined Use Cases. Participants performed the fastest for tasks involving identifying details about topics (Tasks 2, 3, and 6), on average taking 10–20 s. Tasks that involved details about the number of tweets in a topic (Task 1) or evaluating the edges in the graph (Tasks 4 and 5) took longer, about 30–50 s on average. Task 7, which required analyzing the document list for a topic, took participants the longest amount of time to accomplish (81.2 s on average). Many participants commented that they would have found it more helpful if the tool allowed the document list to be re-sorted or if retweets were aggregated and displayed only once.

Table 1 Mean (M) and standard deviation (SD) for time, performance, effort, frustration, and effectiveness for each task

Task Number	Time (sec)		Performance		Effort		Frustration		Effectiveness	
	M	SD	M	SD	M	SD	M	SD	M	SD
1	29.8	29.4	17.4	3.6	16.3	3.6	17.3	3.4	17.5	2.7
2	9.2	4.4	19.2	1.6	18.9	1.7	18.2	4.1	19.4	1.3
3	10.4	11.2	18.2	3.9	18.5	2.0	17.2	4.8	19.1	2.2
4	39.7	25.6	17.8	2.9	15.1	3.8	17.3	4.4	15.9	2.0
5	47.3	29.2	18.0	2.8	13.7	5.3	17.2	3.7	15.9	4.4
6	16.9	18.8	18.0	3.7	18.0	3.3	18.7	1.6	18.6	2.5
7	81.2	48.4	14.6	4.8	11.7	4.8	13.8	4.2	13.3	4.5

Time is measured in seconds, and performance, effort, frustration, and effectiveness were measured on a 20-point Likert scale (higher numbers indicate a more favorable rating)

Task Load Index The Task Load Index ratings reflected the results of the time taken for each task. Tasks 2, 3, and 6 had consistently excellent (above 18.0) ratings for all four metrics, while Tasks 1 and 4 had consistently above average ratings (between 15.1–17.8) on all metrics. Task 5 had excellent ratings for performance (18.0), but required much more effort to achieve this level of performance (13.7). Task 7 was consistently the most difficult, with average ratings for each metric (13.3–14.6).

The feedback questionnaire allowed participants of the usability study to provide qualitative comments about TopicFlow’s features. The participants’ favorite features included the responsiveness of the main visualization to interactions (e.g., hovering and clicking for topic information and subgraph highlighting). One participant stated that these features are “very straightforward” and that the tool “answers questions about dominating themes within trends very well.” Participants also appreciated the tooltips when hovering over nodes and edges. Since standard topic modeling does not provide descriptive names for the resulting topics, the users found it helpful that the visualization displays the top words of a topic, so they could quickly understand the topic’s meaning. Similarly, for the edges of the flow diagram, users appreciated the side-by-side bar charts representing the similarity between topics. One user commented that the coloring of the topics facilitated analysis; for example, using the emerging topic color to “find which topics ‘trigger’ other topics.”

Most of the participants noted that the document list pane was their least favorite feature and requested methods for sorting the documents by various metrics (time, number of retweets, etc.). Because of the lack of quantifiable feedback, participants were often not confident in their answers for Task 7 (which was to identify the most re-tweeted tweet in the document list). In addition, participants felt the filter pane needed improvements—updating the graph by the sliders sometimes had a delayed response or choosing a specific value for a filter was imprecise due to lag in the slider feedback.

7 Future Work and Conclusion

Future work for TopicFlow includes modifying the interface to address feedback received from usability study. Although we use time slices for the purpose of this application, binned topic models is a general technique that can be applied to any data source under any binning criteria, such as geographical location or author. To account for the occasionally confusing results of topic modeling, binned topic models could implement a technique such as Interactive Topic Modeling [7], which allows users to refine the topics generated by the model. While TopicFlow garnered particularly favorable reviews for its interface, there were suggestions regarding the document list pane that can be incorporated into future work. Most notably, users requested a way to sort documents by various metadata such as time or author.

The scalability of the TopicFlow system is dependent on the algorithm for generating binned topic models and the interface. Open-source LDA implementations exist that are scalable to very large datasets [24]. The binning technique partitions

the data to allow multiple LDA runs to be done in parallel, which further increases scalability of the algorithm. The TopicFlow visualization is scalable in terms of the number of documents displayed, as paging is used to handle overflow of data to the interface. In the current version, the screen space provides a limit to the number of topics and bins that can be visualized effectively; however, overview visualization methods could be used to support visualizing thousands of topics or bins.

TopicFlow provides a novel visualization of the alignment of topics over time. Our approach applies the statistical NLP method of topic modeling to text data, which allows for richer analysis of “topics” within the data. When LDA is run over an entire corpus, it produces a high-level overview of the corpus’ content. Alternatively, TopicFlow splits the corpus into a set of time slices and applies LDA on each time slice. This method provides for a more granular set of topics and allows for meaningful exploration of topic emergence, convergence, and divergence. Because topics between time slices are not directly correlated, providing our metric for the similarity between two topics allows users to follow the evolution of the word distributions over time. Our evaluation demonstrated that TopicFlow allows users to easily view the frequency of documents relating to a particular topic over time. TopicFlow further facilitates data exploration by providing details-on-demand about automatically extracted topics through hovering and filtering interactions. The use of colors and tooltips provides users with a quick summary of individual topics.

Acknowledgments We would like to thank Timothy Hawes, Cody Dunne, Marc Smith, Jimmy Lin, Jordan Boyd-Graber, Catherine Plaisant, Peter David, and Jim Nolan for their input throughout the design and implementation of this project and thoughtful reviews of this paper. Additionally, we would like to thank Jianyu (Leo) Li and Panagis Papadatos for their assistance in designing, developing, and evaluating the initial version of the tool.

References

1. Blei DM, Lafferty JD (2006) Dynamic topic models. In: Proceedings of 23rd international conference on machine learning. ACM Press, New York, pp 113–120
2. Blei DM, Ng AY, Jordan MI (2003) Latent Dirichlet allocation. *J Mach Learn Res* 3:993–1022
3. Bostock M (2012) Data driven documents (d3). <http://d3js.org>
4. Cui W, Liu S, Tan L, Shi C, Song Y, Gao Z, Qu H, Tong X (2011) TextFlow: towards better understanding of evolving topics in text. *IEEE Trans Vis Comput Graph* 17(12):2412–2421
5. Hart S, Staveland L (1988) Development of NASA-TLX (Task Load Index): results of empirical and theoretical research. *Hum Mental Workload* 1:139–183
6. Havre S, Hetzler B, Nowell L (2000) ThemeRiver: visualizing theme changes over time. In: Proceedings of IEEE symposium on information visualization, pp 115–123
7. Hu Y, Boyd-Graber J, Satinoff B, Smith A (2013) Interactive topic modeling. *Mach Learn J* 95:423–469
8. Kleinberg J (2003) Bursty and hierarchical structure in streams. *Data Min Knowl Discov* 7:373–397 (2003)
9. Kullback S, Leibler RA (1951) On information and sufficiency. *Ann Math Stat* 22:49–86
10. Leskovec J, Backstrom L, Kleinberg J (2009) Meme-tracking and the dynamics of the news cycle. In: Proceedings of 15th ACM SIGKDD international conference on knowledge discovery and data mining, pp 497–506

11. Lin J (1991) Divergence measures based on the shannon entropy. *IEEE Trans Inf Theory* 37(1):145–151
12. Liu Y, Niculescu-Mizil A, Gryc W (2009) Topic-link LDA: joint models of topic and author community. In: *Proceedings of 26th annual international conference on machine learning*. ACM Press, New York, pp 665–672
13. Malik S, Smith A, Hawes T, Dunne C, Papadatos P, Li J, Shneiderman B (2013) Topicflow: visualizing topic alignment of twitter data over time. In: *The 2013 IEEE/ACM international conference on advances in social networks analysis and mining*
14. Mimno D, McCallum A (2007) Organizing the OCA: learning faceted subjects from a library of digital books. In: *Proceedings of the 7th ACM/IEEE-CS joint conference on digital libraries*. ACM Press, New York, pp 376–385
15. Nikulin M (2001) Hazewinkel, Michiel, encyclopaedia of mathematics : an updated and annotated translation of the Soviet. *Mathematical encyclopaedia*. Reidel Sold and distributed in the U.S.A. and Canada. Kluwer Academic, Boston
16. O'Brien WL (2012) Preliminary investigation of the use of Sankey diagrams to enhance building performance simulation-supported design. In: *Proceedings of 2012 symposium on simulation for architecture and urban design*. Society for Computer Simulation International, San Diego, pp 15:1–15:8
17. Ramage D, Hall D, Nallapati R, Manning CD (2009) Labeled LDA: a supervised topic model for credit attribution in multi-labeled corpora. In: *Proceedings of 2009 conference on empirical methods in natural language processing*, vol 1. Association for Computational Linguistics, New York, pp 248–256
18. Shuyo N (2011) LDA implementation. <https://github.com/shuyo/iir/blob/master/lda/lda.py>
19. Sopan A, Rey P, Butler B, Shneiderman B (2012) Monitoring academic conferences: real-time visualization and retrospective analysis of backchannel conversations. In: *ASE international conference on social informatics*, pp 63–69
20. Tan PN, Steinbach M, Kumar V (2005) *Introduction to data mining*, 1st edn. Addison Wesley, New York
21. Teh YW, Jordan MI, Beal MJ, Blei DM (2006) Hierarchical Dirichlet processes. *J Am Stat Assoc* 101:1566–1581
22. Wang X, McCallum A (2006) Topics over time: a non-markov continuous-time model of topical trends. In: *Proceedings of 12th ACM SIGKDD international conference on knowledge discovery and data mining*, pp 424–433
23. Wilbur WJ, Sirotkin K (1992) The automatic identification of stop words. *J Inf Sci* 18(1):45–55
24. Zhai K, Boyd-Graber J, Asadi N, Alkhouja M (2012) Mr. LDA: a flexible large scale topic modeling package using variational inference in mapreduce. In: *ACM international conference on world wide web*

Explaining Scientific and Technical Emergence Forecasting

James R. Michaelis, Deborah L. McGuinness, Cynthia Chang,
John Erickson, Daniel Hunter and Olga Babko-Malaya

Abstract In decision support systems such as those designed to predict scientific and technical emergence based on analysis of collections of data the presentation of provenance lineage records in the form of a human-readable explanation has been shown to be an effective strategy for assisting users in the interpretation of results. This work focuses on the development of a novel infrastructure for enabling the explanation of hybrid intelligence systems including probabilistic models—in the form of Bayes nets—and the presentation of corresponding evidence. Our design leverages Semantic Web technologies—including a family of ontologies—for representing and explaining emergence forecasting for entity prominence. Our infrastructure design has been driven by two goals: first, to provide technology to support transparency into indicator-based forecasting systems; second, to provide analyst users context-aware mechanisms to drill down into evidence underlying presented indicators. The driving use case for our explanation infrastructure has been a specific analysis system designed to automate the forecasting of trends in science and technology based on collections of published patents and scientific journal articles.

J.R. Michaelis (✉) · D.L. McGuinness · C. Chang · J. Erickson
Rensselaer Polytechnic Institute, 110 8th Street, Troy, NY 12180, USA
e-mail: james.michaelis@gmail.com

D.L. McGuinness
e-mail: dlm@cs.rpi.edu

C. Chang
e-mail: csc@cs.rpi.edu

J. Erickson
e-mail: erickj4@rpi.edu

D. Hunter · O. Babko-Malaya
BAE Systems, 6 New England Executive Park, Burlington, MA 01803, USA
e-mail: daniel.hunter@baesystems.com

O. Babko-Malaya
e-mail: olga.babko-malaya@baesystems.com

1 Introduction

Decision support systems have been successfully used to make predictions about constrained information spaces, based on available evidence [7, 13, 24]. Such systems can often be classified as hybrid intelligence systems leveraged one or more methods from artificial intelligence, including both probabilistic and knowledge base driven reasoning.

Users of decision support systems require mechanisms to help them correctly interpret presented findings [7, 12, 30] as a means of enabling both transparency [10, 29] and trust [1, 27]. The presentation of provenance lineage records in the form of a human-readable explanation has been a common and effective strategy for assisting users in their interpretation of results [13, 17, 18, 22].

When decision support software makes predictions, the kinds of explanation required pose unique challenges, rooted in the difficulties inherent in explaining probabilistic reasoning [8] and end-user interpretation of evidence [14]. Both explanation tasks are viewed as fundamentally hard problems and are considered open research areas [25].

This work focuses on the development of a novel infrastructure for enabling the explanation of probabilistic models—in the form of Bayes nets—and the presentation of corresponding evidence. This design is centered on the usage of Semantic Web technologies—including a family of ontologies—for representing and explaining forecasting of scientific and technical emergence forecasting for specified entities—including inventors, terms, and documents. Included are methodologies for capturing and presenting lineage records for emergence predictions, as well as a collection of evidence visualization strategies, tailored to individual kinds of data to be presented.

Our infrastructure design has been driven by two goals: First, to provide technology to support transparency into systems for forecasting scientific and technical emergence based on identified indicators. Here, examples of indicators include trends such as growth in funding agencies or researchers on a given topic. Second, we aim to provide analyst users context-aware mechanisms to drill down into evidence underlying presented indicators. For trend-based indicators, such evidence could include time series data on researcher participation in a topic. While supporting evidence is related to indicators, it is not directly used to explain indicator values. Rather, this supplemental content provides additional information to aid in evaluating relevance of indicators, as well as their overall importance and accuracy.

Our novel approach relies upon a mixture of static and dynamically generated content. Static content, including text-based descriptions of evidence types, is encoded prior to any system executions. Dynamic information such as indicator values are created during runtime. To facilitate organization and retrieval of both static and dynamic content, an ontology-driven encoding strategy implemented in OWL [19, 21] is utilized. To facilitate retrieval of content, a drilldown approach is used to present details of prominence calculation and indicator content. Based on prior research [17, 22, 28], we anticipate utilization of multiple visualization strategies for diverse data spaces will improve data gathering and analysis. As such, we adopted a drilldown

methodology designed to expose evidence-specific visualization strategies selected at runtime based on content from supporting ontologies.

The driving use case for our explanation infrastructure was an analysis system designed to automate the forecasting of trends in science and technology based on previously published materials. The ARBITER¹ system addresses emergence forecasting in a quantifiable manner through analysis of full-text and metadata from journal article and patent collections [2–5, 20, 26]. In ARBITER, feature data extracted from U.S. patents and scientific journal articles is used to calculate indicators of emergence, which can correspond to trends identified within the extracted data; individual or grouped indicators—termed *models*—form the basis of forecasts about near-term changes in prominence of entities (documents, people and terms) that serve as surrogate representations for broader domains of technology. In ARBITER, usage of indicators to predict prominence requires usage of probabilistic models, designed to make estimates based on established indicator values.

This paper describes the application of our explanation infrastructure toward the task of explaining predictions—henceforth termed *forecasts*—of the future impact of science and technology domains. Here, explanations center around predictions of future impact—*prominence*—for entities representing a domain, such as publications, people and terminology.

Section 2 provides background on relevant sections of ARBITER’s system infrastructure. Section 3 discusses ARBITER’s explanation system design—focusing on both the trace explanation and supplemental content design goals. Section 4 provides an overview of recent evaluation work done on ARBITER’s explanation presentations. Finally, Sect. 5 concludes with a discussion of future work and potential follow-up user studies and overall potential and impact of the work.

2 Background on the ARBITER System

To help define our explanation infrastructure, development was done in the context of a system designed to automate the forecasting of trends in science and technology based on previously published materials. ARBITER addresses emergence forecasting in a quantifiable manner through analysis of full-text and metadata from journal article and patent collections.

In this section, we describe sections of ARBITER’s infrastructure relevant to our explanation design needs. We break this discussion into four parts: (i) a high-level overview of ARBITER’s infrastructure, (ii) how prominence forecasts are managed through probabilistic reasoning, (iii) how evidence and hypotheses are formally expressed through ontological modeling, and (iv) how ARBITER records reasoning traces as lineage records to link hypotheses back to evidence.

¹ARBITER: Abductive Reasoning Based on Indicators and Topics of Emergence.

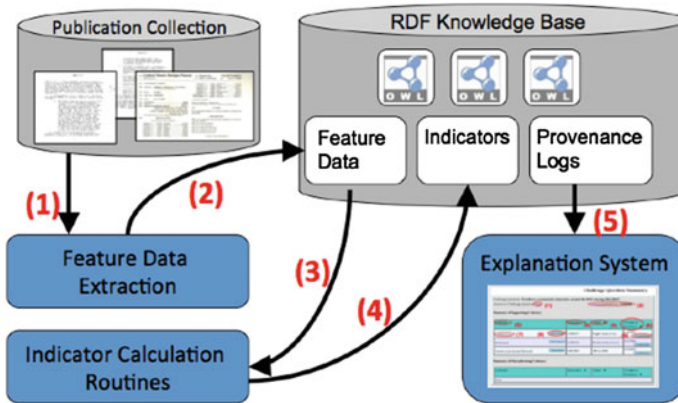


Fig. 1 ARBITER's system architecture

2.1 ARBITER's System Infrastructure

Figure 1 provides a high-level overview of ARBITER's processing routines, carried out in the following sequence:

First, documents are analyzed to obtain feature data, consisting of statistics such as the total number of funding agencies referenced by publication year. Second, feature data is recorded to the RDF knowledge base. Third, feature data is read by a second set of analysis routines to calculate emergence-oriented indicators. Fourth, indicators are recorded back to the RDF knowledge base. Fifth, provenance records that track the indicator derivations are made accessible to the explanation system for analysis.

Once a set of indicators and feature data has been established from the document corpora, ARBITER may then check the prominence of a term/publication person represented within the corpora, based on a supplied query. ARBITER utilizes Bayesian probabilistic models, discussed in Sect. 2.3, to establish prominence scorings through observed indicator values. In turn, to express hypotheses of prominence, evidence for prominence (indicators, features), as well as formal relationships between hypotheses and evidence, an RDF/OWL [21] based encoding strategy is used, discussed in Sect. 2.2.

2.2 Encodings for Evidence and Hypotheses

Content structuring for ARBITER's RDF knowledge base is established using an OWL-based ontology. This ontology serves three purposes: First, to represent hypotheses of prominence. Second, to express traces of reasoning steps applied to establish hypotheses discussed in greater detail in Sect. 2.3. And third, to formally

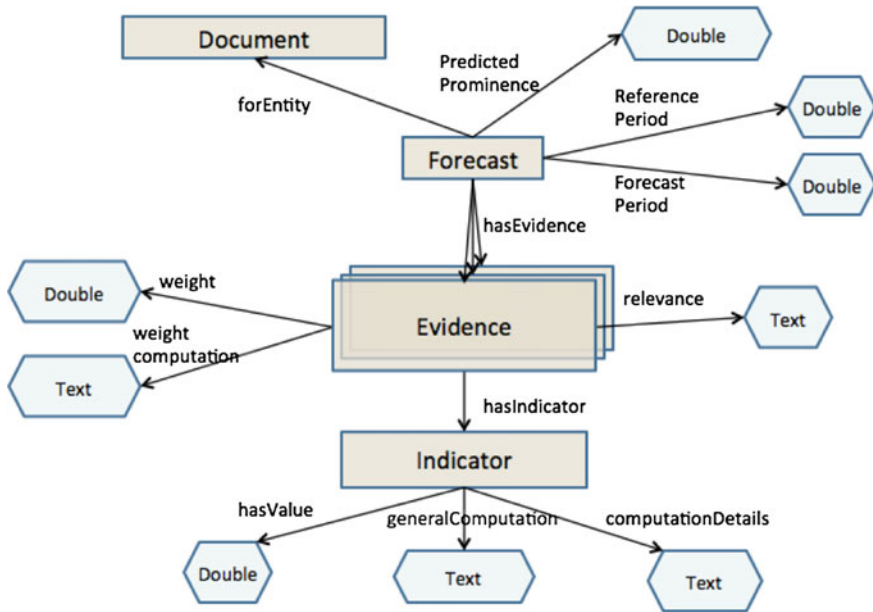


Fig. 2 Key elements of ARBITER’s explanation ontology, designed to express prominence forecasts for documents/people/terms, as well as corresponding evidence for these forecasts

express evidence records based on data identified in publication collections. Combined, these three forms of content are used as input to ARBITER’s explanation component discussed in Sect. 3.

Figure 2 shows the main classes and properties in and their value types. Based on the ARBITER’s ontology, instance data can be defined that corresponds to individual prominence hypotheses, as shown in Fig. 3. Forecast hypotheses are expressed as conclusions about high-level characteristics of a document space, related to estimates of prominence for persons, publications and terms. Instances of the Evidence class encapsulate information about the values of basic indicators or indicator patterns that are relevant to the hypothesis. Each evidence instance corresponds to a unique Indicator and has a generic degree of relevance to the hypothesis as well as a more specific weight in supporting, or disconfirming, the specific hypothesis. In turn, each indicator has a value and is associated with both a generic mode of computation as well as a more detailed computation based on feature values.

In ARBITER, much of the identified evidence centers on publications. Therefore, ARBITER’s ontology imports the Mapekus Publication ontology² [11] for describing documents and adds representations for document features, indicators, and high level network characteristics.

²Mapekus Publication ontology: <http://mapekus.fiit.stuba.sk/?page=ontologies>.

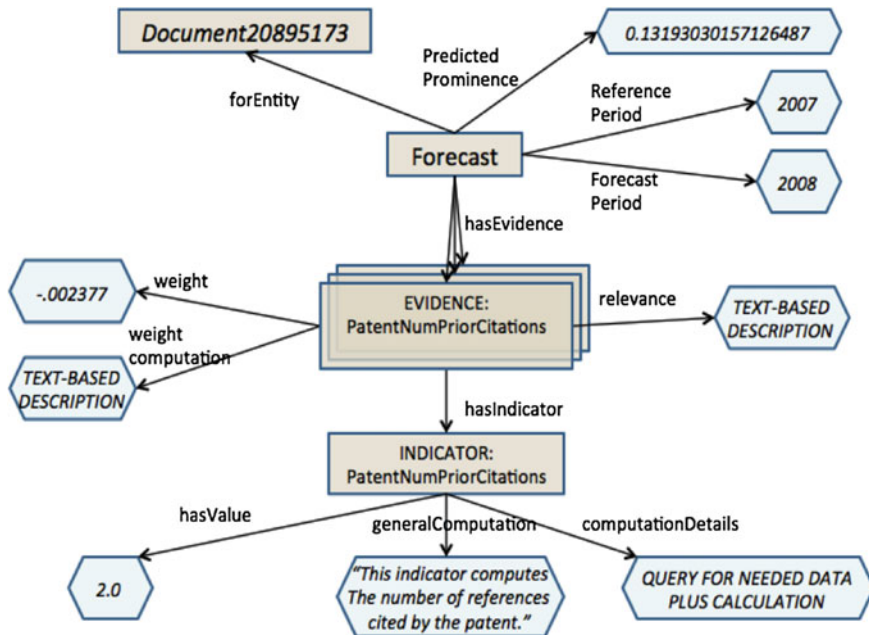


Fig. 3 Example instance data based on ARBITER’s explanation ontology, corresponding to a prominence calculation for a document with ID *Document20895173*

Therefore, ARIBTER’s ontology imports the Mapekus Publication ontology [11] for describing documents and adds representations for document features, indicators, indicator patterns, and high level network characteristics.

2.3 Probabilistic Modeling

In general, explaining probabilistic reasoning is a difficult problem because the detailed calculations leading to a probabilistic result are difficult to interpret. At the same time, explanation of probabilistic reasoning holds the potential for illuminating certain modes of reasoning not easily captured in pure rule-based reasoning.

One approach to overcoming the opaqueness of probability calculations is to impose some structure on the probability model [9, 23, 31]. Bayesian networks are a step in this direction since they encode meaningful relations of dependence or independence between variables. Bayesian networks are directed graphs in which the nodes represent random variables and the links capture probabilistic relations among variables. Nodes in Bayesian networks are typically categorized as observable or unobservable. Observable nodes correspond to variables whose values can be determined by observation or measurement and which serve as evidence. Unobservable nodes represent theoretical states or processes causally linked to observable variables, either as effects or as causes.

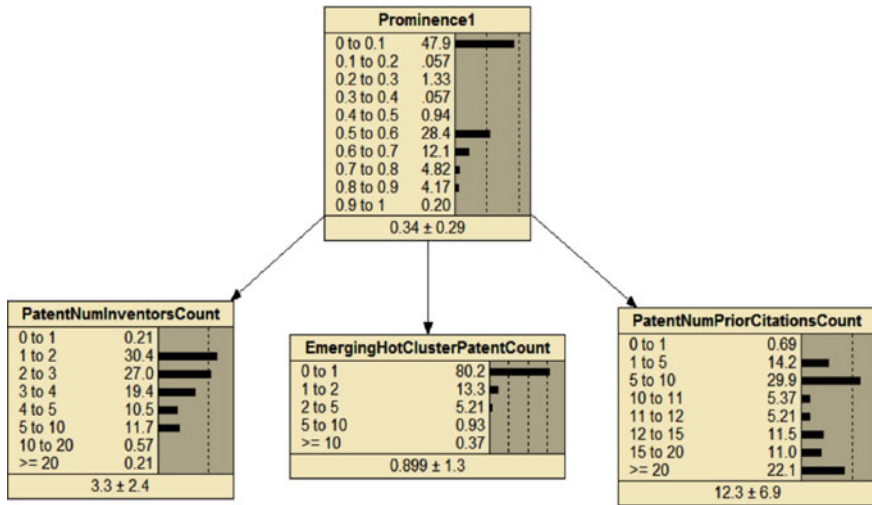


Fig. 4 Naive Bayesian network for predicting future prominence of a patent document

Figure 4 shows an example of a Bayes net model for predicting the future prominence of a patent document. Prominence1, the variable to be predicted, represents the prominence of the document one year into the future. This variable is the root of a tree whose children are variables representing observable indicators of future prominence. The prior distribution for each variable is shown as a horizontal bar graph with the probabilities given as percentages. All variables are continuous but have been discretized for efficient computation. Bayes nets with such a two level tree structure are called Naive Bayes nets.

Let PROM be the prominence variable that is the root of a given Naive Bayes net model and let $IND_1, IND_2, \dots, IND_n$ be the indicator variables for PROM. A Naive Bayes net specifies the conditional probability distribution $p(IND_i | PROM)$ for each indicator variable IND_i . The structure of a Naive Bayes net implies that the child variables are probabilistically independent of one another conditional on the root variable. This implies that we can compute the posterior probability of the prominence variable given values for the indicator variables by the formula:

$$\begin{aligned}
 & p(PROM = x | IND_1 = v_1, \dots, IND_n = v_n) \\
 &= \frac{p(PROM = x) \prod_{i=1}^n p(IND_i = v_i | PROM = x)}{\int_0^1 p(PROM = y) \prod_{i=1}^n p(IND_i = v_i | PROM = y) dy}
 \end{aligned}$$

Here we have assumed that prominence values range continuously from 0 to 1, but the Bayes net will use some discretization of a range that may include negative values. Our Bayesian networks implement efficient computation of the posterior probability for prominence, given indicator values in accordance with the above formula.

The indicator variables in the above model are computable from document features and metadata. *PatentNumInventorsCount* is the number of inventors listed in the

patent (which is metadata); *PatentNumPriorCitationsCount* is the number of citations of the patent by other patents up to the current time. *EmergingHotClusterPatentCount* determines whether the patent is in an emerging cluster. Emerging clusters are a subset of next generation clusters that have characteristics suggesting they are particularly likely to describe emerging technologies.

The variables in the above Bayes net were selected from a large set of patent related variables based on an analysis of their rank correlation with the prominence variable and on sensitivity experiments within a model. Models for other types of entities use up to eight indicator variables. When the target entity is a term, for example, we compute multiple time-series involving the term, e.g., the frequency of occurrence of the term in documents or in particular sections of documents, and use statistics on those time-series, such as average growth, as indicator variables.

Naive Bayes nets are known to perform well as classifiers, i.e. correctly predicting the class to which an entity belongs, but do poorly when measured by the accuracy of their probability estimates. The poor performance on probability estimation is due to the fact that Naive Bayes nets assume the predictor variables are conditionally independent given the target (root) variable and this assumption is typically not correct. In the nets we use for prominence prediction, the target variable is continuous (degree of prominence) rather than categorical as in a classification problem. This raises the question as to why a Naive Bayes net would be appropriate for prominence prediction.

There are two answers to this question. One is that an important sense in which we can predict future prominence of entities is by ranking them in terms of their predicted future prominence. How entities rank in terms of future prominence is arguably a more important question than the precise prominence value assigned. Empirically, Naive Bayes nets do very well at ranking entities in terms of future prominence, despite the fact that the mean squared error between expected posterior prominence and ground truth prominence is large. In tests conducted by an external evaluator, our median rank correlation (Spearman correlation) for both terms and documents was 0.54 (a correlation of 0 would be expected by random guessing). A second reason for the choice of Naive Bayes as a baseline system is that prominence prediction can be turned into a classification problem by choosing a prominence value such that entities with prominence at or above that threshold are deemed to be of interest.

The Naive Bayes nets serve as baselines giving competitive performance on the prediction problems of interest in ARBITER. In more recent work, we have implemented Tree Augmented Naive Bayes models, which remove the assumption of conditional independence among the indicator variables.

2.4 Trace Generation

Hypotheses of entity prominence are evaluated in ARBITER by choosing an appropriate Bayes net model. That is, a model whose hypothesis node corresponds to a selected entity forecasting strategy. Following model selection, evidence—in the form of indicator variables—is entered into that Bayes net. The prominence fore-

casting hypothesis is whichever value of the hypothesis variables has the greater probability after all the evidence is entered.

A record of this reasoning consists of a description of what evidence was entered, which nodes in the Bayes net the entered evidence immediately affected, and what the weight of evidence is for each piece of evidence entered. When a piece of evidence is entered into the Bayes net, in the form of an assignment of a value to an indicator variable, a record is made of the propagation of that evidence through the net. We record the value assigned to the indicator node, the change in the probabilities for the parent pattern variable, and the consequent effect on the probabilities for the hypothesis variable. This information is used to generate a trace of the reasoning in RDF, expressed using the vocabulary ARBITER's explanation ontology.

One crucial difference between our approach to probabilistic explanation and that of similar hierarchical approaches [23, 31] is that we do not confine the explanation elements to information that can be derived from the Bayes net alone. Instead, we supplement the trace of the Bayes net inferencing with English language descriptions of indicators and patterns that provide the user with valuable background information about why they are relevant to the question being answered and why they have the influence the model assumes they do. These description templates along with explanation component relationships are stored in one of the ARBITER ontologies.

3 Explaining Prominence Forecasting

Science and technology analysis systems such as ARBITER have been successfully used to make predictions about constrained information spaces based on available evidence [24, 30]. In these systems, the presentation of lineage records in the form of a human-readable explanation has been a common and effective strategy for assisting users in their interpretation of results [16].

Our infrastructure design has been driven by two goals: utilize lineage records to support transparency into emergence forecasting by exposing both indicator values and probabilistic reasoning steps applied, and develop strategies for providing context-aware drilldowns into evidence underlying presented indicators

In this section we present an explanation strategy based on OWL-encoded data to achieve these goals. Our objective is to expose the steps applied through probabilistic reasoning to reach end results. Across our design, a mixture of content types is required: static content, based on defined routines for calculating and interpreting evidence, and dynamic content, based on individual system runs.

3.1 Trace-Based Explanation

For trace-based explanation, information is extracted directly from lineage records and directly presented to an end-user. For the ARBITER system, a custom-tailored user interface was utilized—designed to present information from records of the form given in Figs. 2 and 3.

Patent Forecast

Patent: 7111146 - Method and system for providing hardware support for memory protection and virtual memory address translation for a virtual machine

Reference Period: 2007 (1) Forecast Period: 2008 (2) Predicted Prominence: 0.131930 (3)

Applicants Assignees Citations

Evidence (5)	(6)	Value (6)	(7)	Weight of Evidence
Number of prior art references (8)		2.000000		-0.002377
Qualification for Emerging Cluster		0.000000		-0.016758

(4)

Fig. 5 Opening screen for ARBITER’s explanation module. Here, a general example is provided, which displays forms of content accessible for forecasts of patents, persons and terms

Figure 5 displays the interface for ARBITERs explanation module. An analyst user will start their explanation session at a screen that provides both a prominence scoring for a given entity (in this case, “Patent: 7111146—Method and system for providing hardware support for memory protection and virtual memory address translation for a virtual machine”), and evidence applied to calculate this scoring. Specifically, the following content is made accessible:

1. **Reference Period:** Denotes the most recent year in which documents from ARBITER’s corpus are considered.
2. **Forecast Period:** The year the prominence prediction is made for.
3. **Predicted Prominence:** A scoring of the prominence for a given patent, term or organization.
4. **Evidence Table:** A table of all indicators considered by ARBITER in making the prominence prediction.
5. **Evidence:** This column of the evidence table lists names of indicators considered.
6. **Value:** This column lists indicator values calculated.
7. **Weight of Evidence:** Weight of evidence is based on the amount the probability of the conclusion increased or decreased upon finding out the value of a given indicator. Positive weights of evidence indicate that the evidence raises our prior expectation of prominence while negative values indicate that the evidence lowers our expectation of prominence.
8. **Evidence Entry:** These correspond to individual indicators applied as evidence.

The items listed correspond to dynamic content generated by ARBITER during runtime.

3.2 Drilling Down into Indicator Evidence

To facilitate retrieval of supplemental content for lineage records, a drilldown approach is used to present details of prominence calculation and indicator content. Based on prior research [17, 22], we anticipate utilization of multiple explanation strategies for diverse data spaces will improve data gathering and analysis. Two drill-down strategies—both incorporating information supplementing lineage records—were considered for presenting evidence records: In the first, the data used to calculate individual evidence values was exposed through evidence-specific visualization strategies; in the second, evidence values and supporting data across information spaces were compared.

Figure 6 provides an example drilldown visualization for an indicator tracking growth in the number of organizations represented in a document corpus. Here, our system is able to determine an appropriate visualization strategy, using a bar chart based on the OWL class of the indicator present in ARBITER’s ontology. The data used to generate the bar chart is retrieved from ARBITER’s dynamically generated knowledge base at runtime. In addition to providing visualization strategies, ARBITER’s ontology provides written statements about indicators, such as general descriptions and their rationale for usage. This information can be retrieved on-demand from an indicator’s drilldown page, as shown in Fig. 7.

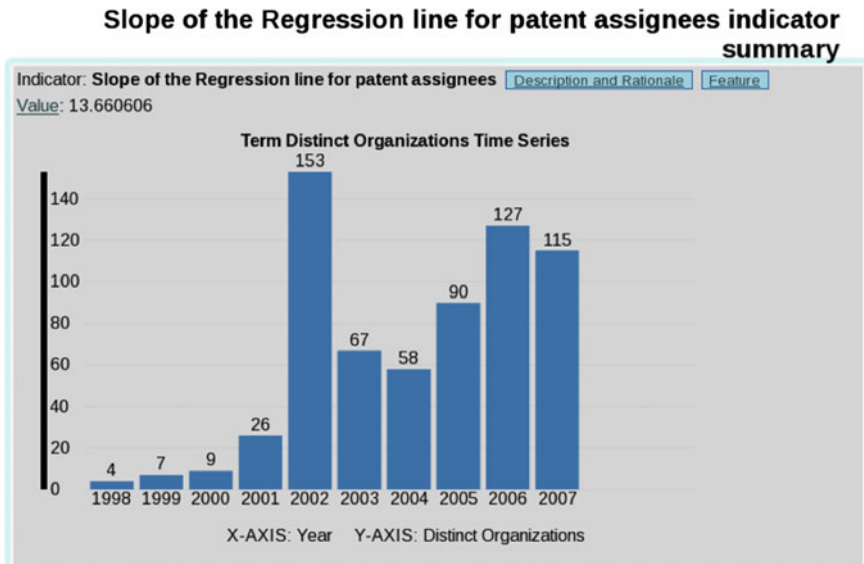


Fig. 6 An indicator drilldown visualization, providing a time series for distinct organizations working in the area of the patent from Fig. 5

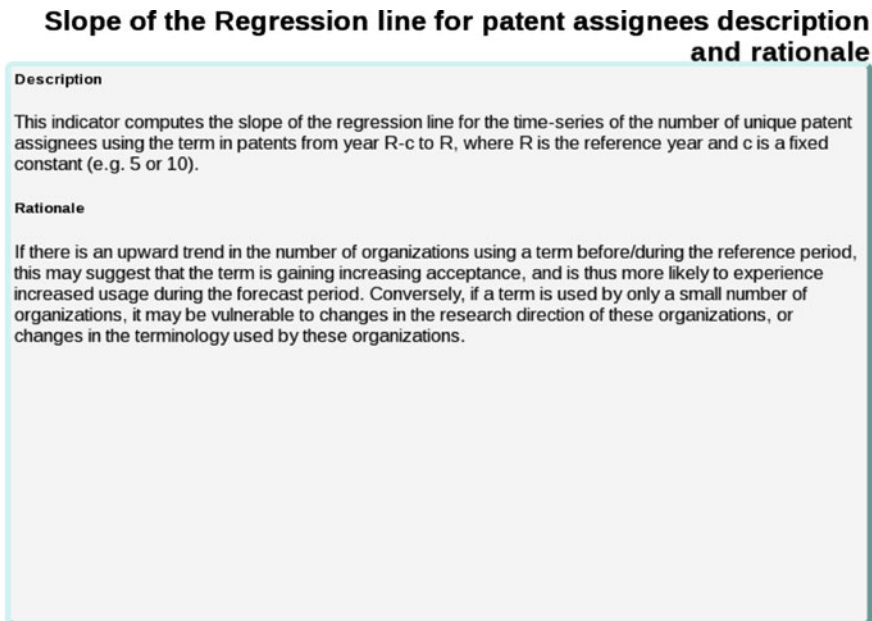


Fig. 7 Statements giving a description and rationale for the indicator from Fig. 6

4 Evaluation

Our explanation infrastructure was evaluated in the context of end-user interactions with ARBITER. The original design for ARBITER's explanation module assumed that users would have minimal knowledge of the both the processing infrastructure and scientific/technical domains under review. Under these conditions, ARBITER would be expected to make explanations available for prominence forecasts and corresponding evidence, as well as supplemental content to facilitate evidence interpretation.

This section presents the protocol and results for an evaluation of ARBITER's explanation module, centered on a user study focusing specifically on review of prominence hypotheses and corresponding evidence. Our evaluation was based upon a set of metrics provided by an independent contractor, listed in Fig. 8. These metrics focused on evaluating the clarity of a set of presented prominence hypotheses, as well as the clarity of corresponding indicators and their supporting data.

Our evaluation was conducted by participants recruited jointly by RPI and BAE Systems across two sessions, each involving three subjects. For each session, participants were assigned a common set of four explanation records supplied by the independent contractor; two gauging the prominence of particular terms, and two for gauging prominence of documents.

1	Entity Forecast Clarity
1.1	The entity forecast answer is clearly expressed.
1.2	The approach used to determine the entity forecast answer is clearly expressed.
2	Indicator Clarity
2.1	Indicator descriptions are clearly expressed.
2.2	Indicator values are clearly expressed.
2.3	The approach used to determine indicator values is clearly expressed.
2.4	The relevance of indicators is clear.
3	Supporting Data Clarity
3.1	Supporting data is clearly expressed.
3.2	The relevance of supporting data is clearly expressed.

Fig. 8 ARBITER’s explanation evaluation metrics

Metric	Page 1-1	Page 1-2	Page 1-3	Page 1-4	Average
1.1	4	4	4.33	4.33	4.16
1.2	4	4	4	4	4
2.1	4.33	4.33	4.66	4.66	4.5
2.2	3.66	3.33	4.66	4.66	4.0825
2.3	4.33	4.33	4.66	4.66	4.5
2.4	4.66	4.66	5	4.33	4.66
3.1	4.33	4.33	4.66	4.66	4.5
3.2	4	4	5	5	4.5

Fig. 9 Metric scorings from Session 1 of evaluation

The participants were asked to review each record individually, i.e., one at a time, and assign the explanation a scoring on a scale of 1–5 for each of the eight metrics. Here, a score of 1 would mean the metric was not met at all, and a score of 5 would mean the metric was completely met. To aid in interpretation of metric scorings, participants were additionally requested to provide statements of rationale—one or two sentences each—for each of their provided scorings.

In general, most of the assigned metrics received high scores from participants. However, in cases where scores were lower, e.g., an average below 4.5, the statements of rationale provided by participants give additional insight. In Session 1, some participants noted difficulty in interpreting hypothesis values (Metric 1.1) as well as how they were calculated (Metric 1.2). Additionally, for Metric 2.2 in Session 1, some participants noted difficulty in interpreting indicator values. Likewise, for Metric 2.3 in Session 2, other participants mentioned being unclear on how presented data was used to calculate indicator values. These statements appear to indicate an assumption in our design of background knowledge which several participants appeared to lack. Given the complexity of ARBITER’s infrastructure and calculation

Metric	Page 2-1	Page 2-2	Page 2-3	Page 2-4	Average
1.1	4.33	4.33	4.33	4	4.25
1.2	4.66	4.33	4.33	4.33	4.4125
2.1	5	5	5	4.66	4.915
2.2	4.33	4.33	4.66	4.66	4.5
2.3	4.33	3.66	4.33	4	4.0825
2.4	4.66	4.66	5	4.66	4.75
3.1	5	3.33	5	3.66	4.25
3.2	4.66	3.66	5	4.66	4.5

Fig. 10 Metric scorings from Session 2 of evaluation

routines, determining how best to address these statements is a nontrivial task. However, for individuals with higher levels of background knowledge in system-relevant areas, i.e., probabilistic modeling, it is foreseeable that refinements could be made to ARBITER's explanation design to meet their needs.

5 Discussion and Conclusion

Based on the application of work conducted within the context of ARBITER, we have produced a novel explanation infrastructure that can be generalized to other analysis and decision support software, especially to systems for indicator-based emergence forecasting and tracking in document collections. Our contribution includes methodologies for applying Semantic Web technologies—including a family of ontologies—towards representing and explaining emergence forecasting for entity prominence. Included are techniques for capturing and presenting lineage records for emergence predictions, as well as a collection of evidence visualization strategies, tailored to individual kinds of data to be presented.

Evidence obtained through the user centered evaluation in Sect. 4 suggests that our interface design will be acceptable for future users of ARBITER with minimal knowledge of its processing infrastructure and with limited knowledge of the scientific and technical domains under review. Comments collected from our test users during evaluation highlighted additional opportunities for improvement of our explanation presentation approach. The impact of additional supplemental content on user comprehension of explanation requires further investigation. In particular, an assessment is needed on how the presentation of value ranges for indicators can impact user understanding of how indicators compare across related topics and years. Additionally, the impact of multiple visualization strategies on user comprehension of individual indicators, i.e., variation in content presentation modalities and data exposed, should be investigated.

In future work, as a means of facilitating interpretation of indicator values, we would like to allow for users to compare both indicator values and supporting data for these indicators across ARBITER's information spaces. We expect to accomplish this by using technologies based on existing OLAP approaches [6], including pivot-based interaction [15].

Acknowledgments Support has been provided by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior National Business Center contract number D11PC20154. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.

Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoI/NBC, or the U.S. Government.

References

1. Artz D, Gil Y (2007) A survey of trust in computer science and the semantic web. *Web Semant Sci Serv Agents World Wide Web* 5(2):58–71
2. Babko-Malaya O, Hunter D, Amis G, Meyers A, Thomas P, Pustejovsky J, Verhagen M (2013) Characterizing communities of practice in emerging science and technology fields. In: 2013 international conference on social intelligence and technology (SOCIETY). IEEE, 2013, pp 37–46
3. Babko-Malaya O, Hunter D, Seidel A, Barlos F (2013) Flexible creation of indicators of scientific and technological emergence. In: Proceedings of 14th international society of scientometrics and informatics conference (ISSI 2013)
4. Babko-Malaya O, Pustejovsky J, Verhagen M, Meyers A (2013) Modeling debate within a scientific community. In: 2013 international conference on social intelligence and technology (SOCIETY). IEEE, 2013, pp 57–63
5. Brock DC, Babko-Malaya O, Pustejovsky J, Thomas P, Stromsten S, Barlos F (2012) Applied actant-network theory: toward the automated detection of technoscientific emergence from full-text publications and patents. In: AAAI fall symposium on social networks and social contagion
6. Chaudhuri S, Dayal U (1997) An overview of data warehousing and OLAP technology. *ACM Sigmod Rec* 26(1):65–74
7. Cowell A, McGuinness DL, Varley C, Thurman D (2006) Knowledge-worker requirements for next generation query answering and explanation systems. In: Proceedings of the workshop on intelligent user interfaces for intelligence analysis, international conference on intelligent user interfaces
8. Druzdzel MJ (1996) Explanation in probabilistic systems: is it feasible? Will it work. In: Proceedings of the fifth international workshop on intelligent information systems (WIS-96). Citeseer
9. Fenton NE, Neil MD (2012) Risk assessment and decision analysis with Bayesian networks. CRC Press, London
10. Frew J, Metzger D, Slaughter P (2008) Automatic capture and reconstruction of computational provenance. *Concurrency Comput. Pract. Experience* 20(5):485–496
11. Frivolt G, Suchal J, Veselý R, Vojtek P, Vozár O, Bieliková M (2008) Creation, population and preprocessing of experimental data sets for evaluation of applications for the semantic web. In: SOFSEM 2008: theory and practice of computer science. Springer, pp 684–695
12. Heuer RJ Jr (1999) Psychology of intelligence analysis. Center for the Study of Intelligence, Central Intelligence Agency, Washington

13. Jackson P (1998) Introduction to expert systems. Addison-Wesley Longman Publishing Co., Inc, Boston
14. Keim D, Andrienko G, Fekete J-D, Görg C, Kohlhammer J, Melançon G (2008) Visual analytics: definition, process, and challenges. Springer, Berlin
15. Larson B (2006) Delivering business intelligence with Microsoft SQL Server 2005. Osborne/McGraw-Hill, New York
16. Leake DB (1991) Goal-based explanation evaluation. *Cogn Sci* 15(4):509–545
17. McGuinness DL, Ding L, Glass A, Chang C, Zeng H, Furtado V (2006) Explanation interfaces for the semantic web: issues and models. In: Proceedings of the 3rd international semantic web user interaction workshop (SWUI'06)
18. McGuinness DL, Pinheiro da Silva P (2004) Explaining answers from the semantic web: the inference web approach. *Web Semant Sci Serv Agents World Wide Web* 1(4):397–413
19. McGuinness DL, Van Harmelen F et al (2004) OWL web ontology language overview. *W3C Recommendation* 10(2004–03):10
20. Michaelis JR, McGuinness DL, Chang C, Hunter D, Babko-Malaya O (2013) Towards explanation of scientific and technological emergence. In: 2013 IEEE/ACM international conference on advances in social networks analysis and mining, Niagara Falls, ON, Canada
21. Motik B, Patel-Schneider PF, Parsia B, Bock C, Fokoue A, Haase P, Hoekstra R, Horrocks I, Ruttenberg A, Sattler U, Smith M (2009) OWL 2 web ontology language: structural specification and functional-style syntax. *W3C Recommendation* 27:17
22. Murdock JW, McGuinness DL, Pinheiro da Silva P, Welty C, Ferrucci D (2006) Explaining conclusions from diverse knowledge sources. In: The semantic web-ISWC. Springer, pp 861–872
23. Šutovský P, Cooper GF (2008) Hierarchical explanation of inference in Bayesian networks that represent a population of independent agents. In: Proceedings of 18th European conference on artificial intelligence, 21–25 July 2008, Patras, Greece. *Prestigious Applications of Intelligent Systems (PAIS 2008)*, vol 178. IOS Press, p 214
24. Swartout W, Paris C, Moore J (1991) Explanations in knowledge systems: design for explainable expert systems. *IEEE Expert* 6(3):58–64
25. Thomas JJ, Cook KA (2005) Illuminating the path: the research and development agenda for visual analytics. IEEE Computer Society Press, Los Alamitos
26. Thomas P, Babko-Malaya O, Hunter D, Meyers A, Verhagen M (2013) Identifying emerging research fields with practical applications via analysis of scientific and technical documents. In: Proceedings of 14th international society of scientometrics and informatics conference (ISSI 2013)
27. Tintarev N, Masthoff J (2007) A survey of explanations in recommender systems. In: 2007 IEEE 23rd international conference on data engineering workshop. IEEE, pp 801–810
28. Viegas FB, Wattenberg M, Van Ham F, Kriss J, McKeon M (2007) Manyeyes: a site for visualization at internet scale. *IEEE Trans Vis Comput Graph* 13(6):1121–1128
29. Weitzner DJ, Abelson H, Berners-Lee T, Feigenbaum J, Hendler J, Sussman GJ (2008) Information accountability. *Commun ACM* 51(6):82–87
30. Welty C, Murdock JW, Pinheiro da Silva P, McGuinness DL, Ferrucci D, Fikes R (2005) Tracking information extraction from intelligence documents. In: Proceedings of the 2005 international conference on intelligence analysis (IA 2005). Citeseer, pp 2–6
31. Yap G-E, Tan A-H, Pang H-H (2008) Explaining inferences in Bayesian networks. *Appl Intell* 29(3):263–278

Combining Social, Audiovisual and Experiment Content for Enhanced Cultural Experiences

Kleopatra Konstanteli, Athanasios Voulodimos, Georgios Palaiokrassas, Konstantinos Psychas, Simon Crowle, David Salama Osborne, Efstathia Chatzi and Theodora Varvarigou

Abstract The rapid penetration of social media in many aspects of today's personal and professional life has created an environment where users within this emerging social cyberspace expect the kind of media experiences they are accustomed to in their daily lives with the community type of communications being at the central stage. In that context social networks and multimedia can be combined and instrumented by monitoring mechanisms to offer new experiences in many areas, among which the cultural and educational sector has a special position. This paper presents an innovative experimental system that enhances the experience of visitors in cultural centers. Apart from offering a modern, immersive, richer experience to the visitors,

K. Konstanteli · A. Voulodimos · G. Palaiokrassas (✉) · K. Psychas · T. Varvarigou
National Technical University of Athens, Athens, Greece
e-mail: geopal@mail.ntua.gr

K. Konstanteli
e-mail: kkonst@mail.ntua.gr

A. Voulodimos
e-mail: thanosv@mail.ntua.gr

K. Psychas
e-mail: kpsychas@mail.ntua.gr

T. Varvarigou
e-mail: dora@telecom.ntua.gr

S. Crowle
IT Innovation Centre, University of Southampton, Southampton, UK
e-mail: sgc@it-innovation.soton.ac.uk

D.S. Osborne
Atos, Spain
e-mail: david.salama@atos.net

E. Chatzi
Foundation of the Hellenic World, Athens, Greece
e-mail: echatzi@fhw.gr

the proposed system involves significant benefits for the organizers as well, supplying them with valuable feedback extracted from the monitoring and analysis mechanisms. The experimental system has been successfully deployed and used in the Foundation of the Hellenic World cultural center.

Keywords Social media · Multimedia content production and delivery · Quality of experience, QoE

1 Introduction

It is a fact that social media have become a part of our lives. They consist of interactive forms of media allowing users to express and share their creativity, gather and share information and experiences concerning every aspect of our everyday lives. Mobile and online users extensively use them in work, education and even leisure. Mobile devices enable them to log into social networks (SNs) during leisure activities such as attending concerts and sports events or watching TV.

Therefore the users' role is more proactive and interactive. They can discover and access additional information, related to the content they are watching in their "first screen" (e.g. TV), using their second screen devices such as smartphones, tablets or laptops. This recent phenomenon, as Google reports in a recent study [5], is known as "second screen" and offers a more personalized and enhanced experience since users can express their likes or dislikes, ask any related questions that might come up, draw their friends' attention to a snapshot they missed, etc. It is implied that users seem to enjoy undertaking leisure activities and simultaneously interact through SNs as this apparently contributes to a richer and more fun experience. In this way content is published once but could be reused several times by different applications, channels and in different formats providing the ability to capture and analyze information, extract metadata and even track people's emotions in their comments.

Furthermore, social-media marketing is an excellent way for businesses to implement their marketing strategies [6]. Its effective usage offers businesses great opportunities to broaden their exposure to the public, form a bond of trust with customers and reach a global audience. Businesses through SNs have the ability to target advertising to precise groups of users applying demographic criteria, or criteria concerning users' interests, allowing users to get to know the business more intimately and increasing the chances for a more favorably response. Besides the possibility to broaden their exposure to the public and market their products and services, businesses are able to find contacts via professional groups and improve their customer service by establishing more intimate conversations with customers based on the knowledge acquired from the SNs.

The ability to receive feedback is one of the most important features of SNs. Stakeholders (producers, performers, event organizers, etc.) gain valuable information and come to conclusions using the feedback circulating in the social media as response to a show, an exhibition or an event, as it is spontaneous, candid and abundant. As

corroborating evidence, Bluefin Labs, a social TV analytics company that specializes in mining and analyzing of SNs conversations, has been recently acquired by Twitter for nearly \$100 million [10]. Therefore, the integration of social media in leisure or even educational activities can produce significant additional value not only for the end-user but for the producer/organizer as well. Although there is much information to be gained by analyzing this feedback, this procedure is very complex as it requires advanced mining and social network analysis tools as well as large-scale data analysis. However these technologies have not yet reached the desired level of progress in order to be effectively used.

In addition the audience's experience could be revamped in several aspects, by the use of SNs. Throughout this paper we will focus on educational cultural centers, such as museums, galleries, special exhibitions. Museums offer a great cultural and learning experience to their visitors. However, taking into consideration that people tend to visit museums in groups and have shared experiences, there is a social aspect in a museum visit that should not be underestimated. This visit could become even more interesting among young people who use SNs as a main mean of socializing and are very familiar with high-end technology and applications.

Another direction in which social networks could modernize and enhance cultural experience, besides peer interaction, is that of dynamic information. When only static exhibitions are available, the derived information regardless of the rarity and importance of the exhibits cannot always be considered as exciting for the general public. The use of museum guides, who have some sort of expertise on the exhibits seemed as a good solution to overcome this issue. In spite of their ability to provide information in a more lively and intuitive manner to visitors, typically they are not true experts but rather individuals who have studied a predefined presentation script and have little ability to operate outside of it. The optimal museum visit would require various true experts as tour guides because a truly intuitive and free navigation, which would allow the visitor to wonder and receive information of any type and on any topic conceivable, would require the involvement of multiple experts in the guidance of each individual group. However, even if they are available, hiring true experts as museum guides is not an option and even more having multiple experts for each group is simply unrealistic. Their knowledge and expertise is too valuable and expensive for them to be on standby at the museum in case they are needed.

Another effective approach is the use of streaming technology which would provide multiple benefits both for the venue and its visitors. It is an ideal tool when there is a need to synchronously transmit heavy streams of information to multiple remote locations. It would also enable the monitoring of the on-going visits and when it is required, the various experts employed by the cultural and educational center could step in to provide specialized information. As a result visitors would have a more enhanced and personalized experience as they would be given more control over the direction of their visit and the venue would be able to provide less structured and therefore considerably more engaging and stimulating experiences.

To this direction, a distributed multimedia social-aware system is introduced in this paper. This system brings together the museum educator with a panel of experts that may be geographically dispersed. This way the audience members interact with each

other as well as with the panel of experts and are engaged in a social media experience. During the experiment, data are exchanged among the panel of experts and the visitors via the SNs. These data are being collected and automatically processed in the background giving valuable information and used to calculate metrics of interest.

In this context, the work presented in this paper has a threefold contribution: (1) the design and real-world deployment of a system that combines heterogeneous technologies, such as video streaming, virtual reality and social networking, to offer a unique enhanced interactive cultural and educational experience to the visitor, (2) the leveraging of social networking on the side of the producers/organizers to collect and process live valuable, comprehensive and accurate feedback which can significantly help them improve the offered experience, and (3) a first attempt on conducting large-scale experimentation on a unique technological facility in an endeavour to (re-)define and progress the concept of the Future Media Internet.

This paper is organized as follows. A description of the system's building blocks is given in Sect. 2, whereas Sect. 3 presents in detail the experimental setup that was deployed in the real-world cultural center settings of the Foundation of the Hellenic World. In Sect. 4 we discuss the results obtained, and, finally, conclusions are drawn in Sect. 5

2 System Components

2.1 Social Network Integration

In order for an application to interact with SNs it has to use the interface that they expose. For this purpose we developed the SocialIntegrator, which provides an easy mechanism to build social applications. Basically, it offers an API that enables user authentication and sharing updates through different SNs while hiding all the intricacies of generating signatures and tokens, doing security handshakes, etc. It extends the functionality of the SocialAuth Core and Android Java Libraries,¹ by customizing them to support the needs of SN related experiments, i.e. by enabling users to join social events, post messages on custom group walls, and send direct messages. The extended functionality is implemented for Facebook and Twitter. This functionality is used by two different applications in the experiment: the visitors' mobile application and the expert's web-application, as explained in detail in Sect. 3.4.

In addition, the SocialIntegrator includes a sub-part responsible for retrieving data of interest from the SNs during the experiment and using them to calculate Quality of Experience (QoE) metrics of interest to the experimenter. This part of the SocialIntegrator is a stand-alone Java application that uses credentials of SNs' accounts with sufficient permissions and access to targeted social activity.

¹<http://code.google.com/p/socialauth>.

2.2 Audio-Visual Content Production and Delivery

Audio Visual Content Component (AVCC) has as main objective to provide high quality support to every part of the experimental application that requires multimedia content handling. In order to achieve this goal, the AVCC consists of several components, which work in a collaborative way to provide a set of services. These services include content acquisition from a media producer, adaptation and distribution of the content to different platforms, live edition and realisation, etc. Another significant feature of AVCC, is its ability to synchronize data and metadata before distribution, which is scalable, since the connection to content delivery networks (CDNs) is permitted (Fig. 1).

Furthermore, AVCC similarly to the SocialIntegrator records several quality of service (QoS) metrics from the experts' interfaces and its streaming server. These multimedia related quality of service metrics include fps, video res. and I/O Bytes Rate per Stream.

AVC component architecture consists of the following components:

- **Input Manager:** This module manages the reception of all live content including audio, video and metadata from the live metadata acquisition management. This module can receive as input video and audio from multiple formats, providing as output a decapsulated stream.
- **Device Adaptation Manager:** Adapts the media content to different platforms without changing the audio or video codecs. The content is fragmented and encapsulated for different devices and platforms.
- **Multiquality manager:** The main objective of this module is to perform transcoding in order to provide multi-quality support, proving the same feed at different bitrates but aligning the GoPs (groups of pictures) in such a way that the player is able to continuously adapt to network capabilities. Besides, it coordinates the multi-quality content generation.
- **Media Distribution:** This module is in charge of the actual content delivery, which includes the continuous generation of the manifest, final packaging of the content and transport protocols. It also produces all multiplexes media output of the main distribution.
- **Timeshift Manager:** Continuously records live streams for immediate playback on deferred. It allows the user to have DVR experiences such as rewind, pause or fast forward. The content is received from Device Adaptation Manager and it flows both to the AV repository and the Media Distribution Module.
- **VoD Manager:** This module delivers pre-recorded video and audio streams to the Input Manager from a single set of source files. It also can record a live stream to a file available for later playing it on-demand, allowing to record an entire live stream into a single file, segment it into multiple files for chapter replay, or start and stop recording at predetermined points for partial archiving. The content is temporally stored in MP4 container including video, audio and metadata.

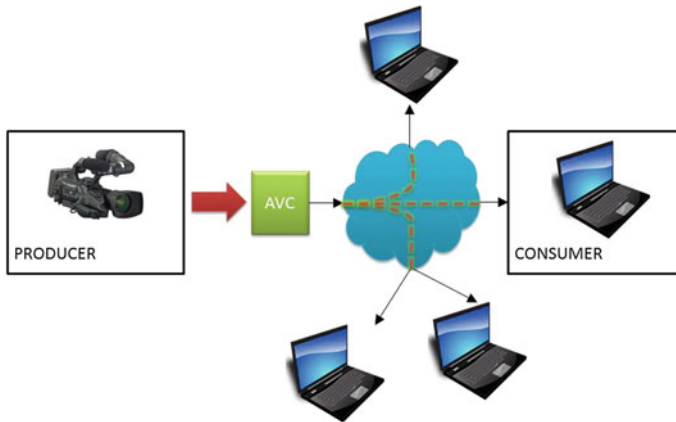


Fig. 1 Simple usage scenario for the audiovisual content component

2.3 Experiment Content Monitoring

The Experiment Content Component (ECC) manages the monitoring of the QoE and QoS metrics generated by the SocialIntegrator and the AVCC. The ECC queries the SocialIntegrator and the AVCC for metrics at the start of the experiment and gathers that data at run-time periodically. It is designed in a generic way, to allow the collection of monitoring data from different sources without needing in advance the details of these sources' design and operation. Within the ECC, the collected data is stored according to a metric data model into a PostgreSQL² database. The ECC also includes a web-based GUI that the experimenters use to get a graphical view on the collected data.

The ECC consists of five interacting sub-components:

- Experiment Specification (ES)
- Experiment Security Correctness (ESC)
- Experiment Deployment and Configuration (EDC)
- Experiment Monitoring (EM)
- Experiment Data Management (EDM)

An overview of the composition of these sub-components, along with their (routed) external exposure to clients of the ECC is presented in Fig. 2. The roles of each sub-component are summarized in Table 1.

Inter and intra-component communications (AMQP bus)

The primary channel for experiment data delivery to the ECC is based on the Advanced Message Queuing Protocol (AMQP)³ since it offers a relatively low technical dependency level and at the same time provides a well-defined, accessible

²<http://www.postgresql.org/>.

³<http://www.amqp.org/specification/0-9-1/amqp-org-download>.

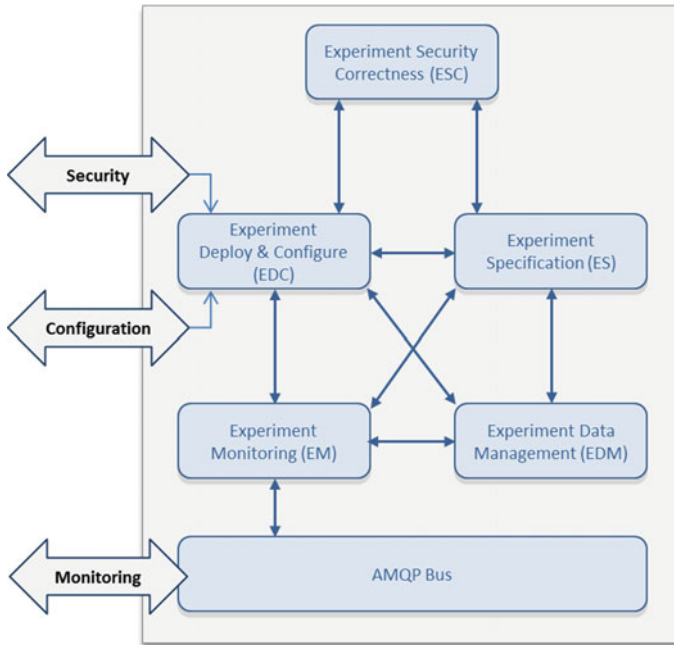


Fig. 2 The ECC architecture

and robust communication method. The ECC component will use the RabbitMQ⁴ AMQP implementation which offers developers a wide range of server and client implementations.

An overview of the interactions between each of these ECC sub-components is provided below.

EDC/ES interactions

The interactions between the EDC and ES are predominantly for the purpose of ‘boot-strapping’ the technical resources required for an experiment to take place. The experimenter will use the ES specification files to describe the location of resources (software installation locations) and their intended targets (cloud based compute and storage/networked hardware), as well as specifying the configuration elements for each of these. The EDC will then handle the execution of deployment, configuration, and tear-down procedures based on this specification and report the results.

EDC/ESC interactions

The deployment and specification process will interact with the security sub-component; the EDC will provide the security models that are associated with experiment technology resources it manages to the ESC. Subsequently, these security

⁴<http://www.rabbitmq.com/>.

Table 1 Roles of ECC sub-components

Sub-component	Role
ES	This component encapsulates the specification of resources required for the deployment of the ECC on a target platform (by the EDC) and any additional resources required by integrating components (such as the connection end-point to the EM)
EDC	The principal role of this component is the automatic (or semi-automatic) deployment and configuration of experimental resources specified by the ES. The EDC manages the boot-strapping process of resource installation and subsequent configuration in collaboration with the experimenter
ESC	Components connected to the ECC have associated security models that the ESC is able to check and evaluate security threats for a particular experimental specification. Experimenters are able to explore a variety of attack scenarios using these models
EM	Monitoring of both the experimental resource status and the experimental metrics generated by those resources is managed by the EM. It queries components deployed by the EDC for metric generating capabilities and gathers that data at run-time
EDM	The EDM manages the access and persistence of metric data collected by the EM. Storage of this data is allocated on a per-experiment basis and metrics can be accessed within a time-frame for subsequent analysis

models can be viewed by the experimenter using the ESC component and evaluated for potential threats (see below).

ESC/ES interactions

The ES component encapsulates a collection of security models generated during the experiment deployment and configuration process; these models can be checked by the experimenter using the security correctness tool. Additionally, the experimenter will be able to explore various ‘attack scenarios’ based on the experimental set-up he has specified using the ESC component.

ES/EDM interactions

Internally, the ES component supports the deployment and configuration process associated with the EDM by specifying the data persistence resources (such as a Postgres database service) that can be used to store experimental data (the installation source/target information of which must be provided by the experimenter).

EM/EDC interactions

The EM component deploys a number of monitoring views for the experimenter to review during the course of an experiment. Some of these views (for example, the experiment resource view) may be implemented using a third party system (such as

NAGIOS⁵). To this end, the EDC is used to deploy and configure these monitoring systems such that the EM can direct monitoring data traffic to them.

EDC/EDM interactions

As described above (see ES/EDM interactions), the EDC plays an important role in the deployment and configuration of data management systems that will allow the EDM to store and provide access to experimental (metric) data. During experiment initialization, the EDM is configured by the EDC to use the database resources specified by the experimenter.

ES/EM interactions

The relationship between the ES and EM is principally concerned with the specification of the technical systems that will be used during the monitoring run-time of the experiment. In a similar pattern to that described above, the ES specifies the EM installation source/target information required for its deployment, including the provision of the AMQP bus service.

3 The Experiment

An experimental setup of the system was deployed at Hellenic Cosmos,⁶ a cultural and educational center in Athens privately owned by the Foundation of the Hellenic World (FHW). A series of exhibitions, both traditional and technological, are presented at the premises of Hellenic Cosmos. The most prominent among those is “Tholos”, the real-time virtual reality (VR) dome theater of FHW. Tholos utilizes a fully digital projection system, configurable in a monoscopic, stereoscopic or a mixed mode of operation. Six pairs of seamlessly blended SXGA+ projectors project the synthesized imagery on a tilted hemispherical reflective surface of 13 m in diameter. The auditorium is designed to host up to 128 visitors at the same time [4]. The in-house developed engine used for creating the VR applications is described in detail in [9].

The Tholos VR system is operated by a single user, namely the educator, via a joystick and manipulator tracker combination. The typical operation of Tholos, as demonstrated in Fig. 3, may be modeled mainly as a one-way communication system, as the educator controls the system, thus determining what the Tholos system will render and project to the visitors, while at the same time commenting on it. As a sole exception to this, visitors can participate in specific electronic polls (through buttons embedded on the seats), which determine the path that the educator will follow, altering in this way the flow of the presentation in real time. The main reason for this extremely structured and predefined approach is that the educator is working with predefined scripts prepared by specialized experts (mainly archaeologists, historians and architects), since the cost of hiring real experts as museum guides is forbidding.

⁵<http://www.nagios.org/>.

⁶<http://www.hellenic-cosmos.gr>.

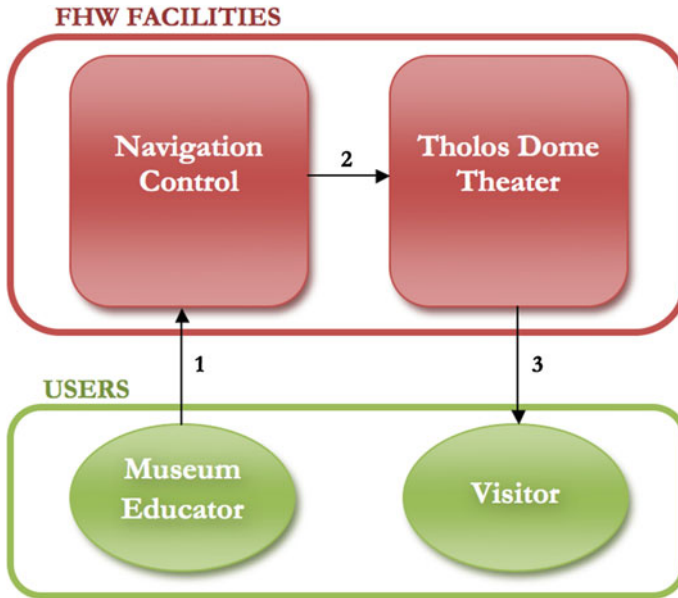


Fig. 3 Tholos operation model before the deployment of the proposed system

These scripts provide information in a specific order and therefore the tour in the VR world has to follow the same order, as the educators have little ability to operate outside it. During the movie, the visitors are not allowed to exchange comments with one another or communicate any questions to the educator since it would disrupt the movie flow and interfere with the experience of the other visitors. Even if they were though, the educator may as well not be in the position to provide sufficiently insightful answers due to lack of in-depth expertise.

3.1 Improving the Quality of Experience

The aim of this experimental system is to enhance the experience of the Tholos visitors by cross-fertilizing the traditional VR Tholos installations with social media and multimedia streaming, so as to provide a shared cultural and educational experience that brings visitors in touch with each other, the educator, as well as other remote individuals with expertise on the field and motivate them to become actively involved and dive deeper into the subject. This is achieved by deploying the distributed application that builds on top of the technologies offered by the toolkit described in Sect. 2

In contrast to the conventional Tholos experience, the deployment of the described application over the FHW facility, as shown in Fig. 4, enables the collaborative pre-

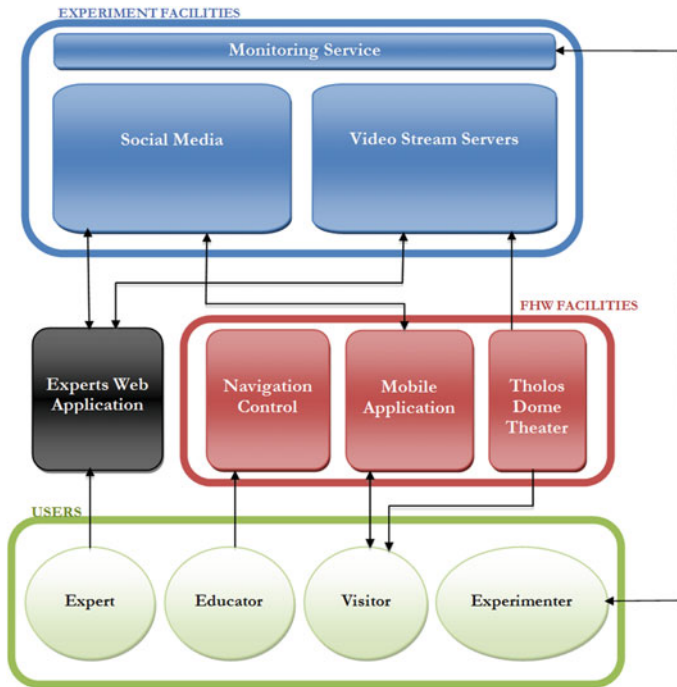


Fig. 4 Tholos operation model after the deployment of the proposed experimental system

sensation of the VR content by allowing a panel of remote experts, who may be geographically dispersed, to “join” the educator, monitor and assist in the presentation by providing specialized information and elaborating on the displayed content. The remote experts use a dedicated web-application, namely the Expert Application (EA). The EA gives them the ability to virtually attend the presented show by utilizing the technology of video streaming. The real-time rendered content of Tholos is captured, multiplexed with the audio from the educator’s microphone and streamed through external streaming servers to the web application. In this way experts have an exact view of the real-time presentation at Tholos. Furthermore the EA allows the experts to publish video and audio from their side.

While their video is only available to the educator’s control pc, their audio is directly passed to the Tholos installation, enabling immediate communication with the educator and the visitors.

In addition to the educator, visitors are also given the ability to interact with the experts through the use of social media by using the Visitor Application (VA). Their experience thus becomes shared and interactive. Visitors can comment and pose questions during the show at specific points. The experts can respond in their turn through the EA, which collects and presents in a structured way the feed from the audience. The Tholos operational model is thus extended through the proposed

experimental system. The following subsections elaborate on the key aspects behind the design and the implementation of this system.

3.2 Extending the Operation of the Tholos Theatre

In order to relay the Tholos view to several remote experts, an extra slave node was added that rendered a slightly bigger frame of the central channel of projection to the experts. Given that the main activity is always at the center of the projection, while the experts do not see the exact same image as the audience, it is sufficient for perceiving the content. The video from the extra node was fed to another computer (STREAM-PC) with a video capture card (AverMedia HD). The STREAM-PC also received the audio from the educators' microphone from the line-in interface of the sound card. From that point on, the rendered view was encoded in H.264 format and multiplexed with the ambient Tholos audio, encoded in AAC and encapsulated in a RTMP stream, which was streamed into the AVCC streaming server (see Sect. 2.2). A visualization of the extended Tholos operation model with the proposed system is shown in Fig. 5.

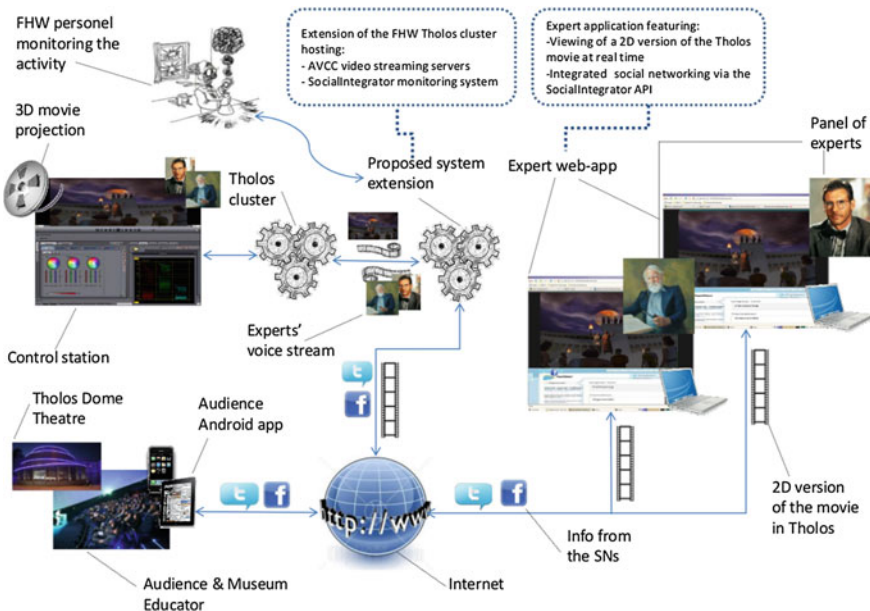


Fig. 5 Proposed system overview

3.3 Leveraging Social Networks in the Tholos Experience

In order to integrate social networking with the visitors' real experience inside Tholos, we envisioned a concept of creating a virtual "event" on Facebook and Twitter for every Tholos show. In the case of Facebook, which supports the concept of events, this was straightforward and involved the creation of a Facebook event under the official Facebook page of FHW. As regards Twitter, which does not support events, we used "hashtags" for simulating an event and managing its content. Once the visitors login to their Facebook or Twitter account using their dedicated VA, they are automatically prompted to attend the specific Facebook or Twitter event respectively, and thus have access to their feed, which includes the comments and questions of the audience, and the answers provided by experts. Likewise, the experts attend this social activity using their own dedicated EA.

There are two issues regarding producing and displaying feed on the SNs in a user-friendly manner. First, the questions of the visitors should be correlated with the appropriate part of the movie that the question pertains to. Second, the experts should be able to view questions grouped under common referenced context and, to avoid confusion, only those that relate to their specific expertise (e.g. an architect might prefer to only view questions on historic buildings). However, as already described, the VR content projected on Tholos is rendered in real time and is dependent on the navigation of the educator, i.e. it depends on the selected course and the speed with which the educator performs the navigation. Moreover, the fact that the visitors may type a question at their own convenience regardless of the content displayed at that moment, as well as the latency introduced when posting/retrieving content to/from the SNs via the Internet, makes timestamp-based solutions inefficient in this case.

To this direction, with each new Tholos show, a list of photos of representative parts of the movie (e.g. historical sights) is added to the corresponding Facebook and Twitter events before the actual show begins. The VA retrieves dynamically these photos during the presentation, thus permitting the visitors to automatically associate questions or comments they have with the particular content of the movie without typing extra text for specifying the context. Meanwhile, the experts attend this social activity and answer to questions of their expertise. It should be noted that the definition of the content of photos and their creation was performed by FHW experts manually. However, this could be achieved in an automated way using video summarization techniques as in [8].

Achieving social network transparency: One of the primary design attributes of SocialIntegrator is SN transparency, i.e. providing the same functionality and user interface regardless of whether visitors log in to their Facebook or Twitter accounts. At the same time, the SN source of the feed should be transparent to the experts. This called for certain design and implementation decisions. In the case of Facebook, each photo has a unique message description. The implementation of audience's comments is connected to Facebook comments posted under the photo. On the other hand, questions, which are not inherently supported per se, are implemented as

separate posts on the feed by concatenating the photo's description with ':', and Facebook comments on those posts are the answers.

With respect to Twitter, the event is identified by a short uncommon hashtag, such as #fhw_ev01, which should be present in every subsequent post related to the event. The photos uploaded by the FHW account should have, similar to Facebook, a unique hashtag, in addition to the event hashtag. Questions or comments on a photo are treated as tweets that have both photo's hashtags and an additional #q or #c hashtag respectively. A reply to a question in this case is simply a reply tweet to a question starting with event hashtag.

3.4 Social Network Activity

The experimental system required the development of applications to facilitate the interactions of the participants, namely the visitors, the experts and the educator.

1. *Visitors of the Tholos*: While watching the 3D movie, the visitors can interact with the panel of experts using the Visitor Application (VA) on their Android smartphones or tablets (Fig. 7). Built on top of the SocialIntegrator software (see Sect. 2.1), this Android application offers the visitors a user-friendly way of interacting with the experts and with each other via the SNs. The initial flow prompts the visitor to 'Sign In' via the supported SN of their choice. As with any application accessing social media, the appropriate permissions will be prompted and validated, passing on the ability to the application to take over permitted actions. The flow of the application is guided to the gallery. The gallery contains the snapshots of the 3D movie, through which the visitors can swipe right or left (Fig. 6 left-hand, top). For each snapshot, the visitor can click 'like', ask a question and view all questions and answers on that photo (see Fig. 6 left-hand, bottom).
2. *Panel of experts and educator*: As already mentioned experts participate in the event remotely. This is made possible by using the Expert Application (EA), a web-based application that was developed using the AVCC and SocialIntegrator software components (see Sects. 2.1 and 2.2) to allow the experts to view the Tholos movie remotely and to access related SN content, respectively.

Figure 8 shows a snapshot of the EA. The left-hand panel shows the 2D stream of the Tholos movie, as well as the expert's own camera feed on the bottom left corner.

The content of SNs appears on the right panel of the application. The experts are expected to login to both Facebook and Twitter so that they can provide answers to every audience member. Through this interface, the experts can see the photos of the event and their related comments, questions and answers. The experts can then choose to answer questions on which they have expertise. Furthermore the number of 'likes' of each question is shown, so that the experts may choose the most popular questions to answer if they need to prioritize.

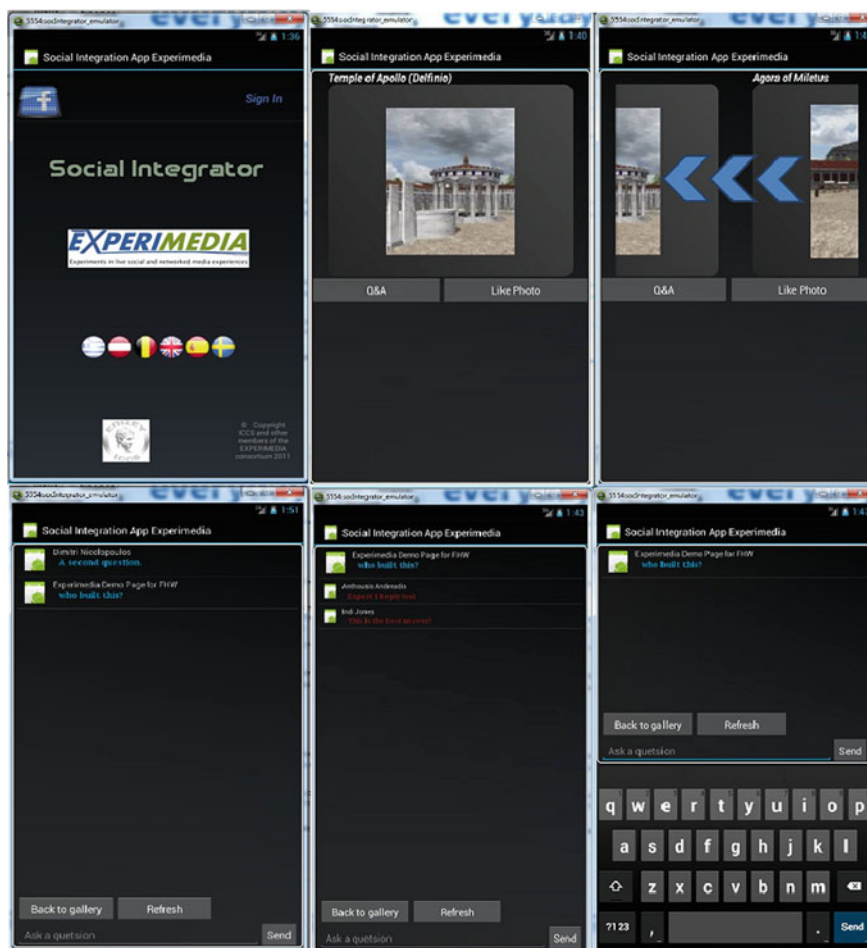


Fig. 6 *Top left* Login menu. *Top middle* Main gallery view. *Top right* Swiping through gallery pictures. *Bottom left* List of questions pertaining to a particular image in the gallery. *Bottom middle* When a prolonged click is performed on a question, if comments exist they appear under the question, in red. *Bottom right* Keyboard that appears when “ask a question” field is clicked

The repertoire of applications is completed with the educator application, which includes the video feeds of the remote experts, as shown in Fig. 9. The EA is also equipped with a Flash object that sends their webcam and audio outputs to educator console in the Tholos. The content is streamed from the EA web interface to a media server, which relays the content to the educator console back in the Tholos, avoiding the need for the museum to have a complex firewall setting. Using this application, the educator can also invite the experts to talk to the audience, over the Tholos audio system.

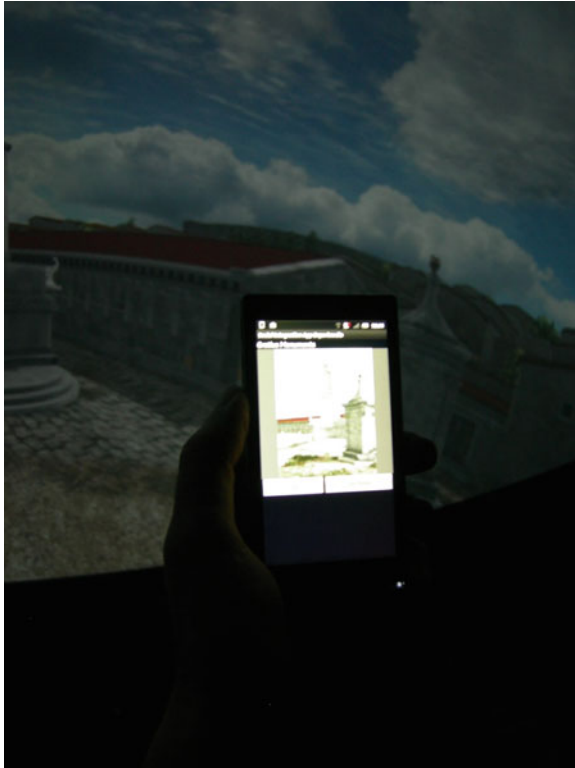


Fig. 7 A visitor using the mobile application inside Tholos

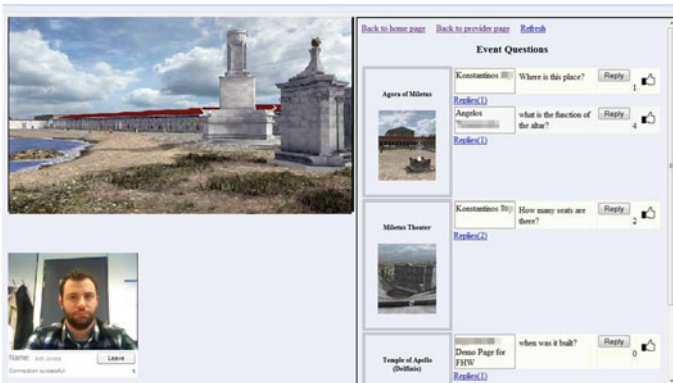
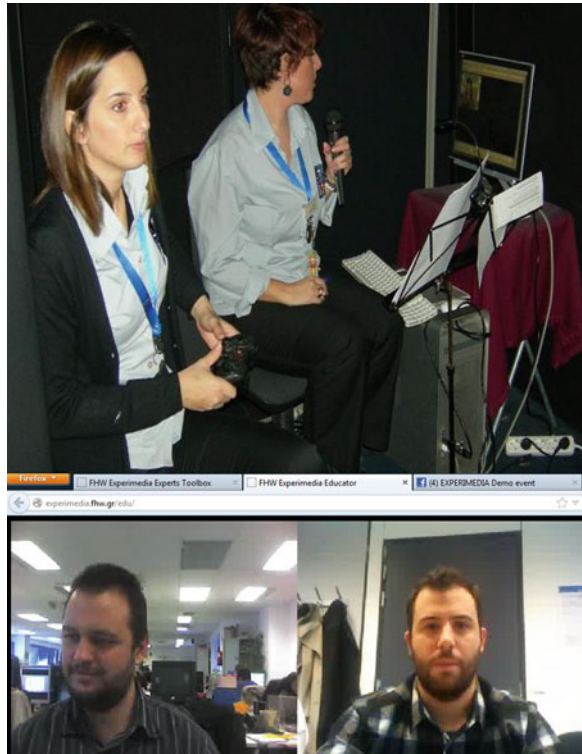


Fig. 8 Expert web-based application

Fig. 9 *Top row* Two museum educators guide the audience through a virtual representation of the ancient City of Miletus inside Tholos. One educator controls the navigation with the joystick, while the other one narrates historical information. The educator app can be seen on the PC on the right. *Bottom row* Close-up view of the educator app with the video feed from the experts. At the particular demonstration, one “expert” was physically located in Madrid, whereas the other one in Brussels



3.5 Monitoring the Experiment

During the experiment and in the background, data that has been exchanged among the participants, i.e. the panel of experts and the visitors, are being collected from the SNs and used to calculate metrics of interest. A sub-part of the SocialIntegrator is responsible for retrieving these data of interest. This sub-part is a stand-alone Java application that uses credentials of SNs’ accounts with sufficient permissions and access to the targeted social activity.

Contrary to other monitoring sources, such as the AVCC, in which the monitoring interval may obtain very low values (<5 s), when it comes to the SocialIntegrator monitoring which monitors the activity of the participants in the social network, such low values are not realistic and simply create unnecessary network traffic. Also, given that the SocialIntegrator Monitoring performs several requests to the social networks via the Internet to retrieve experiment-related data each time it is queried by the ECC EM, in practice we experienced situations whereby the query interval was lower than the time needed to collect the data for the previous query. To this direction, the part of SocialIntegrator Monitoring that is responsible for collecting data from the SNs, has now become a daemon thread that collects data using its own time interval.

Note that this interval is configurable and can be adjusted to the needs of the specific experiment and the speed of the network that is used.

For the specific deployment of experimental system, the hosting venue, FHW, was mostly interested in the following two aspects:

1. **Generic information about the SN activity.** This information helps the FHW understand whether the audience (and what part of the audience i.e. their age group) liked the new experimental system that is offered in the Tholos theatre. To this direction, information about the visitors attending the SN event is being retrieved, and then used to calculate metrics about the overall participants engagement in the social activity, such as the number of attendees, their average age, and the average number of comments/questions per attendee.
2. **Collecting feedback from the visitors on the different parts of the movie.** In order to collect information about the way the audience perceived the movie, each photo hosted in the SN event, which as already described, represents a different part of the movie, becomes a monitoring entity on its own and several attributes are then attached to it, such as number of likes, comments, questions and answers per photo, as well as the top comment, question and answer per photo.

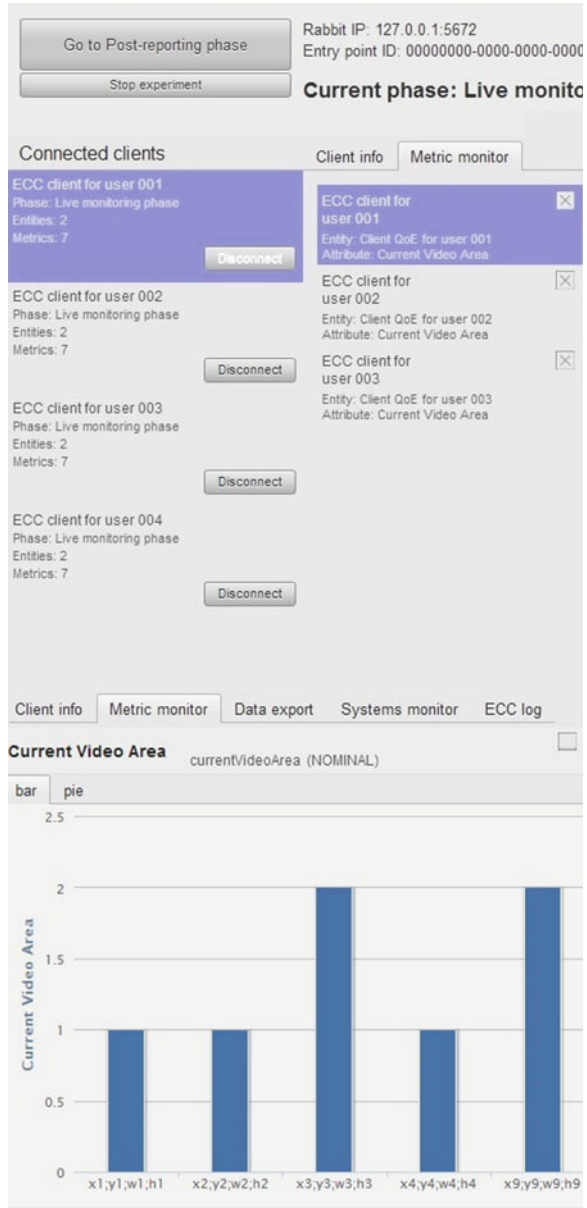
The SocialIntegrator subpart that is responsible for retrieving information from the supported SNs, is calculating the above monitoring metrics and is communicating them to the EM in order to present them to the FHW representative during the experiment. Figures 10, 11 and 12 show snapshots of the EM GUI during the experiment.

4 Collected Results—Discussion

Regarding the details of the experiment, there were three main experiment runs (excluding test runs) over a 2-week period with a total of approximately 70 users, the majority being high-school and university students. Given the users' age and the simplicity of the visitor app, understanding how the system works was rather straightforward. During the 45-min long educational movie the visitors used the android app to comment on the content they were watching and ask questions to remote experts. Although they enjoyed this aspect of the experience, it did not seem distracting. Overall more than 30 questions were asked and even more comments were posted.

After the show has ended, the collected metrics on QoE and QoS are reported and evaluated. Comparing the number of 'likes' among the snapshots can lead to interesting conclusions. For instance, the "Agora of Miletus" collected the most 'likes', which means it was a rather popular part, so the organizers should possibly consider devoting more time to it. On the other hand, a photo with zero or very few 'likes' should motivate the organizers to revise the narrative on that part or even remove it from the show. Moving on, the collected metrics indicate what the

Fig. 10 ECC dashboard screenshot showing multiple clients connected



question with the greater number of ‘likes’ is (i.e. the “top question”), as well as the “top answer” for that question. This means that the audience really wanted to know about that particular topic and was especially satisfied by a specific answer (since multiple answers can be given by experts). An instance that occurred during the experiment run is characteristic: at the point where the film shows an altar in

Social Integrator Monitoring Client
 UUID: fecb388f-7742-48b9-b98a-46a3177709a2

Measurement Set 1:
 Name: Social Integrator Generator
 Description: Metric generator for the Social Integrator
 UUID: 5049c2f6-158b-4a40-bfc2-e55326919d50


Entities	Attributes	Info
<p>FHW photo: Temple of Apollo (Delfinio)</p> <p>FHW photo: Agora of Miletus</p> <p>FHW photo: Miletus Theater</p> <p>FHW Facebook event</p>	<p>Top comment for photo</p> <p>Top answer for top question on photo</p> <p>Number of likes for photo</p> <p>Number of questions for photo</p> <p>Number of answers for top question on photo</p> <p>Top question for photo</p> <p>Number of comments for photo</p> <p>Like count for top comment on photo</p>	<p>UUID: 11cede07-e9be-44b1-b051-825f830f40d1</p> <p>Name: Top comment for photo</p> <p>Description: Comment with more likes for Facebook photo Temple of Apollo (Delfinio)</p> 

Fig. 11 Snapshot of the EM GUI displaying all available photos and their related metrics

the Agora of Miletus, the narration omits to explain its function and the reason why this is considered significant. A member of the audience posted a related question and the expert's answer proved to be the most popular post of the experiment run, being both informative and humorous. The producers therefore decided to include this answer in the narrative.

Of course, a more thorough statistical analysis of the collected measurements could help extract more sophisticated conclusions by applying methods as in [1–3, 7]. For instance, the popularity of specific movie parts or particular types of experts' answers (humorous, elaborate, scholar) could be associated to certain age groups, e.g. younger visitors tend to prefer short, witty answers by experts, rather than heavy-on-historic-information ones. Another more specific interesting conclusion had to do with the attention span of younger visitors: they tended to lose interest when the narrative went into great detail, whereas brief, to-the-point and potentially humorous descriptions and answers keep them engaged. As can be corroborated by the organizing venue (FHW), this can be a valuable source of information that could supersede conventional feedback collection methods, such as post-performance questionnaires. Furthermore, QoS metrics regarding the performance of audiovisual streaming provide interesting feedback on the technical performance of the audiovisual streaming, such as frames per second, playback rate, dropped frames, etc.

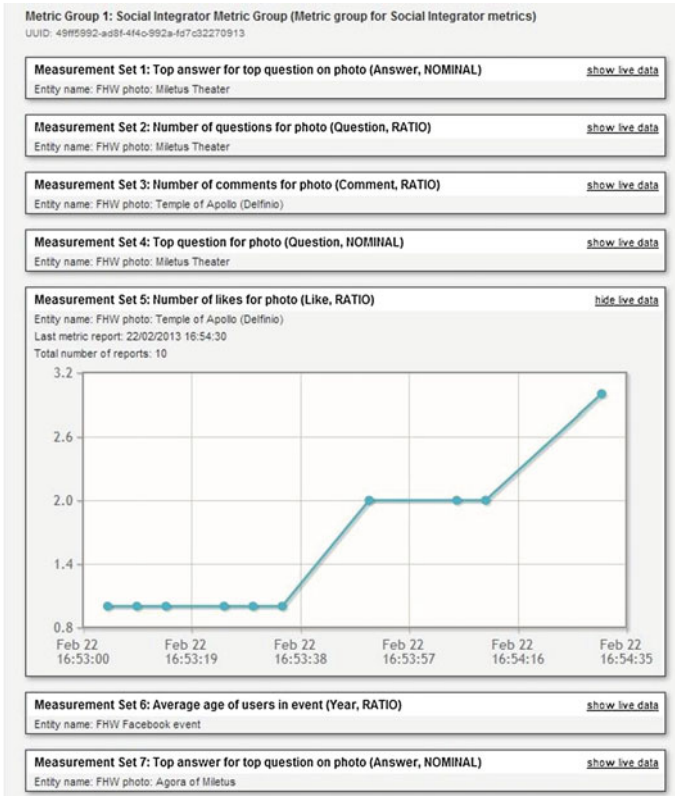


Fig. 12 Snapshot of the EM GUI displaying measurements about the number of likes on photo “Temple of Apollo”

The above results pertain to the added value for the venue/producers. The value for the visitor is more straightforward but equally important: an enhanced, richer, shared experience. A relatively conventional, and sometimes even mundane cultural/educational experience can thus be transformed into an exciting opportunity to discuss one’s questions with field experts, while having fun and communicating with one’s peers. The experts were also very positive about the concept, which enabled them to communicate from their own personal space with students and use their expertise to trigger younger people’s interest on history.

5 Conclusions

We presented an experimental system that provides visitors in cultural and educational events an enhanced, immersive, entertaining, “fresher” experience. We also demonstrated how social networking activity can be leveraged to extract valuable

instant, spontaneous and comprehensive feedback for organizers of such events, thus actually providing a basis for revamping past methods of data collection. The presented framework constitutes a large-scale experiment on a state-of-the-art unique technological facility aiming at investigating the potential of the Future Media Internet.

As future work we will support additional social networks via the SocialIntegrator (e.g. Instagram, Flickr), and we consider extending the current framework to include supplementary functionality (e.g. 3D reconstruction of the expert and superimposition on the live movie) as well as investigating its scalability. The presented system has been successfully deployed and used in the Foundation of the Hellenic World cultural center and is a work in progress to be continued in the years to come. The system can be adapted to fit the scope and needs of other cultural or educational centers, but can also be applied in different contexts, such as pilot screenings or film festivals, which usually involve discussions with the creators. Although the necessary adjustments depend on the particularities of the use case, the application of such a system would presuppose the possibility to locate separate thematic or semantic units in the film or, more generally, the experience offered. This permits “anchoring” the social network activity (comments, questions, answers) exchanged between users and experts to specific key parts of the experience, so that this activity can be easily accessible and exploitable.

Acknowledgments The research leading to these results has received funding from the European Community’s Seventh Framework Programme FP7 under grant agreement n.287966 EXPERIMEDIA—EXPERiments on live social and networked MEDIA experiences.

References

1. Bakshy E, Hofman JM, Mason WA, Watts D (2011) Everyone’s an influencer: quantifying influence on Twitter. In: King I, Nejl W, Li H (eds) WSDM. ACM, pp 65–74
2. Bodendorf F, Kaiser C (2009) Detecting opinion leaders and trends in online social networks. In: Proceedings of the 2nd ACM workshop on social web search and mining, SWSM ’09, pp 65–68, New York, NY USA, 2009. ACM
3. Cha M, Haddadi H, Benevenuto F, Krishna Gummadi P (2010) Measuring user influence in Twitter: the million follower fallacy. In: Proceedings of the ICWSM
4. Gaitatzes A, Papaioannou G, Christopoulos D, Zyba G (2006) Media productions for a dome display system. In: Proceedings of the VRST’06, pp 261–264
5. Google (2012) The new multi-screen world study: understanding cross-platform consumer behavior August 2012. <http://www.thinkwithgoogle.com/research-studies/the-new-multi-screen-world-study.html>
6. Hubspot (2012) The 2012 report on inbound marketing practices and trends. The 2012 state of inbound marketing. http://cdn2.hubspot.net/hub/53/blog/docs/ebooks/the_2012_state_of_inbound_marketing.pdf
7. Kardara M, Papadakis G, Papaioikonomou T, Tserpes K, Varvarigou T (2012) Influence patterns in topic communities of social media. In: Proceedings of the 2nd international conference on web intelligence mining and semantics, WIMS ’12, pp 10:1–10:12, New York, NY, USA, 2012. ACM

8. Kosmopoulos DI, Voulodimos AS, Doulamis AD (2013) A system for multicamera task recognition and summarization for structured environments. *IEEE Trans Ind Inform* 9(1):161–171
9. Papaioannou G, Gaitatzes A, Christopoulos D (2013) Enhancing virtual reality walkthroughs of archaeological sites. In: *Proceedings of the VAST'03*, pp 175–184
10. Stelter B (2013) Twitter buys company that mines chatter about TV. *The New York Times*, 5 Feb 2013

Social Network Analysis for Biometric Template Protection

Padma Polash Paul, Marina Gavrilova and Reda Alhajj

Abstract In this book chapter, novel cancelable biometric template generation algorithm using Social Network Analysis is presented. Two sets of features are fused using Social Network. Fusion technique is cancelable. Eigenvector centrality is used to generate final sets of features from the Virtual Social Network (VSN). From the features, Virtual Social Network generation and analysis is one of the important tasks. Main difficulty of this process is to find the validity of social network feature. Two sets of feature keep the relation among them for the classification. Finding the relation among the features could be done by using social network analysis. A new algorithm to generate VSN from the features is also presented in this chapter. Generated virtual network keeps the discriminability among the features. Once the features are different, the generated networks are different from the previous one. Social network based feature analysis confirms the domain transformation. It is computationally very hard to regenerate the original features from the social network matrix value. This domain transformation confirms the cancelability in biometric template generation. To ensure the multi-level cancelability random projection is used to project social network based feature. Performance analysis for biometric domain data is also presented in this chapter.

Keywords Social network analysis · Virtual social network · Cancelable biometrics · Biometric security · Template protection

1 Introduction

The recent study conducted by McMaster University, found at the Government of Canada Public Safety Web site (<http://www.publicsafety.gc.ca/>), reports over five

P.P. Paul (✉) · M. Gavrilova · R. Alhajj
University of Calgary, 2500 University DR. NW, Calgary, AB, Canada
e-mail: pppaul@ucalgary.ca

M. Gavrilova
e-mail: marina@ucalgary.ca

R. Alhajj
e-mail: alhajj@ucalgary.ca

© Springer International Publishing Switzerland 2015
P. Kazienko and N. Chawla (eds.), *Applications of Social Media and Social Network Analysis*, Lecture Notes in Social Networks,
DOI 10.1007/978-3-319-19003-7_12

million cases a year related to stolen identity, phishing, and direct attacks against individuals in Canada alone. The same study points out those three quarters of large Canadian organizations have been victims of cyber-attacks. Over last few years, numbers of terrorist attacks have increased despite the increased border control and biometric verification. An illegal access to a security system is possible using stolen identity or hacking. Therefore, secure authentication for logical and physical access control is crucial to protect the system and information. Unauthorized access to the system may cause valuable data and information leakage. Typically, authentication systems are based on passwords, pins, smart ID cards or tokens. Unfortunately, hackers can easily break in and get access to password-protected systems through brute-force dictionary attacks [1]. Unauthorized access can also be obtained by stealing a smart card. Biometric systems are more secure than traditional systems [2, 3]. Topology-based methods have been very popular approaches for single biometrics [4, 5]. On the other hand, social interactions are also under investigation for secure biometric system [6]. That is why biometric based system is an established solution for various security applications such as border access control, immigration, national registration, online applications (e-commerce). Nowadays, biometric systems are also suffering from vulnerability because of communication facilities and availability of technologies to hackers. Therefore, protection of a biometric security system is vital. This motivated me to conduct research on template protection in biometric security system.

The concept of cancelable biometric (cancelability) has emerged very recently [7–9] as a powerful tool for template protection [10–12]. Cancelable biometric system is a type of system that can protect the database from template level attack. Biometric system with cancelability, also called the template protection scheme, can defend the biometric database from security threats. This trend focuses on how to transform a biometric data or feature into a new one so that users can easily change their single biometric template in a biometric security system. This newly emerged direction also supports template changes across the application domain, which means a user can use different biometric template for different biometric application. Cancelable biometric is a new field for securing biometric system. Different methods for better performance are still under investigation.

In recent years, SNA was applied for different types of data mining tasks for intelligent applications [13]. There are several method for facial biometric extraction and recognition [14]. The constructed network signifies relative knowledge about the node of the network (facial features). Feature extraction is possible using social network analysis. For cancelable biometric system, random weight can be used when generating social network. This weight can be found based on the correlation values for the folds after cross folding. To regenerate the network, another random weight can be used. This randomness of network allows cancelable fusion. Construction of the network should keep the relationship among the features, i.e., keep the discriminability. Social network analysis can be used for both multimodal approach on single and multiple biometric traits. In this chapter, we have explored the use of social network analysis for single biometric trait.

The rest of the chapter is organized as follows. In Sect. 2, background research on cancelable biometrics is presented. Brief discussion about properties of cancelable biometric system is also presented. In Sect. 3, proposed methodology is presented. The use of social network based cancelable fusion is discussed and validated in Sect. 4. Section 4 also demonstrates secondary template protection scheme using random projections. Finally, in Sect. 5 experimental result is presented.

2 Background Research

Individuals biometric traits are stored in a template database for both the training and the matching processes. The most important part of the biometric system from the point of view of security and privacy is the template database. Previous studies [15] have shown that the raw image or text can be recovered from the template stored in the database. A first approach was to store the transformed version of original template to deal with biometric security and privacy [15]. Ross et al., 2007, reconstructed fingerprint image from stored minutiae points. In previous research on biometric template protection, authors suggested the dependency of cancelable biometric algorithm on security, discriminability, recoverability, performance and diversity of the system [11, 16, 17]. They noted that it should be computationally hard to reconstruct the original template from the transformed template. The discriminability of the original biometric template should not be lost after the cancelable transformation, as well as performance should not degrade. On the other hand, the revocability and diversity are the two most important characteristics of Cancelability. Authors of the book Handbook of Fingerprint Recognition in chapter nine [16] emphasized that a biometric template protection algorithm should satisfy the following four requirements:

Security: Reconstruction of the original biometric template from the transformed biometric template should be computationally hard [16].

Discriminability or Performance: Cancelable transformation should keep the discriminability consistent before and after the transformation. The performance of original and cancelable biometric should not be degraded [16].

Diversity: Cancelable template should not match across the application, i.e., Cancelable template of an individual from one application must not be compatible to another application [16].

Recoverability: An individual can easily revoke compromised biometric template and reissue another cancelable template for the application using the same biometrics. This process should be similar as changing a password as many times as user wishes [16].

Biometric templates are stored in a database in different ways; they can be stored as raw biometric data or as extracted feature vectors. Some researchers suggested to store the encrypted template so that keys are needed to extract the template [18–20]. However, storing the feature vectors in a database is not secure because the original template can be obtained through reverse engineering [15]. In 2003, Adler [15] showed that it is possible to reconstruct the original face from the face-recognition

template stored in the database. Ross and his colleagues presented fingerprint template reconstruction from minutiae points [21]. Scheirer and Boulton presented a system that can crack fuzzy vaults and encrypted biometric [22].

In 2005, Adler described vulnerabilities of biometric encryption [23]. He presented the case that even if biometric templates are protected using encryption, it is still possible to break the system [23]. From the research on template protection, it is found that templates are not secure even after encryption. Leaking of biometric template violates the privacy and security of an individual and biometric system becomes powerless. On the other hand, if the template is compromised, the access to a particular security application becomes impossible for the victim, as his biometric traits are unique and impossible to change or replace.

3 Methodology

In the first stage of the transformation is the 2-Fold random cross folding. The outcome of the process is two sets of a feature that are cancelable. Random indexes are used to generate two folds. Distance based features are calculated from Fold 1 and Fold 2. Distance features are then projected using random projection that transforms the original m -dimensional data to n -dimensional. Random indexes are transformed using Gram-Schmidt transformation [24] to an orthogonal random matrix. This random matrix is used for first random projection. In the second random projection, a random matrix is transformed using Gram-Schmidt transformation. The Gram-Schmidt transformation is used to orthonormalize the random matrix so that it keeps the distance of the projected features similar in the Euclidean space. Transformed matrix is used as a random projection matrix. Figure 1 shows the proposed system architecture of social network based cancelable biometric system Fig. 2.

To enhance the discriminability of the feature, Linear Discrimination Analysis (LDA) is used next to find discriminant feature from randomly projected features. Finally, LDA features are classified using k-Nearest Neighbor (k-NN) classifier. Features projected using LDA are cancelable, because they come from two levels of random projection and initial cross folding between face and ear template.

If the indexes of a cross-folding and random projection matrix are changed, the cancelable template can be built for multiple biometrics. Furthermore, by changing the random projection matrix, new cancelable template can be generated for different application. Multiorder of cancelability is achieved using both random cross folding and two levels of random projection.

In the proposed method, we cannot take the feature vector because features can be reverse engineered to get the original template. One the original template is reconstructed it is compromised based on the system setup and from the main idea of cancelable biometrics. The threshold is required to take the relation among the features that are closely related. If we do not use the threshold, it keeps all the feature relation that is generalization of the feature vector. Therefore, the chance of reconstruction of original feature vector is possible from the constructed network.

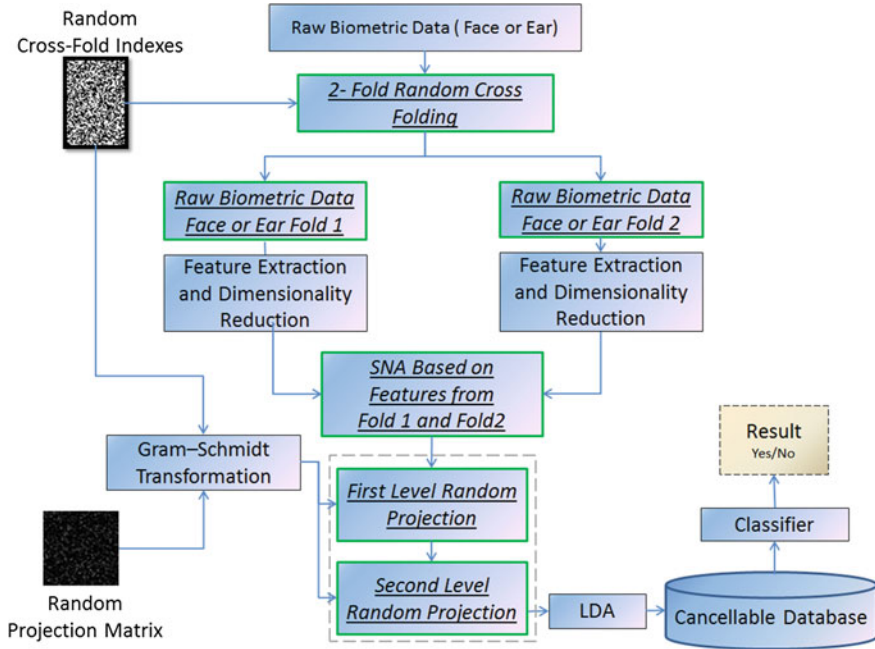
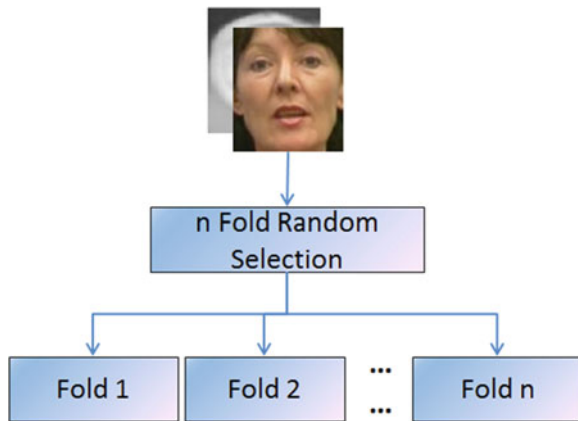


Fig. 1 Proposed system architecture using 2 Fold Random cross folding for single biometric trait. Modules with underlined text are proposed method

Fig. 2 n-Fold Random selection of biometric trait for multimodal approach using single biometric trait



3.1 2-Fold Random Cross Folding

In this step of the proposed system, biometric raw data are randomly split into n blocks. These blocks are then processed. Figure 3 shows the general block diagram for random splitting. Raw face or ear biometric features are divided into two equal parts.

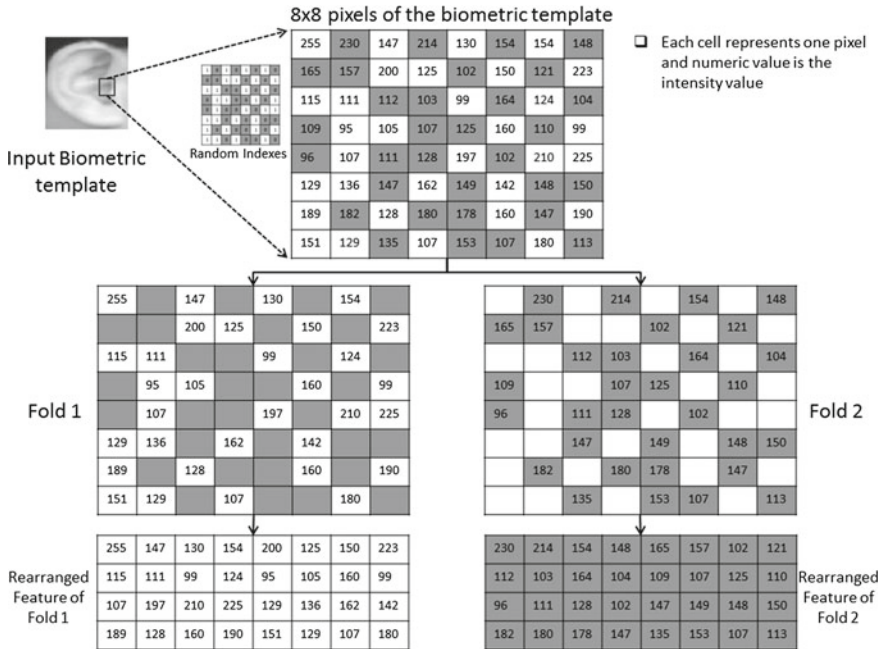


Fig. 3 Pictorial representation 2- Fold cross folding for single biometric trait. An example block of 8×8 pixels form the template is shown in this figure

A pseudorandom number generation algorithm is used to split the raw features into two parts. In the Fig. 3, we have presented the method of random selection. From $N \times N$ face template, m random features are selected and named as Fold 1. Other m features are Fold 2, where $m = N \times N / 2$. We have applied the same cancelable transformation on both folds (Fold 1 and Fold 2). Instead of selecting local features of face like lips, eye etc., we have selected random features. Randomness of the features ensures some properties of cancelability. If we do not select exact indices, it is almost impossible to authenticate an individual. These two folds are then processed separately to obtain different domain feature using another set of cancelable transformation. Finally, transformed features from two folds are projected using random projection matrix. Discrimination analysis is conducted on the randomly projected feature.

An example of 2-Fold Cross-Folding is shown in Fig. 3. This example takes 8×8 blocks of pixels. The partition of the block is based on the random indexes. Each cell of the block represents a pixel.

3.2 Feature Extraction Using Fisherface

In 1991, Pentland and Turk introduced Eigenface method to address two-dimensional recognition problems in their fundamental paper Eigenface for Recognition [25].

They have used Principal Component Analysis (PCA) to extract features for face detection and recognition system. From that work, PCA becomes a standard tool for modern data analysis in different fields such as bio-metrics, image processing, machine learning, etc. For the complex dataset, PCA is a simple, non-parametric method to extract features [26].

In 1936, R. A. Fisher introduced a statistical method to discriminate the features to classify four flowers using a linear system [27]. In his article “The Use of Multiple Measures in Taxonomic Problems”, he tried to maximize the difference between the classes of flowers. From the idea of R. A. Fisher, Belhumeur et al., 1997 inspired to establish Fisherface (FLDA) method for face recognition in different illumination condition where Eigenface method can fail. They have designed a class specific linear projection method for face recognition to maximize the interclass variation and minimize the intra class similarity [28].

Eigenface method finds total variation of data regardless of the class specification [25]. Using Fisherface method allows us to identify the discriminative class specific feature extraction. Because of the class specification and discriminability of feature extraction, we have used Fisherface method instead of Eigenface as a tool to find the features from cross-folded cancelable biometric data.

3.3 Social Network Analysis

Generating the social network for cross-folded biometric feature is an important task. Euclidian distances among the randomly selected features of each biometric trait are calculated. Distance values are then converted into similarity values and normalized in the range of zero to one. Correlation coefficients are calculated from the similarity values. If the correlation coefficient values are greater than the threshold, a link is assigned among the person. Proposed steps of virtual social network construction for biometric data are given below. There are number of metrics that keep the network characteristics. Centrality measures [29, 30] are very common and important ones. Betweenness [31–33] is one of the important centrality measure techniques. Another very important centrality value is eigenvector centrality [34, 35]. In this chapter, we have studied the eigenvector centrality of the nodes of the network for final feature.

3.4 Social Network Analysis

Performance of the biometric feature depends on constructed network. Therefore, construction of social network from the feature is very important task. Mechanism for virtual social network construction should keep the discriminability of the features. Social Network is constructed from the Fold 1 and Fold 2 features. In the first step correlation among the features are computed. Correlation coefficients are taken because it is the relation between the two features sets. A threshold is applied on

the coefficients to find the social network. Correlation coefficients are computed for all features, and it gives an adjacency matrix of relation. L1 or L2 distances are not taken because of the feature extensive information loss. Using correlation instead can keep the spatial relation among the feature, which can be the shape of the human face or ear. We have taken two different networks from the adjacency matrix. It can be done by using upper and lower triangulation.

We named two networks as Social Network-1 and Social Network-2 (Fig. 4). A weight is calculated from features of Fold 1 and Fold 2. Distance based features are used to generate weights. Eigenvalue centrality is measured for both networks. The cross products of both features are taken to fuse them together. Figure 4 is the block diagram of the proposed social network based cancelable fusion.

This process is cancelable because it satisfies the property of cancelable features. First, it is computationally hard to reconstruct the original feature from the calculated features. Second, once the cross fold indexes are changed, therefore, the feature relations as well as the constructed network changed. Finally, features from social network are non-invertible which increases the template security.

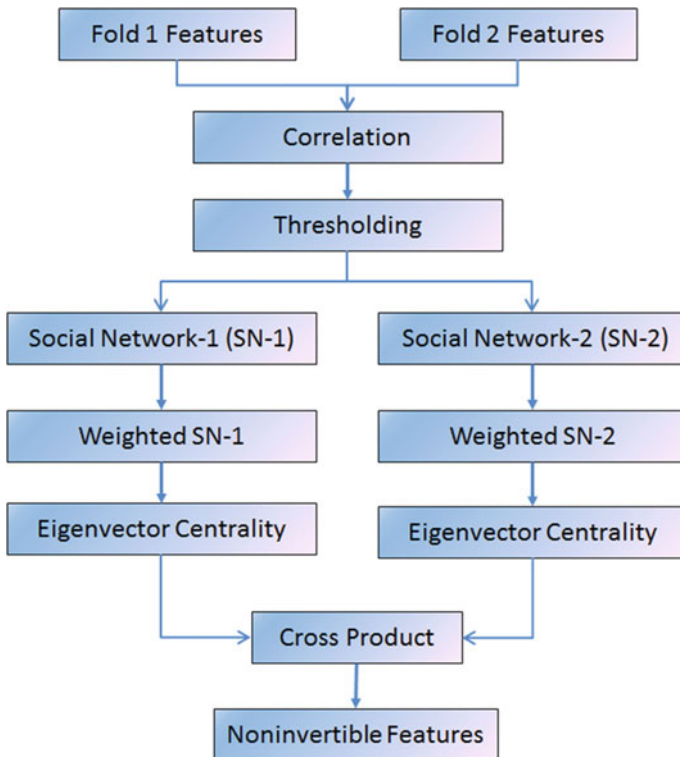


Fig. 4 Pictorial representation 2- Fold cross folding for single biometric trait. An example block of 8 × 8 pixels form the template is shown in this figure

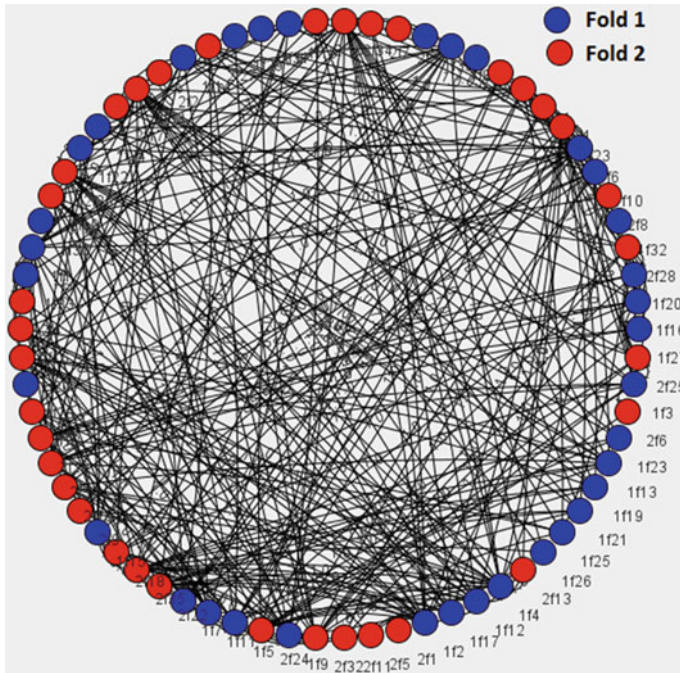


Fig. 5 Social Network-1 from *Fold 1* and *Fold 2* features of face biometrics

Selecting the threshold is one of the important tasks to generate virtual social network. In the following section, we will validate the network construction and use of threshold. Figures 5 and 6 show the social network constructed from the cross folded feature of two folds of face biometrics. Both figures are generated using NetDriller [36].

3.5 Validation of Virtual Social Network

Validation of the constructed social network can be done based on several values such as betweenness centrality, clustering coefficient, degree centrality, eigenvector centrality etc. From the analysis of the metric values we found that different metric gives better distribution of values for different threshold values. We found the best betweenness for 65 % of correlation values. For clustering coefficient, degree centrality and eigenvector centrality 60 % of the correlation values gives the best distribution of social network metric values. In the following sections, we have presented the computation for each of the centrality measures.

In 1972, Bonacich suggested that the eigenvector could be a good network centrality measure [35]. It can be computed from the largest eigenvalue of an adjacency

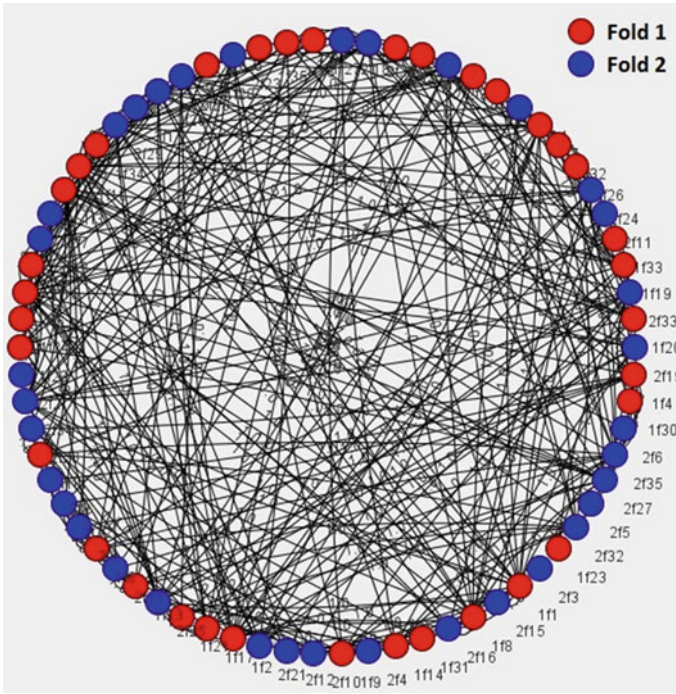


Fig. 6 Social Network-2 from Fold 1 and Fold 2 features of face biometrics

matrix [35]. It assigns relative scores to all nodes in the network. Assigning the score depends on the high-scoring and low-scoring nodes [34].

For a social network $G(V, E)$, if $|V|$ is the number of vertices and A is the adjacency matrix the eigenvector centrality of node v can be computed using Eqs. (1) and (2).

$$x_v = \frac{1}{\lambda} \sum_{t \in M(v)} x_t = \frac{1}{\lambda} \sum_{t \in G} x_{v,t} x_t \tag{1}$$

where $M(v)$ is a set of neighbor node v and is constant can be computed form Eq. (2).

$$Ax = \lambda x \tag{2}$$

In eigenvector centrality measure, eigenvector for highest eigenvalues are taken. Figure 7 shows the eigenvector centrality distribution for different correlation coefficient threshold. Network construction using 60% of the correlation coefficient gives better convergence for the centrality of the node (person). Uniqueness of the centrality measure is an important to provide feedback to the classifier.

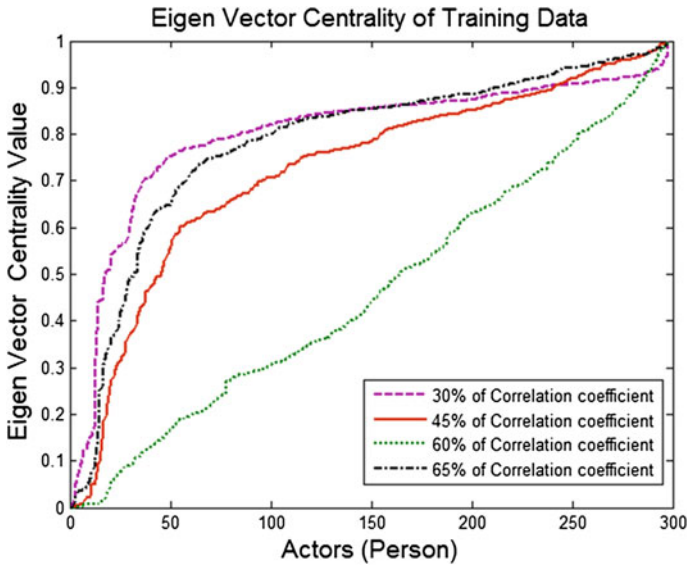


Fig. 7 Distribution of Eigen vector centrality for face training data. Values are normalize in the range 0–1

3.6 Random Projection and Classification

Johnson and Lindenstrauss [37] first develop the idea of random projection. Number of researchers used random projection for cancelable biometric system [11, 38, 39]. Random projection technique is used as an alternative of PCA for dimension reduction of data [40]. The main goal of random projection is to project vector on to a reduced dimensional space called Euclidian space [37]. The main property of the random projection is to keep the Euclidean distance similar in some extent before and after the projection of vector. Random projection changes the vector and transforms into new vector, but it keeps the statistical property of the original [39]. In the proposed method, random projection matrix is calculated in two steps. In the first step, a random matrix is generated based on the random seed. In the second step, random matrix is transformed into orthogonal matrix using Gram-Schmidt orthogonal transformation. The Gram-Schmidt transformation is used to orthonormalize a set of vectors (2D matrix) in Euclidean space R_n [24]. This transformation allows the projection to keep the distance of the projected features same in Euclidean space.

The k-nearest neighbor (k-NN) is one of the simplest and powerful classifier for pattern recognition. k-NN uses topological similarity in the feature space for object recognition [41]. It uses a majority vote of the neighbors to establish the out-put. k is a small positive integer. If $k = 1$, it uses similarity distance between two objects. It is better to use odd numbers of k to address the voting tie. There is different distance

functions used in k-NN classifier. The similarity value of k-NN algorithm can be calculated using Euclidean distance, cosine distance, city block distance etc. [41]. In k-NN classifier, value of k is very important. For all types of system, optimization of k is an important factor. Finding k can be brute force search or k means clustering. In brute force search, systems are modeled for different values of k. The model that gives the best performance is selected for k. Another way of optimization of k is using k means clustering. If a cluster is optimal, it is possible to get best k from the distribution of classes in a cluster. After understanding the feature characteristics optimal k is selected.

4 Experiment

4.1 Experimental Setup

The proposed cancelable system is preliminary study for cancelable biometric and researchers are still trying to find reliable method to protect the template. This book chapter focuses on the main idea and feasibility of the system but not the experimental result or performance comprehensively. The experiment is conducted both for face and ear database. For face, FERET [42], VidTIMIT [43] and Olivetti Research Lab Database [44] were chosen. The Facial Recognition Technology (FERET) database is widely used database to evaluate a face-recognition system. The database was collected at George Mason University and the US Army Research Laboratory facilities [42]. The images of the FERET database were collected using the 35 mm film camera with Kodak-ultra color film. Images are transferred onto a CD-ROM through multiresolution technique of Kodak. Color images are then converted into a gray-scale image of resolution 256×384 and publicly available for testing face recognition system [42]. The FERET database has 1199 subjects, and each subject has multiple facial images.

The VidTIMIT database consists of 43 subjects. This is a video database. Each subject has images of different view. To generate the virtual database, we have taken images that include different partial views of a subject. The videos are stored in a sequence of JPEG images of 512×384 resolutions. The quality of the JPEG images is 90 % [43].

The Olivetti Research Lab Database, also known as AT&T database of faces, contains a set of face images of 40 subjects. The database was used in the context of a face-recognition project carried out in collaboration with the Speech; there are ten unlike images of each of 40 subjects. The images were taken at different times, varying the lighting, facial expressions (open/closed eyes, smiling/not smiling) and facial details (glasses/no glasses). The size of each image is 92×112 pixels, with 256 gray levels per pixel [44].

Two databases called University of Science and Technology Beijing (USTB) Image Database-I & II [45] for ear were selected to generate virtual multimodal Face-

Ear database. Database-I contains 66 subjects, and each subject has three images of gray-scale. Database II contains 77 subjects, and each subject has three 300×400 images of gray scale [45].

4.2 Experimental Result

To validate the proposed method, first we have tested the cancelability of the system. The main goal for this test is to check the performance of the cancelable features and their characteristics such as interclass variability, improvement of performance etc. We have applied FLDA and double random projection on single biometric trait to get the cancelable template. We also applied LDA and k-NN on face and ear biometric data to test the original template performance. We have applied general fusion method instead of SNA. The general fusion indicates concatenation of the features. It also implies taking all the features or taking the distance of the features. Similarly, same experiment is conducted for ear biometric template. From the analysis of results, it is found that using cancelable ear feature is better than using original feature. Classification accuracy is improved by approximately 4% for cancelable transformation. Figure 9 shows the ROC curve for cancelable and the original ear biometric template.

We have proposed the new methodology using Social Network Analysis (SNA) to establish the hypothesis that SNA is feasible for cancelability. The main goal is to establish that social network analysis can be applied in biometric feature transformation and domain conversion, it keeps the feature relation, and transformed features can be used for biometric identification. We can accept social network as a new tool of biometric template protection even if it gives us similar performance as original biometric traits. However, in our experiment we have found that the performance is better than the original biometric traits.

Another experiment is conducted for single biometric trait. Instead of using the entire feature from a trait, the raw features are divided randomly in two parts to generate feature sets. Two raw sets are then processed using Fisherface to generate the discriminative feature. Instead of Fisherface, it is possible to use Social Network Analysis (SNA). In this experiment, instead of using distance feature between two folds SNA is used. SNA extract the relationships among the features of two folds. These relationships are then randomly projected to ensure the cancelability. Feature fusion using SNA is very effective and improved. Cancelable templates are achieved by fusing two feature sets to generate final cancelable template.

This system is tested for both face and ear biometric database. From the result of the experiment, it is found that social network based fusion approach can improve the performance of cancelable biometric system. The graphs shown in Figs. 8 and 9 are the results for cancelable fusion approach for single biometric trait. The performance of SNA based cancelable biometric system using only face improved by 2% over general cancelable biometric system. The ROC curve for cancelable biometric system using only face template is presented. Applying the same system architecture for ear

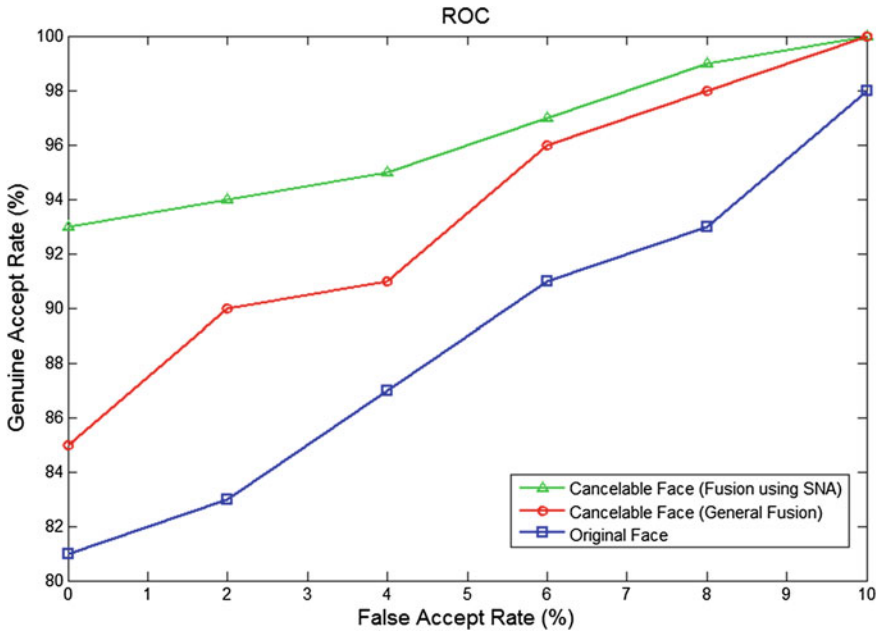


Fig. 8 ROC for original face versus cancelable face template using Social Network Analysis

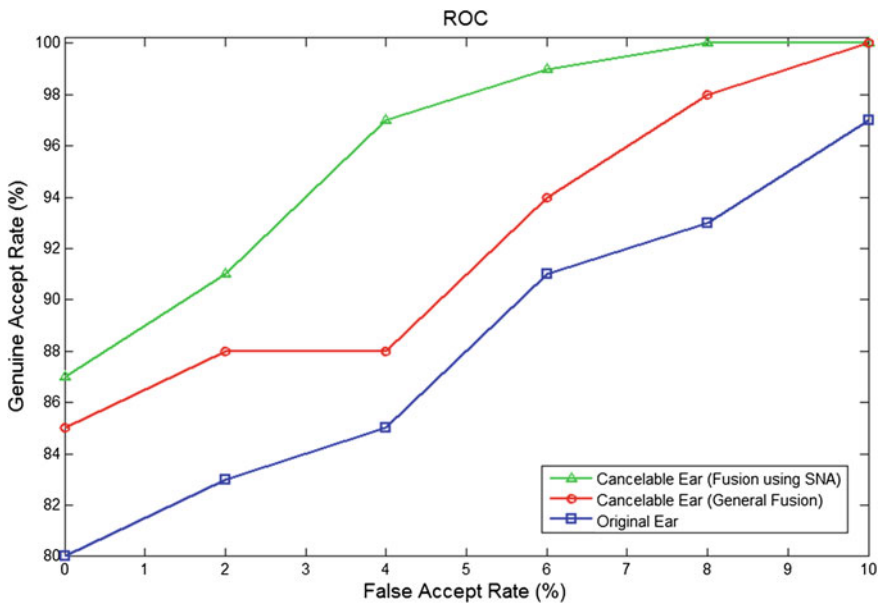


Fig. 9 ROC for original ear versus cancelable ear template using Social Network Analysis

biometric template similar result is found. SNA based cancelable biometric template performance improved over 2% compared with unimodal cancelable template of ear. For face, the overall performance improved over the original template is more than 10%. Similar scenario is found for ear as well. For ear, the overall performance is improved by more than 8% compared to the original template. Finally, it is clear that applying social network analysis improves biometric system protection and performance.

4.3 Cancelability Analysis

Template protection can be achieved by using social network based fusion. This fusion is cancelable. Once the template is compromised it is computationally hard to reconstruct the Fold 1 and Fold 2 features. If the features are non-invertible re-construction of original template is impossible. Furthermore, if the random indexes of the 2-Fold cross folding are compromised still template is secure. The Cancelability of the proposed system is self-explanatory because when we combine randomly selected features and converted the features in to a new domain without keeping any information of the original domain then it is hard to reconstruct the feature from the converted domain. Random projection is also cancelable because of the domain transformation. Now in case of social network analysis since networks are generated from random features it extends another level of domain transformation keeping the original relation but not the original feature.

To improve the confidence of the system, random projections are used. Without the random projection templates are secure using social network analysis. Even of the projection matrix are disclose to the attacker, it is hard to reverse engineer the original template.

5 Conclusion

In this chapter, multilevel cancelability is presented. New method is proposed to generate cancelable template. Cancelable fusion is presented for the first time in this paper. The performance is improved over original and general fusion methods. Specifically, in this chapter, new cancelable template protection architecture is presented. A novel biometric cancelable fusion is applied to achieve first level cancelability. Three levels of cancelability is incorporated in a single system to ensure security of the template. Random cross-folding approach for face or ear biometric traits is introduced to obtain the cancelability. Random indexes are generated using pseudo random number generator. These indexes are then used to split the biometric traits into folds. Cancelability is ensured by random fold generation and fusion. Experimental results reported in this chapter satisfy the properties of cancelable biometric systems.

Acknowledgments The authors would like to thank NSERC DISCOVERY program grant RT731064, URG, and NSERC ENGAGE for partial support of this project.

References

1. Ratha NK, Connell JH, Bolle RM (2001) An analysis of minutiae matching strength. In: Audio- and video-based biometric person authentication. Lecture notes in computer science, vol 2091. Springer, New York, pp 223–228
2. Ross A, Nandakumar K, Jain AK (2006) Handbook of multibiometrics. Springer, New York
3. Schneier B (1999) The uses and abuses of biometrics. *Commun ACM* 42(8):136
4. Wang C, Gavrilova M, Luo Y, Rokne J (2006) An efficient algorithm for fingerprint matching. In: Proceedings of the 18th international conference on pattern recognition, pp 1034–1037
5. Wang C, Gavrilova M (2006) Delaunay triangulation algorithm for fingerprint matching. In: Proceedings of the 3rd international symposium on voronoi diagrams in science and engineering, pp 208–216
6. Sultana M, Paul PP, Gavrilova ML (2014) Online user interaction traits in web-based social biometrics. *Comput Vis Image Process Intell Syst Multimedia Technol* pp 177–190
7. PP Paul, M Gavrilova (2012) Multimodal cancelable biometrics. In: 11th IEEE international conference on cognitive informatics and cognitive computing, pp 43–49
8. Paul PP, Gavrilova M (2012) A novel cross folding algorithm for multimodal cancelable biometrics. *Int J Softw Sci Comput Intell* 4(3):20–37
9. Paul PP, Gavrilova M, Klimenko S (2013) Situation awareness of cancelable biometric system, *Visual Comput* 30(9):1059–1067
10. Matsumoto T, Hirabayashi M, Sato K (2004) A vulnerability evaluation of iris matching. In: Proceedings of the symposium on cryptography and information security. Iwate, Japan
11. Feng YC, Yuen PC, Jain AK (2010) A hybrid approach for generating secure and discriminating face template. *IEEE Trans Inf Forensics Secur* 5(1):103–117
12. Ratha N, Chikkerur S, Connell J, Bolle R (2007) Generating cancelable fingerprint templates. *IEEE Trans Pattern Anal Mach Intell* 29(4):561–572
13. Memon N, Alhadj R (2010) Social networks: a powerful model for serving a wide range of domains. From sociology to computing in social networks, Part-1. Springer, Vienna, pp 1–9
14. Paul PP, Monwar M, Gavrilova M, Wang P (2010) Rotation invariant multi-view face detection using skin color regressive model and support vector regression. *Int J Pattern Recogn Artif Intell* 24(8):1261–1280
15. Alder A (2003) Sample images can be independently restored from face recognition templates. *Elect Comput Eng* 2:1163–1166
16. Maltoni D, Maio D, Jain AK, Prabhakar S (2003) Handbook of fingerprint recognition. Springer, Berlin
17. Jain AK, Nandakumar K, Nagar A (2008) Biometric template security. *EURASIP J Adv Signal Process* 2008:579416
18. Soutar C, Roberge D, Stoianov A, Gilroy R, Vijaya Kumar BVK (1998) Biometric encryption using image processing. In: Proceedings of the SPIE, optical security and counterfeit deterrence techniques II
19. Juels A, Wattenberg M (1999) A fuzzy commitment scheme. In: Proceedings of the sixth ACM conference on computer and communication security
20. Juels A, Sudan M (2002) A fuzzy vault scheme. In: IEEE international symposium on information theory
21. Ross A, Shah J, Jain AK (2007) From template to image: reconstructing fingerprints from minutiae points. *IEEE Trans Pattern Anal Mach Intell* 29(4):544–560
22. Scheirer WJ, Boult TE (2007) Cracking fuzzy vaults and biometric encryption. In: Proceedings of biometrics symposium

23. Adler A (2005) Vulnerabilities in biometric encryption system. In: Proceedings of the audio- and video-based biometric person authentication, pp 1100–1109
24. Soliverez CE, Gagliano E (1985) Orthonormalization on the plane: a geometric approach. *Rev Mex J Phys* 31(4):743–758
25. Turk M, Pentland A (1991) Eigenfaces for recognition. *J Cogn Neurosci* 3(1):71–86
26. Randall D, Martinez RT (2003) The general inefficiency of batch training for gradient descent learning. *Neural Netw* 16(10):1429–1451
27. Fisher RA (1936) The use of multiple measurements in taxonomic problems. *Ann Eugen* 7(2):179–188
28. Belhumeur PN, Hespanha JP, Kriegman DJ (1997) Eigenfaces vs. fisherfaces: recognition using class specific linear projection. *IEEE Trans Pattern Anal Mach Intell* 19(7):711–720
29. Brandes U, Pich C (2007) Centrality estimation in large networks. *Int J Bifurcation Chaos* 17(7):2303–2318
30. Freeman L (1979) Centrality in networks conceptual clarification. *Soc Netw* 1(1):215–239
31. Newman M (2010) *Networks: an introduction*. Oxford University Press, Oxford
32. Freeman L (1977) A set of measures of centrality based upon betweenness. *Sociometry* 40:35–41
33. Brandes U (2001) A faster algorithm for betweenness centrality. *J Math Sociol* 25:163–177
34. Bonacich P (2007) Some unique properties of eigenvector centrality. *Soc Netw* 29:555–564
35. Bonacich P (1972) Factoring and weighting approaches to clique identification. *J Math Sociol* 2:113–120
36. Koochakzadeh N, Sarraf A, Kianmehr K, Rokne JG, Alhaji R (2011) NetDriller: a powerful social network analysis tool. In: *ICDM workshops*, pp 1235–1238
37. Johnson WB, Lindenstrauss J (1984) Extensions of Lipschitz mappings into a Hilbert space. In: *Proceedings of the international conference on modern analysis and probability*
38. Teoh ABJ, Ngo D, Goh A (2004) Biohashing: two factor authentication featuring fingerprint data and tokenised random number. *Pattern Recognit* 37(11):2245–2255
39. Pillai J, Patel V, Chellappa R, Ratha N (2010) Sectorised random projections for cancelable iris biometrics. In: *Proceedings of the IEEE international conference on acoustics speech and signal processing*, pp 1838–1841
40. Achlioptas D (2011) Database-friendly random projections. In: *ACM SIGACT-SIGMOD-SIGART symposium on principles of database systems*
41. Darrell S, Indyk P (2005) *Nearest-neighbor methods in learning and vision*. MIT Press, Cambridge. ISBN 0-262-19547
42. Phillips JP, Wechsler H, Rauss P (1998) The FERET database and evaluation procedure for face-recognition algorithms. *Image Vis Comput* 16(5):295–306
43. Sanderson C, Paliwal K (2002) Polynomial features for robust face authentication. *IEEE Int Conf Image Process* 3:997–1000
44. Samaria F, Harter A (1994) Parameterization of a stochastic model for human face identification. In: *Proceedings of the 2nd IEEE workshop on application of computer vision, Sarasota, FL*
45. Zhichun M, Zhengguang X, Yuan L, Wang Z (2003) A new technology of biometric identification-ear recognition. In: *Proceedings of the 4th Chinese conference on biometric recognition, Beijing*

Glossary

AN Affiliation Network—a social network of authors and links that represent undirected co-authorship relationships

BA model Barabási-Albert algorithm for generating random scale-free social networks

Bag of words A simplified representation where a text is represented as a set of its words, disregarding grammar and word order.

Biometric system A system that enables identification or authentication based on biometric individual characteristic like face image or fingerprint

Biometric template A pattern extracted from biometric data used at comparison between a given and reference template

BNB Bernoulli Naive Bayes

Categorization Presenting the search results in categories rather than as an undifferentiated mass

Commuter flow centrality Refers to a network centrality quantifying node structural position—the importance using commuter volume

Cosine similarity Cosine value between two vectors that is used to quantify similarity level between them both

Cross-validation A classification model validation technique; the dataset is randomly split into n sets; typically $n = 10$; every set is used once for testing—validation, while the other $n - 1$ sets are utilized for training; the process is repeated n times so it is also called n -fold cross-validation, e.g. 10-fold cross-validation

Data leakage The creation of unexpected additional information in the training data, allowing a machine learning algorithm to make unrealistically good predictions

- Degree** The number of links incident to the node in the undirected social network; equivalent to the number of nearest neighbors; one of network centrality measures for nodes
- Delay flow centrality** Refers to a network centrality that measures structural node position—the importance using the delay incurred to commuters when a node fails
- Eigenvector centrality** One of the network centrality/influence measures for nodes; connections to nodes with high eigenvector value contribute more to this measure than connections to nodes with its low value; iteratively computed
- Extrinsic Time** Refers to time related to dynamics; it increments if the social network has changed
- FN** False Negative
- FP** False Positive
- Global network analysis** Studies the connectivity among individuals in the whole community
- GUI** Graphical User Interface
- Indegree** The number of neighboring nodes having connections to a given node; one of network centrality measures for nodes
- Intrinsic time** Refers to absolute time; in opposite to extrinsic Time
- k*-NN** The *k*-Nearest Neighbors algorithm used for classification
- LDA** Latent Dirichlet Allocation
- Local network analysis** Studies the connectivity among members in a team working on a particular project; a team is a subset of a community
- MNB** Multinomial Naive Bayes
- Natural-language processing** Determining the meaning of written text taking into account its context, grammar, colloquialisms, etc.
- NC** The Nearest Centroid
- Naïve Bayes Net** a.k.a. Bayesian Network, Bayes Network—a statistical model that represents variables and their conditional dependencies by means of directed acyclic graph
- Network centrality** Refers to a structural measure describing the structural position of a node in the social network; often used as an indicator of the node importance in a network
- Network sampling** Selection of a subset of nodes from the social network together with all links between them
- N-gram** A sequence of *n* contiguous items—typically terms from a given text
- OLAP** OnLine Analytical Processing
- One versus all** A binary classification scheme which distinguishes between one class of labels and the rest

- Opinion mining** A process for extracting, generally from a text, of the judgment or the mood of a person about a certain topic, product, etc.
- OSS** Open Source Software—a computer program with source code available to the general public for use and modification
- Outdegree** The number of neighboring nodes linked from a given node; one of the network centrality measures for nodes
- OWL** Web Ontology Language
- PA** Preferential Attachment process
- PCA** Principal Component Analysis
- Polarity of words** Refers to the strength in the classification of a word in a range of feeling going from positive to negative.
- Polarity-Length** A linguistic processing method that uses the natural relation between the polarity of a text and its length.
- Random social network** A social network with a given number of nodes and links randomly generated
- QoE** Quality of Experience
- RDF** Resource Description Framework
- REST** Representational State Transfer
- RPC** Remote Procedure Call
- SGD** Stochastic Gradient Descent
- SI** Susceptible—Infected; an epidemic model
- SIR** Susceptible—Infected—Removed; an epidemic model
- SIS** Susceptible—Infected—Susceptible; an epidemic model
- SN** Social Network is a network of (1) nodes (a.k.a. actors, vertex, members, users) that represent to social entities. i.e. individuals, groups of people, organizations or organizational units as well as (2) social relations (a.k.a. edges, binds, connections, links) that correspond to directed or undirected, weighted or unweighted relationships existing between nodes in the network; typically a link binds a pair of nodes; SN is commonly represented by a graph or adjacency matrix; in some applications, the nodes may correspond to other objects like biometric features or terms
- SNA** Social Network Analysis; a diverse bunch of methods, techniques, approaches and also measures that are used to analyze social network data
- Stemming** A process of reducing the words to their stem, base or root form
- SVM** Support Vector Machine
- TF-IDF** Term Frequency—Inverse Document Frequency; a measure quantifying importance of the term in the collection of documents
- TN** True Negative
- TP** True Positive

VR Virtual Reality

Visual analytics A visualization technique used to present the results of data analytics or to support interactive exploration of data and/or results

VSN Virtual Social Network

Index

A

Affiliation network, 63, 73
Autonomous system, 2, 10

B

Bag-of-words, 146, 149
Best friends, 41
Biometric fusion, 217, 229
Blog, 78, 91, 113, 114, 117, 119

C

Cancelable template, 219, 229, 231
Community, 59, 60, 104, 182
Commuter flow centrality, 27, 32
Complex network, 83, 103, 104
Contact network, 104, 116, 121, 123

D

Degree, 3, 6, 14, 15, 17, 18, 21, 24, 26, 31, 41, 43–47, 49–53, 59, 60, 63–67, 74, 95
Degree distribution, 43, 44, 52–54, 63, 64, 85, 86, 97, 98
Delay flow centrality, 25, 27–29, 33, 35, 36
Diffusion, 79, 98, 104, 105, 107, 109, 110, 113, 118
Dimensionality reduction, 227
Dynamics, 78, 81

E

Emergence, 178, 185, 190
Emotion recognition, 126, 127, 129–131, 141

F

Facebook, 78, 196, 205
Friendship ranking, 43

G

Global network, 59, 61, 71
Gram-Schmidt transformation, 220, 227
Graph, 104, 105, 107, 108, 111, 113, 114

H

Heuristic, 146, 157

I

Indifferent attachment, 52
Intelligence, 177, 191
Intrinsic time, 105–108, 111, 114, 117, 118, 120, 121, 123

L

Language, 143, 145, 146, 148
Learning, 149, 151, 153–156
Lexicon, 144–146, 149, 150, 153, 154
Local network, 59, 71

M

Machine learning, 132, 134, 144
Micro-blog, 78, 91
Multimedia, 195, 202
MySpace, 43, 44

N

Natural language, 160
 Network centrality, 24, 26, 38

O

Open source, 178
 Opinion, 143–148
 OSS, 58–61, 63, 65, 67, 71, 74
 Outbreak threshold, 79, 81–83

P

Pattern, 181, 185
 Polarity, 144–147, 150, 151, 155, 156
 Polarity-length, 148, 149, 153, 156, 157
 Preferential attachment, 43, 54
 Propagation, 2, 14, 21
 Provenance, 180
 Psycho-linguistic, 144, 148, 156

Q

Quality of Experience (QoE), 196, 202
 Quality of Service (QoS), 197

R

Random cross folding, 220, 221
 Random projection, 217, 219, 220, 222, 227,
 229, 231
 Raw biometrics, 219
 Relative rank, 48, 49
 Reputation, 2–4, 7–10, 12

RSS, 78

Rumor, 78–83, 85, 91–93, 95, 98, 100

S

Scale-free network, 79, 82, 83, 85, 93
 Sina Weibo, 79, 85, 91, 95, 100
 SIR, 78–81, 94
 SIS, 81
 Social media, 194, 195, 202
 Social network, 58, 59, 69, 74, 78–81, 92,
 95, 100
 Social network analysis (SNA), 126
 Social network integration, 196
 Software, 58
 SPNR, 78, 79, 82, 85, 86, 89, 95, 98, 100

T

Template protection, 218–220, 231
 Temporal network, 106
 Ties, 58, 70, 72, 73
 Top Friends, 43, 46, 52
 Topic, 159–167, 170–174
 TopicFlow, 159–162, 164–174
 Transparency, 178, 185
 Transportation network, 24, 25, 28, 38
 Trust, 6
 Twitter, 78, 95, 97

V

Visual analytics, 25, 35, 38