Modeling Information Diffusion in Networks with Unobserved Links

Conference Paper · October 2011

DOI: 10.1109/PASSAT/SocialCom.2011.50 · Source: DBLP

3 authors, including:

Satinder Singh
University of Michigan
228 PUBLICATIONS **12,910** CITATIONS

SEE PROFILE

The fifth SNAKDD Workshop 2011 on

Social Network Mining and Analysis

Held in conjunction with SIGKDD conference, August 21, 2011

Workshop Organizers

C. Lee Giles
Juanzi Li
Rong Yan
John Yen
Haizheng Zhang

Table of Contents

Graph Embedding Framework for Link Prediction and Vertex Behavior Modeling in Temporal Social Networks

Chunsheng Fang[1,3] Mojtaba Kohram[1] Xiangxiang Meng[2,3] Anca Ralescu[1]

1 Dept of Computer Science, University of Cincinnati, Ohio, USA, 45221 fangcg@mail.uc.edu

2 Dept of Mathematical Sciences, University of Cincinnati, Ohio, USA, 45221 kohramma@mail.uc.edu

3 Cincinnati Children's Hospital Medical Center, Cincinnati, Ohio, USA, 45229 mengxa@mail.uc.edu Anca.ralescu@mail.uc.edu

ABSTRACT

We present a novel framework in which the link prediction problem in temporal social networks is formulated as trajectory prediction in a continuous space. Four major modules constitute this framework: (1) **graph embedding**: the discrete space of graphs is mapped into a continuous space while preserving distances for a given graph kernel; (2) **manifold alignment**: graph embeddings corresponding to different time points are aligned to achieve low variance trajectories for a more reliable prediction; (3) **trajectory prediction**: the temporal graphs form a time series, to which various prediction models (e.g. regression) can be applied; (4) **graph reconstruction** from the predicted graph embedding which is invariant against scale, translation and rotation. Furthermore, this framework enables an innovative way to analyze temporal graph vertex behaviors and visualization. Extensive preliminary results on real world data sets demonstrate the promises of this proposed approach.

Keywords

Link prediction, social network, algorithmic framework, graph embedding.

1. INTRODUCTION

With the recent advent of large-scale social networks and biological networks etc, link prediction in temporal networks emerges as a more and more important research problem. Several survey papers have summarized some recent progress in link prediction [1] and link mining [2]. A thorough survey paper [1] summarizes various "static" methods that explore different graph distance metrics on one network snapshot and try to predict the next one.

Alternatively, recent research focuses more on formulating the link prediction problem in a "dynamic" way, for example, as a time series regression model to accommodate historical data [2,3]. This is done in a more general framework by modeling the whole graph historical dynamics to provide a more insightful understanding into the link prediction problem.

The first important issue in developing such a framework is to decide exactly what needs to be extracted from network such that (I) it can be effectively tracked, and predicted in time, and (II) to use the predicted structure to (re)construct a network state consistent with the network evolution. For example, various features, such as node degrees of the graph representing the network can be used in conjunction with an iterative regression solver to predict the graph features as the network evolves in time [3]. Alternatively, spectral approaches attempt model graph evolution using polynomial curves [4,5]. These assume eigenvectors as stable during time. Finally, a combined time series ARMA model and low rank approximation approach for estimating the eigenvectors of the Laplacian matrix from each time point has been proposed in [6, 13].

In this paper, we revisited most of the existing link prediction algorithms and the recent attempts in temporal social networks, to answer the following important questions which will lead to our framework: What key components contribute to a successful link prediction model? What are their contributions to a reliable model estimation and prediction?

As summarized in [1,9], graph distance metric is essential in encoding the local and global consistency, but simply using it is not sufficient to capture evolutions along time.

In order to utilize historical network snapshots under a general framework, a common approach is to convert the discrete graphs into the continuous space while preserving distance constraints among vertices, which can be achieved by **graph embedding**. Most of the existing continuous link prediction algorithms are actually different derivations of graph embedding. MDS [12] using principal component of the distance matrix; Spectral embedding [10] using eigenvectors of the Laplacian matrix; graph feature tracking [3] using linear embedding defined by a linear mapping.

To ensure a **reliable model for estimation and prediction** the stability graph-valued time series must be guaranteed. In this paper we submit that the stability aspect did not attract sufficient attention in previous literature. By contrast, in the framework proposed here, we point out that reliability of prediction is achieved only when a key graph feature is preserved in the process of graph embedding.

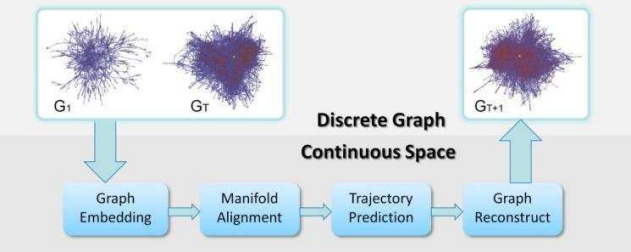


Figure 1. The framework for link prediction problem in temporal social network

Since the graph embedding results in a nonlinear subspace, we refer to this constraint as manifold alignment. More precisely, each graph embedding can be viewed as a smooth **manifold** in the higher dimensional space.

However, the individual graph geometry such as position, orientation, scale, can be arbitrary, making the link prediction problem difficult. The **manifold alignment**, i.e. alignment of the manifolds corresponding to different time points, will result in smaller variance and more stationary time series. This in turn will contribute to more accurate estimation and prediction.

To address all these above, we propose a general framework as in Figure 1 which formulates this problem as **trajectory prediction** in a continuous graph manifold space, and decompose it into four essential components: graph embedding, manifold alignment, trajectory prediction, graph reconstruction.

Extensive experiments in real-world social networks evaluate the effectiveness of each component, which demonstrate the promise of our framework for link prediction. Furthermore, several functionalities derived from our framework, such as visualization of temporal social networks as trajectories, vertex behavior modeling, are demonstrated.

2. PROPOSED FRAMEWORK

2.1 Problem Formulation

Following [6], we start by formally defining the problem as follows. A dynamic graph with horizon T, $\mathcal{G}_T$, is a collection of graphs, G_t indexed by time, t=1,2,…, T. In this work we consider that the set of nodes remains constant over time. More precisely we define

$$\mathcal{G}_T=\{G_t \mid G_t=(V, E_t),\ t=1,..,T;\ E_t \subseteq E_{t+1},\ t=1,\ldots,T-1\}$$

The link prediction problem is then to predict G_{T+1}, which is actually E_{T+1}, based on the $\mathcal{G}_T$.

The proposed framework consists of the following:

2.2 Graph Embedding

Graph embedding is the first component of our framework. The goal of the graph embedding component is to embed the graph as a point cloud in some continuous space where its evolution can be tracked using conventional regression techniques. To achieve this goal many of the available methods in Multi-Dimensional Scaling (MDS) can be used.

MDS is generally used to achieve a visualization of a dissimilarity matrix in some smooth space. A good review of the available methods and techniques can be found in [12]. MDS methods can be classified into metric and non-metric methods. In metric MDS the dissimilarity matrix must represent a metric on the embedding space. The MDS algorithm used here will be classical metric MDS where the metric will be the shortest distance between two nodes in the original graph.

After this step, each graph snapshot G_t is optimally embedded into real numbers X_t with dimension of M vertices by K MDS dimensions, as (1).

$$X_t \in R^{M\times K} \tag{1}$$

2.3 Manifold Alignment

Manifold alignment finds the correspondence between two seemingly disparate datasets [8]. It constitutes a critical component is our proposed framework, due to the variability of manifolds along time points. To see the effect of the alignment, as shown in Figure 2, we consider three alignment procedures: (i) Affine transformation alignment

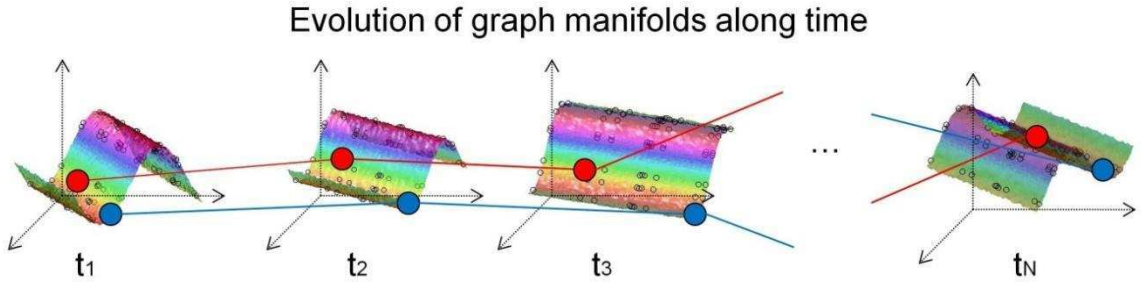


Figure 2. The evolution of graph manifolds along time. Each graph is represented as a manifold embedded in high dimensional space at each time point. The two dots are graph vertices embeddings sampled from the underlying smooth manifolds, whose correspondences are linked by red and blue lines as time series. Note that each manifold only preserves the topology within that time point, but not necessarily corresponds to the neighboring time points. This observation indicates the importance of aligning the manifolds before we model them as time series.

to the previous graph in the sequence, (ii) Affine transformation alignment to the 1st graph, and (iii) Procrustes alignment the 1st graph.

Procrustes alignment [8] seeks the isotropic dilation and the rigid translation, reflection and rotation needed to correspond two embeddings X and Y, by optimizing the following objective function:

$$Q_{optimal} = \text{argmin}_Q \|X - kYQ\|_F \quad (2)$$

where $k = tr(\Sigma)/tr(Y^T Y)$. Σ is the diagonal matrix of SVD of $Y^T X$.

With the Procrustes alignment matrix Q, correspondence between two graph manifolds can be performed with respect to different procedures. We denote the aligned manifold in time point t as :

$$Z_t \in R^{M \times K} \quad (3)$$

2.4 Trajectory Estimation and Prediction

Two standard regression models, linear and quadric, are employed for both trajectory estimation and prediction of the embedded vertex manifold coordinates. Each vertex's coordinate time series are centered and then fitted together into a nested random effect regression model, as follows:

Linear:
$$Z_t(m,k) = \beta_{mk} t + \varepsilon_{tmk}$$
$$\beta_{mk} \sim N(0, \tau_k^2), \ \varepsilon_{tmk} \sim N(0, \sigma_k^2) \quad (4)$$

Quadratic:
$$Z_t(m,k) = \beta_{mk} t + \gamma_{mk} t^2 + \varepsilon_{tmk}$$
$$\beta_{mk} \sim N(0, \tau_k^2), \ \gamma_{mk} \sim N(0, \nu_k^2), \ \varepsilon_{tmk} \sim N(0, \sigma_k^2) \quad (5)$$

for snapshots $t = 1,2,\cdots,T$, authors $m = 1,2,\cdots,M$, and dimensions $k = 1,2,\cdots,K$. Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) serve as likelihood criteria for models selection. To compare and select alignment methods, we define the learning errors as the Euclidean distance between the ground truth and the learned positions of a vertex.

$$e_{mt} = \sqrt{\left(\sum_{k=1}^{K} \left(Z_t(m,k) - \hat{Z}_t(m,k)\right)^2\right)} \quad (6)$$

Furthermore, we propose to use a weighted learning error:

$$e_{mt}^w = \frac{1}{(\sum_{k=1}^{K} Z_t(m,k)^2)^{1/2}} e_{mt} \quad (7)$$

which is more robust to the random errors on the boundaries of the whole network and put more penalty to the prediction algorithm that have relatively worse performance inside the center of the whole data clusters.

2.5 Graph Reconstruction

Prior to this step, an optimal predicted graph embedding X_{T+1} for G_{T+1} has been computed. To reconstruct the graph, we need to ensure that the graph kernel from X_{T+1} is Positive-Semi Definite (PSD) [9] . In our framework, we circumvent this potential issue by building the graph from pair-wise distance in X_{T+1}, which is guaranteed to be PSD.

2.6 ALGORITHM

Input: A dynamic graph, $\{G_t, t = 1,2,\cdots,T\}$.

Output: Predicted G_{T+1}.

1. Define a **graph distance or kernel**; Any PSD graph distance matrix or any Mercer kernel can be adopted. Local and global distance constraints are preserved in this step. Shortest path kernel is adopted in this paper;
2. **Graph Embedding**. For each graph, apply Multi Dimension Scaling, or other graph embedding algorithms (e.g. using graph spectral) to map the graph into Euclidean space (or other continuous space);
3. **Manifold Alignment**. Aligned the each G_t in the embedded space with different choices of alignment algorithms as in Section 2;
4. **Trajectory Prediction**. For each graph vertex in the embedded space, its trajectory during time can be modeled using any time series regression model, e.g. linear model[7], ARMA, etc. After this step, the graph embedding X_{T+1} is optimally predicted;
5. **Graph Reconstruction**. The predicted graph G_{T+1} can be constructed from the pair-wise distances in the embedded space X_{T+1}.

3. EXPERIMENT EVALUATION

The approach described above was implemented on a data set extracted from the DBLP authorship database[1]. This data set is analyzed from 1995 to 2004. An author is included in this set if he/she contributed in at least eight of the ten years. From this set of authors the CORE set is selected to be the largest connected component of the first snapshot (1995). This results in a CORE set of 2,538 authors at 10 different time points.

3.1 Validating Manifold Alignment

Our framework provides an innovative visualization of temporal social networks as Figure 3 illustrates. Each vertex can be viewed as a trajectory along time axis, while each layer represents the graph embedding in each year.

Three important observations can be made from Figure 3:

1. Of all trajectories evolving with time, those without manifold alignment have high temporal variance with large fluctuations;
2. All three alignments have reduced the trajectory variance with different levels;
3. Procrustes alignment performs the best both in magnitude and fluctuation of variance.

[1] http://dblp.uni-trier.de/

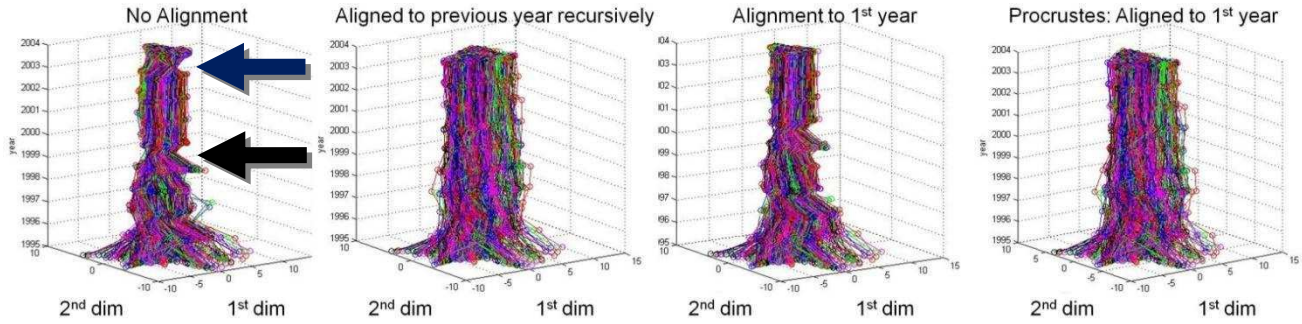


Figure 3. Trajectories of four different alignments for the real-world DBLP data set with 2,538 core authors: No alignment; Alignment to previous year recursively; Aligned to the 1st year; Procrustes alignment to the 1st year. In all panels, each horizontal layer demonstrates the 2D graph embedding of each year. Each corresponding vertex (author) is linked by line segments going upward. As arrows pointed out in the 1st panel, without alignment, the trajectories have huge variance and fluctuate dramatically. All 3 alignments have reduced the trajectory variance with different levels, among which the Procrustes perform the best.

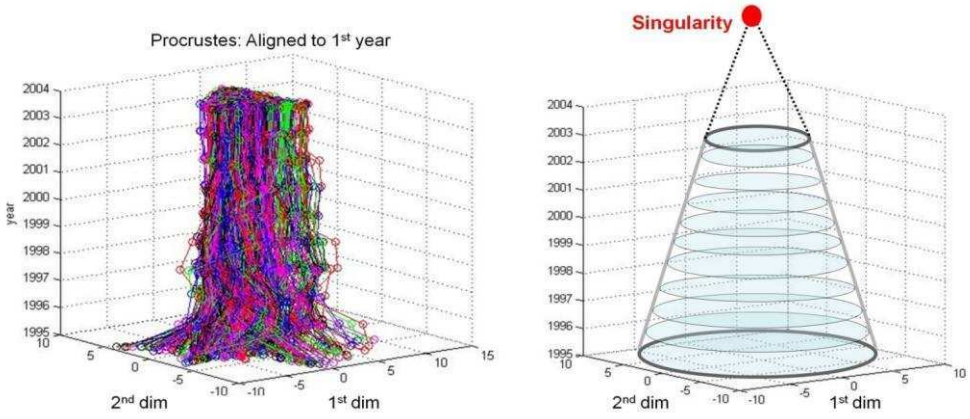


Figure 4. Gravitational collapse of trajectories to singularity. Left panel: trajectories after manifold alignment for the real-world DBLP data set with 2,538 core authors. Right panel: the conceptual idea of all trajectories converges to singularity (complete graph). Refer to Figure 2 for details. The collapsing phenomenon can be visually inspected, which indicates the diameters of the graphs are shrinking.

3.2 Gravitational collapse of trajectories

Another interesting phenomenon observed from Figures 3 and 4 (which is not accounted for by previous methods), is that of gravitational collapse of trajectories.

In the current formulation of the link prediction problem, the dynamic graph has a fixed vertex set. The edge set increases in time as edges are never deleted (for the DBLP example, this means that if two authors have been linked, they remain linked). It then follows that the diameters of successive graphs (successive network snapshots) are eventually decreasing as the number of shortest paths increases. The graph embedding and the trajectories of each vertex reflect this property. By analogy to the astronomical gravity effect which attracts the mass and eventually every atom will collapse into a singularity, we call this gravitational collapse of trajectories. In the social networks context this corresponds to the convergence of the graphs representing the network to a complete graph.

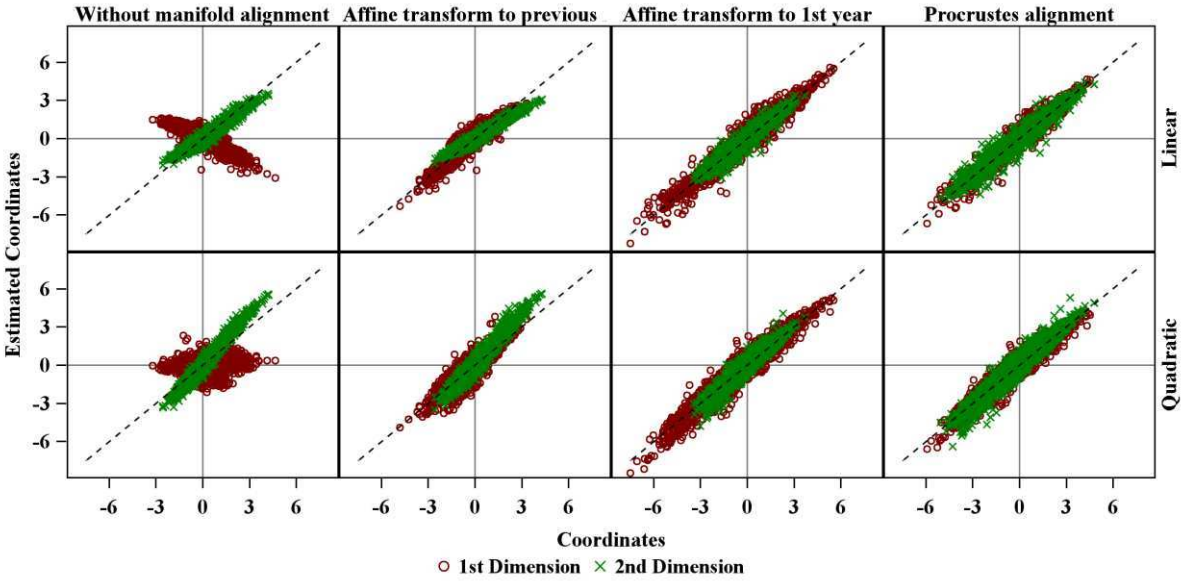
3.3 Trajectory Modeling Results

Both linear and quadratic regressions are applied for the estimation and prediction of the author coordinates in year 2004. We fit the model with the first two primary MDS dimensions. For each type of aligned data, the linear fitting has better AIC and BIC scores. Table 1 summarizes the means and the standard deviations of the learning errors for the four types of alignment algorithms. The Procrustes Alignment yields the smallest mean weighed error. It is not surprised that the data without any alignment has errors significantly larger than those from alignment.

Figure 5 presents the scatter plots of ground truth coordinates vs. estimated and predicted coordinates for the network in 2004. Prediction has wider variance than estimation, due to the exclusion of the last year data. Alignment plays a significant role in modeling the dynamic graph. The Procrustes alignment method with linear random effect regression performs well in both estimation and prediction, which strongly supports the claim that under this framework, the nature of dynamic social network is sufficiently captured by a simple model.

Table 1. Estimation and Prediction Errors for the four alignments in linear or quadratic nested random effect models.

Alignment Method	Regression	AIC	BIC	Estimation Error (6)	Estimation Error, Weighted (7)	Prediction Error (6)	Prediction Error, Weighted (7)
Without manifold alignment	Linear	192926.1	192942.2	1.7473(1.0435)	1.2271(0.7722)	2.3604(1.4096)	1.6576(1.0432)
	Quadratic	209312.2	209328.1	1.1329(0.6530)	0.8544(1.9637)	2.5351(1.4613)	1.9120(4.3943)
Affine transform to previous year	Linear	184649.8	184665.8	0.4283(0.2988)	0.3353(0.5095)	0.5785(0.4036)	0.4529(0.6883)
	Quadratic	203406.2	203422.0	0.6214(0.4009)	0.5524(2.2990)	1.3905(0.8972)	1.2361(5.1446)
Affine transform to the 1st year	Linear	**161244.7**	**161260.7**	**0.4043**(0.3362)	**0.2767**(0.6975)	**0.5461**(0.4541)	**0.3738**(0.9423)
	Quadratic	183140.3	183156.2	0.4938(0.3750)	0.3702(2.0993)	1.1051(0.8392)	0.8283(4.6977)
Procrustes alignment	Linear	**162879.5**	**162895.6**	**0.4071**(0.3324)	**0.2716**(0.6748)	**0.5500**(0.4491)	**0.3669**(0.9116)
	Quadratic	184128.5	184144.3	0.5073(0.3750)	0.3677(1.7034)	1.1352(0.8390)	0.8227(3.8117)



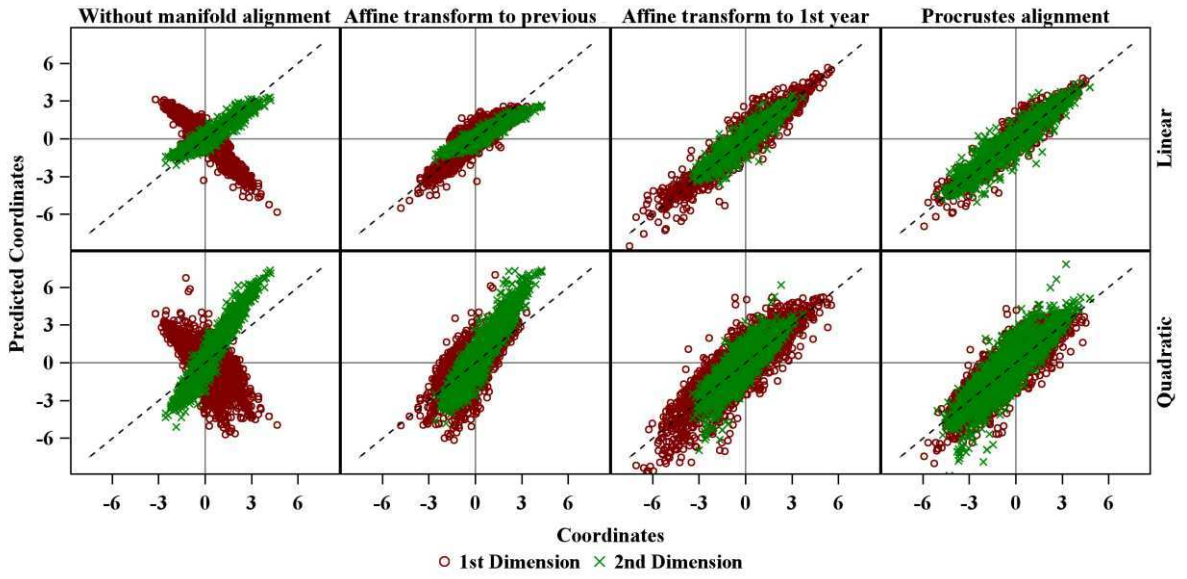


Figure 5. The scatter plots of the estimated and predicted coordinates vs. the true coordinates for the DBLP data in 2004. Top: Estimations of the core author coordinates in 2004 using the whole time series. Bottom: Predictions of the core author coordinates in 2004 using whole time series except 2004. Prediction has wider variance than estimation. Alignment plays a significant role in modeling the dynamic graph, especially Procrustes alignment.

3.4 Vertex Behavior Modeling

Another functionality enable by this framework is vertex behavior modeling, by clustering the trajectories as shown in Figure 6. Using K-means clustering, we compute 10 clusters which grouped authors with similar temporal behaviors together. Similar trajectories indicate the co-evolution patterns of authors.

To get more insights into the clustering, Figure 7 takes a closer look into seven authors' trajectories with highest vertex degrees. Interestingly, five of them have similar trajectories, and they turn out to be all from Israel, compared to the rest two are from different institutes.

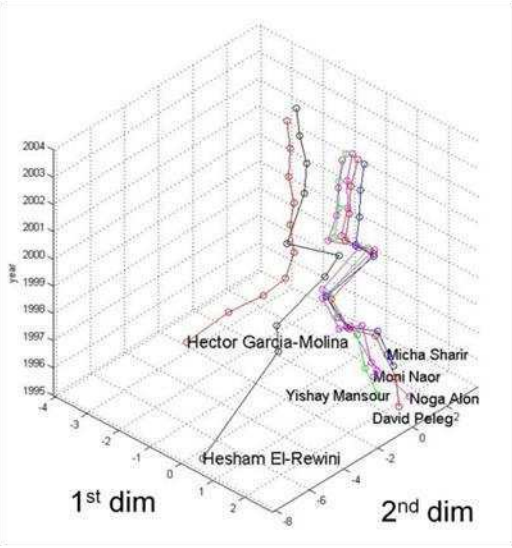


Figure 7. Trajectory behavior analysis. 7 authors' trajectories with highest vertex degrees are inspected. Interestingly, YM, NA, DP, MN, MS are all from Israel, and HGM, HER are from different institutes. We observe that these 5 Israeli authors have similar patterns than the rest.

3.5 Reconstructing the Predicted Network

To reconstruct the predicted network we take the following steps:

4. Predict the graph embedding for the last time point;
5. Collect those pairs of vertices which are not connected by an edge in the graph at the preceding time;
6. Sort the vertices collected at the previous step by their distance: pairs of vertices that are closest in Euclidean space could be potential edges;
7. Prioritize links based on the distances computed above.

The reconstruction results are compared against a set of edges created randomly as shown in Table 2. The results of each embedding method with regression methods are reported in terms of how much better they are than a random guess. The percentage of a correct random guess of an edge existence is 0.013%. The consistent superiority of our framework compared random guess suggests that the trend of graph evolution is meaningfully captured. Lastly, the Procrustes alignment method along with a linear regression method proved to be the best performing predictors, which is consistent with Section 3.4.

4. CONCLUSIONS AND FUTURE WORK

We have innovatively formulated the problem of link prediction in a network which evolves over time. We

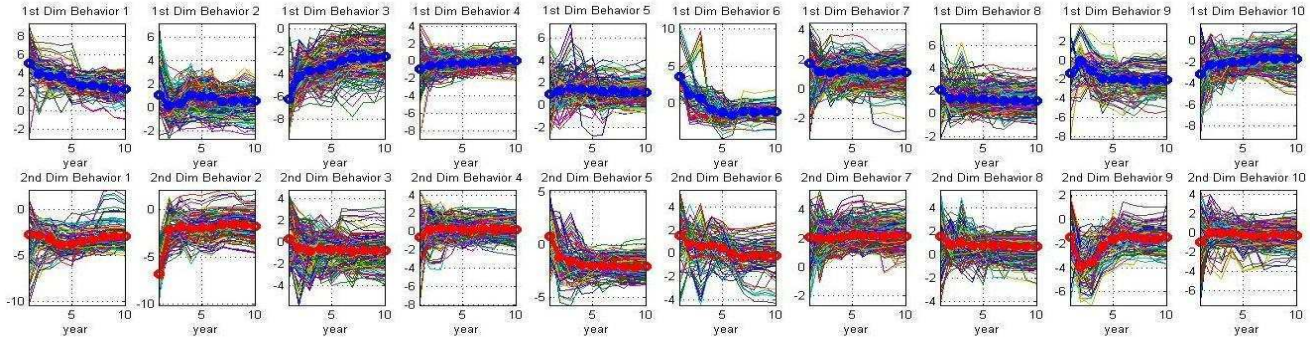


Figure 6. Trajectory behavior clustering for the Procrustes alignment. First row illustrates 10 K-means clusters of temporal trajectory behaviors for DBLP core authors for the 1st MDS dimension. Second row is for the 2nd MDS dimension. The cluster center is superimposed on each plot.

Table 2. Performance of the algorithm with different alignment and regression methods compared to random

	Without manifold alignment		Affine transform to previous year		Affine transform to the 1st year		Procrustes alignment	
	Linear	Quadratic	Linear	Quadratic	Linear	Quadratic	Linear	Quadratic
Factor of improvement over random	4.49	3.91	4.57	3.93	4.78	3.68	**4.81**	3.76

started from the premise that evolution in time of the network requires a dynamic approach, which take into account this evolution (as opposed to approaches based on node similarities in a static, snap shot of the network). This leads us to consider time series approach. However, the challenge for this approach was to extract suitable, useful network characteristics. This was done by embedding the graphs underlying the network (at each time moment) into a continuous space, resulting in a nonlinear subspace or manifold. Essentially for this approach, ensuring reliable prediction and estimation is the step of manifold alignment. Experimental results support this approach both in (i) the need of the alignment, and (ii) estimation and prediction reliability.

This proposed framework identifies key components in constructing a good link prediction model, but has not been thoroughly exploited all its potentials. What combinations of these key components will theoretically guarantee a good link prediction result? Is there a theoretical upper or lower bound for link prediction upon this framework? These question remains as future work.

5. ACKNOWLEDGMENTS

We appreciate the reviewers' precious comments.

6. REFERENCES

[1] David Liben-Nowell, Jon Kleinber, The link-prediction problem for social networks, Journal of the American Society for Information Science and Technology, Volume 58, Issue 7, pages 1019–1031, May 2007.

[2] Lise Getoor, Christopher P. Diehl, Link Mining: A Survey, SIGKDD Explorations, Volume 7, Issue 2

[3] E. Richard, N. Baskiotis, T. Evgeniou, N. Vayatis, Link Discovery using Graph Feature Tracking, NIPS 2010, Vancouver, Canada, December, 2010.

[4] Kunegis and Andreas Lommatzsch. 2009. Learning spectral graph transformations for link prediction. In Proceedings of the 26th Annual International Conference on Machine Learning (ICML '09).

[5] Kunegis, Damien Fay, and Christian Bauckhage. 2010. Network growth and the spectral evolution model. In Proceedings of the 19th ACM international conference on Information and knowledge management (CIKM '10).

[6] Chunsheng Fang, Jason Lu, Anca Ralescu, "Graph Spectra Regression with Low-Rank Approximation for Dynamic Graph Link Prediction", NIPS2010 Workshop on Low-rank Methods for Large-scale Machine Learning, Vancouver, Canada, December, 2010.

[7] Nalini Ravishanker, Dipak Dey , A first course in linear model theory, Chapman and Hall/CRC, 2002.

[8] Chang Wang and Sridhar Mahadevan. 2008. Manifold alignment using Procrustes analysis. In Proceedings of the 25th international conference on Machine learning (ICML '08).

[9] S.V. N. Vishwanathan , et al, Graph Kernels, Journal of Machine Learning Research 2008.

[10] Mikhail Belkin, Partha Niyogi , Laplacian Eigenmaps for Dimensionality Reduction and Data Representation, Neural Computation, 2003.

[11] J. B. Tenenbaum, V. de Silva and J. C. Langford , A Global Geometric Framework for Nonlinear Dimensionality Reduction, Science, 2000.

[12] Young. F. W. and Hamer. R. M. (1994). Theory and Applications of Multidimensional Scaling. Eribaum Associates. Hillsdale, NJ.

[13] Chunsheng Fang, Mojtaba Kohram, Anca Ralescu, Towards a Spectral Regression with Low-Rank Approximation Approach for Link Prediction in Dynamic Graphs, IEEE Intelligent Systems Magazine (To appear, July 2011).

Inferring the Diffusion and Evolution of Topics in Social Communities

Cindy Xide Lin[1] Qiaozhu Mei[2] Yunliang Jiang[1] Jiawei Han[1] Shanxiang Qi[1]

[*1]Department of Computer Science, University of Illinois at Urbana-Champaign,
201 N. Goodwin Ave., Urbana, IL 61801, USA
[2]School of Information, University of Michigan,
1085 S. University Ave., Ann Arbor, MI 48109, USA
xidelin2@uiuc.edu, qmei@umich.edu, jiang8@uiuc.edu, hanj@cs.uiuc.edu, sqi2@uiuc.edu

ABSTRACT

The prevailing of Web 2.0 techniques has led to the boom of various online communities, where topics are spreading ubiquitously among user-generated documents. Together with this diffusion process is the content evolution of the topics, where novel contents are introduced in by documents which adopt the topic. Unlike an explicit user behavior (*e.g.*, buying a DVD), both the diffusion paths and the evolutionary process of a topic are implicit, making them much more challenging to be discovered.

In this paper, we aim to simultaneously track the evolution of any arbitrary topic and reveal the latent diffusion paths of that topic in a social community. A novel and principled probabilistic model is proposed which casts our task as an joint inference problem, taking into consideration of textual documents, social influences, and topic evolution in a unified way. Specifically, a mixture model is introduced to model the generation of text according to the diffusion and the evolution of the topic, while the whole diffusion process is regularized with user-level social influences through a Gaussian Markov Random Field.

Experiments on both synthetic data and real world data show that the discovery of topic diffusion and evolution benefits from this joint inference; and the probabilistic model we propose performs significantly better than existing methods.

Categories and Subject Descriptors: H.2.8 [**Information Systems Applications**]: Database Applications – *data mining*.

General Terms: Algorithm, Experimentation

*This work was supported in part by the U.S. NSF grant IIS-09-05215 and by the Army Research Laboratory accomplished under Cooperative Agreement Number W911NF-09-2-0053. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation stated here on.

The 5th SNA-KDD Workshop '11 August 21, 2011, San Diego, CA, USA.

Keywords: Information diffusion and evolution, Social networks, Topic modeling, Gaussian markov random field

1. INTRODUCTION

The prevailing of Web 2.0 techniques has led to the boom of various online communities. One of the core problems in analyzing such online communities is concerned with understanding the cascading behaviors and the diffusion of information. Epidemic diseases, adoption of innovation, memes of information, and many types of user actions all spread widely in these communities, following the social network of users. The modeling of information diffusion plays a crucial role in many domains. The contagion of disease forms the foundation of epidemics; the social influence in cascading behaviors has been a basic mechanism of viral marketing; and the diffusion of topics is essential to the understanding of scientific innovation. Recently, a large body of research work has been done in the field of social network analysis, aiming to describe the macro-level dynamics and characteristics of information diffusion [21, 13], reveal key factors that affect the adoption of behaviors [1, 22], and design contagion models that simulate the diffusion process [34, 47].

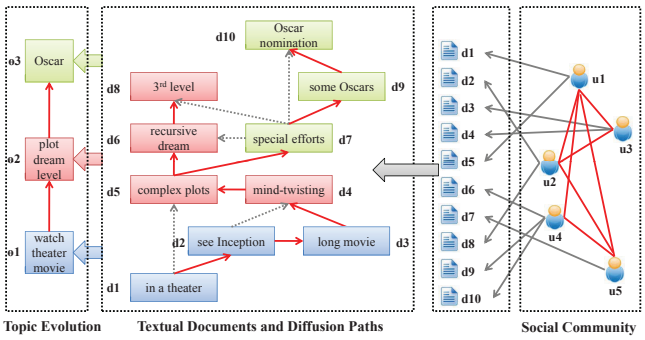


Figure 1: Example of Topic Diffusion and Evolution

Many conclusions in this line of work are motivated and validated in scenarios where the actual contagion/diffusion paths are observed. Such an assumption, which is considered as a common practice in user surveys and controlled user studies, does not apply to large scale online communities however. While the adoptions of behaviors are relatively easy to observe (based on which most macro-level descriptive statistics are computed), the evidence

of actual contagion and influence tend to be vague. Who infected whom? Who got the gossip from whom? Who influenced whose research? There are still substantial challenges in this micro-level analysis of information diffusion in large scale social networks. Indeed, users who joined a community or purchased an iPad usually won't explain which particular friends have influenced them; rumor spreaders tend to cover the source of the information; a researcher cites many references in her paper, without labeling the top three that have the most salient influence on her work. The identification of contagion is difficult even if the general social network structure is observed. It is a non-trivial task to detect the actual diffusion paths of user behaviors merely based on the time of adoption and the social network structure, known as the problem of diffusion (or influence) inference [9].

Inference of diffusion becomes even more challenging when the behaviors themselves are subtle. The adoption of explicit behaviors can be easily identified, for instance buying a DVD, joining a community, or using a hashtag in a tweet. Some behaviors are however implicit, such as writing about of a topic, holding an opinion, or having a particular mood. In this paper, we focus on the diffusion of topics in social communities. Inferring topic diffusion has introduced several additional challenges on top of the diffusion inference of explicit behaviors. First, topics are implicit and abstract concepts used in natural language. The adoption of topics cannot be directly identified, instead has to be inferred from the user-generated contents. Second, the meaning of a topic is evolving over time. A smart system should understand that *'MSN search'*, *'Live search'*, and *'Bing'* all refer to the same topic *'the Microsoft search engine'*, with unique aspects at different time; and it should be able to track and adapt to this content change in the the inference of topic diffusion. Third, information transmission is a complex social-psychological behavior [28], so the diffusion process of contents is inevitably influenced by the social relationships of the users. Moreover, the evolution and the diffusion of topics are compound processes: indeed, when a topic spreads from one user to another, new perspectives or new focus is introduced to the topic; and an outbreak of a topic is usually accompanied by a shift of the meaning of the topic. Although there has been a line of work on the diffusion inference of explicit behaviors recently [6, 11, 12, 20, 44, 15, 34, 16, 17, 6, 13, 1, 24, 22], none of this work addresses these challenges, making the existing methods incapable to accurately infer the diffusion paths of topics.

In this paper, we address these challenges by studying the joint inference of topic diffusion and evolution in social communities. Content and linkage in user-generated text information, together with social network structures, are used to facilitate the identification of topic adoption, the tracking of topic evolution, and the estimation of actual diffusion paths of any arbitrary topic. Our intuition is illustrated in Figure 1.

When a topic is introduced into the community by a user, other users read the documents she wrote (*e.g.*, tweets, blogs, scientific papers, *etc*) and adopt the topic by writing about it in their own articles. They may or may not cite the original document, or they may cite it together with other documents. Although topics are spread among documents instead of through social connections, we consider it is much more likely for users to adopt ideas from their social connections (*e.g.*, friends, people they follow, or people they have cited before) than from a stranger. Each document can not only adopt content from documents that influenced it, but also include novel perspectives into the topic, and pass on the *'innovation'* to other documents. The meaning of the topic is thus evolving over time. The goal of the joint inference of topic diffusion and topic evolution is to identify the *'real'* paths through which the topic propagates (red edges among documents in Figure 1), and also identify the time specific versions of the topic.

In this paper, we propose a novel statistical model for topic-based information diffusion and evolution (TIDE). Specifically, a mixture model is introduced to model the generation of text according to the diffusion and the evolution of the topic, while the whole diffusion process is regularized with user-level social influences through a Gaussian Markov Random Field. The discovery of novel aspects and the diffusion paths of the topic can be done by the joint inference of topic diffusion and evolution in TIDE.

Organization. The rest of this paper is organized as follows: Section 2 formally defines the problem of TIDE, as the solution of which a statistical model is proposed in Section 3. We present experiments and results in Section 4, discuss the related work in Section 5, and conclude in Section 6.

2. PROBLEM FORMULATION

In this section, we formally define the task of inferring the diffusion process and tracking the evolution of topics in social communities. We begin with a few key concepts as follows.

Definition 2.1. **Social Network**. A *network* is a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is a set of vertices and $\mathcal{E}$ is a set of edges among V. Particularly in a *social network*, a vertex corresponds to a user, and an edge $e = (i, j)$ stands for a connection (or a tie) between two users i and j. The strength of the tie (i, j) is defined as a non-negative value $g(i, j)$. An edge can be either *directed* or *undirected*.

Definition 2.2. **Document Collection**. A textual document d_i in a *document collection* $\mathcal{D} = \{d_i\}_{i=1}^{M}$ is defined as a bag of words from a fixed vocabulary $\mathcal{W} = \{w_k\}_{k=1}^{L}$. That is, $d_i = \{c(d_i, w_k)\}_{k=1}^{L}$, where $c(d, w)$ denotes the number of occurrences of word w in d.

Definition 2.3. **Social Community**. A *social community* is defined as the union of a social network $\mathcal{G}$ and a user-generated document collection $\mathcal{D}$, saying $\{\mathcal{G}, \mathcal{D}\}$. Each document $d_i \in \mathcal{D}$ is associated with an *author* a_i in $\mathcal{G}$ and a *time-stamp*s $t_i \in 1..T$.

Definition 2.4. **Topic**. A semantic *topic* θ observed in a particular time period is defined as a multinomial distribution of words $\{p(w|\theta)\}_{w \in \mathcal{W}}$ with the constraint $\sum_{w \in \mathcal{W}} p(w|\theta) = 1$.

Definition 2.5. **Theme**. We define a general and coherent *theme* discussed in a social community as a stream of time-stamped topics $\Theta = \{\theta_t\}_{t=0}^{T}$. We call θ_0 the *primitive topic*, which represents the original content of the theme prior to the discussions of the social community. $\theta_{t>0}$ are time variant versions of θ_0, which are gradually developed in the discussions of the social community. That is, θ_t is the snapshot of Θ at time t, which represents the novel aspect of the theme appearing at time t. Altogether Θ represents the origin and evolution of the contents of the theme over time.

While the text content of individual document can be explicitly observed, the general semantics of the time-variant topics and the adoption of the topic(s) in a document is implicit. What else remains implicit is the source adopted in a document. There could naturally be multiple sources: a document can be influenced by a few other documents, thus inherit the topic from those documents. The influence of some sources can be more salient than the influence of others. A document could also introduce original perspectives of the topic without being influenced by any existing document. The existence and strengths of the influence among docu-

ments assemble the actual diffusion process of the topic, which is formally defined as a *diffusion graph*.

Definition 2.6. **Diffusion Graph**. Given a theme Θ, we define a *diffusion flow* from one document d_j to another d_i ($t_j < t_i$) as the likelihood that d_i adopted the topic of Θ due to the influence of d_j. The strength of such a diffusion flow is denoted as a positive value $\pi_{i,j}$. Note that d_i can also introduces its novel perspective to Θ. In this case, we assume there is a diffusion flow into d_i from the time-stamped topic θ_{t_i}, with a strength $\pi_{i,\theta}$. Therefore, we define *diffusion vector* π_i as a vector of the strength of all the diffusion flows into d_i, *i.e.*, $\pi(i) = \{\pi_{i,j}\}_{d_j \in \mathcal{D}} \cup \{\pi_{i,\theta}\}$, with the constraint $\sum_{d_j \in \mathcal{D}} \pi_{i,j} + \pi_{i,\theta} = 1$. The union of diffusion flows into all documents in $\mathcal{D}$ assembles the *diffusion graph*, *i.e.*, $\Pi = \{\pi(i)\}_{d_i \in \mathcal{D}}$. Clearly, the graph Π is both weighted and directed.

Although the actual diffusion graph is unobserved, there are proxy networks that convey weaker signals in social communities. In many scenarios, a reference network (denoted as $\mathcal{R}$) of the documents can be observed, for example the citation network of scientific publications, the hyperlink network of blog articles, or the tweet network posted by follower-followees. Intuitively, the diffusion network should correlate well with such a reference network because a document is very likely to be influenced by documents it cites. However, the actual diffusion network could still be substantially different from $\mathcal{R}$, because many real influential references are covered up, and many explicitly cited ones do not represent the true influence. Another signal is the social network structure. An author is likely to follow the work of his social connections, such likely to adopt topics and ideas from the documents they generate [30, 40]. We call the set of documents pointing to d_i in the reference network as d_i's *reference set*, denoted as $r(i) \subset \mathcal{D}$. When no signal of citation or social communication is available, r_i can be simply instantiated as all documents with a time stampe prior to t_i. When such a reference network is available, we assume $\pi_{i,j} = 0$ if $j \notin r(i)$. Clearly, we also have $\pi_{i,i} = 0$.

Based on the definitions of concepts above, we can formalize the two major tasks of tracking **the diffusion and evolution of topics in social communities**. Given the input of a social community $\mathcal{G}$, a user-generated document collection $\mathcal{D}$, and the primitive topic θ_0 defining a theme, we aim to:

Task 1: Infer the Diffusion Graph. In this task, the goal is to discover the latent diffusion flow graph documents (and topics) (*i.e.* Π). The result of this task can be used to answer (i) *the source(s) of topic in a document*: to what extent the document is influenced by other documents, and (ii) *the degree of originality in a document*: how much novel perspetives the document introduces to the topic.

Task 2: Track Topic Evolution. In this task, the goal is to infer the time-variant versions of topics (*i.e.*, $\{\theta_t\}_{t=1}^{T}$) of a theme. By inferring Θ given θ_0, we expect to keep track of the new developments of the theme, understand its evolution over time, and better understand how it influences documents, *etc.*

The two tasks are challenging in many ways. First, although recently there are extensive studies on inferring the social influence on explicit behaviors at the user level [6, 42, 12, 11, 45], there is limited progress in the analysis of the influence on the adoption of latent topics at document level [20, 44]. Even though topic diffusion occurs at document level, the influence along the social network structure is playing an non-negligible role. There is however little existing wisdom on how to bridge document networks with social networks. The implicity nature of topics have made the problem even harder. Second, the inference of topic diffusion cannot be done independently to the tracking of topic evolution. Along with the diffusion process of the topic, new contents are introduced into the topic, making the semantics of the topic evolving over time. Without understanding the shift of topic contents, it is impossible to accurately detect the adoption of the topic in documents especially after a substantially long time. Moreover, since usually there are limited labeled examples, the solution model should be unsupervised. All of these challenges require us to propose a unified model that takes social connections, textual contents, influence among documents, and temporal information into consideration. In the following section, we propose such an integrative model and present the joint inference of topic diffusion and evolution.

3. PROPOSED MODELS

In this section, we propose a novel and integrative probabilistic model of Text-based Information Diffusion and Evolution (TIDE) in social communities. Based on TIDE, we present the joint inference of the diffusion graph and the evolution of arbitrary topics.

3.1 Intuitions and the General Model

The general model of TIDE is designed based on a few key observations in social communities.

Observation 1. *Diffusion and Contents*. When there is a significant diffusive flow between two documents, or there is a significant influence on one document on the other, the content of these two documents tend to be highly related. On the other hand, if two documents talk about different subjects, there is unlikely salient influence or significant diffusion flow between them even if one cites the other [36]. *W.l.o.g.*, we can assume that the content of a document depends on the documents which have influenced it.

Observation 2. *Diffusion and Social Connections*. Information transmission is a complex social-psychological behavior [28], *e.g.*, there exist persistency interests of users [32]. The diffusion process among documents is likely to be regularized by social connections of their authors. Indeed, an author is more likely to follow the work of her friends and thus adopt topics and ideas from a friend instead of from a random author. The diffusion flows among documents are thus dependent to the social network of authors.

Observation 3. *Diffusion and Evolution*. As the diffusion proceeds, both the semantics of the topic and the regularization effect of the social network of users evolve over time. If an aspect in a document never appear in any of its potential references (either papers it cites or all existing papers exposed to its author), it is likely to be original ideas introduced by the document, which contributes to the evolution of the general theme. Meanwhile, the strength of influence through old social connections would decay after a reasonably long time.

Given a collection of authored and time-stamped documents $\mathcal{D}$, a social community $\mathcal{G}$ of users who published these documents, and a primitive topic θ_0 representing the original semantics of a theme, we aim at inferring the latent stream of topics Θ and the diffusion graph Π. Based on our observations above, the task of TIDE is then cast as the joint inference of the posterior of Θ and Π:

Formally, our object becomes to infer:

$$P(\Pi, \Theta|\mathcal{G}, \mathcal{D}, \theta_0) \propto P(\Theta|\Pi, \mathcal{D}, \theta_0) \cdot P(\Pi|\mathcal{G}) \quad (1)$$

Based on our observations, here we assume that the generation of the diffusion graph (only) depends on the social network structure, while the evolution of topics depends on the documents, the diffusion process, and of course the original version of the topic. We denote the first component of Equation 1 as the *topic model* and the second as the *diffusion model*. Please note that although TIDE can

be easily extended to model the mixture of multiple topics (similar to LDA [3]), we only present the primitive case to model only one given topic. Our focus is to model the diffusion and evolution of any given topic instead of the discovery of multiple topics. We leave the modeling of multiple topics in our future work.

In the topic model, a *mixture model* is designed to extract the topic snapshots (time-variant versions) of the theme (Section 3.2). In the diffusion model, we introduce a *Gaussian markov random field* based on *graph projection* to model the dependency of diffusion flows on social connections (Section 3.3). Finally, the inference of the combined model is discussed in Section 3.4.

3.2 The Topic Model

It is difficult to directly compute the posterior of topics Θ. We make the following transformation such that

$$P(\Theta|\Pi, \mathcal{D}, \theta_0) \propto P(\mathcal{D}|\Theta, \Pi, \theta_0) \cdot P(\Theta|\theta_0), \tag{2}$$

where the introduction of new aspects to the topic (*i.e.*, the time-variant topic snapshots) does not depend on the diffusion flows.

We consider a typical generative process of $\mathcal{D}$: each document d_i is generated from a mixture model. When writing each word in d_i, one first chooses a component model from the mixture with a certain probability; once the component model θ is selected, a word is sampled according to the word distribution of θ.

We first introduce a background component model θ_B estimated from the entire collection that explains the generation of common English words in the document d_i. The rest component models are designed based on the diffusion flows. Specifically, we introduce a component model for each document d_j that could have potentially influenced d_i. There is a non-trivial diffusion flow from d_j to d_i, and d_i could inherit the topic of d_j according to the strength of this diffusion. These component models can be estimated simply using a maximum likelihood estimator on the corresponding d_j. Finally, we introduce a component model to explain the novel aspects introduced by the document d_i, *i.e.*, the aspect that is not influenced by any existing document. We assume that this aspect is generated directly from the latent topic at the time that d_i is written (θ_{t_i}). In other words, the original content is diffused from the topic directly to the document instead of from other documents. We assume that the probability of choosing each component is proportional to the strength of the diffusion vector, *i.e.*, $\pi(i)$.

Formally, the probability of generating a word w in d_i is:

$$\begin{aligned} &p(w|d_i) \\ &= (1-\lambda_B)(\sum_{j\in r(i)} \pi_{i,j} p(w|\theta_{d_j}) + \pi_{i,\theta} p(w|\theta_{t_i})) + \lambda_B p(w|\theta_B) \end{aligned}$$

where λ_B is a predefined parameter that fixes the sampling probability of the background model. Note that for documents $d_j \notin r(i)$, we have $\pi_{i,j} = 0$. The likelihood of the collection $\mathcal{D}$ is given as:

$$P(\mathcal{D}|\Pi, \Theta, \theta_0) = \prod_{d_i \in \mathcal{D}} \prod_{w \in \mathcal{W}} p(w|d_i)^{c(w,d_i)}$$

We then consider the generation of the time-variant versions of the topic, Θ. In TIDE, the primitive topic θ_0 is realized as a conjugate Dirichlet prior of the time-variant topic model θ_t: $Dir(\{1 + \mu_E p(w|\theta_0)\}_{w\in\mathcal{W}})$. By doing so, we regularize these time-variant topic snapshots so that they can reflect the novel aspects of the theme, but do not shift away from it. μ_E indicates how much we rely on the prior. Formally,

$$P(\Theta|\Pi) = \prod_{t\in 1..T} p(\theta_t|\theta_0) = \prod_{t\in 1..T} \prod_{w\in\mathcal{W}} p(w|\theta_t)^{\mu_E p(w|\theta_0)}$$

3.3 The Diffusion Model

Comparing to the modeling of topic evolution, the modeling of diffusion graph ($P(\Pi|\mathcal{G})$) is less straightforward. Intuitively, the diffusion graph Π should be regularized by the social network $\mathcal{G}$, as social influence plays an important role in topic diffusion. However, Π is a network of *documents* while $\mathcal{G}$ is a network of users. This makes it hard to model the regulation effect of $\mathcal{G}$ on Π. We need a bridge between the two heterogenous networks, for which we introduce the operation of *graph projection*.

Definition 3.1. **Graph Projection**. Let $\mathcal{G}_1$ and $\mathcal{G}_2$ be two graphs, a projection $f : \mathcal{G}_1|\mathcal{G}_2 \to \mathcal{G}_1'$ is called a *graph projection* if:

1. $\mathcal{V}(\mathcal{G}_1') = \mathcal{V}(\mathcal{G}_2)$.
2. $\forall v \in \mathcal{V}(\mathcal{G}_1'), \exists u \in \mathcal{V}(\mathcal{G}_1)\ s.t.\ v \in f(u)$.
3. $\forall e = (u, v) \in \mathcal{E}(\mathcal{G}_1'), \forall u' \in f(u)\ and\ v' \in f(v), e' = (u', v') \in \mathcal{E}(\mathcal{G}_1')$.

Through graph projection, two networks are endowed with the same vertex set, so that the comparison of them becomes more succinct and natural. Note that there are two asymmetric projection directions: 1) projecting $\mathcal{G}$ into a document network and using it as *a priori* of Π, or 2) projecting Π into a social network and consider the generation of such a social network based on $\mathcal{G}$. Since the document collection $\mathcal{D}$ is commonly much larger than the set of user $\mathcal{V}(\mathcal{G})$, projecting the document network into a social network is at inevitable risk of losing information. Although this doesn't rule out the second direction of graph projection, in this work we consider the first direction: the projection of $\mathcal{G}$ into a document network.

Let's denote Π' as the document network projected from $\mathcal{G}$, s.t.

$$P(\Pi|\mathcal{G}) = P(\Pi|\Pi') = P(\{\pi(i)\}_{d_i\in\mathcal{D}}|\Pi').$$

The remaining issue is how to fold the $\mathcal{G}$ into Π' and how to model the generation of Π based on Π'. Note that like Π, we can also denote $\Pi' = \pi'(i)_{d_i\in\mathcal{D}}$. We start with the generative model $P(\Pi|\Pi')$.

Gaussian Graphicl Models (GGM) [46] are classical models used to explain the generation of networks, which could be an ideal solution of our problem. In a typical GGM, each nodes in the graph is modeled as a random variable, for example a vector of k features. In our scenario, such a vector can be implemented as the diffusion vector $\pi(i)$. The joint distribution of all these variables (in our case, $P(\pi(i))$) is assumed to be a multivariate Gaussian. Each edge in Π' stands for the conditional dependency between two Gaussian variables, thus the graph structure Π' corresponds to the inverse covariance matrix.

However, the computational complexity of such a graphical model usually scales cubically with the number of variables, and therefore becomes intolerant even for a moderate size of dataset. To make our model practical, we introduce an independency assumption: the diffusion vector of one document is independent to the others. By doing so, we can simplify the generative model of Π as

$$P(\Pi|\Pi') = \prod_{d_i\in\mathcal{D}} P(\pi(i)|\pi'(i)) \tag{3}$$

Here $\pi'(i) = \{\pi'_{i,j}\}_{j\in r(i)} \cup \{\pi'_{i,\theta}\}$ is a conjugate prior vector, indicating the expected value of $\pi(i)$. Since Π' is projected from $\mathcal{G}$, $\pi'_{i,j}$ represents the social influence between a_j (the author of d_j) to a_i, which decays over time. By doing this, the document-level influence is regulated by the social tie at the user level.

Formally, we define $\pi'_{i,j} = \frac{1}{Z(\pi'(i))} g(a_i, a_j) \cdot e^{-\frac{t_i - t_j}{\alpha}}$ by consolidating an exponential time model with $\mathcal{G}$ [1]. Intuitively, doc-

[1]Other decay functions are also applicable [7].

uments with higher authority is likely to introduce more original content. We thus define $\pi'_{i,\theta} = \frac{1}{Z(\pi'(i))} Aut(a_i)$, where $Aut(a_i)$ is an estimation of the authority of d_i. $Z(\pi'(i))$ is a normalization factor such that $\sum_{d_j \in \mathcal{D}} \pi'_{i,j} + \pi'_{i,\theta} = 1$.

Given the design of $\pi' i)$, the computation of $P(\pi(i)|\pi'(i))$ is still non-trivial because of the dependency between the dimensions of $\pi(i)$. We introduce a *Gaussian Markov Random Field* [33] to model the conditional probability $P(\pi(i)|\pi'(i))$ for each d_i.

Definition 3.2. **Gaussian Markov Random Field (GMRF)** . A random vector $\S = (x_1, x_2, \cdots, x_n)^T$ is called a GMRF *w.r.t.*the graph $\mathcal{G} = (\mathcal{V} = \{1, 2, \cdots, n\}, \mathcal{E})$ with the mean μ and the precision matrix $\mathcal{Q}_\S$, iff the density of $\S$ has the form

$$P(\S) = (2\pi)^{-n/2} |\mathcal{Q}_\S|^{1/2} e^{-\frac{1}{2}(\S-\mu)^T \mathcal{Q}_\S (\S-\mu)}$$

and $\mathcal{Q}_\S(i,j) \neq 0 \Leftrightarrow (i,j) \in \mathcal{E}$ for all $i \neq j$.

In our case, the random vector is the diffusion vector $\pi(i)$, with the mean as the prior vector $\pi'(i)$. The precision matrix $\mathcal{Q}_{\pi(i)}$ corresponds to the similarities between the dimensions of $\pi(i)$ (documents and topic snapshots), which can be realized as the content similarities of corresponding θ_{d_j}'s and θ_t's. Computationally, $P(\pi(i)|\pi'(i))$ is defined as:

$$P(\pi(i)|\pi'(i)) \propto e^{-\frac{1}{2} \sum_{i',j' \in \{r(i)\} \cup \{\theta\}} (\pi_{i,i'} - \mu_{i,i'}) \mathcal{Q}_{\pi(i)}(i',j') (\pi_{i,j'} - \mu_{i,j'})}$$

3.4 Parameter Estimation

Given our model defined above, we can fit the model to the data and estimate the parameters using a Maximum A Posterior estimator [38]. Expectation Maximization (EM) algorithm [29] is applied, which iteratively computes a local maximum of the posterior. Computationally, the log likelihood we want to maximize is:

$$\begin{aligned} & E_{\Lambda^{(n-1)}} \{\log p(C|\Lambda) p(\Lambda)\} \propto \quad (4) \\ & \sum_{d_i, w, d_j \in r(i)} c(d_i, w)(1 - z^{(n)}_{d_i,w}(\theta_B)) z^{(n)}_{d_i,w}(\theta_{d_j}) \log((1-\lambda_B)\pi_{i,j} p(w|\theta_{d_j})) \\ & + \sum_{d_i, w} c(d_i, w)(1 - z^{(n)}_{d_i,w}(\theta_B)) z^{(n)}_{d_i,w}(\theta_{t_i}) \log((1-\lambda_B)\pi_{i,E} p(w|\theta_{t_i})) \\ & + \sum_{d_i, w} c(d_i, w) z^{(n)}_{d_i,w}(\theta_B) \log(\lambda_B p(w|\theta_B)) + \mu_E \sum_{\theta_t, w} p(w|\theta_0) \log p(w|\theta_t) \\ & - \frac{\mu_G}{2} \sum_{d_i} \sum_{i', j' \in \mathcal{N}(i)} (\pi_{i,i'} - \mu_{i,i'}) \mathcal{Q}_{pi(i)}(i', j') (\pi_{i,j'} - \mu_{i,j'}) \end{aligned}$$

Here μ_G is a weight combining two components, and we use terms $z_{d_i,w}(\cdot)$ instead of $p(z_{d_i,w} = \cdot)$ for better equation display.

In the E-Step, we compute the expectation of the hidden variables:

$$z^{(n)}_{d_i,w}(\theta_{d_j}) = \frac{\pi^{(n-1)}_{i,j} p(w|\theta_{d_j})}{\sum_{j' \in r(i)} \pi^{(n-1)}_{i,j'} p(w|\theta_{d_{j'}}) + \pi^{(n-1)}_{i,\theta} p(w|\theta_{t_i})}$$

$$z^{(n)}_{d_i,w}(\theta_{t_i}) = \frac{\pi^{(n-1)}_{i,\theta} p^{(n-1)}(w|\theta_{t_i})}{\sum_{j' \in r(i)} \pi^{(n-1)}_{i,j'} p(w|\theta_{d_{j'}}) + \pi^{(n-1)}_{i,\theta} p(w|\theta_{t_i})}$$

$$z^{(n)}_{d_i,w}(\theta_B) = \frac{\lambda_B p(w|\theta_B)}{(1-\lambda_B)(\sum_{j' \in r(i)} \pi^{(n-1)}_{i,j'} p(w|\theta_{d_{j'}}) + \pi^{(n-1)}_{i,\theta} p(w|\theta_{t_i})) + \lambda_B p(w|\theta_B)}$$

In the M-step, given the expectation of the hidden variables, we get the best parameters $p(w|\theta_t)$ as:

$$p(w|\theta_t) = \frac{\sum_{d_i, t_i = t} c(d_i, w)(1 - z^{(n)}_{d_i,w}(\theta_B)) z^{(n)}_{d_i,w}(\theta_t) + \mu_E p(w|\theta_0)}{\sum_{w'} \sum_{d_i, t_i = t} c(d_i, w')(1 - z^{(n)}_{d_i,w}(\theta_B)) z^{(n)}_{d_i,w'}(\theta_t) + \mu_E p(w'|\theta_0)}$$

By integrating Lagrange multipliers [29] f_i for each $d_i \in \mathcal{D}$, the inference of $\pi(i)$ boils down to solve a group of cubic equations:

$$\pi^2_{i,*} + \beta_{i,*} \pi_{i,*} + \gamma_{i,*} = 0, \quad * \in r(i) \cup \{\theta\} \quad (5)$$

where

$$\begin{aligned} \beta_{i,*} &= \frac{\sum_{*' \neq *} (\mathcal{Q}_{\pi(i)}(*,*') + \mathcal{Q}_{\pi(i)}(*',*))(\pi^{(n-1)}_{i,*'} - \mu_{i,*'})}{2\mathcal{Q}_{\pi(i)}(*,*)} \\ &- \mu_{i,*} + \frac{f_i}{\mu_G \mathcal{Q}(i)_{*,*}} \\ \gamma_{i,*} &= -\frac{\sum_w c(d_i, w)(1 - z^{(n)}_{d_i,w}(\theta_B)) z^{(n)}_{d_i,w}(\theta_{d_j})}{\mu_G \mathcal{Q}(i)_{*,*}} \end{aligned}$$

Let $\pi_{i,*} = \frac{-\beta_{i,*} + \sqrt{\beta^2_{i,*} - 4\gamma_{i,*}}}{2}$ be the root of Equation 5. It is easy to prove that $\pi_{i,*}$ can be arbitrarily close to zero when $f_i \to +\infty$ and arbitrarily large when $f_i \to -\infty$. Also, the derivative $\frac{\partial \pi_{i,*}}{\partial f_i} = \frac{1}{2}\left(-1 + \frac{\beta_{i,*}}{\sqrt{\beta^2_{i,*} - 4\gamma_{i,*}}}\right) < 0$. Hence, it is guaranteed that there exist valid solutions for the group of equations that satisfy the constraint $\sum_{* \in r(i) \cup \{\theta\}} \pi_{i,*} = 1$ for each d_i in $\mathcal{D}$.

4. EXPERIMENTS

In this section, we evaluate the effectiveness of our TIDE model on synthetic datasets as well as data collected from two real-world social communities, *i.e.*, *DBLP* [43] and *Twitter* [25].

4.1 Experimental Setup

4.1.1 Data Collections

The DBLP Dataset ([43]). The Digital Bibliography and Library Project (DBLP) is a web accessible database of the bibliographic information of computer science publications. In this experiment, we use a collection of DBLP articles augmented with citation information, released by the ArnetMiner group [2], which contains $1,632,442$ publications by $1,741,170$ researchers with $2,327,450$ citations. After filtering out papers without text or citation information, $243,425$ papers and $246,839$ authors are retained. This dataset represents a typical academic community, with a social network of authors (with coauthoring and citation relations) and a collection of scientific papers.

The Twitter Dataset ([25]). Twitter is a well known social networking and microblogging community. In this experiment, the Twitter dataset was crawled down by the DAIS group at University of Illinois , which contains $5,000$ socially connected users and their most recent 200 tweets posted before Nov. 23, 2010. Totally, there are $103,968$ oneway following relations, and $51,032$ pairs of friends (mutual following relations). This dataset represents a typical social community with a directed social network (defined by following relations) and a collection of tweets.

[2]http://arnetminer.org/DBLP_Citation

Synthetic Dataset. The lack of ground truth on real world dataset makes it hard to evaluate the model performance quantitatively. To achieve quantitative evaluation, we construct a synthetic dataset which simulates the diffusion of $1,000$ themes. For each theme, we extract a subgraph of $1,000$ authors from the *DBLP* dataset using breath first search from a random seed author. This subgraph is used to simulate the social network in which the theme diffuses. We then randomly attach $1,859$ empty and time-stamped documents to the authors in this network [3]. We then simulate a diffusion graph of the $1,859$ documents that is regularized by the simulated social network structure. Specially, we first randomly generate a network of the $1,859$ documents using Erdos/Renyi model, with the average degree of 5 (consistent with the real statistics in the DBLP dataset). The direction of each edge is determined by the time stamps of the documents (always points to a "newer" document). We then weight each edge based on the social connections of the authors of the two document plus a random effect. This directed and reweighed random network simulates the real diffusion network among documents. For each theme, we also simulate a sequence of 10 evolving topic snapshots based on the dynamic topic models [2]. Finally, the text content of each document is generated by a simple mixture model with all documents that have "influenced" this document as well as the corresponding topic snapshot.

4.1.2 Baselines

The NetInf Model [11]. NetInf is a typical model that infers the diffusion network of explicit user behaviors. Given the time stamps at which individuals adopt a behavior, *NetInf* identifies the optimal general network of users that best explains the observed adoptions. Comparing to TIDE, *NetInf* is trying to infer the general social network structure according to the observation of the propagations of a group of events, while *TIDE* infers the theme-specific diffusion graph with the help of a general social network. Note that *NetInf* doesn't consider text information, thus cannot track topic evolution.

If we treat each term with a positive probability in the primitive topic as an explicit event/behavior, then a document adopts that behavior explicitly if the term appears in the document. We are then able to infer the optimal document network using *NetInf*. This optimal network is easily converted into a diffusion graph by endowing each edge with equal flow volume.

The IndCas Model [34]. The second baseline is a deviation of the independent cascade model stated in [34], where the probability for an active document to infect another is proportional to the strength of the social connection between their authors with an exponential decay effect [7] (see Section 3.3). We convert these probabilities into a diffusion graph where the diffusion flow from d_j to d_i is proportional to the probability that d_j infects d_i.

The TIDE- Model. To evaluate the effectiveness of social connections in our models, we implement a special version of *TIDE* by removing the regularization term with the network structure, *i.e.*, by setting $\mu_G = 0$.

We believe *NetInf* and *IndCas* are good representatives of diffusion inference models of explicit behaviors, which do not consider textual information or the evolution of topics. *TIDE-* on the other hand ignores the effects of social connections.

4.2 Experiments on Synthetic Data

The goal of the experiments on synthetic data is to quantitatively evaluate how well each method can (i) infer diffusion graphs, (ii) estimate contribution of novelty (if possible), and (iii) discover snapshots in topic evolution (if possible). Given the simulated social community (the social network, the document collection, and the primitive topic), our goal is to recover the diffusion graph and the topic snapshots. The parameters in the *TIDE* model are set empirically as $\mu_E = 10$, $\alpha = 30$, and $\mu_G = 10$.

4.2.1 Analysis on Information Diffusion

We first look at how successful models are at inferring diffusion graphs. Let us first introduce the evaluation metrics.

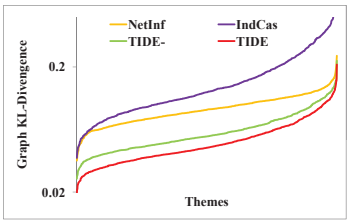
Definition 4.1. **Graph KL-Divergence**. The symmetrized Kullback-Leibler divergence [19] is a classic measure of the difference between two probability distributions. We extend the SKLD and define an evaluation metric to measure the discrepancy between two diffusion graphs $\Pi_{\mathcal{P}}$ and $\Pi_{\mathcal{Q}}$ on the same document collection $\mathcal{D}$:

$$GD_{KL}(\Pi_{\mathcal{P}}, \Pi_{\mathcal{Q}}) = \frac{\sum_{d_i \in \mathcal{D}} (D_{KL}(\pi_{\mathcal{P}}(i)||\pi_{\mathcal{Q}}(i)) + D_{KL}(\pi_{\mathcal{Q}}(i)||\pi_{\mathcal{P}}(i)))}{2|\mathcal{D}|}$$

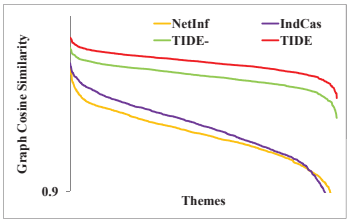
Definition 4.2. **Graph Cosine Similarity**. We also define a metric of similarity between two diffusion graphs $\Pi_{\mathcal{P}}$ and $\Pi_{\mathcal{Q}}$, as the average cosine similarity [41] between their diffusion vectors.

$$Cos(\Pi_{\mathcal{P}}, \Pi_{\mathcal{Q}}) = \frac{1}{|\mathcal{D}|} \sum_{d_i \in \mathcal{D}} \frac{\pi_{\mathcal{P}}(i) \cdot \pi_{\mathcal{Q}}(i)}{||\pi_{\mathcal{P}}(i)|| \cdot ||\pi_{\mathcal{Q}}(i)||}$$

A better model should infer a diffusion graph that is closer to the "ground truth" (the simulated diffusion network), that is, a lower KL-divergence and a higher Cosine similarity.



(a) Effectiveness (Divengence)



(b) Effectiveness (Similarity)

Figure 2: Diffusion Evaluation on the Synthetic Dataset

In practice, we calculate the two metrics for the result of each method and each theme[4], and connect the KL-divergence scores in decreasing order (Figure 2(a)) and Cosine-similarity scores in increasing order (Figure 2(b)). The aggreated performance of the $1,000$ themes is reported in the 1^{st} and 2^{nd} columns of Table 1.

[3] According to the statistics on our *DBLP* datsset, each researcher has 1.859 first-authored publications in average.

[4] To make the results from all methods comparable, vertices associated with topic snapshots are removed from the diffusion graphs inferred by *TIDE* and *TIDE-*.

We can conclude that *TIDE* achieves the best performance, then *TIDE-*, then *NetInf*, and then *IndCas*.

4.2.2 *Proof of Combined Power*

With this experiment, we can also prove that both social networks and text information play an important role the inference of topic diffusion.

Object	TDG		$\mathcal{H}$		$\mathcal{G}$	
Metric	GKLD	GCS	GKLD	GCS	GKLD	GCS
NetInf	0.0936	0.9359	1.2906	0.8685	0.9490	0.8022
IndCas	0.1601	0.9313	1.2971	0.8550	**0.7210**	**0.8975**
TIDE–	0.0628*	0.9691*	**0.9906**	**0.9494**	0.8757	0.8407
TIDE	**0.0524***	**0.9722***	1.0109	0.9378	0.8459	0.8547

Table 1: Diffusion Evaluation on the Synthetic Dataset (TDG = True Diffusion Graph, GKLD = Graph Kullback-Leibler Divergence, GCS = Graph Cosine Similarity)

We measure the statistical significance of the improvement using the dependent t-test. * means that the improvement (over the row above) hypothesis is accepted at significance level 0.001.

First, we create a document network (denoted as $\mathcal{H}$), where the edge weight is proportional to the content similarities between documents. We compare each inferred diffusion graph with $\mathcal{H}$, and report the aggregated value of the two metrics in the 3^{rd} and 4^{th} columns of Table 1.

Second, we project each diffusion graph Π into a user network (denoted as $f(\Pi)$), compare $f(\Pi)$ with the general social network $\mathcal{G}$, and report the aggregated value of the two metrics in the last two columns of Table 1.

We can observe some phenomena that accord with our hypothesis in designing our model: *TIDE-* infers diffusion graphs only considering textual information without considering the social network structure, while *IndCas* infers the diffusion network purely based on the social influences. Indeed, the diffusion networks inferred by *TIDE-* are significantly biased towards the document similarity networks $\mathcal{H}$, and the diffusion networks inferred by *IndCas* are biased towards the social networks $\mathcal{G}$. **Neither of them infers diffusion networks that are closer to the ground truth than TIDE, which employs both text information and the social network**.

4.2.3 *Analysis on Content Evolution*

In this experiment, we study how successfully TIDE and the baseline models track topic evolution. Since *NetInf* and *IndCas* are not able to handle topics, we compare our models *TIDE* and *TIDE-* with a simple mixture model stated in [48].

We repeat similar experiments as done in Section 4.2.1. We use two similar metrics (*i.e.*, the symmetrized KL-divergence and the Cosine similarity) to measure the closeness of the discovered word distributions of the topic snapshots to the "ground truth" (topic snapshots we construct in the synthetic dataset). The results are reported in Table 4.

As shown above, *TIDE* outperforms the other two methods with sufficient certainty, which proves our statement in Section 1: **the evolution and the diffusion of topics are compound processes; the success of one aspect will help the inference of the other**.

4.3 Experiments on Real Social Networks

Metric	KLD	CS
FM	0.4281	0.7033
TIDE-	0.3301*	0.8622*
TIDE	**0.2893***	**0.8774***

Table 4: Evolution Evaluation on the Synthetic Dataset (KLD = Kullback-Leibler Divergance, CS = Cosine Similarity, FM = Feedback Model [48])

* means the improvement (over the above row) hypothesis is accepted at the significance level 0.001 based on dependent t-test.

We present the experiments on real world social communities in this section. Note that "ground truth" diffusion networks and topic snapshots are usually not available.

4.3.1 *Verifying Motivating Observations*

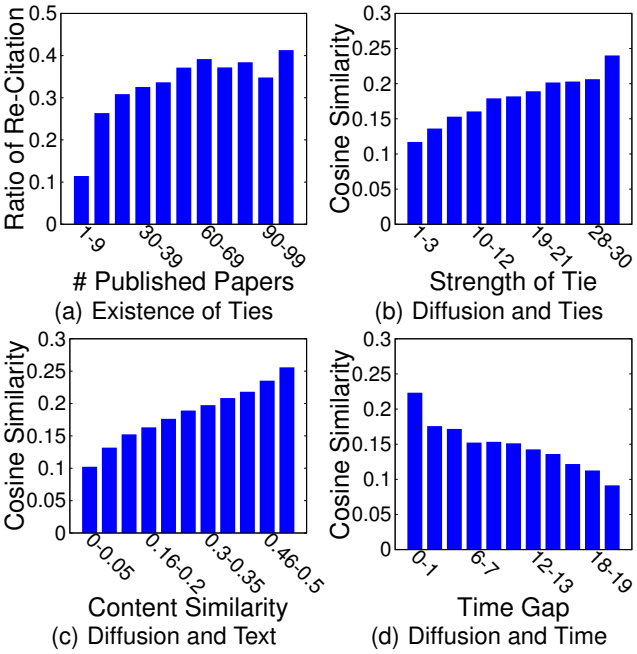


Figure 3: Verifying Observations by DBLP-Citation Dataset

We start with the verification of the authenticity of the three motivating observations stated in Section 3.1. We expect that social influence, so that an author is more likely to adopt topics from the documents of her social connections. If this is the case, an author will pay consistent attention to papers published by authors she knows, or she has cited before. One intuitive way to verify this is through the behavior of 're-citation', *i.e.*, once the author cited one paper, it is likely that she will cite the paper of the same author again. We group authors by the number of publications, and plot the average ratio of re-citation in Figure 3(a). It shows that there are substantial re-citation behaviors, when an author publish more papers, the ratio of re-citation also grows. This verifies the existence of social influence in document-level information diffusion.

Instead of inferring the diffusion, a rough proxy of the influence of a cited paper d_r on a citing paper d_c can be measured by the author's behaviors after the citation. Generally, if the authors of d_c

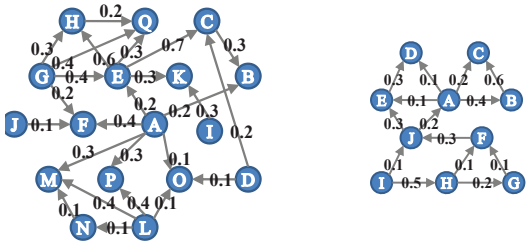
ID	Publication	ID	Publication
A	J. Han, *etc*, SIGMOD'00.	B	A. Khan, *etc*, KDD'10.
C	X. Yan, *etc*, SIGMOD'04.	D	M. Zaki, *etc* KDD'03.
E	X. Yan, *etc*, KDD'03.	F	Y. Chi, *etc*, TKDE'05.
G	M. Zaki, KDD'02.	H	A. Bifet, *etc*, KDD'08.
I	X. Yan, *etc*, KDD'05.	J	U. Rükert, *etc*, SAC'04.
K	C. Chen, *etc*, CIKM'08.	L	J. Wang, *etc*, KDD'03.
M	J. Wang, *etc*, TKDE'05.	N	F. Pan, *etc*, KDD'03
O	A. Lee, *etc*, Infomation System'10.		
P	U. Yun, Knowledge-Based System'08		
Q	J. Balcázar, *etc*, Machine Learning'10.		

Table 2: Publications Shown in Figure 4(a)

ID	Tweet (incomplete)
A	Inception had better special effects than Videodrome.
B	Inception's effects might take some Oscars.
C	I predict Inception's 12 Oscar nominations.
D	It has to be like a 3rd level Inception dream.
E	I wonder what level of recursive dreams.
F	Inception. What a brilliant, mind-twisting movie.
G	Watching inception. Long movie.
H	You'd be odd on twitter if you haven't seen Inception.
I	First time I have seen a movie in a theater in the last 6 months.
J	If you like intelligent movies and complex plots, go to see Inception.

Table 3: Tweets Shown in Figure 4(b)

publish many papers related to d_r after they publish d_c, it is fair to believe d_r is quite influential to d_c. We partition the citations (each of which is recognized by a cited paper d_r and a citing paper d_c) into different groups according to the strength of the social connection between their authors. For each citation, we then compute the average document similarity between d_r and all papers published by d_c's authors after they had published d_c. The aggregated similarity is plotted in Figure 3(b). We repeat the same experiment, but partitions citations by degrees of the content similarity of d_r and d_c (Figure 3(c)), as well as the time gap (3(d)) between d_r and d_c. Figure 3(b)-3(d) prove our motivations that the (proxy) influence between two documents increases with the strength of social ties (Observation 2) and the content similarity, but decays over time (Observation 3).



(a) Theme 1 (DBLP-Citation) (b) Theme 2 (Twitter)

Figure 4: Case Study on Real Networks: Diffusion Graphs

4.3.2 *Case Study*

We select two themes for case study: one is about the research topic *'frequent pattern mining'* on the DBLP-Citation dataset (see the 1^{st} column of Table 5), and the other is about the movie *'inception'* on the Twitter dataset (see the 1^{st} column of Table 7).

Analysis on Information Diffusion. For theme 1, we apply the *TIDE* model on 344 papers published during the past ten years (2000 to 2010), which contain at least three primitive keywords in the title or abstract. A subgraph of the diffusion graph estimated by TIDE is shown in Figure 4(a), on a subset of 17 selected papers (listed in Table 2). The volume of each diffusion flow is marked on the edge. To quantitatively access the result, we compare the graph with three alternative "diffusion graphs." In the first graph, the weight of an edge $d_r \rightarrow d_c$ is set proportional to the total length of citation sentences where d_c mentions d_r. We then employ two experts to manually score the impact of each reference paper in a scale from one to five. The **Mean Absolute Error** [35], as the statistical metric of accuracy, based on each criteria, and the **Cohen's Kappa Coefficiency** [37], as the measure of inter-criteria agreement, are reported in Table 10.

Theme 2 has been used as the running example in Section 1 (see Figure 1), and let us reveal more details. We apply the *TIDE* model on 361 tweets containing the keyword *'inception'*, and draw the diffusion graph on 10 selected tweets (listed in Table 3) in Figure 4(b). We repeat the same evaluation procedure as done for theme 1 (see Table 11), only except that the edge weight of the first criteria graph is decided by whether one tweet was replying the other.

MAE	SL	Exp1	Exp2
TIDE	0.1217	**0.1080**	0.1195

CKC	Exp1	Exp2
SL	0.5019	0.2095
Exp1	–	**0.6333**

Table 10: Theme 1 (DBLP-Citation)

MAE	RR	Exp1	Exp2
TIDE	0.3632	**0.1301**	0.1351

CKC	Exp1	Exp2
RR	0.3583	0.3726
Exp1	–	**0.7500**

Table 11: Theme 2 (Twitter)

Table 12: Case Study on Real Networks: Accuracy Evaluation (MAE = Mean Absolute Error, CKC = Cohen's Kappa Coefficiency, SL = Sentence Length, RR = Replying Relation)

In both cases, the opinions of the first expert gains the most agreement from others, and our result has the highest accuracy against the truth ground supplied by the first expert.

Analysis on Content Evolution. We apply both *TIDE* and the feedback model [48] to extract the topic snapshots for two themes. Top words (with probabilities) of several selected topics are listed in Table 5-8. Note, to demonstrate more results, the word *'fre-*

Primitive Topic		Year 2003		Year 2005		Year 2009	
frequent	0.20	itemset	0.05	itemset	0.04	itemset	0.03
pattern	0.40	GSM	0.03	tree	0.02	tree	0.02
mining	0.20	association	0.02	parallel	0.01	sequence	0.01
graph	0.05	apriori	0.02	graph	0.01	graph	0.01
tree	0.05	tree	0.01	sequence	0.01	slide	0.01
sequence	0.05	graph	0.01	traversal	0.01	gram	0.01
itemset	0.05	subgroup	0.01	optimize	0.01	window	0.01
		sequential	0.01	suffix	0.01	apriori	0.01

Table 5: Topic Snapshots by *TIDE* on Theme 1 (DBLP-Citation)

Year 2003		Year 2005		Year 2009	
efficient	0.02	close	0.01	sequential	0.02
close	0.01	itemset	0.01	itemset	0.01
association	0.01	match	0.01	tree	0.01
support	0.01	tree	0.01	graph	0.01
query	0.01	graph	0.01	database	0.01
temporal	0.01	sequential	0.01	efficient	0.01
graph	0.01	efficient	0.01	rule	0.01
rule	0.01	application	0.01	match	0.01

Table 6: Topic Snapshots by [48] on Theme 1

Primitive Topic		Jul 16-19		Jul 20-23		Jul 24-27	
inception	1.00	watch	0.05	dream	0.06	oscar	0.04
		night	0.05	mind	0.05	effect	0.04
		movie	0.05	level	0.03	dream	0.02
		special	0.03	walk	0.01	clever	0.01
		enjoy	0.01	recursive	0.01	briliant	0.01

Table 7: Topic Snapshots by *TIDE* on Theme 2 (Twitter)

Jul 16-19		Jul 20-23		Jul 24-27	
movie	0.06	type	0.05	oscar	0.03
night	0.06	eye	0.05	act	0.03
special	0.03	watch	0.05	dream	0.03
watch	0.03	night	0.05	strong	0.02
bad	0.03	dream	0.04	night	0.02

Table 8: Topic Snapshots by [48] on Theme 2

Table 9: Case Study on Real Networks: Topic Evolution

quent', *'pattern'* and *'mining'* are eliminated from Table 5 and 6; and the word *'inception'* are eliminated from Table 7 and 8.

As elaborated in Section 3.2, the topic component of *TIDE* utilize (i) a background model to absorb common words, and (ii) reference models to absorb old words, so that topic snapshots would attract more discriminative and meaningful words that describe the novel aspect of a theme. For example, the topic at *'Year 2009'* in Table 6 reveals a new trend of mining patterns up to certain length (*i.e.'gram'*) in a *'sliding' 'window'*, and the topic at *'Jul 20-23'* in Table 8 talks about the movie plots such as the *'level'* of a *'recursive' 'dream'*. However, since [48] only considers the idea of background model, these interesting new words are easily overlooked, because antiquated words, such as *'efficient'* in Table 5 and *'watch'* in Table 7, repeatly appear in lots of topic snapshots.

5. RELATED WORK

TIDE is a novel probabilistic model for the joint inference of diffusion and evolution of topics, by comprehensively considering 1) the generation of text, 2) the effects of social networks, 3) contribution of novelty, and 4) the topic evolution. To the best of our knowledge, there is no existing model that considers all these factors in a unified and principled way. There are, however, several lines of related work.

Information Diffusion. Information diffusion [1, 13, 18, 44, 24, 9, 21, 22] is a classic topic in social network analysis, which models the cascade of behaviors on a network structure. [6, 12, 20, 17] aim at a subset of nodes or links in a network that could maximize (or minimize) the spread. [16, 42, 45, 27] estimate social graphs with edges labeled with probabilities of influence between users. [11] infers a latent network structure, over which events spread well. [18] predicts the sharing scale of an opinion. This line of work, however, usually do not consider textual topics evolving along time, and thus hard to be applied to our tasks.

Topic Modeling. Topic modeling approaches [14, 3] have been developed to mine variations of topics [39, 23, 8, 31, 10]. Specially, incorporating network regularization into topic modeling has become a popular tendency, such as NetPLSA [30], Laplacian PLSI [4], iTopicModel [40] and locally-consistent Models [5, 26]. [30] uses a harmonic function to enforce the constraint that topic distribution on neighboring nodes should be similar, [40] defines a Markov Random Field on the graph to model the influence between nodes in a generative way, and [25] leverages Gibbs Random Field to estimate the popularity index of a textual topic depending on social connections and history. [45, 42] model the influence graph among users by considering both network structures and topics. However, none of these methods aim to infer diffusion paths among documents. Thus they can not be directly applied to our problem.

6. CONCLUSION

In this paper, we propose *TIDE*, a novel probabilistic model for the joint inference of diffusion and evolution of topics in social communities. TIDE integrates the generation of text, the evolution of topics, and the social network structure in a unified model. Given the primitive form of any arbitrary topic, *TIDE* effectively tracks the topic snapshots that evolves along time and reveals the latent diffusion paths of the topic. Comprehensive experiment studies on both synthetic data and two real-world datasets show that TIDE outperforms existing approaches.

One important finding is that the discovery of topic diffusion and topic evolution benefits significantly from the joint inference process. Social influence still plays an important role in the diffusion of topics. Both text information and the general social network structure play an irreplaceable role to the inference process. We expect a future extension of TIDE that model the evolution of the social network structure in addition to the evolution of topics.

7. REFERENCES

[1] L. Backstrom, D. P. Huttenlocher, J. M. Kleinberg, and X. Lan. Group formation in large social networks: membership, growth, and evolution. In *KDD*, pages 44–54, 2006.

[2] D. M. Blei and J. D. Lafferty. Dynamic topic models. In *ICML*, pages 113–120, 2006.

[3] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. In *NIPS*, pages 601–608, 2001.

[4] D. Cai, Q. Mei, J. Han, and C. Zhai. Modeling hidden topics on document manifold. In *CIKM*, pages 911–920, 2008.

[5] D. Cai, X. Wang, and X. He. Probabilistic dyadic data analysis with local and global consistency. In *ICML*, page 14, 2009.

[6] W. Chen, C. Wang, and Y. Wang. Scalable influence maximization for prevalent viral marketing in large-scale social networks. In *KDD*, pages 1029–1038, 2010.

[7] R. Crane and D. Sornette. Robust dynamic classes revealed by measuring the response function of a social system. *the National Academy of Sciences*, 105(41):15649-15653, Oct 2008.

[8] L. Dietz, S. Bickel, and T. Scheffer. Unsupervised prediction of citation influences. In *ICML*, pages 233–240, 2007.

[9] D. Easley and J. Kleinberg. Networks, crowds, and markets: Reasoning about a highly connected world. *Cambridge University Press*, 2010.

[10] S. Gerrish and D. M. Blei. A language-based approach to measuring scholarly impact. In *ICML*, pages 375–382, 2010.

[11] M. Gomez-Rodriguez, J. Leskovec, and A. Krause. Inferring networks of diffusion and influence. In *KDD*, pages 1019–1028, 2010.

[12] A. Goyal, F. Bonchi, and L. V. S. Lakshmanan. Learning influence probabilities in social networks. In *WSDM*, pages 241–250, 2010.

[13] D. Gruhl, R. V. Guha, D. Liben-Nowell, and A. Tomkins. Information diffusion through blogspace. In *WWW*, pages 491–501, 2004.

[14] T. Hofmann. Probabilistic latent semantic indexing. In *SIGIR*, pages 50–57, 1999.

[15] D. Kempe, J. M. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *KDD*, pages 137–146, 2003.

[16] M. Kimura, K. Saito, and H. Motoda. Minimizing the spread of contamination by blocking links in a network. In *AAAI*, pages 1175–1180, 2008.

[17] M. Kimura, K. Saito, and R. Nakano. Extracting influential nodes for information diffusion on a social network. In *AAAI*, pages 1371–1376, 2007.

[18] M. Kimura, K. Saito, K. Ohara, and H. Motoda. Learning to predict opinion share in social networks. In *AAAI*, 2010.

[19] S. Kullback and R. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22(1):79–86, 1951.

[20] C. Lee, H. Kwak, H. Park, and S. B. Moon. Finding influentials based on the temporal order of information adoption in twitter. In *WWW*, pages 1137–1138, 2010.

[21] J. Leskovec, L. A. Adamic, and B. A. Huberman. The dynamics of viral marketing. *TWEB*, 1(1), 2007.

[22] J. Leskovec, M. McGlohon, C. Faloutsos, N. S. Glance, and M. Hurst. Patterns of cascading behavior in large blog graphs. In *SDM*, 2007.

[23] W. Li and A. McCallum. Pachinko allocation: Dag-structured mixture models of topic correlations. In *ICML*, pages 577–584, 2006.

[24] D. Liben-Nowell and J. Kleinberg. Tracing the flow of information on a global scale using internet chain-letter data. *the National Academy of Sciences*, 105(12):4633–4638, 2008.

[25] C. X. Lin, B. Zhao, Q. Mei, and J. Han. Pet: a statistical model for popular events tracking in social communities. In *KDD*, pages 929–938, 2010.

[26] J. Liu, D. Cai, and X. He. Gaussian mixture model with local consistency. In *AAAI*, 2010.

[27] L. Liu, J. Tang, J. Han, M. Jiang, and S. Yang. Mining topic-level influence in heterogeneous networks. In *CIKM*, pages 199–208, 2010.

[28] B. R. M. M. H. MacRoberts. Problems of citation analysis. *Scientometrics*, 36(3):435–444, 1996.

[29] G. McLachlan and T. Krishnan. The em algorithm and extensions. *Wiley series in probability and statistics, Hoboken*, 2008.

[30] Q. Mei, D. Cai, D. Zhang, and C. Zhai. Topic modeling with network regularization. In *WWW*, pages 101–110, 2008.

[31] Q. Mei and C. Zhai. Discovering evolutionary theme patterns from text: an exploration of temporal text mining. In *KDD*, pages 198–207, 2005.

[32] F. Qiu and J. Cho. Automatic identification of user interest for personalized search. In *WWW*, pages 727–736, 2006.

[33] H. Rue and L. Held. Gaussian markov random fields: Theory and applications - theory and application. *Chapman and Hall/CRC*, 2006.

[34] K. Saito, M. Kimura, K. Ohara, and H. Motoda. Selecting information diffusion models over social networks for behavioral analysis. In *ECML/PKDD (3)*, pages 180–195, 2010.

[35] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *WWW*, pages 285–295, 2001.

[36] A. Si, H. V. Leong, and R. W. H. Lau. Check: a document plagiarism detection system. In *SAC*, pages 70–77, 1997.

[37] N. Smeeton. Early history of the kappa statistic. *Biometrics*, 41:795, 1985.

[38] H. W. Sorenson. Parameter estimation: Principles and problems. *Marcel Dekker*, 1980.

[39] M. Steyvers, P. Smyth, M. Rosen-Zvi, and T. L. Griffiths. Probabilistic author-topic models for information discovery. In *KDD*, pages 306–315, 2004.

[40] Y. Sun, J. Han, J. Gao, and Y. Yu. itopicmodel: Information network-integrated topic modeling. In *ICDM*, pages 493–502, 2009.

[41] P. Tan, M. Steinbach, and V. Kumar. Introduction to data mining. volume ISBN:0321321367, 2005.

[42] J. Tang, J. Sun, C. Wang, and Z. Yang. Social influence analysis in large-scale networks. In *KDD*, pages 807–816, 2009.

[43] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su. Arnetminer: extraction and mining of academic social networks. In *KDD*, pages 990–998, 2008.

[44] X. Wan and J. Yang. Learning information diffusion process on the web. In *WWW*, pages 1173–1174, 2007.

[45] J. Weng, E.-P. Lim, J. Jiang, and Q. He. Twitterrank: finding topic-sensitive influential twitterers. In *WSDM*, pages 261–270, 2010.

[46] J. Whittaker. Graphical models in applied multivariate statistics. In *Wiley Publishing*, 2009.

[47] J. Yang and J. Leskovec. Modeling information diffusion in implicit networks. In *ICDM*, pages 599–608, 2010.

[48] C. Zhai and J. D. Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *CIKM*, pages 403–410, 2001.

Multi-Step Community Detection and Hierarchical Time Segmentation in Evolving Networks

Thomas Aynaud
UPMC Univ Paris 06, UMR 7606, LIP6, F-75252, Paris, France
thomas.aynaud@lip6.fr

Jean-Loup Guillaume
UPMC Univ Paris 06, UMR 7606, LIP6, F-75252, Paris, France
jean-loup.guillaume@lip6.fr

ABSTRACT

Many complex systems composed of interacting objects like social networks or the web can be modeled as graphs. They can usually be divided in dense sub-graphs with few links between them, called *communities* and detecting this underlying community structure may have a major impact in the understanding of these systems. We focus here on evolving graphs, for which the usual approach is to represent the state of the system at different time steps and to compute communities independently on the graph obtained at each time step.

We propose in this paper to use a different framework: instead of detecting communities on each time step, we detect a unique decomposition in communities that is relevant for (almost) every time step during a given period called the *time window*. We propose a definition of this new decomposition and two algorithms to detect it quickly. We validate both the approach and the algorithms on three evolving networks of different kinds showing that the quality loss at each time step is very low despite the constraint of maximization on several time steps.

Since the time window length is a crucial parameter of our technique, we also propose an unsupervised hierarchical clustering algorithm to build automatically a hierarchical time segmentation into time windows. This clustering relies on a new similarity measure based on community structure. We show that it is very efficient in detecting meaningful windows.

Categories and Subject Descriptors

[**Data**]: Social networks, Graphs; [**Algorithms/Models**]: Clustering, Classification; [**Application Area**]: Bioinformatics, Web and ecommerce, Humanities and social sciences

General Terms

Community detection, evolving graphs, social networks, complex networks, web, clustering

The 5th SNA-KDD Workshop '11 (SNA-KDD'11), August 21, 2011, San Diego CA USA.

Introduction

Many complex systems are modeled by graphs, or complex networks, which are particularly suited for representing large groups of interacting objects. For instance, the web is a graph of web pages interacting through hyperlinks and the brain is a graph of neurons connected through synapses. Even if they model systems of very different types, these graphs share common non-trivial properties like a low average distance and a small global density balanced with a high local density. This last property means that there are few links inside these graphs but that links are grouped, which suggests that there exist groups of nodes with many links inside the groups but few between them. These groups of nodes are called *communities* and finding and understanding this underlying structure may have a major impact in the global understanding of the modeled systems. For instance, the detection of these communities can be used for graph visualization [2] or to mine various kinds of graphs: they can be groups of interest in a social network [4, 33], web pages dealing with the same subject [10], proteins that share a common function in a metabolic network [36] or modules in a software source code [18].

In the last decade, many definitions have formalized this concept and many algorithms have been proposed to compute these communities in large graphs using only topological information in an efficient way [12, 25, 27]. The Louvain Method [7] in particular allows to detect communities quickly and efficiently with enlightening results.

Complex networks model systems that are usually evolving (pages appear or disappear on the web, contacts between people evolve over time...). However, most of these graphs are still studied as static graphs, *i.e.* without considering time. These studies therefore overlook much information that would be crucial to fully understand phenomena taking place in the network. Now that more and more data containing temporal information becomes available, algorithms devoted to evolving graphs are required. However, finding and analyzing communities on evolving graphs raises new complex issues. For example, an evolving graph can be seen as a sequence of static graphs, but it raises the issue of deciding what happens between snapshots.

In this paper, we propose a novel approach that consists in detecting communities that are satisfying on a given long period. To achieve this, we detect only *one* partition in communities for a given period that will be relevant for every time step of this period and will thus represents the structure of these snapshots at once. Obviously, considering one unique clustering that is relevant for several snapshots can-

not be as good on a given snapshot as partitions found by a static algorithm that specifically optimizes it for this particular snapshot, but we will show that we can find partitions having equivalent quality even during long periods. Relying on an existing static definition allows us to quantify the loss of quality and to validate more efficiently our results. This unique decomposition will represent a more global structure since it will remain valid for a long time called the *time window*.

Choosing a relevant time window is of course an issue since the graph may have a given structure for some time and then change during another period. For instance, a contact graph is different between day and night since people are often with colleagues during daytime and with family during evening and night. Furthermore, the structure during the day also evolves with meetings, lunches or breaks: time windows can therefore contain smaller natural windows. We propose to identify automatically such changes by building a hierarchical decomposition of time in which each period can be associated to a clustering of the graph.

This paper contribution is thus twofold: first a new definition of multi-step communities with two algorithms to detect them on very large datasets and second an automatic method to hierarchically extract meaningful time windows. This paper is organized as follows: we first introduce some background and related works and then we define precisely the concept of multi-step communities. In section 3 we propose two algorithms to efficiently detect such communities and, in section 4, we evaluate these algorithms on three evolving graphs with very different features. Finally, we present and validate the hierarchical time decomposition in meaningful time windows, before concluding and presenting the perspectives of this work.

1. BACKGROUND AND RELATED WORK

1.1 Communities

Detecting communities in complex networks has raised a lot of interest and many algorithms have been proposed. Usually, people try to find a partition π of the nodes which can be evaluated by a *quality function* that gives a score to π. The *modularity* [20] is a widely used quality function which compares, for each community, the proportion of links inside the community with a null model claiming that meaningful communities are more densely connected than a null model would expect. The classical null model is a random graph with the same degree sequence as the graph under study. Using this null model, with L the total number of links, l_s the number of links inside a community s and d_s the total degree of s, the modularity of a partition π is:

$$Q(\pi) = \sum_{s \in \pi} \frac{l_s}{L} - \left(\frac{d_s}{2L}\right)^2$$

Since the modularity is null when all nodes are grouped in a single community, interesting modularities are always positive and the higher the modularity, the better the partition. Readers can refer to [6] for tighter bounds and a more precise study of the modularity and to [26] for expected values on random graphs. The modularity's main drawback is that it favors large communities [6]. However, it is very natural for studying graph structure and it can be easily extended to more complex cases, such as directed or weighted graphs and overlapping communities [17, 21].

Finding the partition that maximizes the modularity on a given graph is NP-hard and many approximation algorithms have been proposed (see [12, 25, 27] for very complete surveys). Modularity is one of the few quality functions that can be optimized on huge graphs. Indeed, networks of billion links have been partitioned in a few hours [7] and this efficiency is required in order to study evolving graphs composed of many huge static graphs. Since we are using heuristic algorithms, a low value of the modularity can mean either that the algorithm does not succeed in finding meaningful structure or that this structure does not exist. With this limitation in mind, we will use in this paper the Louvain Method, which is the fastest modularity optimization algorithm [7] producing very high quality results. We describe it in more details in section 2.2.

1.2 Communities in evolving graphs

Evolving graphs can generally be described as a sequence of static graphs where several modifications occur between consecutive snapshots. On such graphs, two main approaches have been followed to study the evolution of communities: computing communities for each snapshot and then tracking communities among them, or using the temporal information directly during the detection, *i.e.* detecting communities on the evolving graph itself.

The first approach is much simpler since it consists in community detection at each time step, which can be done with any algorithm. However, it requires the tracking of the communities *i.e.* following what happens between two consecutive snapshots. A community may remain stable, split, appear, disappear, or merge with another one for instance. The intuitive method is to compare two communities of consecutive time steps with rules based on the size of their intersection [1, 14, 29]. These rules can be used conjointly with the clustering algorithm [23] or simplified by tracking only specific core nodes such as the ones defined in [34] which would be more representative of their community than others. Many workarounds are required because of stability issues: the static algorithms used on each snapshot are often non-stable and hence produce different results even if the input graph does not change. This produces noise that makes the tracking very difficult. To solve this issue, one may consider only stable communities [14] or use a very constrained algorithm [23].

The second approach consists in directly integrating time into the computation. Different techniques have been proposed, such as probabilistic models [32], modified algorithms like streaming algorithms [22], modification of static algorithms [3], use of new objects like sliced networks [19] and finally new definitions of communities [31]. A new definition proposed in [16] splits the quality function in two terms: one for the quality of a snapshot partition and one for the stability. It can be quickly summarized in defining the quality Q_{dyn} as $Q_{dyn} = Q_{snapshot} + \alpha Q_{stability}$ where $Q_{snapshot}$ is a static quality function, modularity for instance, $Q_{stability}$ is a stability term and α is a parameter that allows changing the importance of the stability. Instead of a stability term, [28] extends this idea and proposes to add an overall quality term. The partition found at time t does not have to be close to the partition at time $t-1$ but must be a good partition at time t and a fairly good partition at time $t-1$.

We propose in this paper to extend this idea by computing communities that are almost always good on a given time period.

Finally, extracting interesting time windows based on structural changes has attracted attention mainly to perform event detection by studying how much new information is brought by the new snapshot [30] and by discovering correlated spatio-temporal changes [9]. We will extend the notion of time windows to a hierarchy of windows that do not have to be consecutive in section 4, based on our notion of multi-step communities.

2. MULTI-STEP COMMUNITIES

2.1 Definition

We consider here that an evolving graph is a succession of static graphs, each of them representing the state of the complex network at a given time. The evolving graph G on a set of snapshots $S = \{1, 2, ..., n\}$ is $G = \{G_1, G_2, ..., G_n\}$ with $G_i = (V_i, E_i)$ the snapshot i with nodes V_i and edges E_i. We denote by $V = \cup_{i\in\{1,...,n\}} V_i$ the set of all the nodes and a clustering, *i.e.* communities, is a partition of V. We also define a time window T as a subset of the possible snapshots, *i.e.* $T \subseteq S$. In many situations, not all snapshots have the same importance and we may assign a weight w_i to snapshot i. A possible use of weights is to consider snapshots that are not regularly spaced in time: for instance if snapshot i represents the state of the network for a long period, then w_i could be related to the length of the period. It is also possible to give an increasing importance to recent snapshots using a weight like: $w_i = \frac{i}{n}$. Then, we define $Q_{avg}(G, \pi, T)$ the average modularity of the partition π of V for a given time window T as:

$$Q_{avg}(G, \pi, T) = \frac{1}{\sum_{i\in T} w_i} \sum_{i\in T} w_i . Q(G_i, \pi)$$

Where $Q(G_i, \pi)$ is the modularity of the partition π on the static graph G_i considering only the nodes in V_i. We call it the static modularity to differentiate it from Q_{avg}, the average modularity, which applies on evolving graphs and is defined on several snapshots. Detecting communities means finding a partition that maximizes the average modularity. As it is a NP-complete problem [8], we will actually try to find partitions that have the higher average modularity possible.

2.2 Detection algorithms

We propose two algorithms to find a partition of high average modularity on a given time window. The first one consists in building a new graph that is an average representation of the evolving graph and then detecting static communities on it. The second method is a modification of the Louvain Method which is a static modularity optimization algorithm.

To be more precise, the Louvain Method is a hierarchical greedy algorithm designed to optimize the modularity on a static graph (weighted or not). It is composed of two phases, executed alternatively. Initially, each node is in a singleton community. Next, during phase 1, nodes are considered one by one [1]. Each one is placed in its neighboring community (including its own community) that maximizes the static modularity gain. This phase is repeated until no node is moved (the obtained decomposition is therefore a local maximum). Then, phase 2 consists in building a new graph between the communities found during phase 1: there is a node in the new graph for each community and, for two communities C and C', there is a link of weight w where $w = \sum_{v,v'\in C\times C'} weight(v, v')$. There is also a loop on C of weight $\sum_{v,v'\in C\times C} weight(v, v')$ [2]. The algorithm then executes phase 1 and 2 alternatively until the static modularity no longer increases.

2.2.1 Sum-method

Rather than directly optimizing the average modularity, we first propose the sum-method to optimize the static modularity on an average representation of the evolving graph. Given an evolving graph $G = \{G_1, G_2, ..., G_n\}$ and a time window $T \subseteq \{1, ..., n\}$, we build a new weighted graph, called the sum graph, which is the union of all the snapshots in T: each edge of the sum graph is weighted by the total time during which this edge exists in T. Since the sum graph is a static weighted graph, we can apply the Louvain Method on it, or any other classical algorithm, and use the result as multi-step communities.

However, the metric optimized this way is slightly different from the average modularity. Given a partition π, we denote by l_{tc} the number of links inside the community c at time t, L_t the number of links of G_t and d_{tc} the total degree of the community c at time t. The total weight of links inside community c is therefore $\sum_{t\in T} l_{tc}$, the total weight of the sum graph is $\sum_{t\in T} L_t$ and the total weighted degree of the community c is $\sum_{t\in T} d_{tc}$. Then, the static modularity of the partition π on the sum graph is equal to:

$$Q_{sum}(\pi, G, T) = \sum_{c\in\Pi} \frac{\sum_{t\in T} l_{tc}}{\sum_{t\in T} L_t} - \left(\frac{\sum_{t\in T} d_{tc}}{2.\sum_{t\in T} L_t}\right)^2$$

If we come back to the definition of average modularity and after reordering the summation, we have:

$$Q_{avg}(\pi, G, T) = \frac{1}{\sum_{i\in T} w_i} \sum_{c\in\Pi} \left(\sum_{t\in T} \frac{l_{tc}}{L_t} - \sum_{t\in T} \left(\frac{d_{tc}}{2.L_t}\right)^2 \right)$$

Therefore the static modularity on the sum graph Q_{sum} is not exactly the average modularity Q_{avg} and we will next propose a second method to directly optimize this quantity. Nevertheless, this first method is very fast since sum graphs are usually much smaller than the sum of the size of all snapshots and we will see that it is a good first approximation since the obtained results have a high average modularity.

2.2.2 Average-method

To optimize the average modularity we will modify the Louvain Method. Two reasons explain the efficiency of the Louvain Method. First, when it must decide into which community a node should be moved, it can compute quickly the static modularity gain of each possible move. Second,

[1] The results depend on the order in which nodes are considered. It does not really matter in our results and we use one predefined order for each graph.

[2] This ensures that the partition found after phase 1 has the same static modularity as the partition of the new graph where each node is put in its own community.

the size of the considered network is quickly reduced at phase 2 and therefore all subsequent phases are really fast. We must change two elements to adapt the Louvain Method to optimize the average modularity during a time window T: the computation of the quality gain in the first phase and how to build the network between communities in the second one.

One must note that the difference between two partitions is the sum of the differences of the static modularity for each snapshot. Therefore, the average modularity gain can also be easily computed locally: it is the average of the static gains for each snapshot of T. The transformation of the network into a network of communities can be modified in the following way: given a partition π of V, we apply the same transformation as for the Louvain Method on every snapshot of T independently (with different weights for each snapshot) to obtain a new evolving network between the communities of π. We call this algorithm the average-method.

Executing the average-method does not take more time in worst cases than executing the classic Louvain Method on each snapshot of T since it consider less nodes during phase 1. Therefore, it remains applicable to huge datasets composed of thousands of nodes and thousands of time steps. An implementation of this algorithm is available at [13]. We will next analyze the results of these two algorithms.

3. RESULTS ANALYSIS

To validate our approach and the two associated algorithms, we use three evolving graphs with very different characteristics, which show the advantages and the limitations of our approach.

Blogs dataset. During four months, about six thousands blogs have been monitored to track posts, comments and citation links between blogs. We study here the aggregated graph between blogs. We start with an empty graph and every day we add the blogs and links seen this day. The obtained evolving graph grows slowly and regularly during 120 time steps. For more details about the measurement, see [11].

Mrinfo dataset. The second graph represents the topology of multicast routers on the Internet measured with the `mrinfo` tool. This tool allows asking to a multicast router all its neighboring multicast routers. Every day, `mrinfo` was run on a first router and then recursively on every neighbor in a breadth first search fashion. This is not a social networks, but our algorithms and definitions are valid on any kind of evolving graphs and internet topology share many properties with social networks [35, 5]. Thus we consider it as another interesting dataset and include it in our study. The measurement has been performed during several years, yielding a dynamic map of the multicast routers topology. For more details about the measurement, see [24]. We focus here on year 2005, which represents 365 snapshots containing about 3100 nodes on average. The evolution of this graph is divided in three very distinct phases. The first phase spans from the beginning up to day 52. During this phase the graph is very unstable and contains many events. The second phase is between days 52 and 117 and is more stable. Finally, the third phase lasts from day 117 to the end. This last phase is similar to the first one but is longer and contains fewer events.

Imote dataset. The last graph is a sensor graph representing proximity between participants of the Infocom 2005 conference. Participants received a sensor device capable of recording the presence of nearby devices. The experiment started at the conference registration on the evening before the conference itself, then lasted three nights and days. The measurement ended at the end of the third morning. The measurement is very noisy since bluetooth devices often fail to detect each other. People move and so links are quickly changing, resulting in a graph with many transformations. It is composed of 41 nodes and 25000 snapshots, which are not regularly distributed in time. This allowed us to weight snapshots with the effective duration between them. Note that the graph is very stable during night, with only a few connections, and very dynamic during days. For more details see [15].

The three graphs have very different characteristics. Mrinfo and Blogs are quite stable and Imote is very dynamic; Imote contains many snapshots and Blogs only a few. Finally, Blogs is growing (no link is ever removed) and therefore the last snapshot contains all the information. We will see in the sequel that the two algorithms provide results which are related to these characteristics.

3.1 Average modularity

To study the efficiency of the algorithms we compare the quality of several partitions during the whole measurement and the time window T is then $\{1, ..., n\}$. The s-partition is the result of the sum-method (see Section 2.2.1) and the a-partition the result of the average-method (see Section 2.2.2). We have also computed the partitions found by the classic Louvain Method for every snapshot independently. These are the static partitions. As they have the best static modularity we may expect, we compare their average static modularities to the values obtained with both our methods.

	s-partition	a-partition	Average static
Blogs	0.6 (2min 20s)	0.61	0.62 (2min 40s)
Mrinfo	0.91 (28s)	0.92	0.923 (34s)
Imote	0.38 (16s)	0.39	0.56 (8s)

Table 1: Average modularities of studied partitions during whole measurement and time of the optimization on a regular desktop personal computer. a-partitionnever takes more than a few seconds.

Table 1 presents the obtained modularities for the 3 methods. While a-partition and s-partition have very similar quality, the a-partition is always better. As the s-partition is faster to compute, it remains interesting, but from now on we will always use the a-partitions and call them multi-step partitions.

Finally, one can see that the multi-step partition has a comparable average modularity to the maximum one in the Blogs and Mrinfo graphs. We insist that this maximum cannot be achieved because this would require a snapshot so that the unique partition optimized for every snapshot has a better static modularity on this snapshot than the partition specifically optimized for it without consideration of others. This is not realistic since you cannot find higher static modularity by adding more constraint. If the evolving graph is composed of only one snapshot, both methods are exactly the classic Louvain Method.

The most difficult case for our algorithm is when the community structure changes drastically, for instance when a

few nodes that are strongly connected stop being connected and change their links to nodes outside their community. It happens in the Imote graph and we will see more precisely in the sequel some cases where the algorithm finds a good structure and some where it fails.

3.2 Static modularity of multi-step partitions

To obtain more precise information about the quality of partitions, Figure 1 presents the static modularity of these partitions over time. We have also added the best static partition which is the partition among all the static partitions which has the highest average modularity. Their average modularity are 0.6 for Blogs, 0.61 for Mrinfo and 0.34 for Imote. Excepted for Blogs, the best static partitions are not good compared to s-partition and a-partition.

In the Blogs graph, multi-step partition and static partition quality are regularly getting closer confirming that static and average structure are getting closer. Similarly, the best static partition challenges other approaches. This is because this graph is always growing. Therefore, the last snapshots are very similar to the sum graph and then the last static partitions have a high average modularity.

In the Mrinfo graph, the multi-step partition has almost always the same quality as the static. Thus, we capture the overall structure of this graph with only one partition. It works even during phase 2 between days 52 and 117. It seems that parts of the set of nodes change during this period, and thus this event does not affect the partition of the other nodes. The best static partition here clearly fails in discovering a global structure since it is good only during a few snapshots. This shows the interest of looking for a relevant multi-step partition on the whole measurement rather than just selecting one snapshot which cannot be representative of the whole.

In the Imote graph, it appears that the structure changes a lot over time and there is a clear day/night effect. The multi-step partition is a lot more effective during night, since the graph is almost static and groups are clearly separated. Conversely it fails during the day when the structure is less stable and thus more difficult to identify. The best static partition suffers from same default and has a lower quality.

Finally, the efficiency of the average modularity depends on graph characteristics. With Mrinfo and Imote, the multi-step partition gives new information since there is no representative time of the average structure whereas for Blogs detecting the structure of the last snapshots is sufficient.

3.3 Nodes existence

An interesting insight is the existence or not of nodes during the time window. The partition is unique and thus static, but the graph evolves. Nodes and links appear and disappear and although we only have one partition, communities still evolve. Some nodes may even never exist at the same time but be strongly connected to a very stable core in a multi-step community. To represent this, we have drawn a picture for every community where each line corresponds to a node and each column to a time step (see Figure 2). If the $n-th$ node of the community exists at time t, the point (t, n) is white and conversely the point is black if it does not exist. To have a result which does not look noisy and is understandable, we must order the nodes carefully. We have chosen one simple order which gives good results: nodes are first ordered according to their first appearance time

Figure 2: Nodes existence diagrams in Mrinfo dataset for two interesting communities.

since nodes that appear together are often similar, and then sorted according to the number of snapshots during which they exist (it appears that this approach puts together nodes involved in the same events).

This graphical representation is not interesting for all graphs and communities since we cannot display many nodes nor many snapshots. Furthermore the multi-step partitions are not always meaningful and then trying to display them is not very useful. In Blogs, communities grow slowly and there is nothing really interesting to show. For Imote, there are too many snapshots divided in phases where we do not find relevant average structure so we do not represent it. The most interesting results are obtained on Mrinfo, see Figure 2.

The community (a) illustrates the simplest case: all nodes exist almost during the same time except a few outliers. Nodes appear and disappear together which is revealing that the grouping is relevant. The community (b) is a more complex case and allows the identification of sub-clusters: a core group composed of nodes that are the most present and two groups that exist at different moments. Thus at the beginning, group 1 and core group exist and disappear during phase 2 between days 52 and 117 before reappearing for a few days. Then, group 1 disappears and group 2 replaces it.

Such drawings illustrate that even if the partition is unique, nodes may exist or disappear during the measurement and therefore the communities are evolving. This reveals new information about community structure, hence looking locally inside a community allows us to study an interesting evolution.

4. TIME WINDOW AND HIERARCHY

Until now, we have always optimized the average modularity over the whole experiment time and we have seen that in some cases this leads to communities of lower interest since the average structure changes too much. We will now exploit the fact that we can choose the time window T. It can be an interval if we are interested only in consecutive snapshots, but it is not mandatory. For example, one may imagine a repeated social structure every day, which would therefore not be composed of consecutive snapshots, and detecting it would be interesting.

4.1 Hierarchy

In Imote the structure changes between days and nights, but it changes also during days with several sessions, lunches, keynote, etc. Thus, time windows can naturally contain

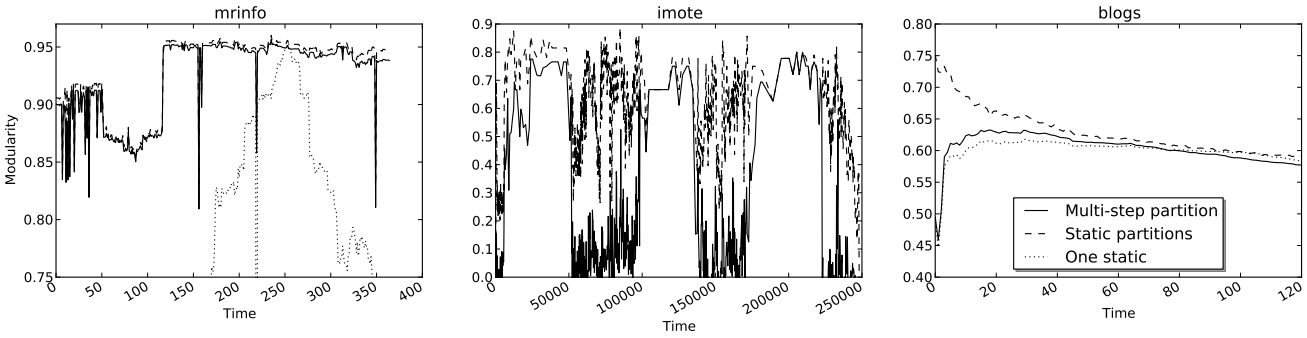


Figure 1: Static modularities of the studied partitions on each snapshot.

smaller windows. This kind of hierarchical segmentation is usually represented as a tree in which the leaves are the individual time steps, which are hierarchically grouped using a similarity criterion. We propose an agglomerative hierarchical time clustering algorithm (algorithm 1): at the beginning, every snapshot is alone and we group recursively the two most similar windows until no possibility remains. This algorithm requires a similarity measure to decide which time windows should be merged.

Algorithm 1 Hierarchical time window merge algorithm.

```
G the initial graph
L the list of potential snapshots, initially empty
for all snapshots t of G do
    Compute the communities π_{t} on the snapshot t
    Put {t} in L
end for
while L not empty do
    Find T_i and T_j in L which maximize Sim(T_i, T_j)
    Remove T_i and T_j of L and add T = T_i ∪ T_j in L
    Output that T_i and T_j have been merged
    Compute π_T the communities of G on the window T
end while
```

4.1.1 Similarity

To define a similarity between time windows, we will rely on the associated partitions. We claim that if two time windows are structurally similar, the multi-step partition of one, which summarizes its structure, will be a relevant decomposition for the other and conversely. This can be evaluated for two time windows T_i and T_j associated to the partitions π_i and π_j by the similarity:

$$Sim(T_i, T_j) = Q_{avg}(G, \pi_i, T_j) + Q_{avg}(G, \pi_j, T_i)$$

We do not perform direct comparison of the partitions because of the instability of the decomposition algorithms as discussed earlier and in [3]. Conversely, the modularity is very stable and therefore we use it to test if both time windows have a similar structure.

Some extensions of the algorithm are possible. For example, one may want to force the merged time windows to be consecutive: in this case, the results are easier to interpret but repeated structures will be missed if they are not consecutive. In the following we will also forbid the merge of time windows whose similarity is negative [3]. With these rules, the algorithm can produce several disjoint trees rather than a simple one.

This algorithm can produce time segmentation quickly. If the evolving graph is made of n snapshots, there are initially n community detections to perform and one more community detection for each merge, *i.e.* at most n more. Thus, there are $2n$ executions of the sum-method (but most on smaller time windows than the whole dataset) and each partition must be evaluated on each snapshot, which is equivalent to iterating over all the edges. The exact complexity of this algorithm relies on the complexity of the Louvain Method, which is unknown. For instance, the times segmentation takes 1 hour and 30 minutes for Mrinfo, 25 minutes for Blogs and 8 hours for Imote (the high number of snapshots induces a very large tree). If we use the constraint of merging only consecutive time windows, these lengths drop to 17 minutes for Mrinfo, 11 minutes for Blogs and 1 minute 50 seconds for Imote.

4.2 Time window hierarchies

Validation of the results is an important problem and is now performed experimentally. We have checked that time windows identified by the algorithm correspond to understandable phenomena in the graphs. This validates both the algorithm and the similarity function. We have built the tree for each dataset with and without forcing time windows to be consecutive. Note that if we force time windows to be consecutive, a strong modification of the structure during one time step prevents time windows before and after this event to be merged and results in a split. This validation process is limited, but there do not exist an objective benchmark for dynamic community detection.

With the Mrinfo dataset using consecutive time windows (Figure 3 top), the highest level, which contains the largest time windows, is more granular than the 3 phases described before. Indeed, there are many events during the first phase which therefore cannot be identified as one consecutive time window. The group of snapshots between 52 and 116 corresponds to phase 2 and the time windows after correspond to the phase 3 also separated in four sub-windows by small events. The first level hence reflects almost exactly what we were expecting and we can explain many separations at

[3] the trivial partition where all nodes are in one community has a modularity of 0, so a negative modularity describes a worse decomposition!

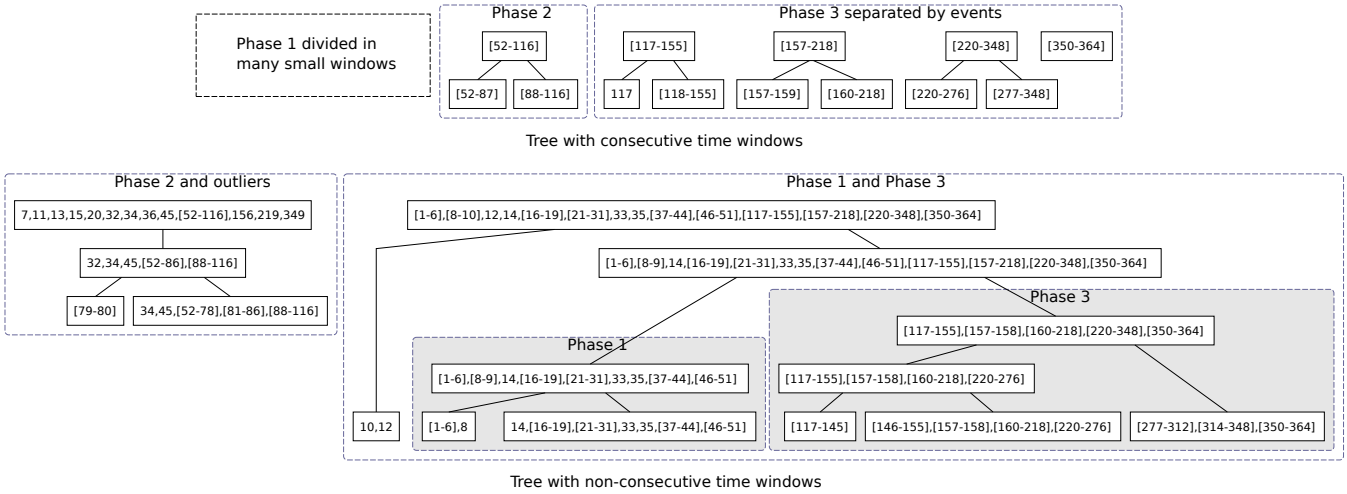


Figure 3: First levels of the hierarchical time decomposition on the Mrinfo dataset. Numbers in tree nodes represent time windows. For instance, "32,34,45,[52-86],[88-116]" corresponds to the time window containing days 32, 34, 45, from 52 to 86 and from 88 to 116. We have grouped time windows into blocks with potential explanations when possible.

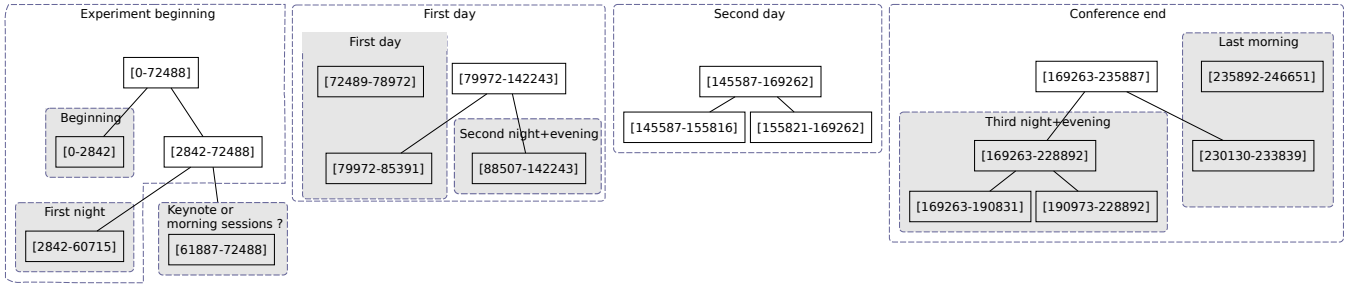


Figure 4: First levels of the hierarchical time decomposition on Imote dataset using consecutive time windows. Times are in seconds.

lower levels.

When considering the decomposition obtained with non-consecutive time windows (Figure 3 bottom), the effect caused by events disappears. The first level is composed of two groups: one containing phase 2 and some outliers and one containing phase 1 and 3, which are quite similar structurally. This last group is then subdivided into phase 1 and phase 3. Sub-levels are separated by events. Moreover, without any constraint of consecutive grouping, we remark that time windows are often almost consecutive except for events, showing that the algorithm tends to merge similar time windows anyway.

Imote dataset presents similar results. Using consecutive time windows, the highest level contains several time windows that correspond to real moments (see Figure 4). The first one corresponds to the beginning of the experiment containing three groups: 45 minutes after the beginning, a first long period corresponding to the first night and then another 2 hours and a half, which may correspond to the breakfast and keynote speech (we do not have the exact experiment timing so we can only conjecture). Then there is a small group that may correspond to a part of the day 1. The next window is composed of another part of the day 1 and of the second night. The following time window contains mostly the second day, which is then divided in two parts that al-

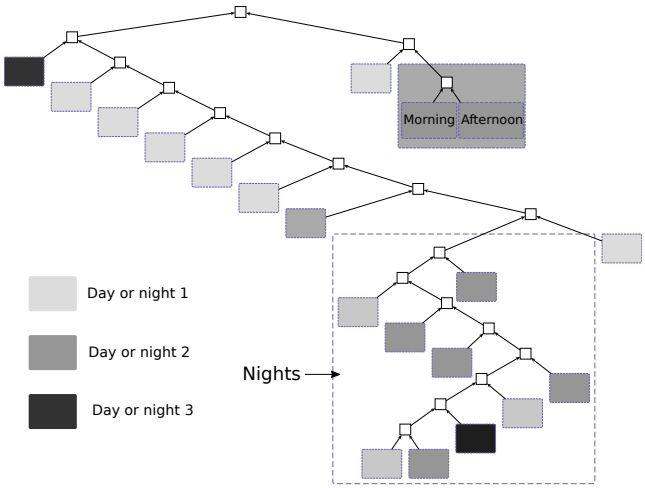


Figure 5: Biggest tree of the hierarchical time segmentation of the Imote dataset using non-consecutive time windows. Blocks represent big subtrees to be able to represent the whole structure.

most match the morning and the afternoon. The next time window is composed of the last night and a part of the last morning. The last window contains the remaining moments of the last morning. Thus, the time segmentation algorithm seems to detect easily explainable time windows.

We present a sketch of the structure of the largest tree (the others are very small) of the decomposition built using non-consecutive time windows on Figure 5. The tree is huge and there are many sub-levels thus we can only give an idea of the whole structure and each block is in fact a big sub-tree. Sub-trees are composed of snapshots of the same day, showing that graph structure gives meaningful information. The two biggest groups are the day 2 and the nights. Day 2 is then divided into the morning and the afternoon and the night sub-tree contains many sub-trees depending on the day.

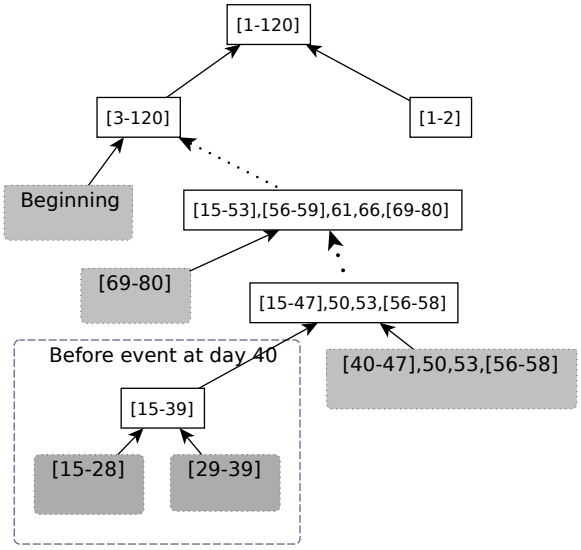


Figure 6: Tree of the hierarchical time segmentation of Blogs. Gray blocks represent sub-trees and dashed arrows long paths with only one snapshot added at each step.

With Blogs, consecutive and non-consecutive trees are very similar. They are unbalanced and almost degenerate into a list which is consistent with the fact that the graph regularly grows: there is no clear notion of time windows. We hence draw only a sketch of the non-consecutive tree on figure 6. There are only few long time windows. The highest windows contain the first snapshots, which are the most different from the average structure. We do not succeed in interpreting the other windows because of a lack of knowledge about the measurement. We only know that a measurement event happened at day 40 that can be seen in the two lowest windows.

Finally, results seem very promising. Time segmentation is often meaningful with graphs that exhibit an evolution we can divide in time windows. Both approaches, using consecutive and non-consecutive time windows are complementary: consecutive time windows are easier to interpret and produce directly interesting results but non-consecutive are less sensitive to events and can therefore detect some long range or periodic connections.

5. CONCLUSIONS AND PERSPECTIVES

We have shown that communities can be detected not only on one given snapshot but also on a given long period by optimizing an average quality function. We have proposed two algorithms to optimize this quality: one is very fast but the other produces better results and remains applicable to huge datasets with many large snapshots.

The new temporal quality function is very general and weights allow to emphasize some snapshots or to consider their duration. We used here the modularity as the basic quality function but others could have been used. Using a known static quality function is an asset to validate our results: if the static quality function is valid and we have the same quality on one snapshot, then our new method is also valid on this snapshot. Having communities that span on several snapshots and diagrams like the nodes existence over time provide new analysis tools to obtain insights about lives of communities.

Optimizing on a given period raises the issue of the relevant length of time windows since using a long period is not always satisfying. We have proposed a method to extract automatically interesting time windows that scales to evolving graphs of thousands of nodes and snapshots. Non-consecutive time windows can be detected using an agglomerative hierarchical time clustering algorithm which uses a similarity function based on the snapshots structure: if two snapshots have the same structure, the multi-step partition of one will be relevant on the other and conversely.

These results give new insights on the complex networks under study. The construction process of Blogs has a significant impact on the conclusions we can draw and maybe some links should be removed some time after their creation to obtain a temporal structure that is more meaningful on such web graphs. Our algorithms are most successful on the Mrinfo and Imote datasets where clear phases exist, which can be detected and described. This experimental validation is still limited and this is a major concern for all dynamic community detection algorithms: there exist no consensual benchmark or dataset with ground truth. As several algorithms now exist, we should develop such tools for objective comparison.

Further studies will also focus on other similarity functions. The current similarity function measures both how similar and modular time windows are and normalization might correct this default. Another limitation of the new average modularity is due to the summation itself. The sum is commutative and thus the order of the snapshots has no influence on the results. This means that there is no causality and that, for instance, the succession of snapshots G_1, G_2, G_3 is strictly equivalent to the succession of snapshots G_3, G_2, G_1. As a first approximation, it is all right, but it would be great to adapt the metric to deal with causality.

Acknowledgments

This work was partly funded by a grant from the *Agence Nationale de la Recherche*, with reference ANR-10-JCJC-0202 and by the project WebFluence #ANR-08-SYSC-009.

6. REFERENCES

[1] S. Asur, S. Parthasarathy, and D. Ucar. An event-based framework for characterizing the evolutionary behavior of interaction graphs. In *Proc. of the 13th ACM SIGKDD*, pages 913–921. ACM, 2007.

[2] D. Auber, Y. Chiricota, F. Jourdan, and G. Melançon. Multiscale visualization of small world networks. In *Proc. IEEE Symposium on Information Visualization*, pages 75–81, 2003.

[3] T. Aynaud and J.-L. Guillaume. Static community detection algorithms for evolving networks. In *WiOpt Workshop on Dynamic Networks*, pages 508–514, 2010.

[4] L. Backstrom, D. Huttenlocher, J. Kleinberg, and X. Lan. Group formation in large social networks: membership, growth, and evolution. *Proc. of the 12th ACM SIGKDD*, pages 44–54, 2006.

[5] A.-L. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509, 1999.

[6] M. Barthélemy and S. Fortunato. Resolution limit in community detection. *Proc. of the National Academy of Sciences of the United States of America*, 104(1):36–41, 2007.

[7] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *J. Stat. Mech*, 10008:1–12, 2008.

[8] U. Brandes, D. Delling, M. Gaertler, R. Goerke, M. Hoefer, Z. Nikoloski, and D. Wagner. Maximizing Modularity is hard. *ArXiv Physics e-prints*, 2006.

[9] J. Chan, J. Bailey, and C. Leckie. Discovering correlated spatio-temporal changes in evolving graphs. *Knowledge and Information Systems*, 16(1):53–96, 2008.

[10] Y. Chi, S. Zhu, X. Song, J. Tatemura, and B. L. Tseng. Structural and temporal analysis of the blogosphere through community factorization. *Proc. of the 13th ACM SIGKDD*, pages 163–172, 2007.

[11] J.-P. Cointet and C. Roth. Socio-semantic Dynamics in a Blog Network. *2009 International Conference on Computational Science and Engineering*, (6):114–121, 2009.

[12] S. Fortunato. Community detection in graphs. *Physics Reports*, (486):75–174, 2010.

[13] J.-l. Guillaume. Web Page, http://jlguillaume.free.fr/www/programs.php.

[14] J. Hopcroft, O. Khan, B. Kulis, and B. Selman. Tracking evolving communities in large linked networks. In *PNAS*, volume 101, pages 5249–5253. National Acad Sciences, 2004.

[15] P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot. Pocket switched networks and human mobility in conference environments. *Proc. of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking - WDTN '05*, pages 244–251, 2005.

[16] R. Kumar, A. Tomkins, and D. Chakrabarti. Evolutionary clustering. In *In Proc. of the 12th ACM SIGKDD*, pages 554–560. ACM Press, 2006.

[17] E. A. Leicht and M. E. J. Newman. Community structure in directed networks. *Physical Review Letters*, 100(11), 2007.

[18] S. Mancoridis, B. Mitchell, C. Rorres, Y. Chen, and E. Gansner. Using automatic clustering to produce high-level system organizations of source code. In *Proc. 6th Intl. Workshop on Program Comprehension*, pages 45–53, 1998.

[19] P. Mucha, T. Richardson, K. Macon, and M. A. Porter. Community structure in time-dependent, multiscale, and multiplex networks. *Science*, 876:10–13, 2010.

[20] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69(2):26113, 2004.

[21] V. Nicosia, G. Mangioni, V. Carchiolo, and M. Malgeri. Extending the definition of modularity to directed graphs with overlapping communities. *J. Stat. Mech.*, 2009:P03024, 2009.

[22] H. Ning, W. Xu, Y. Chi, Y. Gong, and T. Huang. Incremental spectral clustering with application to monitoring of evolving blog communities. *In SIAM Int. Conf. on Data Mining*, 2007.

[23] G. Palla, A.-L. Barabasi, and T. Vicsek. Quantifying social group evolution. *Nature*, 446:664–667, 2007.

[24] J. Pansiot, P. Mérindol, B. Donnet, and O. Bonaventure. Extracting Intra-Domain Topology from mrinfo Probing. In *Passive and Active Measurement*, 2009.

[25] M. A. Porter, P. J. Mucha, and J.-P. Onnela. Communities in Networks. *Notices of the American Mathematical Society*, 56:1082–1097, 2009.

[26] J. Reichardt and S. Bornholdt. When are networks truly modular? *Physica D Nonlinear Phenomena*, 224:20–26, 2006.

[27] S. E. Schaeffer. Graph clustering. *Computer Science Review*, 1:27–64, 2007.

[28] X. Song, Y. Chi, B. L. Tseng, D. Zhou, and K. Hino. Evolutionary spectral clustering by incorporating temporal smoothness. In *Proc. of the 13th ACM SIGKDD*, pages 153–162. ACM, 2007.

[29] M. Spiliopoulou, I. Ntoutsi, Y. Theodoridis, and R. Schult. Monic: modeling and monitoring cluster transitions. In *Proc. of the 12th ACM SIGKDD*, pages 706–711. ACM, 2006.

[30] J. Sun, C. Faloutsos, S. Papadimitriou, and P. Yu. Graphscope: parameter-free mining of large time-evolving graphs. In *Proc. of the 13th ACM SIGKDD*, pages 687–696. ACM, 2007.

[31] C. Tantipathananandh, T. Berger-Wolf, and D. Kempe. A framework for community identification in dynamic social networks. *Proc. of the 13th ACM SIGKDD*, pages 717–726, 2007.

[32] B. L. Tseng, Y.-R. Lin, Y. Chi, S. Zhu, and H. Sundaram. Analyzing communities and their evolutions in dynamic social networks. *ACM Transactions on Knowledge Discovery from Data*, 3(2):1–31, 2009.

[33] M. L. Wallace, Y. Gingras, and R. Duhon. A new approach for detecting scientific specialties from raw cocitation networks. *J. Am. Soc. Inf. Sci. Technol.*, 60:240–246, 2009.

[34] Y. Wang, B. Wu, and N. Du. Community Evolution of Social Network: Feature, Algorithm and Model. *Science And Technology*, (60402011), 2008.

[35] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442,

1998.

[36] J. Zhao, H. Yu, J. Luo, Z. Cao, and Y. Li. Hierarchical modularity of nested bow-ties in metabolic networks. *BMC bioinformatics*, 7(386), 2006.

Conductance Facilitates Decentralized Search in Networks

Qing Ke
School of Computer Science
Beijing University of Posts and Telecommunications
Beijing 100876, China
keqing.echolife@gmail.com

Yuxiao Dong
School of Computer Science
Beijing University of Posts and Telecommunications
Beijing 100876, China
ericdongyx@gmail.com

Bin Wu
School of Computer Science
Beijing University of Posts and Telecommunications
Beijing 100876, China
wubin@bupt.edu.cn

ABSTRACT

We study decentralized search by routing queries in networks from a novel perspective of community structure. We first find that through maximizing sample conductance which is widely used as the metric for graph clustering and community detection, we can get high coverage samples. Then, based on this result, we propose a new decentralized search strategy named *Conductance Search* which tries to efficiently find the nodes belonging to different communities. The results of scalability evaluation indicate that this algorithm is sub-linear in the number of nodes in the network. Finally, comparing the strategy with other common strategies, we show that the conductance search has better performance than other well-known strategies in the number of steps to find the target and in time complexity. These results shed light on understanding the function of current and prospective distributed systems (e.g. social networks, peer-to-peer file sharing networks) that route messages by using only local information.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications-Data mining; E.1 [**Data Structures**]: Graphs and networks; H.3.3 [**Information Search and Retrieval**]: Retrieval Models; G.2.2 [**Graph Theory**]: Graph algorithms, Network problems

General Terms

Algorithms; Experimentation; Measurement

Keywords

decentralized search, social network analysis, graph mining, complex networks, community structure, conductance, peer-to-peer networks

1. INTRODUCTION

In this paper, we explore the algorithmic problem of decentralized search in large networks from a novel perspective of community structure, and propose an efficient search algorithm.

The 5th SNA-KDD Workshop '11 (SNA-KDD'11), August 21, 2011, San Diego CA USA.

1.1 Complex Networks and Search

Many social systems can be represented as complex interaction networks where nodes represent entities and edges mimic the interactions among them. These interaction networks arise from online communities networks describing the friendship among persons, e-mail networks and call graphs abstracting the communication mode to co-authorship and citation networks representing research teams and topics. The study of these complex interaction networks can provide insight into their structure, properties and behavior.

One particular line of network research is concerned with small-world phenomena and decentralized search algorithms. Decentralized means that we do not have the full knowledge of the global network topology. So starting from some initial source node, we forward the messages from one node to another according to the adopted search strategy until the target node is found. Efficient search in networks has a large number of practical applications, including shortest paths search in social network relationship [14], Web pages search in WWW, querying files in peer-to-peer systems and focused Web crawlers design [14]. And it also can be used to better understand the functioning of current and prospective distributed systems that route messages by using local information. These systems include social networks routing messages, referral systems for finding informed experts, and also technological systems for routing messages on the Internet, ad hoc wireless networking [6].

1.2 Overview of our approach

We explore the decentralized search problem from the perspective of community structure which real-world networks often exhibit. Intuitively, a community in a network is a cluster of nodes more densely connected to each other than other nodes. By this intuitive definition, nodes in the same community will be expected to share more neighbors than nodes in different communities. As a result, if we can visit a small number of nodes from many different communities, many nodes would be discovered in the search, hence the search will be very efficient. An illustration is depicted in Figure 1. In the figure, there are 3 communities and starting from the source node s, if we want to search node t, we try to locate the nodes that belong to different communities like node $b1$ and $b2$ which are responsible for delivering the querying messages to target node and serve as a bridge role.

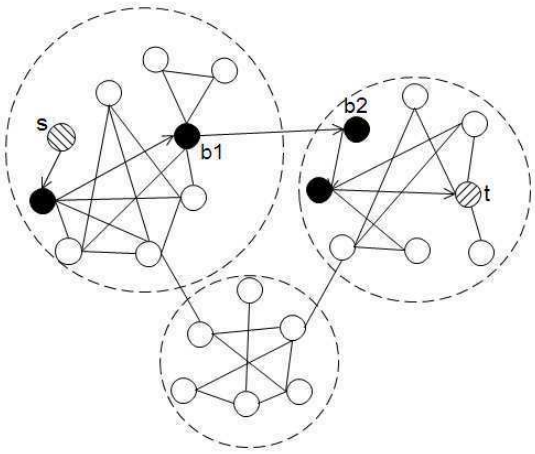


Figure 1: Sample network illustrated our method

How can we fast locate these nodes? We choose a natural and widely-adopted notion of community goodness called *conductance*, also known as the normalized cut metric [12], as objective function. In every step of the search process, we try to optimize this metric. Whether maximize or minimize, we will further discuss in Section 4. Because we do not know the global network topology, we cannot use the existed both theoretical and practical algorithms to optimize this quantity. We propose an approximation greedy algorithm that only uses local information to optimize this function. We will show the search strategy based on this greed algorithm is very efficient.

1.3 Contributions

Our main contributions and findings include the following:

- We define the concept of *sample coverage*, which concisely captures size of the sample and its neighbor. Then we find that, through *maximize* rather than *minimize* sample conductance, we can construct high coverage samples, and relatively small samples of nodes can exhibit significantly high coverage.
- Based on sample conductance, we propose a new decentralized search strategy which explicitly tries to locate the nodes which belong to different communities.
- We compare the proposed strategy to many other search algorithms on both random and real-world networks. And the results show that our strategy generally outperforms others.

The rest of the paper is structured as follows. Section 2 describes related work. Section 3 gives some notations, definitions and datasets which are used in this paper. Section 4 describes how to construct high coverage samples. In Section 5, we give the detail description of conductance search strategy. And in Section 6, we evaluate the proposed strategy, and compare to other strategies. Finally in Section 7, we draw the conclusions and discuss future work.

2. RELATED WORK

2.1 Real-world Experiment

One of the first experiments on decentralized search in networks was conducted by sociologist S. Milgram [19] in the 1960s. The experiments quantified the small-world phenomenon, that is, the principle that all people are linked by short chains of acquaintances. Milgram's findings were subsequently confirmed by others like [8]. Adamic [1] simulated short paths finding experiments which using only local information on a network of actual email contacts within an organization. And the results indicated that small world search strategies using a contact's position in physical space or in an organizational hierarchy relative to the target can effectively be used to locate most individuals.

2.2 Theoretical Model

In 2000s, Kleinberg [13] proposed a mathematical graph model for studying the aspect of small worlds. In his model, individuals are the nodes of a two-dimensional grid, which is augmented with some additional long-range edges probabilistically. Under such form of correlation, an algorithm with knowledge of the target's location can find the target in polylogarithmic time. Banerjee and Basu [4] introduced a novel Social Query Model (SQM) for decentralized search, which factored in realistic elements such as expertise levels and response rates of nodes, and had the Pagerank model and certain Markov Decision Processes as special cases. Recently, Fraigniaud and Giakkoupis [10] proposed a model that based on *arbitrary* graphs rather than 2-D grid, then they gave out the long-range contact scheme and corresponding search strategy where the next hop was to a distant node only if it yields sufficient progress towards the target, finally they shed light on the upper and lower bound of expected number of search steps.

However, these real-world experiments and theoretical models are unavailable as search aids in many real-world scenarios such as unstructured peer-to-peer file sharing networks because they need *other* information. For example, in the Milgram experiment, the people know the geographic information of their friends. Also, in Adamic simulation, the people know the physical space or organizational hierarchy of the target. And in Kleinberg's and Fraigniaud's model, the nodes know the local contacts of *all* nodes in the network.

Hangal et al. [11] proposed search in social networks could be made more effective by incorporating weighted and directed influence edges in the social graph, thus capturing both tie strength and asymmetry. However, it was global social search rather than decentralized search because all network information was available to the search algorithm.

Simsek [27] proposed a new algorithm for finding a target node in a network whose topology is known only locally by formulating the searching task as a problem of decision making under uncertainty and used the statistical properties of the graph to guide this decision. However, this algorithm also need other information like node attribute values.

2.3 Search Strategy

As described above, all the theoretical modes are unavailable as search aids in real-world applications. Instead, typical

approaches in practical applications resort to the following strategy:

Breadth-First Search (BFS). In BFS strategy, each node sends a copy of the query to each of its neighbors. This flooding and broadcast method finds targets quickly. But it is highly unscalable to network size due to the tremendous overhead incurred from redundant forwards and can cause network congestion.

Random Walks (RW). In RW [22] search strategy, the current node forwards the query to exactly one randomly selected neighbor. And the expected search steps may be very large.

Degree Search (DS). In DS [2], the next node in the search is the neighbor of the current node with highest degree. This strategy is efficient for power-law networks but fails to search efficiently in the heterogeneous networks.

Expansion Search (XS). Recently, Maiya and Berger-Wolf [18] proposed the expansion search strategy borrowing from concepts in expander graphs.

Our focus in this work is to investigate efficient search in networks and propose a new search strategy which outperforms other strategies in many both random and real-world networks data.

3. PRELIMINARIES

3.1 Notations and Definitions

We first briefly describe some notations and definitions used in this paper.

Definition 1. $G = (V, E)$ is a undirected *network* or *graph* where V is a set of nodes and $E \subseteq V \times V$ is a set of undirected edges. In this paper, we use the terms *network* and *graph* interchangeably.

Definition 2. A *sample* S is a subset of nodes where $S \subset V$, and S is assumed to contain no more than half of all the nodes.

Definition 3. $N(S)$ is the *neighbors* of S if $N(S) = \{w \in V - S : \exists v \in S \text{ s.t. } (v, w) \in E\}$.

Definition 4. The *conductance* of a sample S is:

$$\phi(S) = \frac{c(S)}{c(S) + 2e(S)}$$

where $c(S)$ is the number of edges with one endpoint in S and one endpoint in $\bar{S}$ (the complement of S), $e(S)$ is the number of edges with both endpoints in S.

Definition 5. The *maximum conductance sample* of size k is a sample S of size k with the maximum conductance:

$$S^* = \underset{S:|S|=k}{\operatorname{argmax}} \frac{c(S)}{c(S) + 2e(S)}$$

Definition 6. The *coverage* of a sample S is $\frac{|S \cup N(S)|}{|V|}$.

We can see that the coverage of a sample S measures the number of the sample and its neighbors. So it concisely captures the number of nodes with one hop of the sample.

3.2 Clusters and communities in networks

A community in a network is a subgroup of relatively densely connected nodes S. More formally, if L is the adjacency matrix of the graph G, then:

$$\phi(S) = \frac{\sum_{i \in S, j \notin S} L_{ij}}{\min\{L(S), L(\bar{S})\}}$$

where $L(S) = \sum_{i \in S} \sum_{j \in V} L_{ij}$. In this case, the conductance of the graph G is:

$$\phi_G = \underset{S \subset V}{\operatorname{argmin}}\, \phi(S)$$

The conductance of a sample provides a measure for the quality of the cut $(S, \bar{S})$, or the goodness of a community S. It is often noted that communities have more and better intra-connections than inter-connections, so samples that are densely linked inside and sparsely linked to the outside have *small* conductance. The extraction of communities in networks is important since they often represent real social and functional groups, or similarity, so when interested in detecting communities and evaluating their quality, we prefer samples with *small* conductance. Although there are many measures that have been proposed for evaluating how a set of nodes is like community, it is commonly noted that conductance captures the notion of clustering [12, 26] and so it has been widely-used for graph clustering and community detection [25].

3.3 Datasets

We assess the performance of different decentralized search strategies on a total of nine different networks: three random graph models and six real-world network data. We describe each dataset in detail as follows.

Erdos-Renyi Model (ER). It is the first random graph model which was proposed by Erdos and Renyi [9]. The model produces a random graph of n nodes with each of the possible edges existing with probability p. The graphs generated by this model exhibit short average path lengths, but lack the high clustering and heavy-tailed degree distribution found in reality. For our experiments described in 6, we generated ER networks with $n = 40,000$, $p = 0.001$.

Barabasi-Albert Model (BA). The model [5] generates a graph of n nodes in a sequential mode. Each subsequent node of m edges is preferentially attached to previously added nodes with high degree. The generated graphs exhibit power law degree distributions and short average path lengths, but lack the high clustering found in real networks. Also, we choose parameters $n = 40,000$ and $m = 20$.

Watts-Strogtz Model (WS). The model [28] starts with a ring of n nodes, each connected to its k nearest neighbors by undirected edges. Then a node and the edge that connects it to its *nearest* neighbor in a clockwise sense are chosen.

Table 1: Networks Basic Structural Properties

Random Graphs	N	M	d	k	D	L	C	a
Erdos-Renyi	40,000	800,921	0.001	40.0460	4	3.1642	0.0010	-0.0003
Barabasi-Albert	40,000	797,526	0.001	39.8763	4	2.9510	0.0058	-0.0061
Watts-Strogtz	40,000	800,000	0.001	40.0000	5	3.9575	0.5426	-0.0047
Real world	N	M	d	k	D	L	C	a
CondMat	21,363	91,286	0.0004	8.5462	15	5.3522	0.6417	0.1253
Enron	33,696	180,811	0.00032	10.7319	13	4.0252	0.5092	-0.1165
Epinions	75,877	405,739	0.00014	10.6946	15	4.3078	0.1378	-0.0406
Gnutella	62,586	147,892	0.00008	4.7260	11	5.9355	0.0055	-0.0926
HepPh	11,204	117,619	0.00187	20.9959	13	4.6727	0.6216	0.62950
Slashdot	82,168	504,230	0.00646	12.2731	13	4.0691	0.0603	-0.0738

With probability p, this edge is reconnected to a node chosen uniformly at random over the entire ring, with duplicate edges forbidden. This process is repeated by moving clockwise around the ring, considering each node in turn until one lap is completed. Next, we consider the edges that connect nodes to their *second-nearest* neighbors clockwise and continue the similar process described above. The generated graphs exhibit short average path lengths, and high clustering, but lack power law degree distribution found in real networks. To generate the networks, we choose $n = 40,000$, $k = 20$ and $p = 0.1$.

CondMat. It is arXiv Condense Matter Physics *collaboration* network from the e-print arXiv and covers scientific collaborations between authors' papers submitted to Condense Matter category [16].

Enron. It is Enron *email* communication network which covers all the email communication within a dataset of around half million emails [15, 16].

Epinions. This is a who-trust-whom *online* social network of a general consumer review site Epinions.com [23].

Gnutella. This is a snapshot of the Gnutella *peer-to-peer* network in August 31 2002 [16, 24].

HepPh. It is arXiv High Energy Physics Phenomenology *collaboration* network from the e-print arXiv and covers scientific collaborations between authors' papers submitted to High Energy Physics Phenomenology category [16].

Slashdot. The network contains friend/foe links between the users of Slashdot. The network was obtained in February 2009 [17].

It should be noted that some of the networks described above are not connected. And for the purpose of experiments described in Section 6, we extract the largest connected component so that any node can be reached from any other node in the extracted subgraph. Table 1 shows the basic structural and statistical characteristics of each network's extracted subgraph. In Table 1, N is number of nodes, M is number of edges, d is density, k is average degree, D is diameter, L is average path length or characteristic path length, C is clustering coefficient, and finally a is associative coefficient.

4. SAMPLE COVERAGE

In this section, we see that high coverage samples can be got through maximizing conductance. Intuitively, from Section 3.2, we know that the smaller conductance of the sample is, the better "gestalt" notion of community is. Conversely, we want to sample the nodes from different communities, so we will *maximize* the conductance of the sample. And we will verify this intuition through comparing the sample coverage of different sample size. We begin with problem formalization.

4.1 Problem Formalization

Definition 7. (*Maximum Conductance Problem*) Given a graph $G(V, E)$, and a sample size $k < |V|$, the Maximum Conductance Problem (MCP) is to find a sample $S \subset V$ of size k with the maximum conductance, $\frac{c(S)}{c(S)+2e(S)}$. That is, find:

$$\underset{S:|S|=k}{\operatorname{argmax}} \frac{c(S)}{c(S) + 2e(S)}$$

The MCP is a NP-hard problem, and there exist a rich suite of both theoretical and practical algorithms to optimize the conductance quantity [3, 7]. One typically resorts to spectral analysis, as the spectrum of a graph can be computed in polynomial time. However, our ultimate goal is to locate the overlapping nodes during the course of a decentralized search and we do not know the full topology of network. So spectral analysis may not be useful here. Thus, we approximate conductance using a simple greedy algorithm. At each iteration, we greedily select the node that maximizes the conductance of the currently constructed sample. Before formally proposing the algorithm, we first give a proposition and proof it.

PROPOSITION 1. *There exist networks for which the conductance,* $\frac{c(S)}{c(S)+2e(S)}$*, is non-monotonic as the sample size* $|S|$ *grows.*

PROOF. Consider a current sample S. Assume that the next node selected for inclusion in the sample is v. Then, the new sample is $S \cup \{v\}$. By the conductance definition, the conductance of S is $\frac{c(S)}{c(S)+2e(S)}$ and the conductance of $S \cup \{v\}$ is $\frac{c(S\cup\{v\})}{c(S\cup\{v\})+2e(S\cup\{v\})}$. The conductance will increase

when:

$$\frac{c(S \cup \{v\})}{c(S \cup \{v\}) + 2e(S \cup \{v\})} > \frac{c(S)}{c(S) + 2e(S)}$$

$$c(S \cup \{v\}) > c(S) \cdot \frac{c(S \cup \{v\}) + 2e(S \cup \{v\})}{c(S) + 2e(S)}$$

because $c(S \cup \{v\}) = c(S) - k_{v,S} + d_v - k_{v,S}$, and $e(S \cup \{v\}) = e(S) + k_{v,S}$, where $k_{v,S}$ is the number of edges with one endpoint being node v and the other endpoint in S, and d_v is the number of neighbors or degree of node v.

$$c(S \cup \{v\}) > c(S) + c(S) \cdot \frac{d_v}{c(S) + 2e(S)}$$

$$\frac{c(S \cup \{v\}) - c(S)}{d_v} > \frac{c(S)}{c(S) + 2e(S)}$$

$$1 - 2 \cdot \frac{k_{v,S}}{d_v} > \frac{c(S)}{c(S) + 2e(S)}$$

Conversely, conductance will decrease when:

$$1 - 2 \cdot \frac{k_{v,S}}{d_v} < \frac{c(S)}{c(S) + 2e(S)}$$

The increase or decrease in conductance, then, is a function of $\frac{k_{v,S}}{d_v}$. □

From the proof of proposition 1, we can know that if we want to maximize the conductance, we only have to, for every step of iteration, *minimize* $\frac{k_{v,S}}{d_v}$, as shown in Algorithm 1.

Algorithm 1 Greedy Algorithm for MCP.

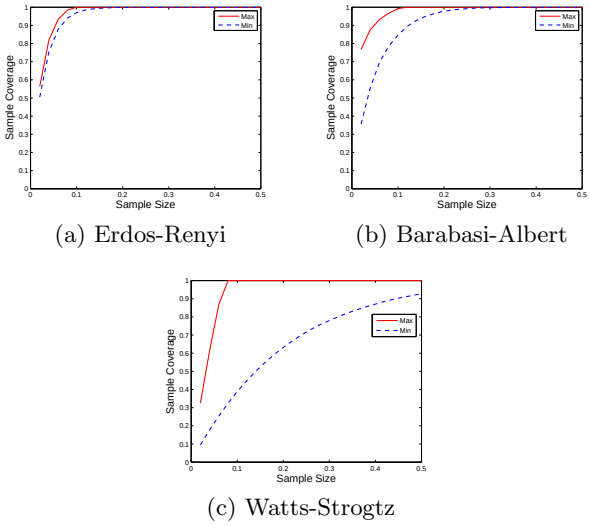
Input:
Graph $G = (V, E)$;
the sample size k.
Output:
constructed sample S.

```
S = ∅;
u = argmax_{v∈V} |N({v})|;
S = S ∪ {u};
while |S| ≤ k do
    v = argmin_{v∈V−S} k_{v,S}/d_v;
    S = S ∪ {v};
end while
return S;
```

We now use the greedy algorithm to construct sample coverage for both synthetic random graphs and real-world networks.

4.2 Coverage Results

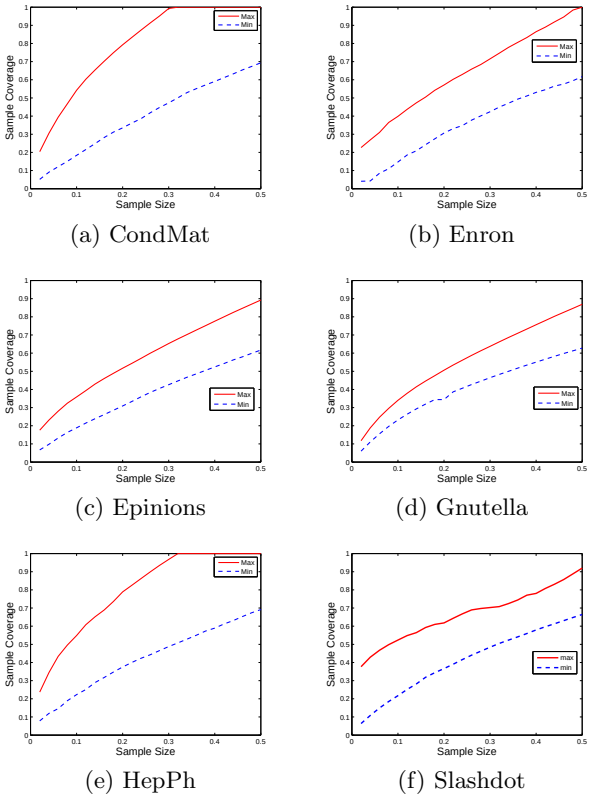
We construct sample coverage of 3 well-known artificial networks and 6 real-world networks described in Section 3.3. The results for each network are shown in Figure 2 and 3. We can immediately see that for all the networks it is the maximum rather than minimum conductance that has higher sample coverage. And the results demonstrate our ideas. We also see that except the Erdos-Renyi network, the coverage is much higher through maximizing than minimizing



(a) Erdos-Renyi (b) Barabasi-Albert

(c) Watts-Strogtz

Figure 2: Coverage results for artificial networks.



(a) CondMat (b) Enron

(c) Epinions (d) Gnutella

(e) HepPh (f) Slashdot

Figure 3: Coverage results for real-world networks.

the conductance, and for the Erdos-Renyi network, the coverage is a little higher. For the 3 artificial networks, it is clear that they all exhibit rapidly increasing maximum conductance, and the BA model exhibits higher sample coverage than the other 2 networks.

For the 6 different real-world networks, firstly, we can easily see that they all have a lower and less rapidly increasing maximum conductance than the 3 artificial networks. Secondly, different types of networks exhibit very different coverage properties. For instance, the size scale required to obtain sample coverage of 1 in the CondMat, HepPh and HepTh networks is near 0.3. For the Epinions and Gnutella networks, it still does not reach 1 when the sample size is 0.5.

5. CONDUCTANCE SEARCH

5.1 Search Strategy

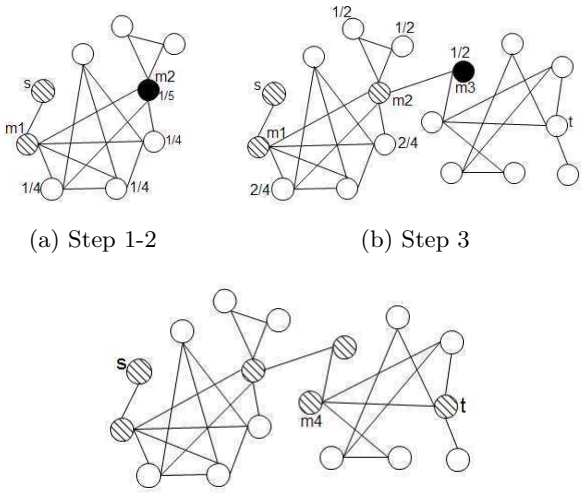
In Section 4, through constructing sample coverage, we have seen that it is often a relatively small set of nodes that is connected to a large portion of the network. If one were able to easily locate this set of nodes, then the efficiency of search might vastly be improved because this would quickly take us within one hop of many other nodes. How can these nodes be accessed during the course of a decentralized search? We use the approximation algorithm from Section 4. We refer to this as *conductance search* (CS). In the conductance search, the next node in the search is selected so as to maximize the conductance. Let S be the set of nodes visited thus far, and let c be the current node (most recently visited node, $c \in S$). Then in the conductance search, the next hop is selected from among the unvisited neighbors of c. That is, the next node is v where:

$$v = \underset{v \in N(\{c\}) - S}{\operatorname{argmin}} \frac{k_{v,S}}{d_v}$$

Recall that $k_{v,S}$ is the number of edges with one endpoint being node v and the other endpoint in S, and d_v is the number of neighbors or degree of node v. The key difference from the approximation algorithm described in Algorithm 1 is that the next hop is selected from the neighborhood of the current node c rather than all of $V - S$. Occasionally, if more than one neighbor satisfies the criteria, the strategy selects randomly among them.

Actually, there are 3 variations for a certain search strategy. They are (i) *unrestricted* where the current node may ignore prior visitations and can forward the message to one of its neighbors, (ii) *no-retracing* where the current node c can forward the message to one of its neighbors *except* the last visited nodes and (iii) *self-avoiding* where the current node can forward the message to one of its *unvisited* neighbors. Hence, our search strategy is self-avoiding conductance search. It is possible to construct other variations of search strategies, but the self-avoiding version described above is consistently outperformed other 2 variations in our simulations. Therefore we do not discuss them any further.

We use the sample network depicted in Figure 1 to clearly illustrate the proposed conductance search algorithm step-by-step. Starting from node s, we have to forward to $m1$ which then advance to $m2$ because it has the smallest of $\frac{k_{v,S}}{d_v}$ (see Figure 4(a)). Then the unvisited neighbors of node



(a) Step 1-2 (b) Step 3

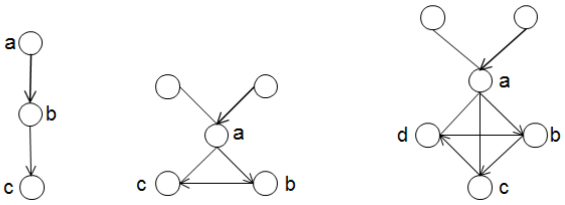
(c) Step 4-5

Figure 4: Step-by-step to illustrate CS. The number next to the nodes are the value of $\frac{k_{v,S}}{d_v}$.

$m2$ have the same value of $\frac{k_{v,S}}{d_v} = 1/2$, due to the space limitations, we suppose that the next node is happened to be node $m3$ (see Figure 4(b)). Finally, $m3$ passes the message to $m4$ who forwards directly to target node t (see Figure 4(c)). Then we complete the search process from s to t.

We conclude this section with an important remark. As mentioned previously, during the search unvisited nodes are always preferentially chosen over visited nodes. But it may be the case that all the neighbors of a current node are already visited. There are several approaches to dealing with this situation. For example, the next step might be chosen uniformly at random from among the visited neighbors, which used in [18] and [27]. But during our preliminary testing, we found that this approach may cause bad performance due to the small local structures shown in Figure 5. For instance, in Figure 5(c), if we visit nodes from a, b, c to d, and all the neighbors of d have been visited, then d may randomly choose b who faces the same situation as d. This random choice scene will continue unless a who has at least one unvisited neighbor is selected. The worst situation is the search process traps into an loop among b, c and d. So the approach of randomly selecting visited neighbors is not a good choice. We adopt an approach where we retrace the message to the last node and this approach is realistic because, in the message passing scenario, if Jason just received a message from Eric but found he had no neighbor nodes to forward the message, he had to return to Eric.

The pseudocode of conductance search is shown in Algorithm 2. We use a stack T to store the visiting sequence. Every time we forward the query message to a new node, we push it into T (see Statement 13). And every time we face the situation that all the neighbors of the current node have been visited, we retrace the message to the last node. This equals to pop from T (see Statement 10). The search process will not be terminated until the target node t is the neighbor of the current node c.



(a) two nodes (b) 3-clique (c) 4-clique

Figure 5: Small structures may cause bad performance.

Algorithm 2 Self-avoiding Conductance Search.

Input:
Graph $G = (V, E)$;
initial source node s;
target node t.
Output:
number of steps.

```
1: if t ∈ N({s}) then
2:    return 0;
3: end if
4: S = ∅, T = [];
5: steps= 0, c = s;
6: S = S ∪ {s}, push(T, c);
7: while t ∉ N({c}) do
8:    c =peek(T)
9:    if ∀u ∈ N(c), u ∈ S then
10:      pop(T);
11:   else
12:      v = argmin_{v∈N({c})−S} k_{v,S}/d_v;
13:      push(T, v), S = S ∪ {v};
14:      steps++, c = v;
15:   end if
16: end while
17: return steps;
```

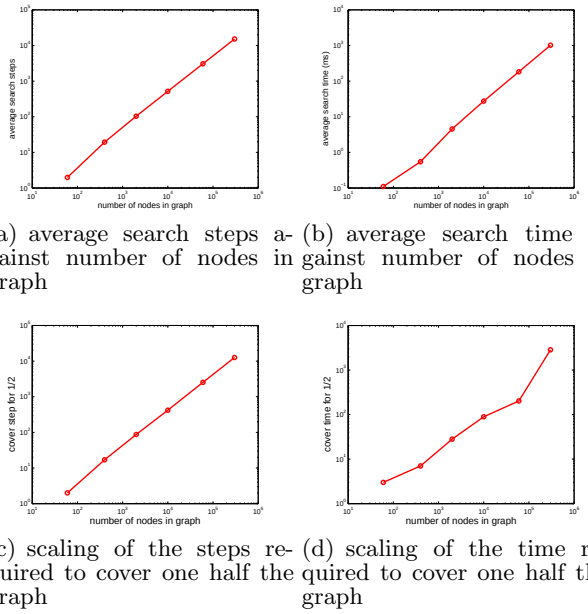
5.2 Scalability Evaluation

We have been confronted by the existing network data on a scale previously unimagined, so we are expecting the conductance search has the property of high scalability. To examine it, we conducted 2,000 randomly selected search tasks on different size of Watts-Strogtz networks. The results are shown in Figure 6. Figure 6(a) and Figure 6(b) show the scaling of the average search step and average search time with the size of the graph. Consider, on the other hand, the number of steps and spending time it takes to cover half the graph. Figure 6(c) and Figure 6(d) just show these two measures. As can be seen, the conductance search is well explained by a polylogarithmic relation to network size (i.e., the number of nodes), indicating a high scalability potential for searching in a growing information space.

6. EXPERIMENTAL EVALUATION

In this section, we begin to evaluate conductance search strategy with other existing strategies on various network datasets.

6.1 Experimental Setup



(a) average search steps against number of nodes in graph (b) average search time against number of nodes in graph

(c) scaling of the steps required to cover one half the graph (d) scaling of the time required to cover one half the graph

Figure 6: Scalability Evaluation Results

We evaluate each search strategy on each network described in Section 3.3. In our simulation, we assume that each node knows its neighbors and passes received messages to them based on one of the search strategies. The search ends when the message is passed to a neighbor of the target, at which time the message-holder can pass the message directly to its destination. We compare the conductance search strategy to several popular search strategies in complex networks. These strategies include degree search, breadth-first search, random walk and recently proposed expansion search:

Degree Search (DS). In the degree search [2], the next node in the search is the *unvisited* neighbor of the current node c with highest degree. That is, we visit node v where:

$$v = \underset{v \in N(\{c\})-S}{\operatorname{argmax}} |N(\{v\})|$$

Adamic et al. [2] analytically and empirically showed that the strategy is efficient for power-law networks.

Expansion Search (XS). The expansion search strategy was proposed by Maiya and Berger-Wolf [18]. At each step in the search, the search query is forwarded from current node c to the *unvisited* neighbor v where:

$$v = \underset{v \in N(\{c\})-S}{\operatorname{argmax}} |N(\{v\}) - (N(S) \cup S)|$$

Random Walk (RW). In random walk [22] search strategy, the current node forwards the query to exactly one randomly selected *unvisited* neighbor.

Breadth-First Search (BFS). In BFS strategy, each node sends a copy of the query to each of its neighbors. In our simulation, we evaluate a hypothetical BFS strategy in which each message holder forwards a copy of the query only to those neighbors who have not yet received it because it is difficult to determine which neighbors have already seen

the query due to no information transfer between the various copies of the query. It is also difficult to know when one of the copies reaches the target node, so all copies of the query terminate as soon as at least one copy of the query is successfully reaches its destination. It should be noted that this strategy needs less steps than real BFS, but we can test the true performance of flooding-based strategies because if this strategy, with better power than real BFS, still cannot match the performance of other strategies, then BFS strategy may not be useful.

For each network we randomly choose 500 nodes as source nodes, and for every source node we choose 20 nodes as target nodes. Then we complete 10,000 search tasks and compute the average search step. Here, the step means a single hop taken by a single query message in the search process. If there are multiple copies of the message, as in the case of BFS, then the number of steps is defined as the total number of hops taken by all copies of the message.

6.2 Experimental Results

In this section, we show the experimental results and then analyze the results from two perspectives. They are (i) performance of search strategies, this analysis focuses on different strategies on the same network; (ii) the searchability of networks, on the other hand, it sheds light on a certain strategy on different networks. We discuss each separately.

6.2.1 On the Performance of Search Strategies

Figure 7 shows the average number of steps to find the target node divided by number of nodes in networks. As can be seen, the conductance search (CS) and expansion search exhibit the best overall performance of average steps number. But from Table 2 we can see that the average search time of expansion search is much higher than the other strategies. So we can get the conclusion that *conductance search* has better performance than other search strategies from time complexity and number of search steps. We also find that the conventional search strategies (BFS and RW) and the newly proposed approaches (CS, XS and DS) have a clear performance difference. Overall, these new approaches have much better performance than traditional strategies.

Conventionally, BFS and RW strategies are wildly used in searching networks such as P2P systems, Web crawling and graph sampling. For example, BFS based flooding strategies are used in Gnutella to find answers quickly, and it also used in web crawling to yield high PageRank Web pages (see e.g. [20]). But the results shows that our hypothetical BFS strategy, which need less steps to find the targets, still cannot outperform other approaches.

6.2.2 On the Searchability of Networks

Network searchability describes the extent to which a network can be efficiently searched. Typically, on any given good searchability network, all search strategies fare relatively well. On the other hand, all search strategies perform bad. A direct illustration is shown in Figure 8. In Figure 8, for every network we draw the best performance (i.e. the lowest) number of steps divided by number of nodes (*Portion*). As can be clearly seen, the 3 artificial networks are much more searchable than the 6 real-world networks. For

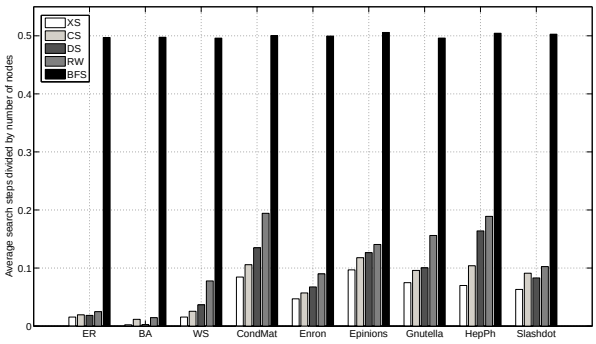


Figure 7: Average search steps to find the target node.

Table 2: Average time(s) for one search task.

	XS	CS	DS	RW	BFS
Erdos-Renyi	1.6229	0.1892	0.0242	0.0066	0.0054
Barabasi-Albert	0.8813	0.3040	0.0310	0.0067	0.0056
Watts-Strogtz	1.1137	0.1241	0.0242	0.0154	0.0093
CondMat	4.1091	0.0609	0.0246	0.0126	0.0074
Enron	8.1252	0.1813	0.0395	0.0194	0.0104
Epinions	91.1339	1.7335	0.1549	0.0602	0.0338
Gnutella	34.0146	0.1307	0.0528	0.0272	0.0163
HepPh	0.9505	0.1957	0.0310	0.0095	0.0065
Slashdot	70.1447	1.6501	0.1525	0.0567	0.0346

the 3 artificial networks, ER and WS seem less searchable than BA. The explanation is that the BA model is a scale-free network. Scale-free networks are those networks with a power-law degree distribution, which means that the probability of a given degree d is proportional to $d^{-\beta}$, where β is a parameter called the degree exponent. Most nodes in such networks have only a few edges, but a few nodes have much higher degree. And the degree exponent of BA model is $\beta = 3$. [2] proved that DS (or XS) is very efficient in large degree exponent networks. Actually, $\beta = 3$ is rather large because the exponent of most real-world networks is between 2 and 3 [21].

For the six real-world networks, Enron and Slashdot are more searchable than other 4 networks. This leads us to a natural question: what causes performance differences of the same strategy (CS) on different networks, that is, which properties affect network searchability? We discuss from several structural properties. (i) *density and average degree.* Denser networks seems to be more searchable than extremely sparse networks because nodes have larger neighborhoods (larger degree) in denser networks are easier to explore. For example, the 3 artificial networks have larger density and average degree than the other 6 real-world networks, and as analyzed above, the artificial networks are more searchable than others. But density and average degree fails to fully explain network searchability. For instance, although all the 3 artificial networks have very similar density and average degree (see Table 1), the BA network is more searchable than ER and WS networks (see Figure 8). (ii) *average path length.* Apparently, this property fails to explain searchability. The

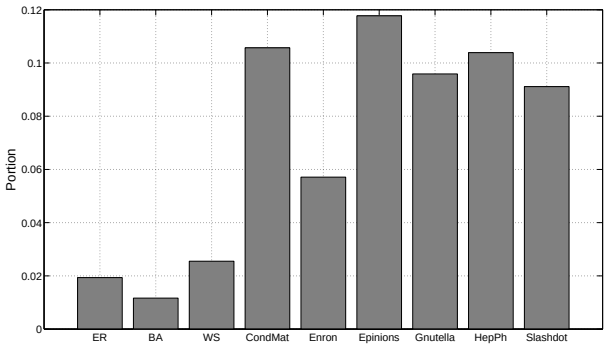


Figure 8: Searchability of different networks

networks we experimented on have very similar average path lengths (see Table 1), but they exhibit very different levels of searchability (see Figure 8). (iii) *clustering coefficient.* We think that high clustering which real-world networks often exhibit can facilitate searchability. That is, high clustering networks are more searchable. Firstly, a cluster in a network exhibit higher clustering coefficients and nodes in the same cluster share more neighbors than nodes in different clusters. Then based on this, if we can visit the nodes from many different clusters during the search course, we can get more nodes within one hop.

7. CONCLUSIONS

In this paper, we have shown that through introducing the concept of sample coverage, we can get high coverage samples by maximizing the sample conductance. Moreover, based on sample conductance, we propose a search strategy who has a better performance than other typical strategies in average number of steps to find the target nodes and in time complexity. In this strategy, for every step during the course of search, we choose a node that can maximize the conductance of the visited sample. Finally, we discuss the relationship between network searchability and structural properties, and we think that high clustering coefficient or community structure can facilitate decentralized search. For future work, we plan to further investigate the interplay between the search performance and various graph structural properties and their effect on search.

8. ACKNOWLEDGMENTS

This work is supported by the National Natural Science Foundation of China (Grant No.60905025, 90924029, 61074128).

9. REFERENCES

[1] L. Adamic and E. Adar. How to search a social network. *Social Networks*, 27(3):187–203, 2005.

[2] L. A. Adamic, R. M. Lukose, A. R. Puniyani, and B. A. Huberman. Search in power-law networks. *Physical Review E*, 64(4):046135, 2001.

[3] S. Arora, S. Rao, and U. Vazirani. Expander flows, geometric embeddings and graph partitioning. *Journal of the ACM*, 56(2):1–37, 2009.

[4] A. Banerjee and S. Basu. A social query model for decentralized search. In *Workshop on Social Network Mining & Analysis, ACM KDD*, 2008.

[5] A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[6] O. Şimşek and D. Jensen. Navigating networks by using homophily and degree. *Proceedings of the National Academy of Sciences*, 105(35):12758–12762, 2008.

[7] I. S. Dhillon, Y. Guan, and B. Kulis. Weighted graph cuts without eigenvectors: A multilevel approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29:2007, 2007.

[8] P. S. Dodds, R. Muhamad, and D. J. Watts. An experimental study of search in global social networks. *Science*, 301(5634):827–829, 2003.

[9] P. Erdös and A. Rényi. On random graphs, i. *Publicationes Mathematicae (Debrecen)*, 6:290–297, 1959.

[10] P. Fraigniaud and G. Giakkoupis. On the searchability of small-world networks with arbitrary underlying structure. In *Proceedings of the 42nd ACM Symposium on Theory of Computing*, pages 389–398, 2010.

[11] S. Hangal, D. MacLean, M. S. Lam, and J. Heer. All friends are not equal: Using weights in social graphs to improve search. In *Workshop on Social Network Mining & Analysis, ACM KDD*, 2010.

[12] R. Kannan, S. Vempala, and A. Vetta. On clusterings: Good, bad and spectral. *Journal of the ACM*, 51(3):497–515, 2004.

[13] J. Kleinberg. The small-world phenomenon: An algorithmic perspective. In *Proceedings of the 32nd ACM Symposium on Theory of Computing*, pages 163–170, 2000.

[14] J. Kleinberg. Complex networks and decentralized search algorithms. In *Proceedings of the International Congress of Mathematicians (ICM)*, 2006.

[15] B. Klimt and Y. Yang. Introducing the enron corpus. In *CEAS*, 2004.

[16] J. Leskovec, J. Kleinberg, and C. Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data*, 1(1):2+, 2007.

[17] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. Technical report, 2008.

[18] A. S. Maiya and T. Y. Berger-Wolf. Expansion and search in networks. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, pages 239–248, 2010.

[19] S. Milgram. The small world problem. *Psychology Today*, 2:60–67, 1967.

[20] M. Najork. Breadth-first search crawling yields high-quality pages. In *Proceedings of the 10th International World Wide Web Conference*, pages 114–118, 2001.

[21] M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45(2):167–256, 2003.

[22] J. D. Noh and H. Rieger. Random walks on complex networks. *Physical Review Letters*, 92:118701, 2004.

[23] M. Richardson, R. Agrawal, and P. Domingos. Trust management for the semantic web. In *Proceedings of the 2nd Internatioanl Semantic Web Conference*, pages 351–368, 2003.

[24] M. Ripeanu, I. Foster, and A. Iamnitchi. Mapping the gnutella network: Properties of large-scale peer-to-peer systems and implications for system. *IEEE Internet Computing Journal*, 6:2002, 2002.

[25] S. Schaeffer. Graph clustering. *Computer Science Review*, 1(1):27–64, 2007.

[26] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transcations of Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[27] O. Simsek and D. Jensen. Decentralized search in networks using homophily and degree disparity. In *19th International Joint Conference on Artificial Intelligence (IJCAI)*, 2005.

[28] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, 1998.

Combining Feature Weighting and Semantic Similarity Measure for a Hybrid Movie Recommender System

Ugur Ceylan
Department of Computer Engineering,
Baskent University
Ankara, Turkey
uceylan@baskent.edu.tr

Aysenur Birturk
Department of Computer Engineering,
METU
Ankara, Turkey
birturk@metu.edu.tr

ABSTRACT

In this study, we propose a hybrid movie recommender system that is based on content-based collaborative filtering paradigm in order to cope with sparsity and item cold-start problems of pure collaborative filtering. The content-based part of the system is based on a priori defined ontology-based metadata and user models. Content-based user models are generated using the multiple criteria approach in order to determine the weights of features that affect users' decision process. By using ontology-based metadata and user models, the system finds the similarities between items and recommends items to users using similarities between items and users' explicit ratings. Additionally, a feature selection method, namely sequential forward selection, is applied in order to improve the quality of recommendations.

Keywords

content-boosted collaborative filtering, semantic similarity, ontology, sparsity, item cold-start, multiple criteria

1. INTRODUCTION AND RELATED WORK

Collaborative filtering systems suffer from some problems such as sparsity and item cold-start problems [20, 7]. The simplest technique used for overcoming the sparsity problem is default voting [4]. In this technique, in order to increase the number of overlapping rated items by users, a default rating is added for items which don't have rating values. Thus, success of the algorithm that finds the similarity of users is improved. Default voting can also be used for the item-cold start problem by adding default rating to items that has been rated by insufficient number of users. Another technique for dealing with the sparsity problem is using dimensionality reduction techniques such as Singular Value Decomposition (SVD) [23]. By applying SVD, user-item rating matrix may become less sparse.

Hybrid recommendation approach is generally implemented by combining collaborative and content-based filtering approaches to cope with the drawbacks of these two filtering methods [3]. Different from the collaborative filtering, in content-based filtering, the system tries to recommend items that have similar contents with the items liked by the active user. Some hybrid approaches are as follows [5]: In one approach, independent recommendations are generated by using content-based and collaborative filtering separately and then these recommendations are combined [22]. The other approach gives some weight values to collaborative and content-based predictions. Then recommendations are made by using the weighted sum of the predicted ratings [7]. In another hybrid approach, content-based filtering is used for enhancing user-item rating matrix by predicting the unknown ratings. Then collaborative filtering approach is applied on enhanced user-item rating matrix. This hybrid approach is called content-boosted collaborative filtering [20].

In this paper, we propose a hybrid approach based on content-boosted collaborative filtering presented in [20] to cope with sparsity and item cold-start problems of collaborative filtering. The contribution of our study is that the content-based filtering in our approach uses user models and semantic similarity measures on ontology-based metadata, based on the studies in [6, 18, 17], instead of naive Bayesian classifier [21] which is mentioned in [20]. The content-based part of our approach utilizes user models to determine the feature-weights. These feature weights is used for calculating the semantic similarities of items using ontology-based metadata in movie domain. After finding similarities of the movie items, user-item matrix is enhanced using these similarities and explicit ratings of the active user. And then, collaborative filtering is applied on enhanced user-item matrix to generate recommendations.

2. PROPOSED APPROACH

In order to reduce the sparsity and item cold-start problem in collaborative filtering, we propose a hybrid approach, called $SEMCBCF$, that uses semantic similarities between items using ontology-based metadata and content-based user models in the movie domain. The flow diagram of our approach is shown in the Figure 1.

$SEMCBCF$ is based on content-boosted collaborative filtering paradigm in [20]. The content-based filtering part ($SEMCBF$) of the hybrid approach is used for enhancing the user-item rating matrix by making predictions for unknown user-item pairs. Afterwards, $SEMCBCF$ applies collaborative filtering to enhanced user-item rating matrix to generate recommendations to an active user. $SEMCBCF$ consists of five phases:

The 5th SNA-KDD Workshop '11 (SNA-KDD'11), August 21, 2011, San Diego CA USA . Copyright 2011 ACM 978-1-4503-0225-8 ...$5.00.

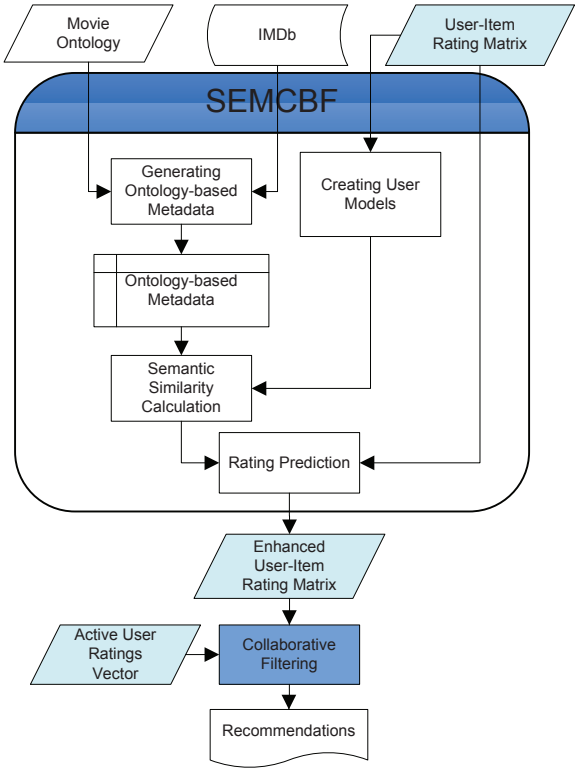


Figure 1: Flow diagram of SemCBCF.

1. Generating ontology-based metadata
2. Creating user models
3. Semantic similarity calculation
4. Rating prediction
5. Collaborative filtering

2.1 Generating Ontology-based Metadata

In order to find semantic similarities between items, first, ontology model and metadata model have to be defined. Because the movie domain is chosen as recommendation domain of our approach, a *movie ontology* has to be defined. In third phase, based on the studies in [18] and [17], our approach finds similarities between items that are movies in the movie domain.

Ontology model is defined as follows [18]:

$$O := \{\tilde{C}, \tilde{P}, \tilde{A}, H^c, prop, att\} \quad (1)$$

For ontology model, $\tilde{C}$,$\tilde{P}$ and $\tilde{A}$ are sets which consist of concepts,relation and attributes identifiers respectively. H^c is called concept taxonomy which defines the hierarchical relations between concepts. *prop* and *att* are functions that define non-taxonomical relations between concepts and literal values of concepts.

Metadata represent the instances and relationships between them in a domain. Metadata model is defined as follows

$$MD := \{O, \tilde{I}, \tilde{L}, inst, instl, instr\} \quad (2)$$

For metadata model, $\tilde{I}$ and $\tilde{L}$ are sets which consist of instances and literal values respectively. *inst*, *instr*, *instl* are functions that define concept instantiation, relation instantiation and attribute instantiation. Predicates that are used in the following sections are shown in the Table 1 with their meanings and examples for movie ontology shown in Figure 2.

Table 1: Predicates, Meanings and Examples.

Predicate	Meaning	Example
$H^c(C_1, C_2)$	Concept C_1 is a subconcept of concept C_2	$H^c(Director, Person)$
$P(C_1, C_2)$	P is a relation with domain C_1 and range C_2	$hasCast(Movie, Cast$
$A(C_1)$	A is an attribute of C_1	$Rating(Movie)$
$C(I)$	I is an instance of concept C	$Action(Men\ In\ Black)$
$P(I_1, I_2)$	Instance I_1 has a P relation to instance I_2	$hasCast(Men\ In\ Black, Will\ Smith)$
$A(I_1, L)$	Instance I_1 has attribute an A with value of L	$Rating(Men\ In\ Black,$ "7")

The extracted information of a movie consist of a set of *values* and each value belongs to a $feature$. Thus, for each movie, the web crawler parses the IMDb web page of that movie according to the predefined features. In $SEMCBCF$, 10 features of the movie domain are used. These are *cast*, *director*, *writer*, *language*, *genre*, *runtime*, *releasedate*, *country*, *color* and $IMDb\ rating$. Some features are multi-valued. For example, if a movie has 2 writers, the number of the features that belongs to *writer* dimension is 2.

A free, open source ontology editor and knowledge-base framework called Protege (*http://protege.stanford.edu*) is used for defining movie ontology. After defining the movie ontology, the metadata are generated based on the defined movie ontology and the content of movies extracted from IMDb (The Internet Movie Database, $http://www.imdb.com$). For the simplicity, in the rest of this paper, when we say ontology, unless explicitly expressed, we mean ontology and ontology-based metadata models. A part of the movie ontology used in our approach, is shown in the Figure 2.

$Movie$ and $Feature$ concepts are sub-concepts of *root* concept. The values of *genre* feature are represented by concepts and these concepts are linked to $Movie$ concept by $subConceptOf$ links. Hierarchy (taxonomy) of the descendants of $Movie$ concept is created by the help of a group of students studying Radio-Television and Cinema. Briefly, the genres that are closer in taxonomy is more similar. It can be seen from Figure 2 that a value of a feature other than *genre* can be an instance or a literal in movie ontology. For example, $English$, which is value of language feature, is an instance of concept $Language$. On the other hand, 1997, which is value of release date feature, is a literal. The features and their representation formats in movie ontology are shown in Table 2.

2.2 Creating User Models

In $SEMCBF$, content-based user models are used to calculate the similarity of items in the system. The reason for

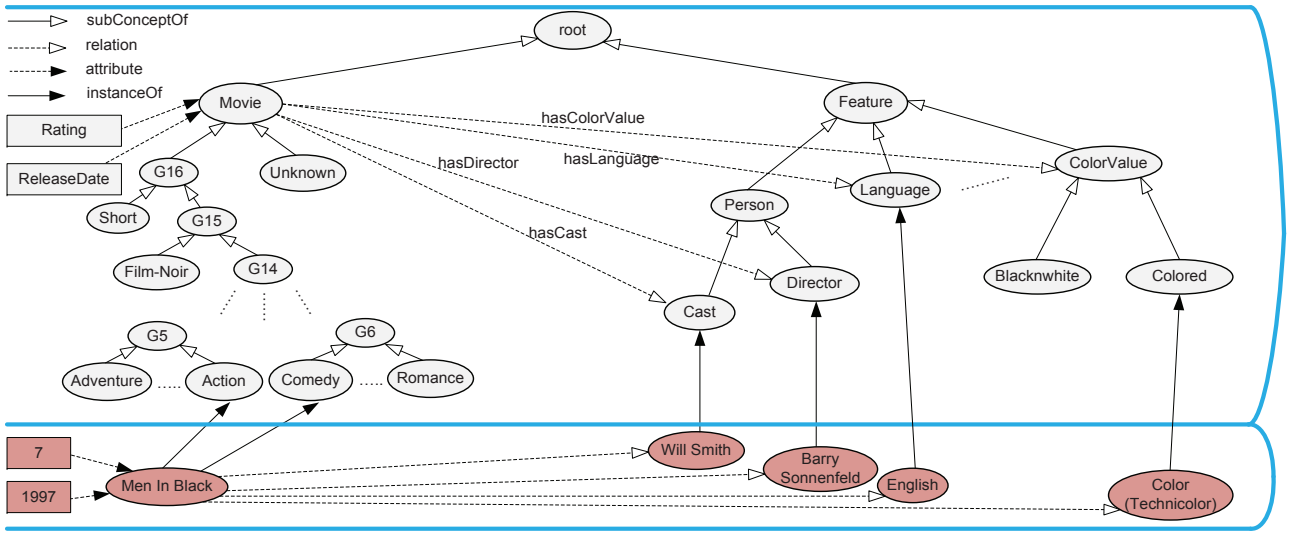


Figure 2: Movie ontology.

Table 2: Features and Representations in Movie Ontology.

Features whose Values are Instances	Features whose Values are Literals	Features whose Values are Concepts
cast, director, writer, language, country, color	runtime, release date, IMDb rating	genre

creating user models is to discover effects of each feature to the preference of the users. For instance, some users consider *writer* feature of the movie important more than *director* feature of the movie. For such users, when calculating the similarities of movies, our approach gives more weight to the similarities of the *writer* feature of the movies than the *director* feature of the movies. In order to determine the feature weights of users, we use multiple criteria approach in [6].

Every movie in the system is represented as a vector called movie feature vector (MFV) which is as follows:

$$MFV_m = ((f_{1,1}, ..., f_{1,k_1}), ..., (f_{N,1}, ..., f_{N,k_N})) \quad (3)$$

Each value has an index number for its corresponding feature. This index number has a range between 1 and the number of values belonging to corresponding feature. And also each feature has a number in the system. In MFV, $f_{i,j}$ represents the value with the index j of the feature i, k_i represents the number of possible values for the feature i and N represents the number of features. For instance, *genre* feature, which is the 3rd feature in our approach, has 24 values such as *comedy*, *action*, *horror* etc. in our system. And *comedy* is the 5th value for *genre* feature. Then, if a movie is not a comedy movie than $f_{3,5}$ is set to 0. But if a movie has a feature-value that is represented by $f_{i,j}$, $f_{i,j}$ is set a value that depends on the selected method. We use three types of methods to set $f_{i,j}$ in MFV of a movie: $Default$, Idf, Ifw.

Simplest method used to construct MFV vector is $Default$. In this method if a movie has a feature-value that is represented by $f_{i,j}$, than $f_{i,j}$ is set to 1. In the second method, *inverse document frequency* (idf) of the feature values, which is mentioned in [1], is used. Idf is defined as follows:

$$Idf_{f_{i,j}} = \log \frac{M}{m_{f_{i,j}}} \quad (4)$$

where M is the number of all movies and $m_{f_{i,j}}$ is the number of movies that has a feature-value that is represented by $f_{i,j}$. If a movie has a feature-value that is represented by $f_{i,j}$, than $f_{i,j}$ is set to $Idf_{f_{i,j}}$. Similarly, in the third method, $f_{i,j}$ is set to *item feature weight* (ifw) [14] of the feature-value. Ifw is defined as follows:

$$Ifw_{f_{i,j}} = \log \frac{M}{m_{f_{i,j}}} \times \log k_i \quad (5)$$

After constructing the MFVs of all movies in the system, to determine the users' preferences, user preference vector (UPV) of each user is constructed. UPV of the user u is as follows:

$$UPV_u = ((pr_{1,1}, ..., pr_{1,k_1}), ..., (pr_{N,1}, ..., pr_{N,k_N})) \quad (6)$$

In order to construct UPV of a user, MFVs of the movies that has been rated the user and the rating values are used.

$$UPV_u = \frac{\sum_{m \in \hat{M}} 0.2 \times r_{u,m} \times MFV_m}{|\hat{M}|} \quad (7)$$

In the equation 7, 0.2 is used for transforming the real rating value in range 1-5 to normalized rating value in range 0-1.

User weight vectors (UWVs) represents the opinion of users about the features. Weight values of the features is obtained using UPVs. UWV of the user u is follows:

$$UVW_u = (w_1, w_2..., w_N) \quad (8)$$

where $w_i = pr_i / \sum_{i=1}^{N} pr_i$

In order to find the value of w_i in UVW, we use two different methods to set pr_i: $Default$, $Stddev$. When using

$Default$ method pr_i is the biggest value of the values that represents the preference of the user about feature i. In the second method, $Stddev$, pr_i is standard deviation of the values that represents the preference of the user about feature i.

After constructing the UWVs of all users in the system, in order to avoid scalability problem, we apply clustering on UVWs for grouping users that have similar UWVs. For clustering users, we apply *K Means* and *Expectation Maximisation* algorithms separately using $Weka$ API ($http$: $//www.cs.waikato.ac.nz/ml/weka$).

Cluster weight vector (CWV) represents the opinion of the group of users in a cluster about features. The feature-weights of the clusters is calculated by the UWVs in the system. CWV of the cluster Cl_i is as follows:

$$CWV_{Cl_i} = \frac{\sum_{u \in U(Cl_i)} UWV_u}{|U(Cl_i)|} \tag{9}$$

where $U(Cl_i)$ is the users in cluster Cl_i. These feature-weights of the clusters is used in similarity calculation of the movies.

2.3 Semantic Similarity Calculation

In order to recommend items, a similarity measure between items has to be defined first. Every cluster has different feature-weights. By using these weights, for each cluster, different similarity values are found. And then, by using the active user's collaborative user model, which is a vector that consists of user's ratings, and similarities between items for the active user's cluster, we predict the ratings of unrated items which will be given by the active user. In $SEMCBF$, to calculate similarities between items which are described by ontology, we use three types of similarity measures[18]:

- Taxonomy similarity(TS)
- Relation similarity(RS)
- Attribute similarity(AS)

Taxonomy similarity between two instances (TS) is based on their corresponding concepts' positions in concept taxonomy (H^c) which is defined in ontology model. Basically, the idea behind taxonomy similarity is that closer concepts in taxonomy are more similar. Suppose that comedy and horror are sub-concepts of movie concept. MovieA and MovieB are comedy movies and MovieC is a horror movie. In such an ontology-based metadata, taxonomy similarity between MovieA and MovieB is higher than the taxonomy similarity between MovieA and MovieC.

An instance can be instance-of two different concepts in ontology. For example in Figure 2, $MenInBlack$ is instance-of both *Action* and *Comedy* concepts. Therefore, in order to find taxonomy similarities between instances (TS), first, taxonomy similarities between concepts (TSC) have to be defined. Then, a comparison of instances' corresponding concepts is made to find TS.

4 different methods/measures are used for calculating TSC. Concept match-CM of two concepts [18] is defined as follows:

$$CM(C_1, C_2) = \frac{|UC(C_1, H^c) \cap UC(C_2, H^c)|}{|UC(C_1, H^c) \cup UC(C_2, H^c)|} \tag{10}$$

where UC, upwards cotopy, is the following.

$$UC(C_i, H^c) = \{C_j \in \tilde{C} | H^c(C_i, C_j) \vee C_i = C_j\} \tag{11}$$

Then TSC using concept match, the first method, can be defined.

$$TSC_{CM}(C_i, C_j) = \frac{CM(C_i, C_j)}{2} \tag{12}$$

The second measure used for calculating TSC is a measure proposed by Wu and Palmer in [25]. TSC using Wu and Palmer's measure is the following.

$$TSC_{Wu\&Palmer}(C_i, C_j) = \frac{2 \times N_3}{N_1 + N_2 + 2 \times N_3} \tag{13}$$

where N_1 and N_2 are the number of $subConceptOf$ links from C_i and C_j to their most specific concept C_k that subsumes both of them and N_3 is the number of $subConceptOf$ links from C_k to the root of the ontology($Thing$ concept).

Lin's taxonomy similarity[16] is selected as the third measure in our approach. TSC between C_i and C_j can be calculated as follows:

$$TSC_{Lin}(C_i, C_j) = \frac{2 \times \log P(C_i)}{\log P(C_j) + \log P(C_k)} \tag{14}$$

where $P(C_n)$ is the probability that a randomly selected instance belongs to concept C_n, and C_k is the most specific concept that subsumes both C_i and C_j.

The last measure used in our approach is mentioned in [15]. In [15] different similarity calculation strategies are evaluated and the following similarity measure yields best performance.

$$TSC_{Mclean}(C_i, C_j) = e^{-\alpha l} . \frac{e^{\beta h} - e^{-\beta h}}{e^{\beta h} + e^{-\beta h}} \tag{15}$$

where l is the shortest path length between C_i and C_j, h is the depth of most specific concept in ontology. According to [15], optimal values of parameters α and β is 0.2 and 0.6 respectively.

After defining the taxonomy similarity between concepts, calculating taxonomy similarity between instances is reduced to calculating the similarity of two sets. So TS is defined as follows:

$$TS(I_1, I_2) = SSIM(CSET(I_1), CSET(I_2)) \tag{16}$$

where $CSET(I) = \{C \in \tilde{C} | C(I)\}$.

Similarity between two sets can be found using the similarities between their elements, in this case TSC of concepts, and using different methods. These methods are mentioned later in this section.

The second type of similarity measure using ontology-based metadata is relation similarity. Relation similarity (RS) between two instances is based on their relations to other instances in ontology. Suppose that DirectorX is the director of MovieA and MovieB, DirectorY is the director of MovieC. In this example, relation similarity between MovieA and MovieB is higher than the one between MovieA and MovieC because director of MovieA and MovieB is the same. For relation similarity measure, we use a modified version of Maedche and Zacharias's relation similarity measure [18]. RS between I_1 and I_2 for cluster Cl_i can be calculated

as follows:

$$RS_{Cl_i}(I_1,I_2) = \frac{\sum_{p\in P_{co-I}} w(Cl_i,p)OR(I_1,I_2,p,IN)}{\sum_{p\in P_{co-I}} w(Cl_i,p)+\sum_{p\in P_{co-O}} w(Cl_i,p)} + \frac{\sum_{p\in P_{co-O}} w(Cl_i,p)OR(I_1,I_2,p,OUT)}{\sum_{p\in P_{co-I}} w(Cl_i,p)+\sum_{p\in P_{co-O}} w(Cl_i,p)} \quad (17)$$

where $w(Cl_i,p)$ is the weight value of the feature, which is represented by the relation p, in CWV_{Cl_i}. P_{co-I} stands for 'incoming relations' and is the set of relations that allows $UC(C(I_1),H^c)$ and $UC(C(I_2),H^c)$ as range. P_{co-O} stands for 'outgoing relations' and is the set of relations that allows $UC(C(I_1),H^c)$ and $UC(C(I_2),H^c)$ as domain.

$OR(I_1,I_2,p,DIR)$ stands for the similarity for relation p and direction DIR between instances I_1 and I_2 where $DIR \in \{IN,OUT\}$. $OR(I_1,I_2,p,DIR)$ can be calculated by considering associated instances of I_1 and I_2 with respect to relation P and direction DIR. For example, if we consider the similarity for the relation *hasDirector* and direction OUT between two movie instances, we consider the directors of two movies. Similarly, if we consider the similarity for the relation *hasDirector* and direction IN between two directors, we consider the movies directed by these directors. Associated instances (A_s) of instance I_i with respect to relation P and direction DIR is as follows:

$$A_s(P,I_i,DIR) = \begin{cases} \{I_j : I_j \in \tilde{I} \wedge (P(I_j,I_i)\}, & \text{if DIR = IN} \\ \{I_j : I_j \in \tilde{I} \wedge (P(I_i,I_j)\}, & \text{if DIR = OUT} \end{cases} \quad (18)$$

After defining A_s, calculating $OR(I_1,I_2,p,DIR)$ is reduced to calculating the similarity of two sets that contains associated instances. So OR is defined as follows:

$$OR(I_1,I_2,p,DIR) = SSIM(A_S(P,I_1,DIR),A_S(P,I_2,DIR)) \quad (19)$$

If $A_S(P,I_1,DIR) = \emptyset$ or $A_S(P,I_2,DIR)) = \emptyset$ then OR is 0. As mentioned before, to find a similarity between two sets, similarities of their elements are considered. In this case, semantic similarities (SS) of elements of associated instances are used. The problem is that to calculate SSs of instances, RS is used and to calculate RSs of instances, SSs of associated instances are used. In order to avoid infinite cycles, a maximum depth of recursion has to be defined.

The advantage of using relation similarity is that the similarities of metadata, which are feature values of items in the system, are taken into account. Suppose that, in a system, movies have only one feature which is an actor played in movie and we try to find the similarity between two movies, MovieX and MovieY. MovieX has a feature value ActorA and MovieY has a feature value ActorB. If a user only rated movies in which only ActorA played, it is unable to predict the rating of MovieY by using naive Bayesian classifier [21] which is used in [20]. But in $SEMCBF$, similarity of MovieX and MovieY depends of the similarity between ActorA and ActorB. In a recursive manner, the similarity of ActorA and ActorB depends on similarity of other instances which have relations with ActorA and ActorB. Therefore, we can calculate a similarity value between these two movies and a prediction can be made.

Attribute similarity (AS) is the third similarity measure. Like relation similarity, attribute similarity(AS) between two instances depends on their attribute values. AS between instances I_1 and I_2 for cluster Cl_i is as follows:

$$AS_{Cl_i}(I_1,I_2) = \frac{\sum_{a\in P_A} w(Cl_i,a)OA(I_1,I_2,a)}{\sum_{a\in P_A} w(Cl_i,a)} \quad (20)$$

where $w(Cl_i,a)$ is the weight value of the feature, which is represented by the attribute a, in CWV_{Cl_i}. P_A represents the set of attributes that are attributes of both $UC(C(I_1),H^c)$ and $UC(C(I_2),H^c)$. $OA(I_1,I_2,a)$ is the similarity for attribute a between instances I_1 and I_2. Like calculation of OR, $OA(I_1,I_2,a)$ is calculated by considering associated literals of I_1 and I_2 with respect to the attribute a. For example if the attribute is *ReleaseDate*, associated literals of two movies are release dates of these movies. Associated literal (A_l) of I_i with respect to the attribute A is the following:

$$A_l(A,I_i) = \begin{cases} L_x, & \text{if } L_x \in \tilde{L} \wedge A(I_i,L_x) \\ \emptyset, & \text{otherwise} \end{cases} \quad (21)$$

The difference between A_s and A_l is that A_l can contain at most one literal unlike A_s. The reason of this difference is the characteristics of the movie ontology used in $SemCBF$. In movie ontology, an instance can have at most one attribute value (literal) with respect to an attribute. For instance, a movie has more than one director, but has at most one release date. In order that, rather than calculating similarity of two sets, similarity between attribute values is focused in order to calculate OA which is as follows:

$$OA(I_1,I_2,a) = LSIM(L_1,L_2,a) \quad (22)$$

where $L_1 = A_l(a,I_1)$ and $L_2 = A_l(a,I_2)$. If $L_1 = \emptyset$ or $L_2 = \emptyset$ then $OA(I_1,I_2,a)$ is 0. In our approach, all of the attributes used are numeric features of the instances like release date (as a year), runtime etc. Thus, we have to translate difference of numeric values to a similarity value that is between 0 and 1. For this translation, maximum difference of a numeric attribute A ($MDIF(A)$) have to be defined.

$$MDIF(A) = max\{(L_i - L_j) : A(I_1,L_i) \wedge A(I_2,L_j) \wedge I_1,I_2 \in \tilde{I}\} \quad (23)$$

After defining maximum difference of a numeric attribute, we can define the similarity between two numeric values (L_1 and L_2) of an attribute (a).

$$LSIM(L_1,L_2,a) = 1 - \frac{|L_1 - L_2|}{MDIF(a)} \quad (24)$$

In order to calculate TS and RS, we have to define the similarity between sets of elements. As mentioned before, similarity between two sets depends on the similarities between their elements and using different methods. Elements of these sets are *concepts* for TS calculation and *instances* for RS calculation. Similarities of elements are TSCs for TS and SSs for RS. The first method ($SSIM_1$) used for calculating the similarity between sets is mentioned in [18] in order to calculate the similarity between two sets of ele-

ments. $SSIM_1(S1, S2)$ is the following:

$$SSIM_1(S1,S2) = \begin{cases} \frac{\sum_{a \in S1} max\{SIM(a,b)|b \in S2\}}{|S1|}, & \text{if } |S1| \geq |S2| \\ \frac{\sum_{a \in S2} max\{SIM(a,b)|b \in S1\}}{|S2|}, & \text{otherwise} \end{cases} \quad (25)$$

The method mentioned in [24] is the second method ($SSIM_2$) used for calculating the similarity between two sets of elements.

$$SSIM_2(S1,S2) = \frac{\sum_{a \in S1} max\{SIM(a,b)|b \in S2\}}{|S1|+|S2|} + \frac{\sum_{b \in S2} max\{SIM(b,a)|a \in S1\}}{|S1|+|S2|} \quad (26)$$

The third method, which is used for calculating similarity between sets, forms pairs of elements in these two sets [2]. Pairing of elements is done by selecting two elements a and b, whose $SIM(a, b)$ is maximum, from sets $S1$ and $S2$ respectively. Then a and b are removed from their belonging sets. This pairing process is done until one of these sets has no more elements. $SSIM_3(S1, S2)$ using element pairing is the following:

$$SSIM_3(S1,S2) = \frac{\sum_{(a,b) \in Pairs(S1,S2)} SIM(a,b)}{max(|S1|,|S2|)} \quad (27)$$

where $Pairs(S1, S2)$ contains the pairs of elements that are formed by the above process.

The other methods used for calculating the similarity between sets are based on the methods used for calculating the distance of pair of clusters in hierarchical clustering algorithms. These methods are single-link ($SSIM_S$), complete-link ($SSIM_C$), and average-link ($SSIM_A$) [19].

In the single-link hierarchical clustering, distance between two clusters is the *minimum* distance between the pairs of elements belong these clusters. Since distance and similarity are inversely proportional, similarity between two sets is the *maximum* similarity between elements from these sets which is the following:

$$SSIM_S(S1,S2) = max\{SIM(a,b)|a \in S1 \wedge b \in S2\} \quad (28)$$

In the complete-link hierarchical clustering, distance between two clusters is the *maximum* distance between the pairs of elements of these clusters. As mentioned before, we take the *minimum* similarity between elements of two sets to determine the similarity between these sets.

$$SSIM_C(S1,S2) = min\{SIM(a,b)|a \in S1 \wedge b \in S2\} \quad (29)$$

The last method used for calculating the similarity between sets is average-link. In the average-link hierarchical clustering, distance between two clusters is the *average* of all distances between the pairs of elements of these clusters. By using this definition, similarity between two sets can be calculated as follows:

$$SSIM_A(S1,S2) = \frac{\sum_{a \in S1} \sum_{b \in S2} SIM(a,b)}{|S1||S2|} \quad (30)$$

So far the taxonomy, relation and attribute similarities between instances are defined. Now, we can combine these measures by giving them some weight values. Semantic similarity (SS) between two instances for cluster Cl_i is defined as follows:

$$SS_{Cl_i}(I_1,I_2) = \frac{aTS(I_1,I_2) + bRS_{Cl_i}(I_1,I_2) + cAS_{Cl_i}(I_1,I_2)}{a+b+c} \quad (31)$$

2.4 Rating prediction

The last step of $SEMCBF$ is prediction of the unknown ratings of the items given by users. In order to predict the unknown ratings, our approach uses the calculated semantic similarities (SS) between items and collaborative user model which consists of ratings given by the user on a neighborhood-based method[10]. To compute a prediction, two prediction functions can be used after selecting the k most similar items (k nearest neighbors). By using the first prediction function, the predicted rating of user u on item i can be calculated by taking the average of the ratings given by the user u on $\hat{I}$ which are k most similar items to i for the cluster that the user belongs to.

$$pred1_{u,i} = \frac{1}{k} \sum_{\acute{\imath} \in \hat{I}} r_{u,\acute{\imath}} \quad (32)$$

By using the second prediction function, the predicted rating of user u on item i can be calculated by taking the weighted average of the ratings given by the user u on $\hat{I}$ which are k most similar items to i for the cluster that the user belongs to. Weights of the ratings are set according to the similarities between items.

$$pred2_{u,i} = \frac{1}{\sum_{\acute{\imath} \in \hat{I}} SS_{Cl_i}(i,\acute{\imath})} \sum_{\acute{\imath} \in \hat{I}} SS_{Cl_i}(i,\acute{\imath}) r_{u,\acute{\imath}} \quad (33)$$

where Cl_i is the cluster that user u belongs to.

Using user-item rating matrix, $SEMCBF$ creates enhanced user-item rating matrix. In other words, the sparsity of user-item rating matrix is reduced. And also, even if an item has no explicit rating given by any user in the system, by using $SEMCBF$, our approach predicts a rating given by every user for that item. Thus, a new item which has no ratings given by any user in the system has a chance of being recommended.

2.5 Collaborative Filtering

In fourth phase of our approach, a neighborhood-based [10] collaborative filtering algorithm is used on enhanced user-item matrix and active user ratings vector which consists of only actual ratings given by the active user. The collaborative filtering algorithm first computes the similarity between the active user and other users in enhanced user-item matrix. In order to calculate similarities between users Pearson correlation of ratings vector is used. After computing similarities between active user and other users, n number of most similar users is selected. An unknown rating is predicted by calculating the adjusted weighted sum of the nearest neighbors' ratings of active users. The advantage of using adjusted weighted sum is that it also considers difference of rating scales of the users. The weight of nearest neighbors depends on the similarities between users.

At the end of a recommendation process, the system recommends a number of unrated items which have the highest predicted rating to the active user.

Table 3: Parameters and Possible Values.

Parameter	Values
Max. Recursive Depth (rd)	0,**1**,2,3,4,5,6,7,8
Weight of TS (a)	0, 0.1, **0.2**, 0.3, 0.4, 0.5, 0.6, ..., 1.0
Weight of RS (b)	0, 0.1, **0.2**, 0.3, 0.4, 0.5, 0.6, ..., 1.0
Weight of AS (c)	0, 0.1, 0.2, 0.3, 0.4, 0.5, **0.6**, ..., 1.0
Measure for Taxonomy Similarity Between Concepts (TSC)	TSC_{CM} ***TSC*$_{WuPalmer}$** TSC_{Lin} TSC_{Mclean}
$SSIM$ Method for TS ($SSIM_{TS}$)	$SSIM_1$, $SSIM_2$, $SSIM_3$, ***SSIM*$_S$**,
$SSIM$ Method for RS ($SSIM_{RS}$)	$SSIM_C$, $SSIM_A$
Number of Nearest Neighbors (k)	5,**10**,15,20, 30,50,100,200
Prediction Function (PF)	$pred1$, **pred2**

3. EXPERIMENTAL EVALUATION

In order to evaluate proposed approach we use MovieLens 100k dataset (*http://www.grouplens.org*). We apply 5-fold cross-validation on the sets that are provided in MovieLens 100k dataset. We use F-measure performance metric which is the combination of precision and recall metrics. F-measure is commonly used for evaluating the performance of such systems [11].

The evaluation consists of three phases. In the first phase, to find the most appropriate values for $SEMCBF$ parameters, $SEMCBF$ is evaluated without considering the user models. In this phase, all users are grouped in only one cluster with equal feature-weight values. Thus, CWV of this cluster has the value of 1 for weight of each feature. The parameters and their possible values are shown in Table 3.

The parameters a, b, c, satisfies the constraint $(a+b+c) = 1$. For each parameter, $SEMCBF$ is evaluated for each of the values of corresponding parameter and k, while other parameter is set to a constant value. For each analysis of a parameter, the determined values of previously analyzed parameters are kept same. At the beginning of the first phase of the evaluation, parameters are set to underlined values in Table 3. The parameters are analyzed in the following order; PF, TSC, $SSIM_{TS}$, $SSIM_{RS}$, a, b, c, rd. The highest F-measure, 75,2%, is obtained by using the values in bold in Table 3.

After determining the values of these parameters, we apply a feature selection algorithm, *sequential forward selection (SFS)* [8], to find a subset of features that improves the F-measure of $SEMCBF$. Figure 3 shows the selected relations/attributes that represent features and F-measure of the $SEMCBF$ at each iteration. The result of the SFS shows that excluding *runtime*, *color* and *cast* features improves the F-measure of $SEMCBF$.

In the second phase, first, methods for MFV and UWV are determined. For this process, for each methods for MFV and UWV mentioned in Section 2.2 are used for one cluster to evaluate $SEMCBF$. The results are shown in Table 3. The highest F-measure is obtained by using the methods Idf for MFV and $Default$ or $Stddev$ for UWV. Because the precision value of $Stddev$ is higher than the the precision value of $Default$ for UWV method, $Stddev$ method is selected for UWV method and Idf is selected for MFV method.

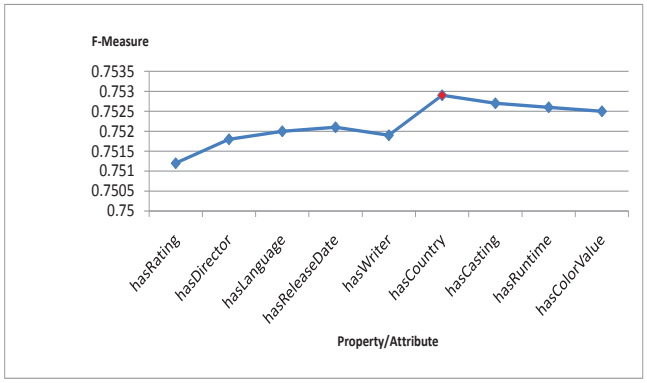


Figure 3: Result of SFS

Table 4: Results of MFV and UWV Methods

MFV Method	UWV Method	F-Measure
Default	Default	75.22
Default	Stddev	75.25
Idf	Default	75.30
Idf	Stddev	75.30
Ifw	Default	75.27
Ifw	Stddev	75.28

After determining the methods for UWV and MFV, UWVs are created. Then, we apply *K Means* and *Expectation Maximisation (EM)* algorithms separately to find CWV of each cluster. The properties of the EM algorithm is the default properties provided by $Weka$ for EM clusterer. The only property that is changed for *K Means* is the number of clusters. We set number of clusters to 15 for *K Means*. F-measures obtained by using EM and *K Means* with $k = 10$ is 75.32% and 75.31% respectively.

In the third phase of the evaluation, the performance of $SEMCBF$ and $SEMCBCF$, in which $SEMCBF$ is used for enhancing user-item matrix, is compared to other approaches. Table 5 gives precision, recall and F-measure metrics of $SEMCBF$, $SEMCBCF$ and some approaches obtained from [12, 13]. Movielens uses collaborative filtering approach. MovieMagician [9] is a hybrid recommender that uses kinds, actors and directors as features of a movie. OPENMORE [13] is a content-based recommender based on fine-tuning the constructed user models. ReMovender [12] is a content-based recommender that is empowered by collaborative missing data prediction. $CBCF$ [20] is a hybrid recommender that uses naive Bayesian classifier as a content-based predictor to enhance user-item rating matrix. It can be seen that $SEMCBF$ and $SEMCBCF$ outperform these approaches, which are shown in Table 5, in F-measure metric.

4. CONCLUSIONS

This paper presents a hybrid approach, $SEMCBCF$, that is based on content-boosted collaborative filtering presented in [20]. The content-based part of our hybrid approach,

Table 5: Comparison of $SEMCBF$ and $SEMCBCF$ with Other Approaches.

Approach	Prec. (%)	Rec. (%)	F-Measure (%)
MovieLens	66	74	69,8
MovieMagician Feature-Based	61	75	67,3
MovieMagician Clique-Based	74	73	73,5
MovieMagician Hybrid	73	56	63,4
OPENMORE	62,0	91,7	74,1
ReMovender	72	78	74,9
CBCF	60	95,2	73,6
SEMCBF	63,7	92,2	75,3
SEMCBCF	63,8	93,1	75,7

$SEMCBF$, utilizes content-based user models to obtain the feature weights that represents the opinion of users on the features. According to these user models, users are clustered and for each cluster, $SEMCBF$ finds semantic similarities between items using feature-weights of cluster and ontology-based metadata. After finding the similarities for each cluster, user-item rating matrix is enhanced using the similarities of items that are calculated for the active user's cluster and active user's explicit ratings. At the last step, collaborative filtering is performed on the enhanced user-item matrix to recommend items.

In the evaluation phase, first, $SEMCBF$ was fine-tuned by determining the values of its parameters, methods for creating user models and clustering algorithm. And also a feature selection algorithm is applied for determining the features that are redundant. Then, determined values and features, $SEMCBF$ and $SEMCBCF$ was evaluated. The results are promising, however, further evaluations are needed.

Besides the properties and attributes of the ontology, the hierarchy of concepts may affect the performance of the system. For further research, a more detailed ontology will be generated and the effects of the hierarchy of concepts will be explored. Additionally, a web application will be developed in order to perform a user evaluation.

5. REFERENCES

[1] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, 2005.

[2] T. L. Bach and R. D. Kuntz. Measuring similarity of elements in owl dl ontologies. In *Proceedings of the AAAI'05 Workshop on Contexts and Ontologies: Theory, Practice and Applications*, pages 96–99, 2005.

[3] M. Balabanovíc and Y. Shoham. Fab: Content-based, collaborative recommendation. *Communications of the ACM*, 40(3):66–72, March 1997.

[4] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence (UAI-98)*, pages 43–52, San Francisco, 1998. Morgan Kaufmann.

[5] R. Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, 2002.

[6] K. Chapphannarungsri and S. Maneeroj. Combining multiple criteria and multidimension for movie recommender system. In *Proceedings of the International MultiConference of Engineers and Computer Scientists (IMECS 2009)*, pages 698–703, 2009.

[7] M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, and M. Sartin. Combining content-based and collaborative filters in an online newspaper. In *Proceedings of ACM SIGIR Workshop on Recommender Systems*, August 1999.

[8] P. Devijver and J. Kittler. *Pattern Recognition: A Statistical Approach*. Prentice Hall, 1982.

[9] S. Grant and G. I. McCalla. A hybrid approach to making recommendations and its application to the movie domain. In *Proceedings of the 14th Biennial Conference of the Canadian Society on Computational Studies of Intelligence: Advances in Artificial Intelligence*, AI '01, pages 257–266, London, UK, 2001. Springer-Verlag.

[10] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '99, pages 230–237, New York, NY, USA, 1999. ACM.

[11] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22:5–53, January 2004.

[12] H. Karaman. Content based movie recommendation system empowered by collaborative missing data prediction. M.Sc. Thesis in Computer Engineering Department of Middle East Technical University, 2010.

[13] O. Kirmemis. Openmore: A content-based movie recommendation system. M.Sc. Thesis in Computer Engineering Department of Middle East Technical University, 2008.

[14] O. Kirmemis and A. Birturk. A content-based user model generation and optimization approach for movie recommendation. In *Proceedings of the AAAI'08 Workshop on Intelligent Techniques for Web Personalization & Recommender Systems*, pages 78–88, 2008.

[15] Y. Li, Z. A. Bandar, and D. McLean. An approach for measuring semantic similarity between words using multiple information sources. *IEEE Transactions on Knowledge and Data Engineering*, 15:871–882, 2003.

[16] D. Lin. An information-theoretic definition of similarity. In *ICML '98: Proceedings of the Fifteenth International Conference on Machine Learning*, pages 296–304, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[17] P. Lula and G. Paliwoda-Pekosz. An ontology-based cluster analysis framework. In *7th International Semantic Web Conference (ISWC2008)*, October 2008.

[18] A. Maedche and V. Zacharias. Clustering ontology-based metadata in the semantic web. In *Proceedings of the 6th European Conference on*

Principles of Data Mining and Knowledge Discovery, PKDD '02, pages 348–360, London, UK, UK, 2002. Springer-Verlag.

[19] O. Maimon. *Decomposition Methodology For Knowledge Discovery And Data Mining: Theory And Applications (Machine Perception and Artificial Intelligence)*. World Scientific Publishing Co., Inc., River Edge, NJ, USA, 2005.

[20] P. Melville, R. J. Mooney, and R. Nagarajan. Content-boosted collaborative filtering for improved recommendations. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence*, pages 187–192, 2002.

[21] T. M. Mitchell. *Machine Learning*. McGraw-Hill International Edit, 1997.

[22] M. J. Pazzani. A framework for collaborative, content-based and demographic filtering. *Artificial Intelligence Review*, 13:393–408, 1999.

[23] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. T. Riedl. Application of dimensionality reduction in recommender systems: A case study. In *WebKDD Workshop at the ACM SIGKKD*, 2000.

[24] N. Tintarev and J. Masthoff. Similarity for news recommender systems. In *Proceedings of the AHŠ06 Workshop on Recommender Systems and Intelligent User Interfaces*, 2006.

[25] Z. Wu and M. Palmer. Verb semantics and lexical selection. In *Proc. of the 32nd annual meeting on Association for Computational Linguistics*, pages 133–138, 1994.

What Your Friends Tell Others About You: Low Cost Linkability of Social Network Profiles

Sebastian Labitzke Irina Taranu Hannes Hartenstein
Steinbuch Centre for Computing & Institute of Telematics
Karlsruhe Institute of Technology (KIT), Germany
sebastian.labitzke@kit.edu, irina.taranu@student.kit.edu, hannes.hartenstein@kit.edu

ABSTRACT

Due to the amount of personally identifiable information shared by users of online social networks (OSNs) and the often not adequately adjusted privacy settings, it is possible to identify a user's several OSN profiles. In this paper, we illustrate that third parties just have to take a small step to link OSN profiles of a user and, consequently, to aggregate various pieces of information shared in several OSNs. Particularly, based on statistical results we illustrate how the often publicly available friends lists can be exploited to link several OSN profiles of a single natural person. The results presented in this paper show that profiles can be linked even at low cost, i.e., without complex correlation techniques or high computational power. To assess the risk of privacy leakage by profile linking, we, additionally, report how often specific pieces of information are made publicly available by users in four of the most popular OSNs. We show that users tend to publish different pieces of information in different OSNs and, thereby, demonstrate that by linking friends lists more information about a user can be gained than the user shared in a single OSN. For the study we analyzed more than 180,000 user profiles and compared more than seven million pairs of profiles to investigate profile linkability.

Keywords

Social Network Analysis, Linkability, Friends Lists

1. INTRODUCTION

Today, online social networks (OSNs) are commonly used to keep in touch with people from all areas of life. Sharing personal data, i.e., personally identifiable information (PII) seems to be so attractive for many OSN members that the recognition of risk regarding privacy often fades into the background. Facebook claims that, on average, every single one of their more than 600 million users provides more than 90 pieces of content per month[1]. The behavior of OSN users and their generosity regarding the amount of shared data has been, for instance, investigated in [5], [14], and [11]. If nowadays third parties know just a few details about a person, the probability to find this person inside one of the mass of OSNs is already high [16] and, thus, more information about this person is accessible. Possibilities to restrict access to published information in terms of privacy settings provided by OSNs are only used by a subset of all OSN members. In most OSNs, just a minority of members hide all of their data from members beyond their friends lists. In Section 3, we illustrate this fact on the basis of analyzed statistical data gathered by 180,000 investigated OSN profiles.

Additionally, publicly available user information is made available by OSN websites that can easily be parsed by software because of their standardized layouts [4]. Consequently, this constitutes an invitation for third parties to crawl OSNs in order to gather as much information about people as possible despite the fact that this might be illegal. If it is then possible to gather information from several OSNs by profile linking and to associate such information with a natural person, the resulting cumulative data set is of a greater value than unlinked and individual pieces of information. Hence, linking of OSN profiles might be profitable for third parties if users provide some information publicly in one OSN and some other in another OSN.

To turn the tables, linking of profiles could also be used to reveal and demonstrate users their virtual appearance across social networks. Bold and simple revealing of their linkability to users might help to motivate a more careful and adequate adjustment of privacy settings. The psychological point of view regarding such opportunity of motivation is investigated in our preceding study [13]. In this study, we also addressed restrictions for OSN analysis with respect to the German data protection act and countermeasures that are implemented by OSNs to disturb crawlers in gathering data via parsing of OSN profiles. However, the focus of the current work is to demonstrate users, who are not aware of the linkability of their OSN profiles, implications for privacy with respect to possible aggregations of information published in several OSNs. Such implications occur if those users do not entirely hide their shared data from strangers. In contrast, if users linked their profiles on their own by use of services such as about.me, the linkability is implicitly given and intended by users.

In this paper, we start by demonstrating the average availability of every type of information published in user profiles of four popular OSNs. The objective of this part of the study is to assess the potential risk regarding profile linking,

[1] https://www.facebook.com/press/info.php?statistics

The 5th SNA-KDD Workshop '11 (SNA-KDD'11), August 21, 2011, San Diego CA USA

or rather, to assess the additional information that can be gained about a user by linking his OSN profiles. Furthermore, we present a "low cost" concept to link OSN profiles of a single natural person. Low cost means that we focus on profile linking on the basis of simple string comparisons of provided data rather than on the use of complex correlation algorithms, such as, for instance, face recognition techniques [17]. In this manner, the analysis of the statistical data shows that especially the often publicly available list of friends can be easily exploited to effectively link OSN profiles of a single natural person. We further constitute that just a low overlap of two friends lists is a sufficient indication to identify a single user as the owner of two corresponding OSN profiles. The underlying linkability investigation is based on seven million comparisons of pairs of profiles. In conclusion, the contributions of this paper are:

- We show a current snapshot (based on an extensive set of measurements) of the availability of information in profiles of four popular OSNs.
- We analyze the diversity of information users share via OSNs with respect to the OSN itself.
- We demonstrate that information can be gathered efficiently by profile linking using only a manageable number of string comparisons on the basis of friends lists.

The paper is structured as follows. Section 2 analyses the related work and point out the unique characteristics of the presented study. Section 3 demonstrates the results according to the investigated availability of specific information inside OSN profiles and formulates associated hypothesis. Furthermore, the derived research questions are highlighted. Section 4 comprises methods, definitions, and fundamentals with respect to the analysis of statistical data regarding the profile linkability. Results of the linkability investigation are demonstrated in Section 5. Furthermore, we discuss the aforementioned results on a meta-level in Section 6, before Section 7 concludes the paper. In the Appendix A, we note essential information on the compliant concept regarding data sampling.

2. RELATED WORK

Publicly available PII and the linkability of OSN profiles have been investigated by other previous studies. In the following, we show the relation and differentiation of these studies and our approach.

As emphasized in the introduction, users reveal a lot of PII via OSN profiles. The "amount of information shared on a user's profile as well as in the process of communication with others" is called self-disclosure [8]. In the past few years, several studies were performed to describe which pieces of PII users reveal (especially via Facebook) and which preventive measures they take to hide their data from third parties. Since the results of such studies changed over time, the evolution of user behavior in Facebook can be derived with respect to the results of related studies and the current status on the availability of users' information presented in this paper. Gross and Acquisti examined the self-disclosure of OSN users based on empirical data of more than 4,000 Facebook profiles of Carnegie Mellon University students in 2005 [5]. At this point in time, only 1.2% of the users prevented others outside the network from finding their profiles and only 0.06% entirely restricted the visibility of their profile. In 2007, Lampe et al. collected about 38,000 Facebook profiles [14]. 19% were set up with access restrictions and 59% of the profile fields contained information. A further study performed by Brown et al. investigated 7,919 profiles from the Facebook network of the Michigan University in 2008 [3]. They found that 68% of users let their profiles visible for everyone from the network. Of these, 86% had a publicly available friends list. In contrast to such related studies, we present a current snapshot of the situation of self-disclosure in Facebook and, additionally, three other popular OSNs based on a more extensive set of measurements. Furthermore, we compare the availability of information within Facebook to these other OSNs. According to this data, it can be seen that the awareness of users regarding the adjustment of privacy settings has grown in the past. Nevertheless, we show that a majority of users is still generous with respect to publicly available PII.

To stay in control of shared data, users have to adjust their privacy settings. In 2008, Krishnamurthy et al. investigated privacy settings of OSNs and showed that most users do not change the default configuration [10]. Only 1% of Twitter users, 25% of Facebook users and 21% of MySpace users changed their privacy settings. Studies such as [11] and [12] also examined the availability of information about users in different OSNs, trying to find out what information OSNs contain and how much information users can protect from strangers or attackers. The authors showed that not every piece of information could be made unavailable for others. In [11] is demonstrated that the friends lists can be accessed in ten of the twelve analyzed OSNs if the users maintain the default privacy settings. We demonstrate in this paper that nowadays in three of four analyzed OSNs still only 7% to 22% of users hide their profile completely from strangers, except information that can not be hidden.

If users do not adjust, or rather, are not able to adjust their privacy settings appropriately, one of the consequences is the linkability of OSN profiles. The authors of [20] demonstrated a strategy to link profiles by revealing user's several names within various web communities, such as OSNs. In 2005 researchers estimated that there is an overlap of profiles in two OSNs of 15% [15]. A study conducted by compete.com in 2007 showed a more major member overlap in OSNs[2]. There was an overlap of Facebook profiles with profiles of MySpace of 64% and, vice versa, only 20%. Besides profile linking, analysis on and comparisons of OSN profiles can be abused for further attacks such as identity clone attacks, as was investigated in [2] or the re-identification of OSN profiles if corresponding users visit a website [19].

However, Motoyama et al. developed a system for matching profiles in OSNs [16]. The goal was to figure out sufficient criteria to find a person in an OSN, so that the search is as effective as possible and the correct profile is found. This has allowed the authors to match user profiles of Facebook with user profiles in MySpace and vice versa. Thereby, they showed that 43% of users with overlapping profiles in both OSNs have similar privacy settings, the rest take different security measures. Most likely MySpace profiles are private while Facebook profiles remain open. This shows that for at least 57% of users new insights can be gained by finding corresponding profiles. This underpins the hypothesis

[2]http://blog.compete.com/2007/11/12/connecting-the-social-graph-member-overlap-at-opensocial-and-facebook

that profile linking is profitable for third parties in light of gathering as much information about a user as possible.

Additionally, Motoyama et al. show only a little overlap of friends lists [16]. We demonstrate in this paper that such merely low overlaps are sufficient to link OSN profiles of a single natural person. Therefore, we disprove the hypothesis that the rare overlap of friends lists can not be (ab-)used to link OSN profiles of a natural person to gain more information about this person.

3. OBSERVATIONS AND RESEARCH QUESTIONS

At first, we assess which and how often pieces of information are shared by users within four popular OSNs. In doing so, we focus on information whose access is not restricted by privacy settings. Table 1 lists attributes published within either of the four evaluated OSNs. For each attribute, we state the ratio of users who did not restrict the attribute's visibility, i.e., attributes can be seen by any logged in user. Hence, the results reveal which information is extractable out of OSN profiles in any case.

The presented statistical data was collected between January and March 2011. The data was gathered by use of a software that we specifically implemented for this purpose. This software is designed to comply to the German data protection act. In the appendix, we note information on the underlying compliant concept of data sampling. By use of this software, we analyzed more than 110.000 Facebook profiles, more than 43.000 profiles of StudiVZ, more than 25.000 MySpace profiles and more than 10.000 profiles of Xing.

The first column of Table 1 classifies the types of information a user can publish via an OSN profile. Information which is shared by using chats or walls also referred to as pinboards have not been considered in this study. The category *general* subsumes the availability of names of users and their friends lists which form the basis of the linkability part of the study presented in Section 5. Moreover, this category indicates how many profiles are completely hidden from strangers by adjusted privacy settings. The category *number of...* illustrates how many profiles we found in a single search request on average and at the maximum. This category also states the number of friends a user has on average, including the corresponding standard deviations, and the size of the largest detected friends lists. Whereas the category *personal* summarizes almost never changing attributes like date of birth or gender, the category *contact* lists all attributes that can be used to locate a user physically. The categories *job* and *higher education* give insights into the occupations of users. Users provide information about their family and relationship background, current frame of mind and physical shape via attributes of the category *oneself and relations*, whereas the attributes of the category *views and attitudes* contain the personal views regarding politics, religion, etc. The category *hobbies* represents attributes to tell others something about further activities whereas *favorites* summarizes favored music, literature, and so on.

Remarkably, the statistical values of Xing differ from others (see Table 1). Some attributes are mandatory to create a Xing profile, such as the type of job, the job description, the company, the area of business, the country and the city. These attributes are available to every Xing user and the access can not be restricted by adjusting privacy settings. In contrast, any contact information can only be seen by direct friends, so that we mention the availability of these attributes at 0%. Currently, further pieces of information are accessible by any Xing user if the user simply published this piece of information.

However, one of the most remarkable observations within the presented statistical data is the diversity of availability of information with respect to the specific OSN. For instance, the date of birth is made available by 64% of all evaluated StudiVZ users. In contrast, only 0.84% of Facebook users share this particular information publicly. A similar difference regarding the availability applies to the university, hometown, current residence, cv, relationship status, and more. In light of the fact that many users participate at more than one OSN, such diversity of the availability of information results in the following worthwhile opportunity for third parties: if it is possible to link several OSN profiles of a single person, a comprehensive set of information can be gathered and correlated by third parties. Therefore, the hypotheses are as follows:

- The different behavior of users regarding specific information shared in different OSNs constitutes a privacy threat for users if profiles are linkable.
- The lower the costs (e.g. computational power, and implementation effort) required for linking a profile are, the broader is the group of third parties that are able to link profiles.

We showed that linking of profiles might be worthwhile for third parties because users share different information in different OSNs. In light of the aforementioned hypotheses, the following research questions are derived:

- Are typically shared attributes sufficient to successfully link profiles?
- Are friends lists, as the most available information despite the specific OSN, sufficient to link profiles?
- Can profiles be linked at low costs, so that a high number of third parties are able to link profiles?

In general, any type of information of OSN profiles might be sufficient to link profiles if these attributes can be compared among each other, such as comparing profile images via face recognition, status messages via text mining, shared PII (see Section 6.1), and so on. However, we are looking for a low cost strategy to link OSN profiles, i.e., we try to establish links between OSN profiles with less computational power and less complex correlation techniques than, for instance, required for face recognitions as well as without complex mining algorithms. Friends lists are publicly available in many OSN profiles (between 40% and 67% of all analzed profiles) and comparisons of two friends lists can be done via simple string comparisons. Therefore, we focus on the investigation of the friends list based linkability and show that string comparisons are sufficient for profile linking.

4. METHODOLOGY AND DEFINITIONS

In this section, we describe the methodology for the investigation of friends list based profile linking. Furthermore, we define terms and metrics necessary to analyze and interpret the statistical data gathered by comparing seven million pairs of profiles during our study.

Category	Attribute	StudiVZ	Facebook	MySpace	Xing
General	name	100.00%	100.00%	100.00%	100.00%
	no further information available	*6.97%*	*21.53%*	*20.34%*	n.a.
	friends lists	**48.13%**	**59.45%**	**67.04%**	**40.98%**
Number of...	...profiles per search request (avg.)	11	20	12	6
	...profiles per search request (max.)	288	545	491	1554
	...friends (avg.)	67	141	21	44
	...friends (std. deviation)	74.01	216.32	75.47	85.39
	...friends (max.)	918	5499	2058	2878
Personal	graduation/title	5.51%	-[1]	-[1]	69.03%
	date of birth (dob)/age	**64.00%** (dob)	0.84% (dob)	32.06% (age)	0%
	zodiac sign	10.51%	-[1]	20.72%	-[1]
	gender	71.06%	49.92%	32.06%	-[1]
Contact	hometown	23.74%	8.77%	6.49%	-[1]
	current residence/region	48.46%	10.32%	32.04%	mandatory (100%)
	homeland or current country	18.69%	n.a.	28.78%	mandatory (100%)
	address	0%	0.11%	-[1]	0%
	email	-[1]	0.62%	-[1]	0%
	mobile phone number	-[1]	1.19%	-[1]	0%
Job	company/occupation	5.75%	9.63%	4.91%	mandatory (100%)
	type of job	7.85%	n.a.	n.a.	mandatory (100%)
	income	-[1]	-[1]	2.21%	-[1]
Higher education	university	51.67%	8.83%	n.a.	n.a.
	field of study or study path	11.21%	n.a.	n.a.	n.a.
	languages	10.04%	-[1]	-[1]	n.a.
	general education/cv	23.71%	2.81%	6.35%	n.a.
	current school	-[1]	16.00%	n.a.	-[1]
Oneself and relations	about myself	16.33%	n.a.	6.22%	-[1]
	relationship status	26.51%	13.12%	n.a.	-[1]
	status message	49.85%	n.a.	20.72%	-[1]
	physique	-[1]	-[1]	6.93%	-[1]
	parentage	-[1]	n.a.	6.02%	-[1]
	children	-[1]	-[1]	7.61%	-[1]
Views and attitudes	sexual orientation	-[1]	8.16%	7.77%	-[1]
	interests/looking for...	31.41%	8.89%	8.92%	-[1]
	political direction	18.42%	0.33%	-[1]	-[1]
	religious views	-[1]	0.46%	4.96%	-[1]
	smoking and imbibing	-[1]	-[1]	5.52%	-[1]
Hobbies	interests/details	25.44%	8.50%	20.72%	n.a.
	clubs/activities/groups	16.57%	14.29%	0.77%	n.a.
Favorites	favorite citation	21.00%	4.78%	-[1]	-[1]
	favorite music	26.02%	16.75%	6.84%	-[1]
	favorite books	19.32%	5.88%	5.20%	-[1]
	favorite movies	22.04%	10.47%	5.90%	-[1]
	favorite TV shows	-[1]	12.73%	5.51%	-[1]

[1] This attribute does not appear in such OSN profile by default

Table 1: Publicly available personally identifiable information in OSNs (unrestricted access) [%]

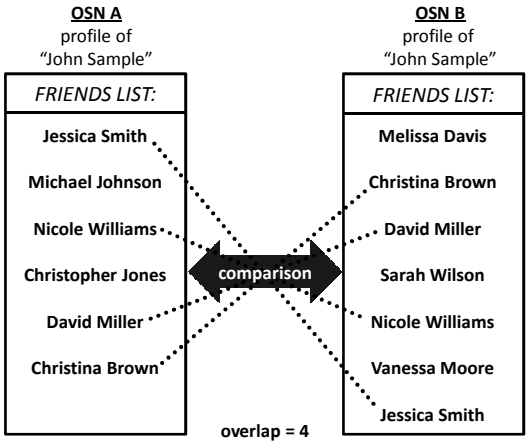
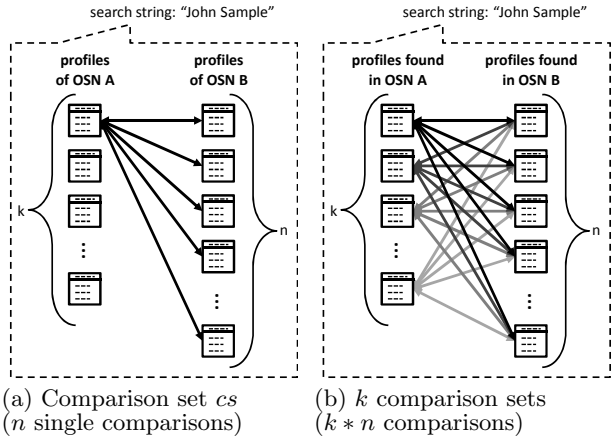


Figure 1: Exemplary comparison of two friends lists



(a) Comparison set cs (n single comparisons)

(b) k comparison sets ($k * n$ comparisons)

Figure 2: Illustration of the comparison set concept

4.1 Overlap metric

By comparing profiles from different OSNs, the objective is to identify users who own a profile in two or more OSNs. In order to compare two given profiles found in different OSNs, or rather, their friends lists, we proceed as follows. First, our analysis software considers each friends list as a set of entries, where each entry is given by the name of a single friend. Then, the intersection of these two sets S_1 and S_2 is determined ($S_1 \cap S_2$). We refer to the number of entries within this intersection as *overlap*, i.e., the overlap comprises the number of names that appeared in both friends lists ($|S_1 \cap S_2|$). A more sophisticated determination of the overlap by calculating, for instance, the Jaccard index [6] is not necessary because the whole number of friends of compared profiles is insignificant for the results presented in Section 5. We refer to the defined "simple" determination of the overlap as a single *comparison* (compare Figure 1). Later on, the overlap is used as one of two correlation metrics that have to be taken into account to decide whether two compared OSN profiles belong to the same natural person.

In the following, we refer to profiles that are identified as profiles of a single user as a *match*. OSNs usually provide a feature to search user profiles and other content of the network by generic search strings. In order to detect a profile that belongs to a user who also owns a specific profile p of OSN A in OSN B, we compare profile p to every possibly matching profile of OSN B. In this context, the characteristic "possibly matching" means any profile of OSN B that can be found with the same search string used for finding p in OSN A. In the appendix, we give more information regarding the selection of search strings.

Since we did a multitude of single comparisons of friends list pairs, we structured these comparisons as follows. A *single* profile which is found by using a specific search string s is compared to every profile that can be found inside another OSN with the same search string. If n constitutes the number of profiles found in OSN B with s, we compare n profiles with profile p (1:n). We refer to such a set of corresponding comparisons as a *comparison set* (cs). Figure 2 illustrates the concept of comparison sets. In Figure 2(a), a single cs is shown. Figure 2(b) depicts several comparison sets assembled with the same search string. Therein, an arrow represents a single comparison of two friends lists and arrows of the same grey-level illustrate a comparison set.

For example, suppose we search for a person named "John Sample" in two OSNs, OSN A and OSN B. Assume that the search returns 200 profiles in OSN A and 100 profiles in OSN B with the name "John Sample". To identify the same user of a specific profile p of OSN A within the 100 profiles of OSN B, we firstly compare the friends list of the profile p with all 100 friends lists of users named "John Sample" from OSN B. These 100 comparisons form a comparison set (cs) consisting of 100 single comparisons (1:100, compare Figure 2(a)). To check each of the 200 profiles found in OSN A, the comparison procedure has to be executed for each of the 200 found profiles, resulting in 200 cs of 100 comparisons each (compare Figure 2(b)).

4.2 Maximum overlap and target comparison

Assuming that comparisons of friends lists are sufficient to identify profiles of a single natural person in several OSNs, the comparison of two friends lists that constitutes the largest overlap inside a cs has the highest likelihood to indicate a "match". In general, resulting overlaps of a single cs, i.e., overlaps that are detected due to n comparisons between a single profile p and each profile found by the same search string within another OSN, are defined by the function:

$$o(cs), cs \in \{comparison_i | 1 \leq i \leq n\}$$

Therefore, we define the largest detected overlap of a cs as $max(o(cs))$, i.e., this function provides the maximum number of equal names detected in a specific comparison of a single cs. A $max(o(cs))$ indicates the overlap of the comparison with the largest overlap inside a cs.

Generally, even more than one comparison of a single cs can result in the same determined overlap. If the maximum overlap is only detected in a single comparison within the cs, we, additionally, denote the corresponding comparison as a *target comparison*. Target comparisons stand out from the others because no other comparison within the corresponding cs resulted in such a high overlap. The function $f(max(o(cs)))$ represents the number of comparisons (of a single cs) that resulted in the maximum overlap.

Figure 3 shows a histogram of an exemplary cs. Therein, the detected overlaps of an exemplary comparison set with twelve single comparisons (1:12) are shown, i.e., twelve profiles of OSN B (found with a search string s) were compared

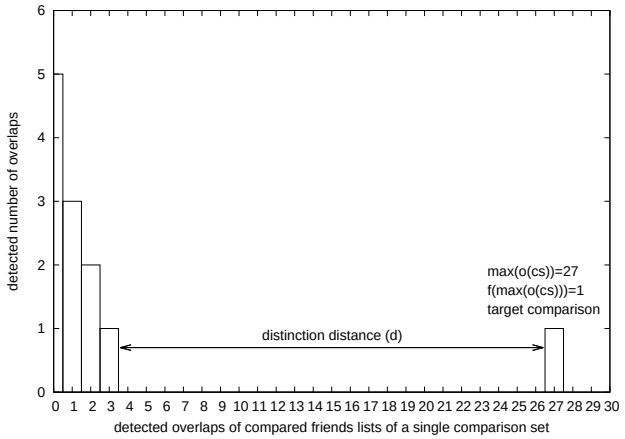


Figure 3: Histogram of detected comparison overlaps for an exemplary (1:12) comparison set cs

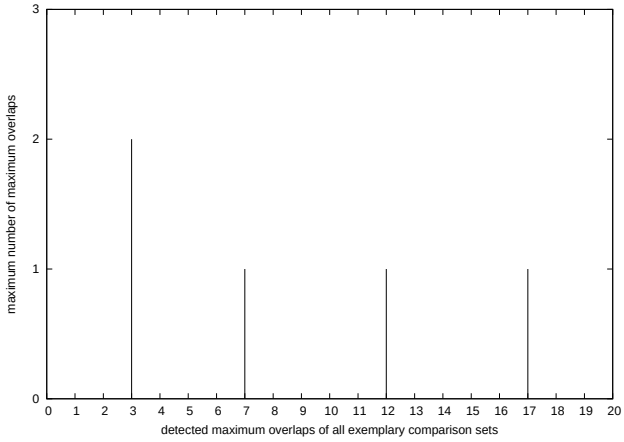


Figure 4: Exemplary aggregated max-overlaps graph (maximum number of maximum overlaps)

to a single profile of OSN A (found also with s). The maximum detected overlap of compared friends lists is at 27 ($max(o(cs)) = 27$), which is a target comparison because it is only detected in a single of the corresponding twelve comparisons ($f(max(o(cs))) = 1$).

4.3 Distinction distance metric

The gap between the overlap value of a target comparison and the next lower detected overlap inside a cs is called the distinction distance d and quantifies how distinct the target comparison stands from the other comparisons (see Figure 3). Therefore, the distinction distance quantifies the discriminative power of a maximum overlap with respect to other comparisons of the corresponding comparison set.

Distinction distances constitute the second correlation metric to decide whether two compared profiles belong to the same natural person. The assumption is: the higher the distinction distance the higher the probability that a match is detected. The distinction distance d of the overlap of the exemplary target comparison is at 24, calculated by 27 minus three (the next lower detected overlap), i.e., by means of d the target comparison distinguishes significantly from other comparisons inside this exemplary cs. Particularly, all other comparisons resulted in an overlap that is significantly lower (here: a maximum of 3 equal names).

4.4 Analysis of statistical data

As motivated before, we concentrate on comparisons that result in a maximal overlap $max(o(cs))$ of friends inside a cs to analyze the linkability of profiles on the basis of friends lists. Thereby, we try to figure out whether these maximum overlaps are overlaps of target comparisons, i.e., whether no other comparison of the same cs result in the same maximum overlap ($f(max(o(cs))) = 1$). Moreover, we investigate the distinction distance of such target comparisons to quantify the significance of the assumption that a target comparison indicates two profiles of the same natural person.

4.4.1 Aggregated graphs

To give an overview of the values and occurrences of detected maximum overlaps, we show graphs for which we consider only the values of $max(o(cs))$ of every cs. Thereby, all $max(o(cs))$ that correspond to comparisons of profiles of a specific OSN A and profiles of a specific OSN B are aggregated in a single graph. The y-value of such a graph indicates the *maximal* number of comparisons that resulted in a $max(o(cs))$ within every cs. For instance, if this y-value equals one, in every cs the corresponding maximum overlap is not detected more often than once. In turn, the corresponding comparisons are *target comparison*. If the y-value is greater than one, the maximum overlap is at least found for two comparisons of a single cs and, therefore, this maximum overlap does not correspond to a target comparison.

An exemplary plot of such a graph is shown in Figure 4. For this diagram assume five comparison sets cs_i, i={1, 2, 3, 4, 5} for which the maximum overlaps are as follows: $max(o(cs_1)) = 3$, $max(o(cs_2)) = 7$, $max(o(cs_3)) = 12$, $max(o(cs_4)) = 12$ as well, and $max(o(cs_5)) = 17$. Assume that the maximum overlap of three is detected in two single comparisons in the first comparison set cs_1. The corresponding aggregated graph shows four peaks, as seen in Figure 4. Three of these peaks indicate a maximum overlap with a maximum number of one because in each of the corresponding four cs these values are not detected or just detected in one single comparison. Note that the peaks indicate the *maximum* number of a maximum overlap and do not indicate the number of cs with a specific determined maximum overlap. In this example, only one comparison within each of two corresponding comparison sets existed whose overlap matched the maximum overlap of 12. In contrast, the peak at three has a y-value of two because we assumed that this maximum overlap is found two times in cs_1 so that the maximum number of such specific maximum overlap three is two. We refer to such graphs as *aggregated max-overlaps graphs*, i.e., graphs in which the maximum overlap of every cs is plotted against its maximum number within every cs.

4.4.2 Illustration of distinction distances

As mentioned above, the expressiveness of the maximum overlap can be estimated by means of its distinction distance d, i.e., the higher the value of d, the more the maximum overlap stands apart from all other detected overlaps of a cs. In turn, a high value of d indicates a high probability that two compared profiles belong to a single natural person because of the associated outlier characteristic of the

maximum overlap. In Section 5, we, therefore, plot the average distinction distances d for every detected maximum overlap and the corresponding standard deviations into the aggregated max-overlaps graphs.

If the maximum overlap stands considerably apart from all other detected overlaps in most analyzed *cs*, the average value of d converges to the value of the overlap. An average d of the same value as a corresponding overlap means that the next lower detected overlap in every *cs* is zero because the distinction distance is at its maximum. Distances which converge to the maximum show that other overlaps are much lower and can be neglected. In contrast, if the average d is much lower than the corresponding overlap or its standard deviation is unusually high, other overlaps were detected that are not much lower than the maximum overlap. If that would be the case, the metrics *maximum overlap* and *distinction distance* are not sufficiently significant to reach conclusions about whether two compared friends lists belong to OSN profiles owned by the same natural person.

5. RESULTS

In this section, we analyze gathered statistical data and demonstrate friends list driven profile linkability for four popular OSNs. The objective is to compare a friends list from a specific user profile of OSN A with the friends lists of profiles from another OSN B. Thereby, we try to find a profile from OSN B which belongs to the same user who owns the profile of OSN A. Firstly, we assume that the user is registered with the same name in both OSNs. A discussion whether the results are transferable to profiles that are not registered with the same name follows in Section 6.2.

5.1 Friends list driven profile linking

In the following, we show aggregated max-overlaps graphs including the illustration of average distinction distances, such as it is introduced in Section 4.4. We compared friends lists of each of four analyzed OSNs with friends lists taken from the other three OSNs and vice versa. Thus, we could present twelve plots in which the maximum overlaps and their average distinction distances are shown. In the following we depict four of these graphs because they already show the gained scientific knowledge. Figure 5(a) shows comparisons between friends lists of profiles of Facebook and StudiVZ, Figure 5(b) visualizes the results of comparisons between StudiVZ and XING, whereas Figure 5(c) and Figure 5(d) present the results of comparisons between Facebook and Xing or MySpace, respectively.

The objective of Figure 5 is to illustrate a phenomenon we observed: beginning at a certain maximum overlap the corresponding maximum number of every larger maximum overlap is one. Rather, low maximum overlaps around one up to three indicate a higher maximum number. Therefore, maximum overlaps higher than three are invariably associated to target comparisons, i.e., every maximum overlap higher than three is detected in none or just a single comparison within every comparison set.

The average distinction distances of maximum overlaps (also shown in Figure 5) strengthen this observation. Considering this distances, two distinctive phenomenons are obvious. Firstly, every average distinction distance d is just a little bit lower than its corresponding maximum overlap. Secondly, it can be noticed that the standard deviations of d are so small that they can hardly be recognized.

Therefore, the interpretation of these effects is as follows: an overlap that is a maximum in a *cs* always stands considerably apart from all other detected overlaps. That means that an associated target comparison of a maximum overlap is in all cases isolated from the rest of detected overlaps. In particular, we determined that maximum overlaps are isolated from other overlaps detected in comparisons of a *cs* if these maximum overlaps are higher than three. In short, the graphs show that each comparison set almost exclusively resulted in very small overlaps of friends lists (see average distinction distance), except a single higher overlap which can be detected in many comparison sets and which stands considerably apart from the others.

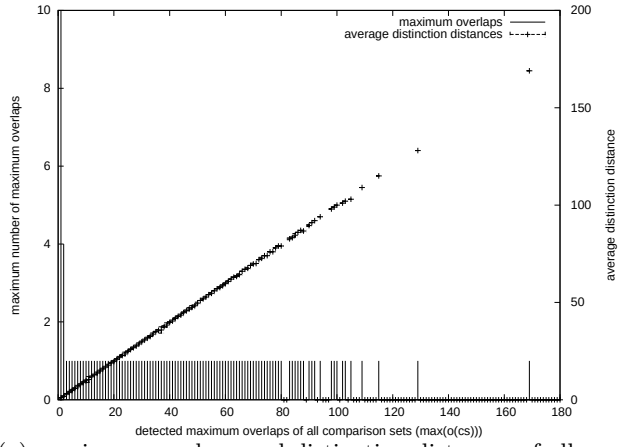
Therefore, we assume that OSN profiles of the same natural person are identified if the comparison of two friends lists results in a higher overlap than all other comparisons of friends lists of "possibly matching" OSN profiles of the same comparison set. The results imply that maximum overlaps of three or lower are not sufficient to decide whether two profiles belong to a single natural person or not, because such maximum overlaps occur more than a single time in some *cs*. In contrast, maximum overlaps higher than three indicate a "match". A discussion with respect to error rates regarding such indication is presented in the following section.

5.2 Evaluation of linkability results

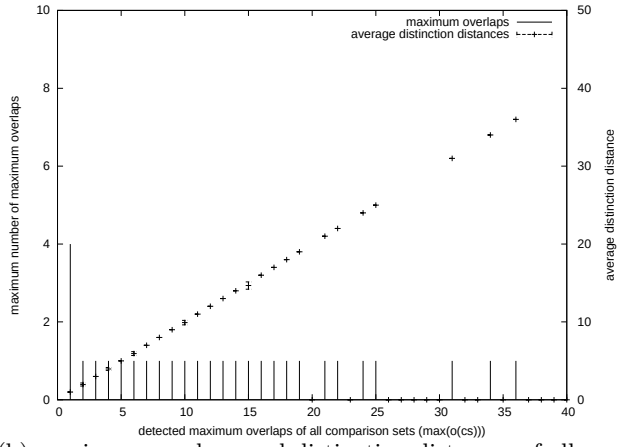
The results indicated that two profiles with a friends lists overlap of more than three belong to the same natural person if this person is registered with the same name in both profiles. This interpretation cannot be proven without talking to the owners of the profiles themselves. However, for being compliant to the German data protection act and for preserving privacy of OSN users, we are unable to contact profile owners. In order to confirm the aforementioned interpretation of the results, we implemented a module that enabled the analysis software to compare additional available information of OSN profiles, such as the given hometown, region, university, date of birth, etc. With this module activated we did another short run with which the software found about 300 comparisons that resulted in an overlap greater than three.

In almost all of these cases the software found a minimum of one other information that appeared in both compared profiles equally. In more than half of all cases the profile image was exactly the same in both profiles. Other attributes that were often available in both profiles were the user's hometown, region, university or the date of birth. In non of the analyzed comparisons the software found two mutually exclusive pieces of information. This fact confirms the hypothesis that two profiles with the same user name and an overlap of friends lists greater than three are owned by the same natural person.

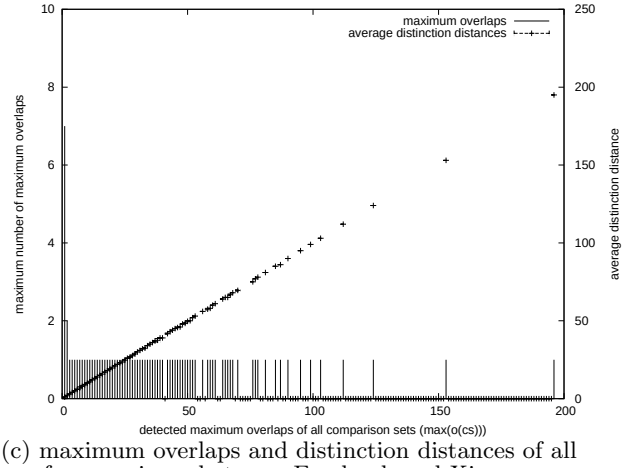
In Section 3 we showed that profile linking might be worthwhile for third parties for gathering more information about natural persons. By use of the presented strategy for profile linking, occurrences of false positives, i.e., accidentally linked profiles owned by different natural persons, cannot be entirely excluded. But, dedicated error rates could not be worked out because of restrictions regarding privacy demanded by the German data protection act. However, it is questionable whether third parties care about any false positives at all. The results of our study hint that the number of false positives is remarkably low. Therefore, potentially



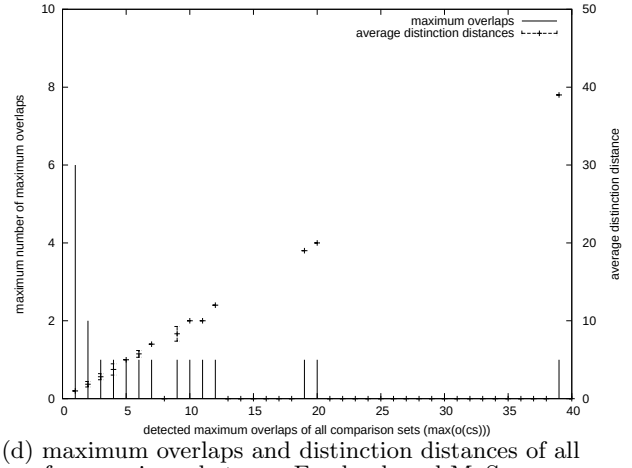
(a) maximum overlaps and distinction distances of all *cs* of comparisons between Facebook and StudiVZ



(b) maximum overlaps and distinction distances of all *cs* of comparisons between StudiVZ and Xing



(c) maximum overlaps and distinction distances of all *cs* of comparisons between Facebook and Xing



(d) maximum overlaps and distinction distances of all *cs* of comparisons between Facebook and MySpace

Figure 5: Aggregated max-overlaps graphs and average distinction distances. Left y-axis: maximum number of each of detected maximum overlaps inside every *cs* with respect to two specific OSNs. Right y-axis: average distinction distances. For further information, see Section 4.4

existing false positives are negligible compared to the number of profiles that third parties are able to link correctly. Furthermore, the expected gain of information (as shown in Section 3) countervails less probably occurrences of false positives from the perspective of third parties.

6. DISCUSSION

In the following, we discuss further options to link profiles as well as the linkability of profiles registered with different names. Moreover, we discuss the current risk awareness of users regarding the linkability on the basis of the presented results and selected statements from authors of related work.

6.1 Linking via other available information

As shown in Table 1 many profiles provide information in addition to friends lists. In the following we shortly discuss which information can also be (ab-)used to link OSN profiles. Because of space restrictions, we do not show details on the implementation of the investigation on the linkability based on user's shared attributes.

Generally, the more information a user shares in OSNs the higher the probability to find a possibly existing profile belonging to the same user in another OSN. As the user's date of birth is made available by most users in StudiVZ and MySpace, it is possible to link profiles based on this attribute. Furthermore, this applies to attributes, such as the current residence, hobbies and maybe information extractable from status messages. These attributes are made publicly available in more than 20% of the analyzed profiles of both mentioned OSNs. Less applicable for profile linking are attributes describing a user's university. This information is made accessible by 51.67% of StudiVZ users but only by 8.83% of Facebook users and it is never directly and publicly available in MySpace.

6.2 Profile linkability with different names

We showed that friends list based profile linking is possible if users are registered with the same name at the OSNs. This assumption is reasonable in light of the fact that, for instance, one OSN recently implemented an algorithm that examines the credibility of user's given name and does not

permit incongruous names. In general, a user's OSN names need not necessarily be the same to be able to link his profiles. It is sufficient to extract a sample of profiles of OSN B that includes the profile of the owner of the investigated profile of OSN A. In this context, Zafarani and Liu demonstrated that a user name from one OSN can be used to identify other user names of the same person in various web communities, such as other OSNs [20]. In this section, we further discuss the linkability of OSN profiles which are configured with different names. Furthermore, we contrast the associated privacy threat to the threat to users registered with the same name in several OSNs.

Certainly, profiles of two differently named users who have the same city or the same hometown in common might have a higher overlap than two profiles that are just set up with the same name. The probability that a user with exactly the same name as another user exists who, additionally, has same friends is probably very low. However, the probability that in an evaluated population of users living in the same city two users with different names can be found who have a high number of overlapping OSN friends is obviously higher because of overlapping groups of friends in "real life". Nevertheless, we are convinced that the metrics *maximum overlap* and *distinction distance* could also be sufficient to link profiles registered with different names.

Referring to table 1, in three of four OSNs the maximum number of OSN profiles returned by a single search request was between 288 and 545 independent of the given search string. These maximal sizes of query results indicate that not always every profile that matches the search string is returned when searching within an OSN. If, for instance, a user searches for the term "New York" (if such search function is provided by the OSN), not every member living in New York is returned but only about 300 to 550 of them. In order to find a person registered with a known name this number of query results is probably large enough to find a sought-after profile. However, the probability to find a specific profile, from which only a few attributes are known, is still low. In contrast, sampling of every member with a given attribute would be necessary to link profiles registered with different names.

However, it is not possible to evaluate high numbers of user profiles (in which a specific attribute is published) by using typical search features provided by today's OSNs. In case OSNs would provide more sophisticated interfaces (such as SQL interfaces) to search for profiles, comparisons of friends lists could be used to link OSN profiles. The threat reduction regarding profile linking by use of different names would nullify in light of such interfaces. This also applies to user names that can be determined by a third party by use of the methods presented in [20].

6.3 Risk awareness of users

Krishnamurthy describes in [9] the significance of preserving privacy in the Internet and especially in OSNs. Krishnamurthy assumes that a possible reason for the complexity of preserving privacy is the ignorance of users regarding the protection of PII. Users share information through OSNs without thinking about possible consequences. Even if they have the possibility to secure their shared information by adjusting privacy settings, they are reluctant, maybe because of the urge for "satisfaction of the needs for belongingness and the esteem needs through self-presentation" [7].

Consequently, the amount of publicly available PII and, thus, options to link several OSN profiles of single natural persons paves the way for third parties to create comprehensive virtual appearances of users. Even though Facebook gains more and more market share, the linking of profiles might still be profitable for third parties due to the facts that other OSNs still exist, many of them are growing as well, and some OSNs are dedicated to a specific purpose (business contacts, dating, etc.). In general, Torkjazi et al. show that social networks tend to experience a phase in which users are moving to another OSN after a phase of an extensive growth [18]. Thereby, most users probably do not delete old and no longer used profiles and some OSNs do not even provide a possibility for deletion.

As early as 2004, Acquisti wrote that only the combination of the aspects *technology* and *risk awareness* has the potential of successfully solving the privacy problem whereas any of these aspects alone will most probably fail [1]. Recently, Krishnamurthy wrote, "From an awareness point of view, the situation is pretty bad" [9]. As we showed in this study (see Section 3), offering the option to adjust privacy settings is not sufficient to encourage people to use such settings adequately. Meanwhile, the media attention regarding privacy risks in OSNs is urging the majority of users to make use of privacy settings for some of the most critical attributes. However, we showed that in most profiles more than enough information remains publicly available to link OSN profiles of the same natural person. This study might increase the awareness of users and might motivate users to adjust privacy setting via corresponding technology provided by OSNs.

7. CONCLUSION

Online social networks allow users to publish personally identifiable information and people are using this opportunity extensively. However, users expect their data to be visible to users of their respective OSN only, e.g., private information being accessible to their Facebook friends only and not to their Xing contacts. In this paper, we demonstrated how several OSN profiles of a user can be linked at low cost to gather pieces of information. Particularly, we showed that by just comparing the names found within an account's friends list profiles can be reliably linked. Additionally, we discussed that by aggregating OSN profiles and combining user specific information found within different OSNs third parties are able to create much more valuable data sets than possible by just extracting information provided in a single OSN profile.

To get an idea of the potential threat regarding profile linking, we reported how often specific pieces of information are made publicly available by users in four of the most popular OSNs. We showed that friends lists are made publicly accessible in between 40% and 67% of all analyzed OSN profiles. Moreover, we found that the availability of attributes differs with respect to the OSN. In turn, this confirms the statement that profile linking is profitable for third parties because of the increased amount of information they can gather and associate to a single natural person.

We further showed that even a small overlap of two compared friends lists is sufficient to assume that associated profiles correspond to the same natural person. We assumed that the registered names of the profile owners are the same

in both profiles[3]. In this case, we identified that more than three friends have to appear in both compared friends lists to decide whether two compared friends lists belong to profiles of the same user. If an overlap higher than three is detected, the information shared in two OSN profiles can easily be linked by third parties. If users registered their OSN profiles with different names, profile linking might also be possible on the basis of comparisons of friends lists. Only the characteristics regarding the introduced distinction distance and the amount of false positives might differ from comparisons under the aforementioned assumption.

It is desirable that evolution of user behavior regarding the participation in OSNs will be closely monitored in the coming years. However, the presented study might motivate to adjust privacy settings more thoughtfully and should motivate more users to not share as much PII publicly.

8. REFERENCES

[1] A. Acquisti. Privacy in electronic commerce and the economics of immediate gratification. In *Proc. of the 5th ACM Conf. on Electronic Commerce*, EC '04, pages 21–29, New York, NY, USA, 2004. ACM.

[2] L. Bilge, T. Strufe, D. Balzarotti, and E. Kirda. All your contacts are belong to us: automated identity theft attacks on social networks. In *Proc. of the 18th Int'l Conf. on World Wide Web*, WWW '09, pages 551–560, New York, NY, USA, 2009. ACM.

[3] G. Brown, T. Howe, M. Ihbe, A. Prakash, and K. Borders. Social networks and context-aware spam. In *Proc. of the 2008 ACM Conf. on Computer Supported Cooperative Work*, pages 403–412, New York, NY, USA, 2008.

[4] D. H. Chau, S. Pandit, S. Wang, and C. Faloutsos. Parallel crawling for online social networks. In *Proc. of the 16th Int'l Conf. on World Wide Web*, WWW '07, pages 1283–1284, New York, NY, USA, 2007. ACM.

[5] R. Gross and A. Acquisti. Information revelation and privacy in online social networks. In *Proc. of the 2005 ACM Wksp. on Privacy in the Electronic Soc.*, WPES, pages 71–80, New York, NY, USA, 2005. ACM.

[6] P. Jaccard. The Distribution of the Flora in the Alpine Zone. *New Phytologist*, 11(2):37–50, 1912.

[7] H. Krasnova, T. Hildebrand, O. Günther, A. Kovrigin, and A. Nowobilska. Why participate in an online social network: An empirical analysis. In *Proc. of the 16th European Conf. on Information Systems*, 2008.

[8] H. Krasnova and N. F. Veltri. Privacy calculus on social networking sites: Explorative evidence from germany and USA. In *Proc. of the 2010 43rd Hawaii Int'l Conference on System Sciences*, HICSS '10, pages 1–10, Washington, DC, USA, 2010. IEEE.

[9] B. Krishnamurthy. I know what you will do next summer. *SIGCOMM Comput. Commun. Rev.*, 40:65–70, Oct. 2010.

[10] B. Krishnamurthy and C. Wills. Characterizing privacy in online social networks. In *Proc. of the 1st Wksp. on Online Social Networks*, WOSP '08, pages 37–42, New York, NY, USA, 2008. ACM.

[11] B. Krishnamurthy and C. Wills. On the leakage of personally identifiable information via online social networks. *SIGCOMM Comput. Com. Rev.*, 40:112–117, Jan. 2010.

[12] B. Krishnamurthy and C. Wills. Privacy leakage in mobile online social networks. In *Proc. of the 3rd Conf. on Online Social Networks*, WOSN'10, pages 4–4, Berkeley, CA, USA, 2010. USENIX Association.

[13] S. Labitzke, J. Dinger, and H. Hartenstein. How I and others can link my various social network profiles as a basis to reveal my virtual appearance. In *LNI - Proc. of the 4th DFN Forum Com. Techn., GI-Edition*, 2011.

[14] C. A. C. Lampe, N. Ellison, and C. Steinfield. A familiar face(book): profile elements as signals in an online social network. In *Proc. of the SIGCHI Conf. on Human Factors in Computing Systems*, CHI '07, pages 435–444, New York, NY, USA, 2007. ACM.

[15] H. Liu and P. Maes. Interestmap: Harvesting social network profiles for recommendations. In *Proc. of the Beyond Personalization Wksp.*, 2005.

[16] M. Motoyama and G. Varghese. I seek you: searching and matching individuals in social networks. In *Proc. of the 11th Int'l Wksp. on Web Information and Data Management*, WIDM '09, pages 67–75, New York, NY, USA, 2009. ACM.

[17] Z. Stone, T. Zickler, and T. Darrell. Autotagging facebook: Social network context improves photo annotation. In *Proc. of CVPR Wksp. on Internet Vision*, 2008.

[18] M. Torkjazi, R. Rejaie, and W. Willinger. Hot today, gone tomorrow: on the migration of MySpace users. In *Proc. of the 2nd ACM Wksp. on Online Social Networks*, WOSN '09, pages 43–48, New York, NY, USA, 2009. ACM.

[19] G. Wondracek, T. Holz, E. Kirda, and C. Kruegel. A practical attack to de-anonymize social network users. In *Proc. of the 2010 IEEE Symp. on Security and Privacy*, SP '10, pages 223–238, Washington, DC, USA, 2010. IEEE Computer Society.

[20] R. Zafarani and H. Liu. Connecting corresponding identities across communities. In *Proc. of the 3rd Int'l Conf. on Weblogs and Social Media, ICWSM*, 2009.

APPENDIX

A. NOTE ON COMPLIANT SAMPLING

As we want to and have to act compliant to the German data protection laws, the designed software used for the presented study preserves a maximum of privacy. The underlying compliant concept is published in a previous paper [13] in which we describe preliminary work for the study presented in this current paper. For instance, to be compliant to the German law, the software generates random name pairs on the basis of large lists of popular German first and last names and uses this unrevealed name pairs as search strings for sampling of OSN profiles. Furthermore, the software discards information about users provided by OSN profiles subsequent to statistical calculation, so that conclusions regarding specific analyzed profiles are not possible on the basis of stored statistical data. Of course, the network load produced by the software has also been taken into account. The analysis software delimits the number of outgoing packets and prevents accidental flooding.

[3]Recently, one OSN implemented an algorithm to check whether user's given name seems to be his full name.

On analysis of complex network dynamics – changes in local topology

Krzysztof Juszczyszyn
Institute of Computer Science,
Wrocław University of Technology,
27, Wyb. Wyspiańskiego Str.
50-370 Wrocław, Poland
+48 71 3202116
krzysztof@pwr.wroc.pl

Katarzyna Musiał
School of Design,
Engineering and Computing,
Bournemouth University, UK
Fern Barrow, Talbot Campus
BH12 5BB, Poole, UK
+44 12029 65795
kmusial@bournemouth.ac.uk

Marcin Budka
School of Design,
Engineering and Computing,
Bournemouth University, UK
Fern Barrow, Talbot Campus
BH12 5BB, Poole, UK
+44 12029 65795
mbudka@bournemouth.ac.uk

ABSTRACT
Social networks created based on data gathered in various computer systems are structures that constantly evolve. The nodes and their connections change because they are influenced by the external to the network events.. In this work we present a new approach to the description and quantification of patterns of complex dynamic social networks illustrated with the data from the Wroclaw University of Technology email dataset. We propose an approach based on discovery of local network connection patterns (in this case triads of nodes) as well as we measure and analyse their transitions during network evolution. We define the Triad Transition Matrix (TTM) containing the probabilities of transitions between triads, after that we show how it can help to discover the dynamic patterns of network evolution. One of the main issues when investigating the dynamical process is the selection of the time window size. Thus, the goal of this paper is also to investigate how the size of time window influences the shape of TTM and how the dynamics of triad number change depending on the window size. We have shown that, however the link stability in the network is low, the dynamic network evolution pattern expressed by the TTMs is relatively stable, and thus forming a background for fine-grained classification of complex networks dynamics. Our results open also vast possibilities of link and structure prediction of dynamic networks. The future research and applications stemming from our approach are also proposed and discussed.

Categories and Subject Descriptors
E.1 [**Data Structures**]: Graphs and Networks; H.4 [**Information Systems Applications**]: Miscellaneous; J.4 [**Social and Behavioral Sciences**]: Sociology;

General Terms
Algorithms, Measurement, Experimentation, Theory, Verification.

Keywords
Complex social networks dynamics, local topology analysis, triad transition matrix.

The 5th SNA-KDD Workshop '11 (SNA-KDD'11), August 21, 2011, San Diego CA USA . Copyright 2011 ACM 978-1-4503-0225-8...$5.00.

1. INTRODUCTION
Complex networked systems attract more and more researchers from different fields. Networked structures are present in our everyday life – power grids, transportation networks, social networks, biological and ecological networks. Changes in the structure of these systems can have a wide range of consequences for individuals, groups, whole companies or even countries. In this paper we focus on social networks but the presented methodology for investigation the changes in local topology can be applied to all types of complex networks. Of course the interpretation of the results will vary from one application to another but the technique remains unchanged.

When investigating the topological properties and structure of complex networks we face a number of complexity–related problems. In large social networks, tasks like evaluating the centrality measures, finding cliques, etc. require significant computing overhead. However, the technology-based social networks add a new dimension to the known problems of network analysis [11]. The existence of link is a result of a series of discrete events (like email exchanges, phone calls, posting of blog entries) which have some distribution in time. As shown in [9] for various kinds of human activities related to communication and information technologies, the probability of inter-event times (periods between the events, like sending an email) may be expressed as: $P(t) \approx t^{-\alpha}$ where typical values of α are between 1.5 and 2.5. This distribution inevitably results with series of consecutive events ("activity bursts") divided by longer periods of inactivity.

These phenomena have serious consequences when we try to apply the classical structural network analysis (SNA) to dynamic networks. The most popular approach to perform SNA on dynamic networks is to divide the time period under consideration into time windows, then run the structural analysis methods on the networks created for each time window separately. This should show how the measures like node centrality, average path length, group partitions etc. change over time, providing an insight into the evolutionary patterns of the network.

However, the bursty behaviour of the users (long inactivity periods mixed with the bursts of communication activities) causes dramatic changes of any measure when switching from one time window to another. There is a trade-off: short windows lead to chaotic and noisy dynamics of network measures, while long windows give us no chance to investigate time evolution of the network [13][14].

In order to address this problem, a number of approaches designed to predict changes in the structure of dynamic networks were

proposed [15][16]. The special case of this family of methods is a so-called link prediction problem – the estimation of probability that a link will emerge/disappear during the next time window [12].

In this work we propose a method of characterizing the dynamic evolutionary patterns of the network by the analysis of changes in the local topology of connections. This approach stems from our previous experience [17] and will be introduced in Sec. 2. Sec. 3 presents the results, showing the possibility of the characterization of network evolutionary schemes with our approach. These concepts are illustrated by the experiments carried on the large e-mail based social networks build from the mail logs of the Wroclaw University of Technology e-mail social network.

2. LOCAL TOPOLOGY OF ONLINE SOCIAL NETWORKS

2.1 Triads and network motifs

For the abovementioned reasons, standard approaches exploiting network analysis by means of listing several common properties, like the degree distribution, clustering coefficient, network diameter or average path lengths often fail when applied to dynamic complex networks [18]. In many cases it is possible to use random algorithms (like standard preferential attachment) to construct networks with for example exactly the same degree distribution whose structure and function differ substantially (we'll comment on this issue in the next section). Huge network structures (like social, biological, gene networks) should be investigated with more precise and structure–sensitive methods [1][4].

For complex networks, we experience a general rule that the global properties like network clusters, diameter, node degree distribution emerge from the local interactions, which constitute the local topology of the network (direct neighbourhood of a node in simplest case). Even simple local rules may lead to the emergence of dense groups, phase transitions or non-trivial network topologies [20].

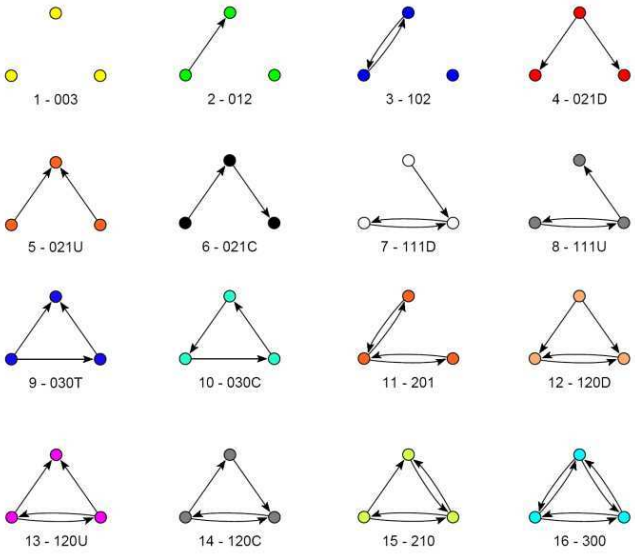


Figure 1 Three-node triads in directed graphs for undistinguishable nodes (picture from [21])

During last years we have experienced the development of a number of methods investigating complex networks by means of their local structure (especially – frequent patterns of connections between nodes). The simplest, and therefore most popular, way to characterize the network in the context of local connections is to examine the links between the smallest non-trivial subgraphs consisting of three nodes – the triads..

A set of 16 triads that do not distinguish between nodes is presented in Figure 1 (Please note that numbers 1, 2, etc will be used further on in this paper when referencing a set of 16 directed triads).

If we want to distinguish between node positions in a triad, there are 64 different triads in a directed graph (Figure 2). In our experiments we distinguish between the nodes, for in our network they are corporate email addresses and, when analysing the connection changes two topologically equivalent subgraphs may in fact represent different behaviour of the users.

Please note the triad ID (the numbers inside the picture of the subgraphs) in Figure 2, as it will be used further on in this paper for identifying the connection patterns. Note also, that there is a correspondence between the IDs and the edit distance between triads – small difference in the ID value in most cases suggests small edit distance (the number of link removal/addition operations needed to transform one triad into another).

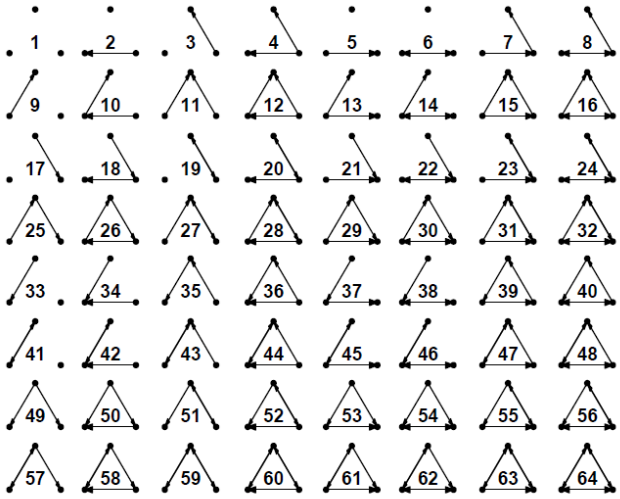


Figure 2 Three-node triads in a directed graph

The basic method utilizing such subgraphs is the well-known triad census, which is enumeration of all triads in the network, allowing to reason about the functional connection patterns of the nodes [18]. Last years have seen the development of more sophisticated approaches, among them motif analysis which aims to characterize the network by the difference between its structures and an ensemble of random networks of the same size and degree distribution. A biased distribution of local network structures (subgraphs) is widely observed in complex biological or technology–based networks. Motif analysis stems from bioinformatics and theoretical biology [1][3], where it was applied to the investigation of huge network structures like transcriptional regulatory networks, gene networks or food webs [4][5]. Although the global topological organization of metabolic networks is well understood, their local organization is still not clear. At the smallest scale, network motifs have been suggested to be the functional building blocks of network biology. So far several interesting properties of large biological network structures were reinterpreted or discovered with the help of motif analysis. There was also one more conclusion: although the properties like node degree distribution, clustering or diameter of real-life networks and their randomly generated counterparts may agree, the local

topology shows distinctive features which are quite different (like the general motif profile of the network expressed by so-called triad significance profile – TSP – a vector of the Z-score measures of the motifs) [6][7][8].
Motif analysis offers low computational overhead and opportunity to gain an insight into the local structure of huge networks which otherwise would require prohibitive computations to investigate. Moreover, the discovered motifs and their numbers enable to assess which patterns of communication appear often in the large social networks and which are rather rare.
In our former research we have investigated the local structure of numerous technology-based networks, among them an e-mail social network of Wroclaw University of Technology (WUT), consisting of more than 5 800 nodes and 140 000 links [2][17]. Our aim was to check if the known properties of local topology in social networks (known on the basis of motif analysis conducted for small non-technology social networks [4]) are also present in large email–based social structures, and if there are some distinct features characteristic to the email communication. The most important conclusion from these experiments was that the TSP of the network is stable over long periods of time. This was confirmed even for periods like summer holidays when the number of links in the university network dropped by 50% and for different link weight thresholds [17]. Summing up – the investigated complex network show statistically stable pattern of connections as a whole, despite the fact that average stability of a single link is quite low: 59% in our case (which means that 41% of the connections will not be present in the next time window). This statement is even more important when we consider that it generally holds regardless of the width of the time window (the link stability of 67% was measured for 30-day time windows). It may be explained by the cumulative effect of the users' activity – for longer time windows the chance that the communication between users will be noticed obviously grows, but on the other hand there are links which will appear only in one of the shorter time windows. Some of the users use their email accounts only occasionally, for example, in our dataset 16% of the users exchanged only one email during the analysed period.
The above observations taken together with the former results cited led to the idea of characterizing the evolutionary patterns of the network by means of the changes in elementary subgraphs, in this particular case – directed triads.
In the next section we introduce the Triad Transition Matrix (TTM) as a basic structure used in our experiments to measure the changes in local topology patterns of the network.

2.2 Triad Transition Matrix

The idea behind the Triad Transition Matrix is to use the data about the history of the network (recorded during past time windows) to derive the probabilities of transitions between triads (patterns of local connections).
The TTM is a g_Δ x g_Δ matrix, where g_Δ is the number of considered subgraphs. For directed triads in our experiments g_Δ = 64 (see Fig.1, however if we decide not to distinguish between the nodes, there will be only 16 possible triads).
The values of TTM entries are defined as follows:

$$TTM_t(i,j) = P(g_i[t] \rightarrow g_j[t+1]) \qquad (1)$$

TTM_t (i,j) is the probability (estimated on the basis of the full subgraph enumeration for networks created from data gathered in time windows [t-1 , t] and [t , t+1]), that a connection pattern g_i detected during [t-1 , t] will transit into g_j during [t , t+1]. In large complex networks (like the one analysed in our experiments) we may expect the occurrence of huge numbers of the triads of all kinds – typically the network comprising of thousands of nodes contains at least million triads.
Our goal was to check if the local network structures (the triads - discussed in the former subsection) show distinguishable evolutionary patterns.

2.3 Size of the Time Window

Selecting the right time window for analysis of network dynamics is a very challenging process, for the reasons briefly presented in the introductory section. There are two standard approaches to split data into time periods:
a) Moving window – length of the time window (e.g. x) and the time interval that is used to move the window (e.g. y) are defined. In order to extract time periods, the time frame of the length x is moved by the factor y. In consequence the whole timeframe under consideration is divided into partially overlapping periods. Note that, the time window and time interval need to be specified in the way that the period from the start date to end date should be completely covered.
b) Equal, separate periods – number of periods e.g. k is set and then the data are divided into k separate, equal periods according to the dates of activity occurrence.
The concepts of both procedures are presented in the Figure 3.

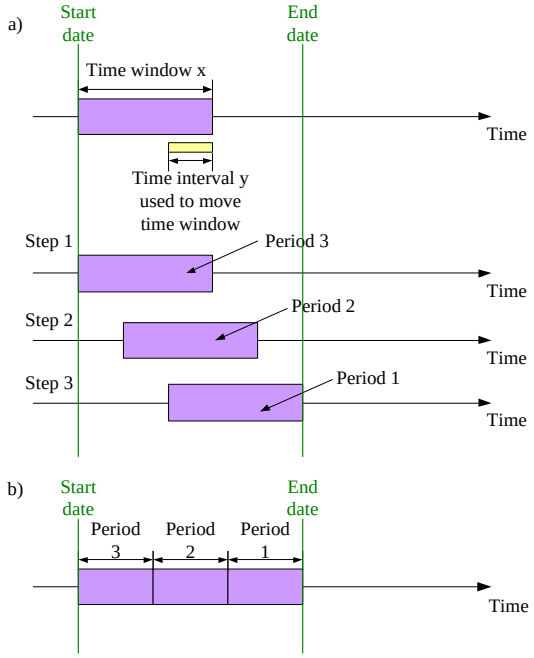


Figure 3 The division of analysed data into time periods using a) the concept of moving window, b) the concept of equal, separate periods.

The most challenging part of both procedures is to decide what should be the length of time window. In this paper, second approach with equal, separate periods has been used, because overlapping windows would make it difficult to uniformly assign the changes in the triad connections to the given time period. Four different time frames have been chosen (1, 3, 7, and 30 days long) and analyses were performed to investigate the influence of time window size on the discovered patterns.

3. EXPERIMENTS

3.1 Data Preparation

The experiments were carried out on the logs from the Wroclaw University of Technology (WUT) mail server, which were pruned to contain only the emails originated from (or: sent to) the staff

members registered at the mail server of the university. First, the data has to be cleansed by removal of spam and unification of duplicated email addresses. There are 5834 active email addresses on the server, which implied that even for the shortest time window of 1 day, there were on average ~2000 active network nodes. For our experiments we used data from a period of 100 days, starting on the 5th of March 2010.

3.2 Experiment Setup

The dataset has been divided into time windows in the following way (note that the length of time windows not always sums to 100 days, but we decided to maintain equal timespans of the windows in order to exclude the possible influence on the results):

- Setup 1: 100 windows – each window 1 day long
- Setup 2: 33 windows – each window 3 days long
- Setup 3: 16 windows – each window 7 days long
- Setup 4: 4 windows – each window 30 days long

In Figure 4 the number of edges for each time window has been presented. The biggest variations in number of links can be observed in the case of time windows of size 1, which corresponds to the phenomena mentioned in Introduction. In particular, the visible drop in the number of links observed every seven days clearly corresponds with Sundays (or, in general, the weekends), when the activity at the university freezes. The situation stabilises when larger time windows are considered. However, the sudden drop of number of links that is visible in windows 25 and 30 (for window size of 3 days), and also for window 13 (for window size of 7 days) and finally window 3 (for window size of 30 days), cannot be properly identified when windows of length 1 day are considered (although it may be associated with student celebration days practically resulting in a few extra holidays in May).

3.3 Frequency of Triad Occurrence

First experiments were performed to check if and how the number of specific triads changes in different windows that are of different size (Figure 5). In this (and only this) analysis we were not distinguishing between the nodes in the triads, which implies that there are only 16 possible directed triads to be considered.

Obviously, the most dynamic changes occur when the time window of size 1 day is investigated. However it can be noticed for 1 day window, that the changes have periodical character. Again, the changes repeat within the cycles which last 7 days each.

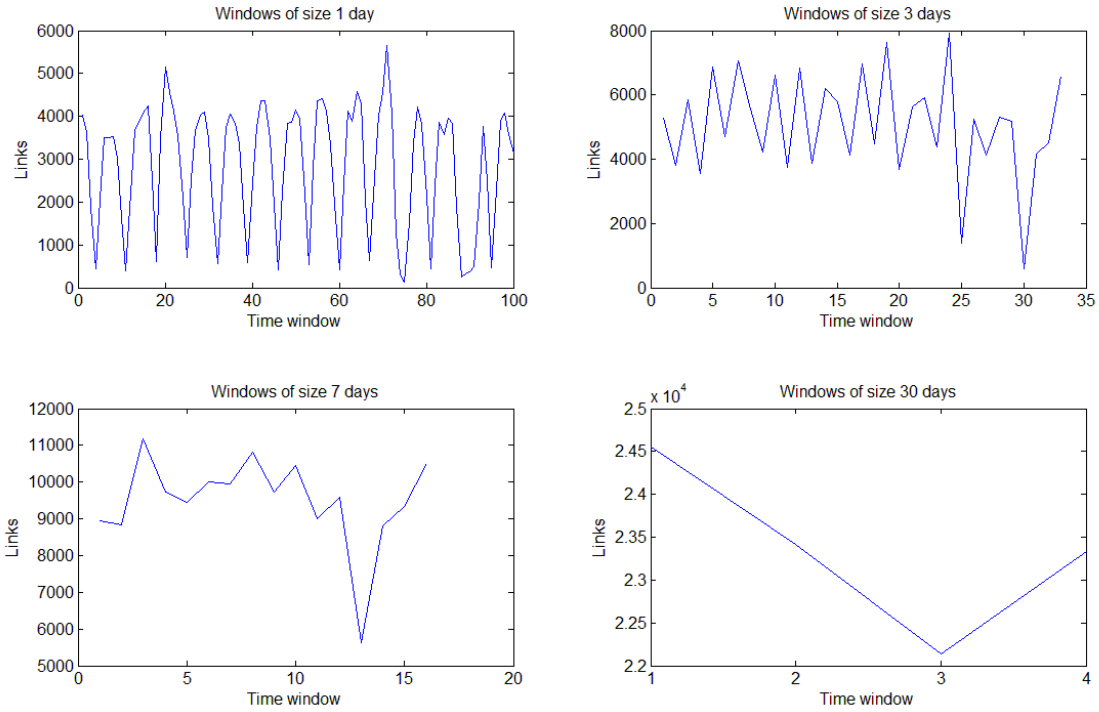


Figure 4 Number of edges for specific time windows

This pattern is still visible when analysing three-day periods but is flatter that in the previous case, since normal communication overlaps with the free time. Analysis of the three-day periods revealed that the number of motifs drops significantly in windows 25 and 30 in comparison to other windows.

Looking at the time windows of the length of 7 days, further flattening of dynamics is clearly visible and it is even more clear for time window of size 1 month.

Figure 5 suggests that in the analysed network one can observe the stability of triad connection pattern. The pattern that covers 7 days and repeats periodically, reflects the changes that occur within the network structure although the rules of these changes can be perceived as stable.

3.4 TTM for different windows size

The number of triads enumerated in the networks derived from different time windows varies from 1.3 million (for 1 day windows) up to 2.2 million (for 30 day windows). The changes in the connections within these triads were used to compute the transition probabilities which constitute the entries of the TTMs.

In Figure 6 the TTM derived on the basis of 4 time windows of size 30 days each are presented (from this point, all results are for

the triads in which the nodes are distinguished from each other, which results in 64 connection schemes between 3 nodes connected by directed links). We may notice that the triad transition probabilities have distinctive form (the distribution of transition probabilities is not flat and looks stable) which reveals some variation but holds the general shape which may be called the evolutionary pattern of the network under consideration –the TTM's entries computed for neighbouring time windows have similar values.

First of all, the value of TTM(1,1) reflects the fact that the network is sparse (the link density is below 1%) which means that most of the possible triads contain no edges. As the result most of the "empty" triads always remain in this state, which gives us a relatively high value of TTM(1,1) for all TTM matrices.

We should also note the high values in the first column of the TTM. This means that when it comes to disappearing of the links, the probability of resetting the entire triad to zero-connection state is relatively high.

On the other hand, it is also visible, that the values on the diagonal of TTM are bigger than most values in their neighbourhood, which shows that the already-formed triads tend (in general) to stay in their current state.

The last important observation is that some triads are special as they show clearly bigger values in their columns of TTM, which means that they are "sinks" of the evolution patterns of connections.

High probability of the fact that triads number 4, 13, and 49 (Figure 1) will not change shows that they may be called stable triads in the analysed network. It reveals the specific characteristic of email networks where many departments exist and there is a lot of broadcast communication (which is not answered) from one person (e.g. secretary) to the large number of recipients.

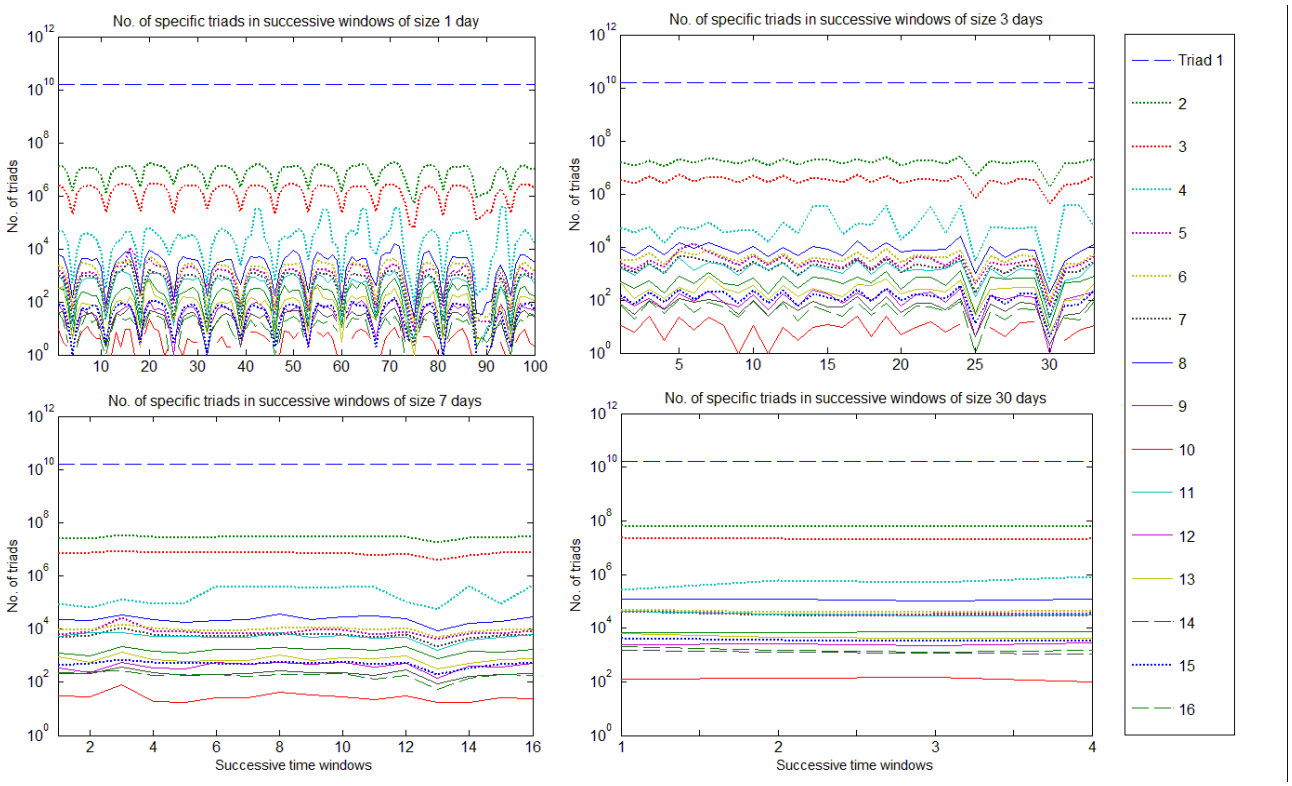


Figure 5 Number of specific motifs (from Figure 1) in successive windows

Interesting situation can be observed in the case of windows of size 1 day. Although most of TTMs are similar to those presented above there is an interesting outlier that reoccur every 7 windows (Figure 7). These Triad Transition Matrices show that almost all network triads disappear as it was pointed out previously. It can reflect the day-off where almost nobody sends emails. The rest of the triads, which do not disappear totally, degenerate to weakly connected triads, e.g. triad number 62 changes into triad number 33.

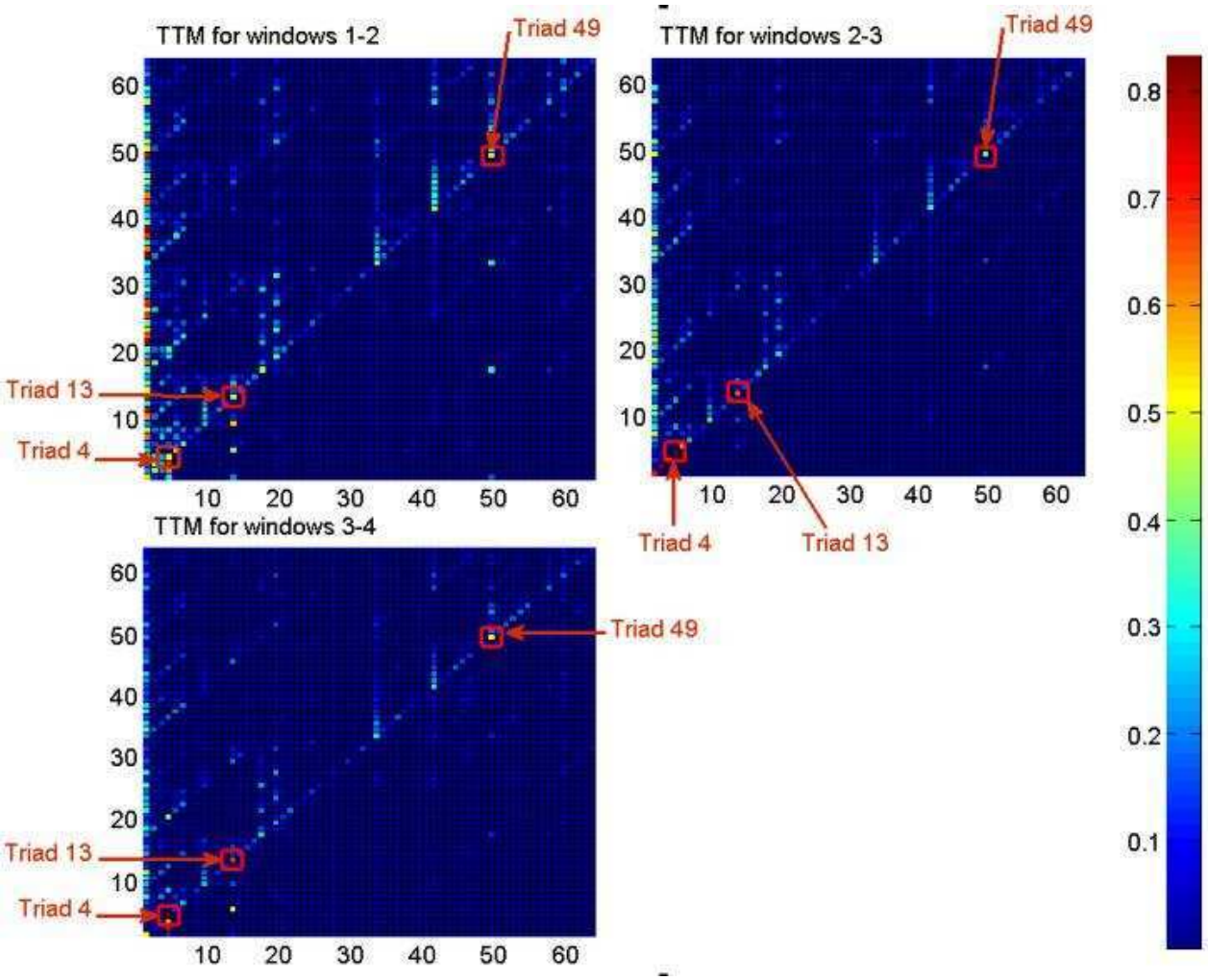


Figure 6 TTM between windows of size 30 days

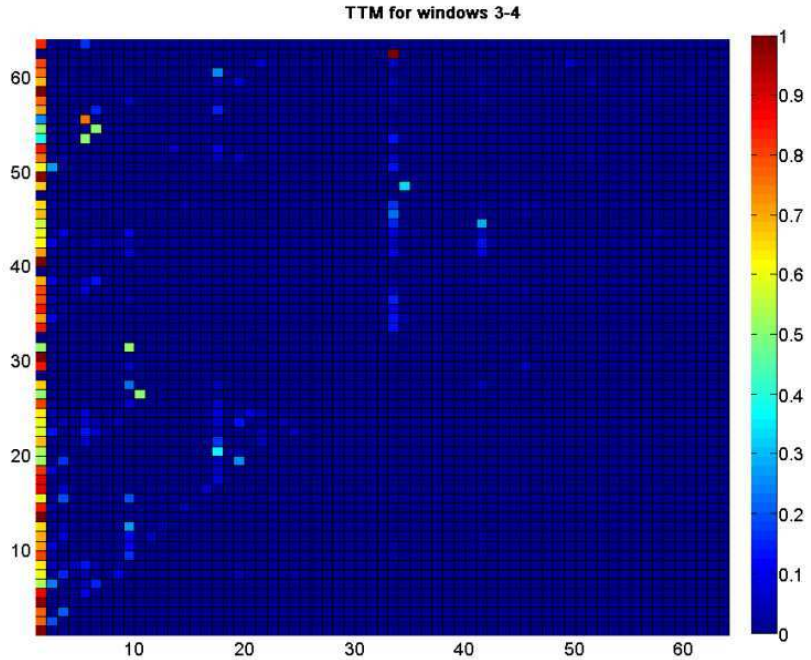


Figure 7 TTM between windows 3 and 4 (1 day time windows)

3.5 Similarity between the TTMs

The analysis of each of the TTM separately is very complex. Thus one of the methods that can be applied to compare TTMs was to calculate the degree to which two Triad Transition Matrices are similar to each other. First step of this approach is to subtract one matrix from another. Then all absolute values of elements from the resulting matrix are summed up. The obtained value is normalised by dividing by the largest possible value – 128 (when two matrices are completely different – the result will be denoted by inv_sim(TTM_1, TTM_2). Finally, the similarity between the matrices, sim(TTM_1, TTM_2), is calculated as:

$$sim(TTM_1, TTM_2) = 1 - inv\_sim(TTM_1, TTM_2) \quad (2)$$

The calculated similarities between 99 TTMs created for time windows of size 1 day are presented in the top-right corner of Figure 8. It can be seen that a repeating pattern can be found. Starting from TTM number 3 and then every 7 days the TTMs are similar to each other. These TTMs (3rd, 10th, 17th, etc.) although very similar to each other are less similar to the rest of the Triad Transition Matrices.

Similarities between TTMs for 3 days' time windows also feature some patterns but they are not as visible as in the case of TTMs for 1 day window. Interesting situation can be observed in the case of time windows of size 3, where the TTMs between window 24 and 25 as well as between 29 and 30 are not similar to other TTMs. It shows that in these TTMs the sudden change of local structure has occurred.

Finally the similarity matrix for TTMs for 7 days time window shows that they are very similar, so the fluctuations in triad profile can be hardly observed.

Next step of the TTMs analysis focused on investigation of the mean and variance of the TTM entries for different sizes of time windows.

The mean values for all experimental setups look very similar (Figure 9). The highest mean values occur in the diagonal and also in the case of loosely connected triads (triads' number 1, 2, 3, 4) of each matrix presented in Figure 9.

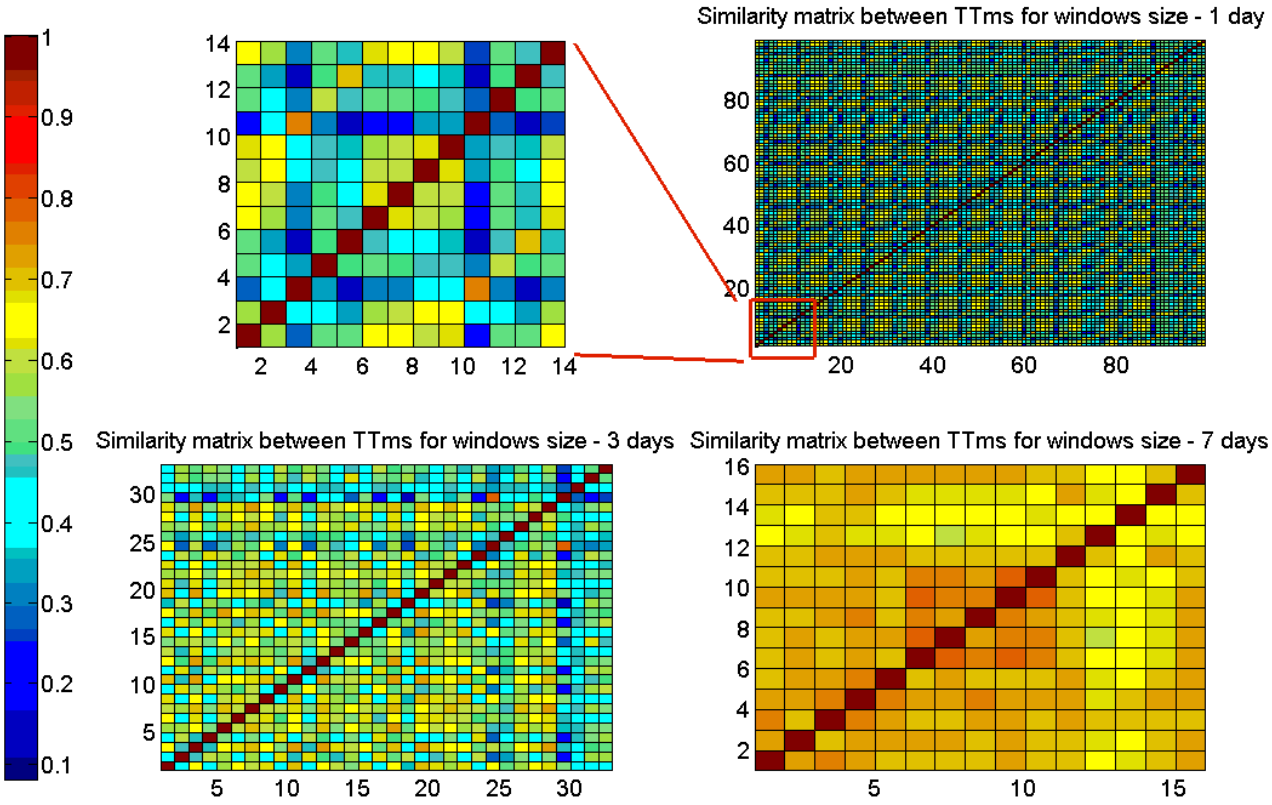


Figure 8 Similarity matrices between TTMs for different sizes of time window

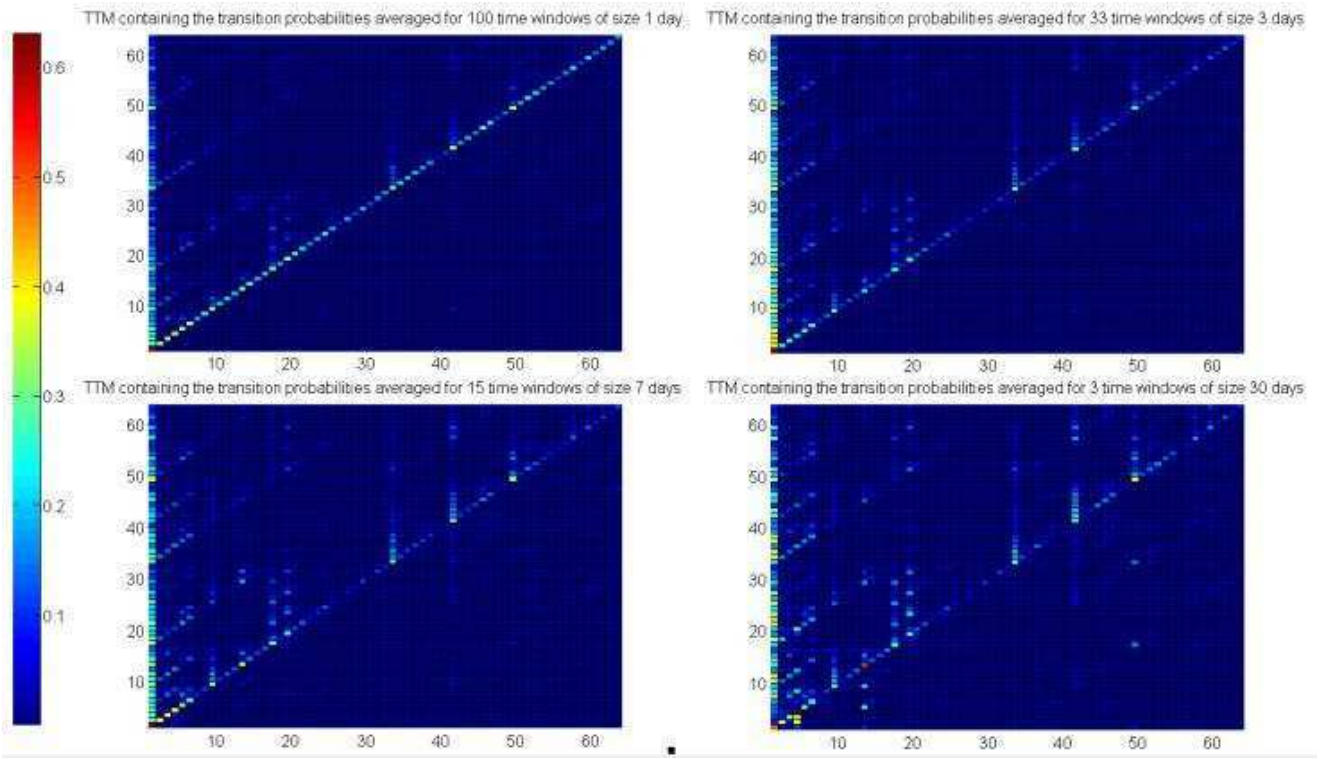


Figure 9 TTMs containing the transition probabilities averaged for number of time windows in a specific setup.

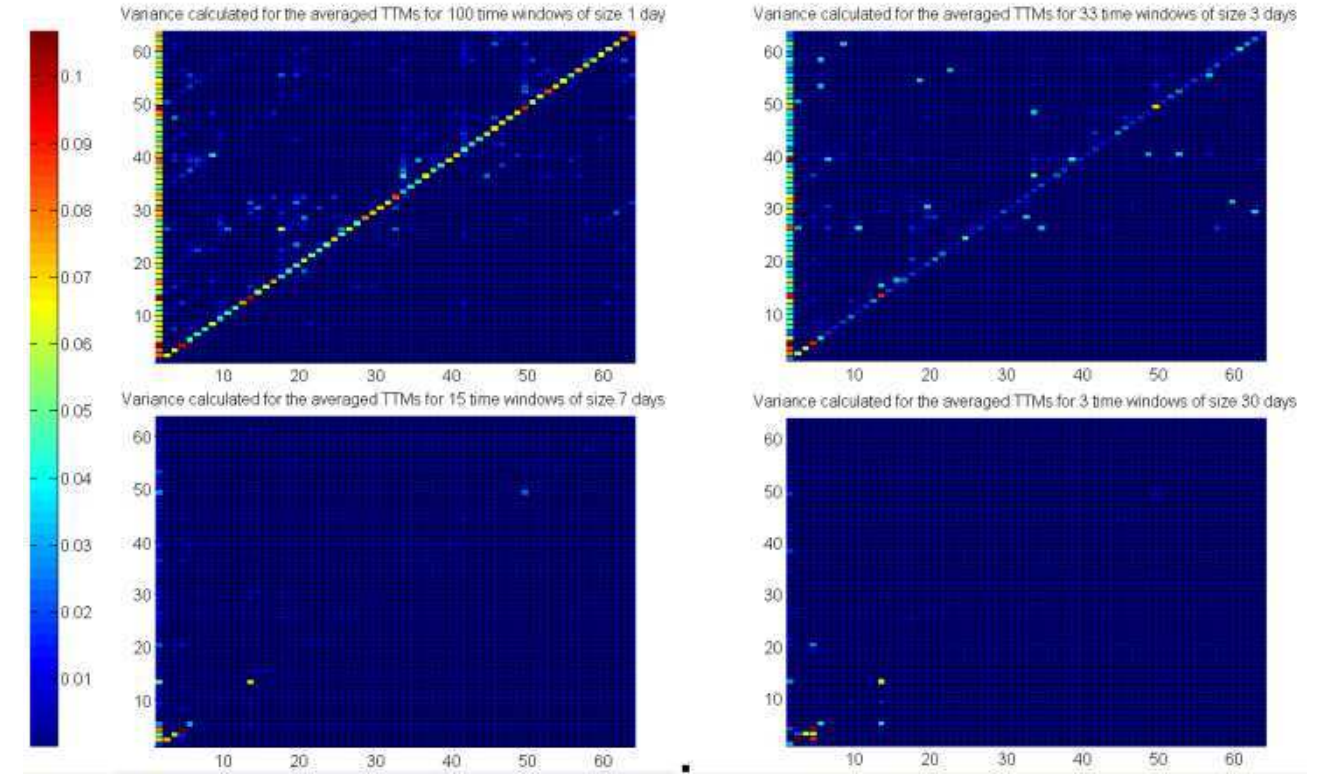


Figure 10 Variance calculated for TTMs containing the transition probabilities averaged for number of time windows in a specific setup.

The variance informs how far the set of numbers lie from the mean (expected value). In Figure 10 a specific variances for corresponding matrices in Figure 9 are presented. As it can be seen the variance is the biggest in the case of TTMs created for windows of size 1 day. However please note that the variance is from the range (0; 0.11) so it is small in general. The variance for TTMs where time windows were either 7 days long or 30 days long is 0 for most of the matrices cells. The variations from mean value exist only in the case of loosely connected triads.

The last fact clearly shows, that for longer time windows the network is still dynamic, there are constant transitions between the connection patterns, however the dynamic changes follow a well-defined pattern.

It is worth to note that although the evolutionary pattern of the network is stable, the links are not, as mentioned in the preceding sections. Thus our approach offers a way to statistically describe the evolutionary patterns in the network and to show that these patterns are relatively stable, even while the links and (in consequence) traditional structural measures of the network are not.

4. CONCLUSIONS AND FUTURE WORK

The approach presented in this paper allows the statistical description of evolutionary patters of complex networks and shows that, for the complex and dynamic social network based on everyday communication in a large company these patterns show stability in spite of link instability. From the other hand, the periodical fluctuations resulting from external events (like weekends, holidays etc.) are also detectable and may be used to quantitatively describe the life of the network.

The results shown in this work open also vast possibilities for future research and building more sophisticated models, allowing also interesting applications. The most interesting and prospective possibilities are:

1. Link and structure prediction: knowing the evolutionary patterns of the network one can predict its future topology, on the level of single link (link prediction) or even global connectivity and cluster structure. Our next step will be the application of the TTMs to link and structure prediction. Our approach seems to be especially promising in the second case, while the triads – topologically - lie between links and network groups. In consequence the TTMs embrace part of group behaviour patterns and may be used to characterize group dynamics as well.

2. Including link weight issues in the analysis. The network analysed in this work was unweighted, but, obviously, not all triads consist of the links of the same intensity. The information about weights may be used as complementary and may help to estimate the triad transition probabilities

3. Reducing the complexity of our method in order to be used to analyse huge networks (like telecom graphs coming from logs of mobile phone operators, who may use the predictions to evaluate the marketing strategies, make customer churn predictions, learn about user behaviour patterns and so on). In practice, for the network of the size used in our experiments the computations were not prohibitive and were made on a state-of-the-art PC (it should be also noted that the method may be easily parallelized, which is not a challenge for algorithms involving triad counting).

4. Network classification according to evolutionary patterns. It will be checked if the complex networks emerging in different areas show stable TTMs and if they can be classified according to them. We have obtained preliminary results showing that the similar patterns (expressed by the stable values of TTMs) exist in other dynamic complex networks – like in communication graphs formed by source-to-destination communication by the computers in distributed systems. We are planning to test the applicability of our approach for the detection of traffic anomalies and network attacks.

5. Tuning of the method by including the periodicity of the changes in TTM entries for consecutive time windows (visible in 1-day windows in our experiments).

5. ACKNOWLEDGMENTS

This work was supported by the Polish Ministry of Science and Higher Education, grant no. N N516 518339.

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 251617.

6. REFERENCES

[1] Itzkovitz S., Milo R., Kashtan N., Ziv G., Alon U. (2003) Subgraphs in random networks. Physical Review E., 68, 026127.

[2] Juszczyszyn K., Musiał K., Kazienko P. (2008), Local Topology of Social Network Based on Motif Analysis, 11th International Conference on Knowledge-Based Intelligent Information & Engineering Systems, KES 2008, Croatia, Springer, LNAI.

[3] Kashtan N., S. Itzkovitz S., Milo R., Alon U. (2004) Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs. Bioinformatics, 20 (11), 1746–1758.

[4] Milo R., Shen-Orr S., Itzkovitz S., Kashtan N., Chklovskii D., Alon U. (2002) Network motifs: simple building blocks of complex networks. Science, 298, 824–827.

[5] Mangan S. Alon U. (2003) Structure and function of the feedforward loop network motif. Proc. of the National Academy of Science, USA, 100 (21), 11980–11985.

[6] Mangan S., Zaslaver A. Alon U. (2003) The coherent feedforward loop serves as a sign-sensitive delay element in transcription networks. J. Molecular Biology, 334, 197–204.

[7] Vazquez, A., Dobrin, R., Sergi, D., Eckmann, J.-P., Oltvai, Z.N., Barabasi, A., 2004. The topological relationship between the large-scale attributes and local interaction patterns of complex networks. Proc. Natl Acad. Sci. USA 101, 17 940.

[8] Young-Ho E., Soojin L., Hawoong J., (2006) Exploring local structural organization of metabolic networks using subgraph patterns, Journal of Theoretical Biology 241, 823–829.

[9] A.-L. Barabási, The origin of bursts and heavy tails in humans dynamics, Nature 435, 207 (2005).

[10] T. Gross, H. Sayama (Eds.): Adaptive networks: Theory, models and applications, Springer: Complexity, Springer-Verlag, Berlin-Heidelberg, 2009.

[11] J. Kleinberg, J. The convergence of social and technological networks. Communications of the ACM Vol. 51, No.11, 66-72, 2008.

[12] D. Lieben-Nowell, J.M. Kleinberg: The link-prediction problem for social networks. JASIST (JASIS) 58(7), pp.1019-1031, 2007.

[13] D.Braha, Y. Bar-Yam, From Centrality to Temporary Fame: Dynamic Centrality in Complex Networks, Complexity, Vol. 12 (2), pp. 59-63, 2006.

[14] D. Kempe, J. Kleinberg, A. Kumar, Connectivity and inference problems for temporal networks. Journal of Computational System Science, 64(4):820–842, 2002.

[15] M. Lahiri, Tanya Y. Berger-Wolf: Mining Periodic Behavior in Dynamic Social Networks. ICDM pp.373-382, 2008.

[16] Lisa Singh, Lise Getoor: Increasing the Predictive Power of Affiliation Networks. IEEE Data Eng. Bull. (DEBU) Vol. 30 No. 2, pp. 41-50, 2007.

[17] K. Juszczyszyn, K. Musial, P. Kazienko, B. Gabrys: Temporal Changes in Local Topology of an Email-Based Social Network. Computing and Informatics 28(6): 763-779 (2009).

[18] S. Wasserman, K. Faust, Social network analysis: Methods and applications, Cambridge University Press, New York, 1994.

[19] Batagelj, V., Mrvar, A., A subquadratic triad census algorithm for large sparse networks with small maximum degree. Social Netw. 23, 237-243, 2001.

[20] T.Gross, H.Sayama (eds)., Adaptive Networks, Springer, New York, 2009.

[21] W. de Nooy, A. Mrvar, V. Batagelj, Exploratory Social Network Analysis with Pajek, Structural Analysis in the Social Sciences 27, Cambridge University Press, 2005.

Learning and Predicting Dynamic Behavior with Graphical Multiagent Models

Quang Duong[†] Michael P. Wellman[†] Satinder Singh[†] Michael Kearns[*]
[†]Computer Science and Engineering, University of Michigan
[*]Computer and Information Science, University of Pennsylvania

ABSTRACT

Factored models of multiagent systems address the complexity of joint behavior by exploiting locality in agent interactions. *History-dependent graphical multiagent models* (hGMMs) further capture dynamics by conditioning behavior on history. We propose a greedy algorithm for learning hGMMs from time-series data, inducing both graphical structure and parameters. To evaluate this learning method, we employ human-subject experiment data for a dynamic consensus scenario, where agents on a network attempt to reach a unanimous vote. We empirically show that the learned hGMMs directly expressing joint behavior outperform alternatives in predicting dynamic voting behavior. Analysis of learned graphs reveals patterns of interdependence relations not directly reflected in the original experiment networks.

1. INTRODUCTION

Modeling dynamic behavior of multiple agents presents inherent scaling problems due to the exponential size of any enumerated representation of joint activity. Even if agents make decisions independently, conditioning actions on each other's prior decisions or on commonly observed history induces interdependencies over time. To address this complexity problem, researchers have exploited the localized effects of agent decisions by employing *graphical models* of multiagent behavior. This approach has produced several (related) graphical representations capturing various facets of multiagent interaction [Koller and Milch, 2003, Kearns et al., 2001, Jiang et al., 2008, Gal and Pfeffer, 2008, Duong et al., 2008]. The *history-dependent graphical multiagent models* (hGMMs) of [Duong et al., 2010] express multiagent behavior on an undirected graph, and capture dynamic relations by conditioning action on history. The authors showed that for a fixed graph structure and history representation, the expressive power to specify local joint behavior provided advantages over models that assume conditional independence given history.

It is not always given or apparent how to organize agents on a sufficiently sparse graph for tractable modeling. Information the modeler may have about the agents' experiment network is not definitive, as the graphical structure of the optimal predictive multiagent model of behavior does not necessarily correspond to the experiment network. Moreover, these networks may be too complex for practical computation without imposing strong independence assumptions on behavior. We thus consider inducing the graphical structure a necessary part of the modeling effort.

We motivate and empirically evaluate our learning technique with the dynamic consensus experiments conducted by Kearns et al. [2009]. The human subjects in these experiments were arranged on a network, specifying for each subject (also called *player*, or *agent*) the set of other players whose voting decisions he or she can observe. The experiment network for this voting scenario provides a basis for expecting that joint agent behavior may exhibit some locality that we can exploit in a graphical model for prediction. However, the graph structure of the optimal predictive model, as noted above, need not mirror the experiment network of the voting scenario, and moreover, the complex experiment network instances we study render computation on the corresponding hGMMs intractable. Unlike previous models of this domain, we aim to capture dynamic voting behavior, with no particular focus on the final voting outcome.

In this study, we propose a greedy algorithm for learning the graphical structure and parameters of an hGMM that can effectively and compactly capture joint dynamic behavior. Moreover, we empirically investigate the learned models' predictions of voting behavior and compare their performance with those of different baseline multiagent models, and demonstrate that models expressing joint behavior outperform the alternatives in predicting voting behavior. We further examine the learned hGMM graphical structures in order to gain better insights and understanding of voting behavior dynamics on networks, as well as the network structure's effect on collective actions. We are particularly interested in examining connections between nodes of different characteristics in the hGMM induced graphical structures. For example, our analysis compares the average number of edges between more and less densely connected nodes in the learned hGMM graphs and their corresponding original experiment networks. We further start a discussion on the challenges and potential solutions of predicting final outcomes of the aforementioned dynamic consensus experiments.

The 5th SNA-KDD Workshop '11 (SNA-KDD'11), August 21, 2011, San Diego CA USA . Copyright 2011 ACM 978-1-4503-0225-8...$5.00.

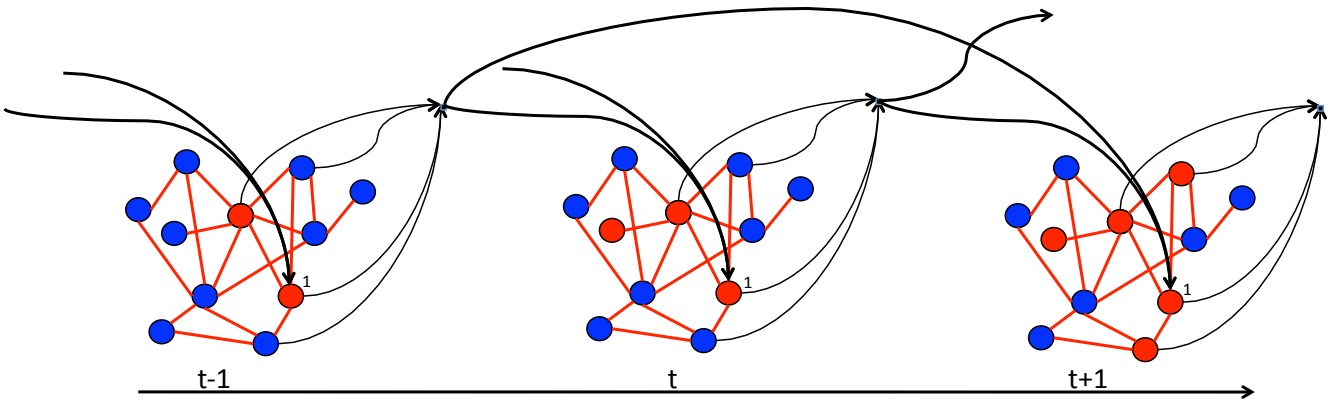


Figure 1: An example hGMM over three time periods. Undirected edges capture the correlation among agents at the present time. Directed edges model the conditioning of an agent's action on others' past actions. Only directed edges pertinent to agent 1 are shown in details: agent 1's current action is conditioned on its neighbors' actions in the previous two time periods.

We provide background information on hGMMs and the dynamic consensus experiments in Sections 2 and 3, respectively. We then present a variety of candidate model forms in Section 4. Section 5 provides motivations and details of our greedy model learning algorithm that simultaneously estimates a model's parameters and constructs its interaction graph. Our empirical study in Section 6 compares different models across three experiment settings, examines the learned graph structures against the original experiment networks, and discusses the problems and potential solutions of predicting these experiments' end state results. We offer concluding remarks and suggest potential extensions in Section 7.

2. HISTORY-DEPENDENT GRAPHICAL MULTIAGENT MODELS

We model behavior of n agents over a time interval divided into discrete periods, $[0, \ldots, T]$. At time period t, agent $i \in \{1, \ldots, n\}$ chooses an action a_i^t from its action domain, A_i, according to its *strategy*, σ_i. Agents can observe others' and their own past actions, as captured in *history* H^t, up to time t. Limited memory capacity or other computational constraints restrict an agent to focus attention on a subset of history H_i^t considered in its probabilistic choice of next action: $a_i^t \sim \sigma_i(H_i^t)$.

A *history-dependent graphical multiagent model* (hGMM) [Duong et al., 2010], $hG = (V, E, A, \pi)$, is a graphical model with graph elements V, a set of vertices representing the n agents, and E, edges capturing pairwise interactions between them. Component $A = (A_i, \ldots, A_n)$ represents the action domains, and $\pi = (\pi_1, \ldots, \pi_n)$ potential functions for each agent. The graph defines a neighborhood for each agent i: $N_i = \{j \mid (i,j) \in E\} \cup \{i\}$, including i and its neighbors $N_{-i} = N_i \setminus \{i\}$.

The hGMM captures agent interactions in dynamic scenarios by conditioning joint agent behavior on an abstracted history of actions H^t. The history available to agent i, $H_{N_i}^t$, is the subset of H^t pertaining to agents in N_i. Each agent i is associated with a potential function $\pi_i(a_{N_i}^t \mid H_{N_i}^t)$: $\prod_{j \in N_i} A_j \rightarrow R^+$. The potential of a local action configuration specifies its likelihood of being included in the global outcome, conditional on history. Specifically, the joint distribution of the system's actions taken at time t is the product of neighbor potentials [Daskalakis and Papadimitriou, 2006, Duong et al., 2010, Kakade et al., 2003]:

$$\Pr(a^t \mid H^t) = \frac{\prod_i \pi_i(a_{N_i}^t \mid H_{N_i}^t)}{Z}. \quad (1)$$

The complexity of computing the normalization factor Z in (1) is exponential in the number of agents, and thus precludes exact inference and learning in large models. Duong et al. [2010] address this problem by approximating Z using the *belief propagation* method [Broadway et al., 2000], which has shown good results with reasonable time in sparse cyclic graphical structures. In particular, we adopt the package libDAI [Mooij, 2010] for approximating Z in our hGMM implementation.

We stress that there are two different types of edges in an hGMM, as depicted in Figure 1. Each node i's undirected edges define its neighborhood N_i^u of the present time's configuration $a_{N_i^u}^t$, and thus model the correlations among their actions. The directed edges ending at i originate from nodes whose actions in the past influence how i chooses its action in the present, and consequently form i's other neighborhood N_i^d. These two different neighborhood definitions lead to a more generalized form of the potential function $\pi_i(s_{N_i^u}^t \mid s_{N_i^d}^{t-1})$. For simplicity, this paper assumes that N_i^u, N_i^d and N_i are the same for all i, and employ the potential function form $\pi_i(s_{N_i}^t \mid s_{N_i}^{t-1})$.

3. DYNAMIC CONSENSUS EXPERIMENTS

We evaluate our approach with human-subject data from the dynamic consensus game introduced and studied by Kearns et al. [2009]. Each agent in this game chooses to vote either blue (0) or red (1), and can change votes at any time. Agents are connected in a network, where each can observe its neigh-

bors' votes. The scenario terminates when: (i) agents converge on action $a \in \{0, 1\}$, in which case agent i receives reward $r_i(a) > 0$, or (ii) they cannot agree by the time limit T, in which case rewards are zero. Figure 2 illustrates the dynamic behavior of an example voting experiment network.

Agents may have different preferences for the available vote options, reflected in their reward functions. As nobody gets any reward without a unanimous vote, agents have to balance effort to promote their own preferred outcomes against the common goal to reach consensus. Another important feature of the dynamic consensus game is that agent i knows the votes of only its neighbors, and its own local graph structure, including its neighbors N_i, the degree of each neighbor $k \in N_i$, and edges between its neighbors. This raises the question of how agents take into account their neighbors' voting patterns and their partial knowledge of experiment network structure.

Kearns et al. [2009] conducted a series of human-subject experiments studying how human agents behave in 81 different instances of the voting game. They varied reward preference assignments and experiment network structure in these experiment instances, and thus were able to collect data about these factors' effects on the consensus voting results, as well as the agent strategies employed. Figure 2 exhibits a run for the experimental network labeled power22, discussed below. Study goals included developing models to predict a given scenario's voting outcome, and if a consensus is reached, its convergence time. This problem also served as the foundation for analysis of adaptive strategies and theoretical constraints on convergence [Kearns and Tan, 2008]. In particular, they were interested in developing models that would predict whether a given scenario would be likely to converge to consensus, and if so, how fast and on which outcome. Exploring the problem also led this group to analyze a family of adaptive strategies, and establish the impossibility of converging to the preferred outcome in the worst case [Kearns and Tan, 2008].

4. MODELING DYNAMIC VOTING BEHAVIOR

We present four multiagent behavior model forms designed to capture voting behavior dynamics in the dynamic consensus experiments. All are expressible as hGMMs. Only the first, however exploits the flexibility of hGMMs to express dependence of actions within a neighborhood given history (1), hence we refer to this as the *joint behavior model* (JBM).

The other three forms model agent behaviors individually: for each agent we specify a probabilistic strategy $\sigma_i(H_i^t) = \Pr(a_i^t \mid H_i^t)$. Such a formulation captures agent interactions by the conditioning of individual behavior on observed history. The agents' actions are probabilistically dependent, but conditionally independent given this common history, yielding the joint distribution

$$\Pr(a^t \mid H^t) = \prod_i \sigma_i(H_i^t). \tag{2}$$

We refer to a dynamic multiagent model expressible by (2) as an *individual behavior hGMM* (IBMM). Conditional independence given history is a compelling assumption for autonomous agents. Indeed, independent choice may even be considered definitional for autonomy. In practice, however, it is often infeasible to specify the entire history for conditioning due to finite memory and computational power, and the assumption may not hold with respect to partial history. History abstraction will generally introduce correlations between agents actions, even if they are independently generated on full history [Duong et al., 2010]. Nevertheless, assuming conditional independence between agents' actions given history exponentially reduces the model's complexity, or more specifically, the computational complexity of the joint probability distribution of the system's actions.

The first of three IBMMs we present is designed as an independent behavior version of JBM; thus, we call it simply the *individual behavior model* (IBM). The remaining two models are based on proposals and observations from the original experimental analysis [Kearns et al., 2009], and are labeled *proportional response model* (PRM) and *sticky proportional response model* (sPRM), respectively.

4.1 JOINT BEHAVIOR MODEL

First, we consider how to summarize a history $H_{N_i}^t$ of length h relevant to agent i. Let indicator $I(a_i, a_k) = 1$ if $a_i = a_k$ and 0 otherwise, and $I(a_{N_i}^{t_1}, a_{N_i}^{t_2}) = 1$ iff $I(a_k^{t_1}, a_k^{t_2}) = 1$ for all $k \in N_k$. We encode the historical frequency of a local configuration a_{N_i} as

$$e(a_{N_i}, H_{N_i}^t) = \frac{\sum_{\tau=t-h}^{t-1} I(a_{N_i}, a_{N_i}^\tau)}{h}.$$

However, this encoding only counts exact matches of a_{N_i} in history H_{N_i}, and thus heavily biases against local configurations that have not happened in the past, effectively hindering a neighborhood of agents with different vote preferences from reach consensus. We introduce an alternative frequency function that captures how close a local configuration to past configurations:

$$f(a_{N_i}, H_{N_i}^t) = \frac{\sum_{\tau=t-h}^{t-1} 0.5^{\sum_{k \in N_i} 1 - I(a_k, a_k^\tau)}}{h},$$

where the exponent $\sum_{k \in N_i} 1 - I(a_k, a_k^\tau)$ basically counts the number of pair-wise mismatches between a_{N_i} and $a_{N_i}^\tau$ and the discount factor is fixed as 0.5. To simplify exposition, we henceforth drop the time superscript t and the neighborhood subscript N_i from $H_{N_i}^t$.

In formulating JBM's potential function, we attempt to capture the impact of past collective choices of i's neighborhood, and i's relative preference for each action. As subjects in the lab experiments were able to keep track of how much time they have left before having to reach a consensus, we also would like to capture the time effect on agents' reasoning in the potential function. We define $r_i(a_{N_i})$ as the product of time-discounted $r(a_i)$ and a heuristic attenuation based on how many neighbors currently vote differently:

$$r_i(a_{N_i}) = \alpha^{\sum_{k \in N_i} (1 - I(a_i, a_k))} r(a_i)^{\frac{T-t}{T}\beta},$$

where $\alpha \in [0, 1]$ and $\beta \geq 0$. Observe that $r_i(a_{N_i})$ as we define it is increasing in the number of iÕs neighbors playing a_i, reflecting the positive influence of neighbor choices on i. The potential function for agent i is given by

$$\pi_i(a_{N_i} \mid H) = r_i(a_{N_i}) f(a_{N_i}, H)^\gamma, \tag{3}$$

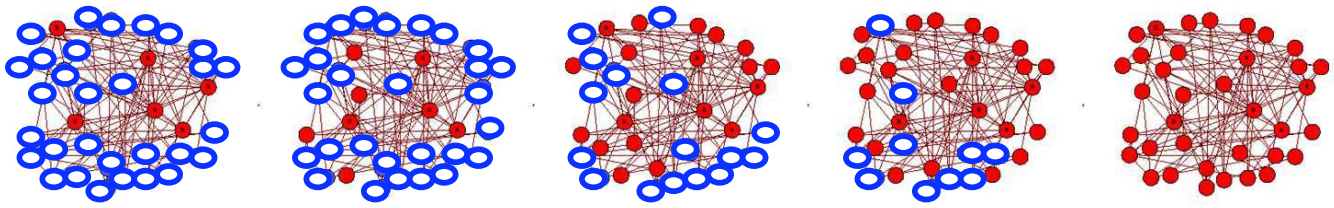


Figure 2: Time snapshots of an experiment run where the densely connected minority group (red) exerts strong influences on others' votes [Kearns et al., 2009].

where $\gamma \geq 0$ denotes the weight or importance of the historical frequency $f(a_{N_i}, H)$ relative to the estimated reward $r_i(a_{N_i})$. The normalized product of these potentials specifies joint behavior as described in (1). The model maintains three parameters α, β and γ.

4.2 INDIVIDUAL BEHAVIOR MODEL

The IBM of the dynamic consensus experiments retains the main elements of JBM (3) with a similar parameter set $\{\alpha, \beta, \gamma\}$, while imposing conditional independence among agents' actions given the common history. Let us define $f(a_i, H^t_{N_i})$ as the frequency of action a_i being chosen by other agents in $H^t_{N_i}$, capturing the degree to which a_i is similar to past choices by i's neighbors in $H^t_{N_i}$.

$$f(a_i, H^t_{N_i}) = \frac{\sum_{k \in N_i - \{i\}} \sum_{\tau=t-h}^{t-1} I(a_i, a^\tau_k) + 1}{h|N_i \setminus \{i\}|} \quad (4)$$

We add one to the numerator to ensure that the corresponding term does not vanish when the configuration a_i does not appear in $H^t_{N_i}$. The probabilistic IBM strategy is then given by:

$$\Pr(a_i \mid H) = \frac{1}{Z_i} r_i(a_i)^{\alpha + \frac{T-t}{T}\beta} f(a_i, H)^\gamma. \quad (5)$$

Z_i is the normalization factor over all $a_i \in A_i$ (this normalization sums only over the actions of a single agent and is, therefore, easy to compute).

4.3 PROPORTIONAL RESPONSE MODEL

We also include in our study the proportional response model, PRM, suggested by Kearns et al. [2009] as a reasonably accurate predictor of their experiments' final outcomes. PRM specifies that voter i chooses action a_i at time t with probability proportional to $r_i(a_i)g(a_i, a^{t-1}_{N_i})$, where $g(a_i, a^{t-1}_{N_i})$ denotes the number of i's neighbors who chose a_i in the last time period,

$$\Pr(a^t_i \mid H^t) \sim r_i(a^t_i) g(a_i, a^{t-1}_{N_i}). \quad (6)$$

4.4 STICKY PROPORTIONAL RESPONSE MODEL

PRM does not capture players' tendency to start with their preferred option, switching their votes only after collecting additional information about their neighbors over several time periods [Kearns et al., 2009]. Therefore, we introduce the *sticky proportional response model*, sPRM, which contains an additional parameter ρ reflecting an agent's stubbornness in voting its preferred option, regardless of its neighbors' past choices. Intuitively, a player's inherent bias toward its preferred option decays exponentially until there is no bias:

$$\Pr(a^t_i \mid H^t) \sim r_i(a^t_i) g(a_i, a^{t-1}_{N_i})(1 + I^{\max}_{a_i} e^{1-t\rho}), \quad (7)$$

where $I^{\max}_{a_i} = 1$ if $a_i = \arg\max r_i(a)$ and 0 otherwise.

5. LEARNING PARAMETERS AND GRAPHICAL STRUCTURES

5.1 PARAMETER LEARNING

We first address the problem of learning the parameters of an hGMM hG given the underlying graphical structure and data in the form of a set of joint actions for m time steps, $X = (a^0, \ldots, a^m)$. For ease of exposition, let θ denote the set of all the parameters that define the hGMM's potential functions. We seek a θ maximizing the log likelihood of X,

$$L_{hG}(X; \theta) = \sum_{k=0}^{m-h} \ln(\Pr{}_{hG}(a^{k+h} \mid (a^k, \ldots, a^{k+h-1})); \theta)).$$

We use gradient ascent to update the parameters: $\theta \leftarrow \theta + \lambda \nabla\theta$, where the gradient is

$$\nabla\theta = \frac{\partial L_{hG}(X; \theta)}{\partial \theta},$$

and λ is the learning rate, stopping when the gradient is below some threshold. We employ this same technique to learn the parameters of the baseline models.

5.2 MODEL LEARNING

Each of the consensus voting experiments involves 36 human players. The largest neighborhood size in these games ranges from 16 to 20, rendering computing exact data likelihood for a joint behavior model of this complexity (required for parameter learning described above) infeasible. Preliminary trials with the belief propagation approximation algorithm [Broadway et al., 2000] on these models indicated that its computational saving would still be insufficient for effective learning. Thus, we need to employ models with simpler graphs in order to take advantage of hGMM's expressiveness in representing joint behavior. Toward this end, we developed a structure learning algorithm that produces graphs for hGMMs within specified complexity constraints.

Though dictated by computational necessity, automated structure learning has additional advantages. First, we observe that there is no inherent reason that the interaction graph should constitute the ideal structure for a predictive graphical model for agent behavior. Even though actual agent behavior is naturally conditioned on its observable history (as captured by the interaction graph), once we abstract the history representation it may well turn out that non-local

historical activity provides more useful predictive information. If so, the structure of the learned graph itself may provide interesting insights on the agents' networked behavior.

Note that the complexity problem does not apply for IBMMs, which are tractable based on their conditional independence assumption. Nevertheless, since as GMMs their graphical structures capture dependence unconditional on history among agents' actions, it may also be interesting to employ our algorithm for learning IBMM structure.

Our learning algorithm combines greedy addition of edges with gradient-descent parameter optimization, as described by the following steps:

1: Start with an hGMM whose graphical structure $(V, E = \emptyset)$ is completely disconnected.
2: Learn θ that maximizes the training data's likelihood $L(X;\theta)$
3: **repeat**
4: **repeat**
5: $\tilde{E} \leftarrow \{(i,j) \mid (i,j) \notin E; |N_i|, |N_j| < d_{\max}; \tilde{L}_{(i,j)} = L_{E\cup(i,j)}(X;\theta) \geq L(X;\theta)\}$
6: $E \leftarrow E \cup (i_{\max}, j_{\max})$ such that $(i_{\max}, j_{\max}) = \arg\max_{(i,j)\in\tilde{E}} \tilde{L}_{(i,j)}$
7: **until** $\tilde{E}$ is empty or the number of added edges is greater than $e_{\max}$
8: Learn θ that maximizes the training data's likelihood $L(X;\theta)$
9: **until** no edges can be added

5.3 EVALUATION

We evaluate the learned multiagent models by their ability to predict future outcomes, as represented by a test set Y. Given two models M_1 and M_2, we compute their corresponding log-likelihood measures for the test data set Y: $L_{M_1}(Y)$ and $L_{M_2}(Y)$. Note that since log-likelihood is negative, we instead examine the negated log-likelihood measures, which means that M_1 is better than M_2 predicting Y if $-L_{M_1}(Y) < -L_{M_2}(Y)$, and vice versa.

6. EMPIRICAL STUDY

We empirically evaluate the predictive power of JBM in comparison with IBM, PRM, and sPRM, using the dynamic consensus experiment data from Kearns et al. [2009]. We also compare JBM against a naive guessing model, nM, which initially assigns each a_i a probability proportional to $r_i(a_i)$ and linearly converges to a uniform distribution of agent actions as the game progresses. We are further interested in examining the graphs induced by structure learning, and relating them to their corresponding original game networks using different statistical measures.

The human-subject experiments are divided into nine different sets, each associated with a network structure. These structures differ qualitatively in various ways, characterized by node degree distribution, ratio of inter-group and intra-group edges, and the existence of a well-connected minority [Kearns et al., 2009]. In particular, networks whose edges are generated by a random Erdos-Renyi (ER) process [Erdos and Renyi, 1959] has a notably more heavy-tailed degree distribution than those generates by a preferential attachment (PA) process [Barabási and Albert, 1999]. For each experimental trial, human subjects were randomly assigned to nodes in the designated network structure, and preferences based on one of three possible incentive schemes. Since players in these experiments can change their votes at any time, the resulting data is a stream of asynchronous vote actions. We discretize these streams for data analysis, recording the players' votes at the end of each time interval of length δ seconds. We experiment with different interval lengths $\delta \in \{0.5, 1.5\}$ in this study.

In our study, we learn predictive models for each network structure, pooling data across subject assignments and incentive schemes. This approach is based on the premise that network structure is the main factor governing the system's collective behavior, in line with the findings of Kearns et al. [2009]. In each experiment set, we use four of the nine trials for training the predictive models for each form. The hGMM graphical structures are learned with node degree constraint $d_{\max} = 10$, while the maximum degree of the individual behavior models is restricted to the greatest node degree of the original network. We also set $e_{\max} = 5$. We then evaluate these models based on their predictions over a test set comprising the other five experimental trials. This process is repeated five times, each of which uses a different training trial set randomly chosen from the original trials. Each data point in our reported experimental results averages over these five repetitions.

We reuse Kearns et al.'s labels for the three experiment networks studied in this analysis, listed in Table 1.

Table 1: Voting Experiment Settings

Label	Strong Minority	Graph Generator Process
coER_2	No	Erdos-Renyi
coPA_2	No	Preferential attachment
power22	Yes	Preferential attachment

6.1 PREDICTIONS

We first examine predictions of players' votes in each time period conditional on available history. Figure 3 shows the negated log-likelihood of the testing data set induced by models JBM,IBM, PRM, sPRM, and nM. Note that the learned graphical structures in JBM are different from the original game experiments' networks, upon which other models are constructed. We observe that JBM performs significantly better than IBM, PRM, sPRM, and nM in predicting dynamic agent behavior in the dynamic consensus experiments for all three experiment sets, given data discretized with two different interval lengths of 0.5 and 1.5 (differences significant at $p < 0.02$. Contrary to the expectation that the less historical information a model uses, the lower its prediction performance, JBM and IBM that employ only the last $h = 3$ periods of historical data generate better predictions than those with $h = 8$. This phenomenon is likely a consequence of the heuristic nature of the frequency functions in Section 4, and moreover may indicate that human subjects take into account only a short history of their neighbors' actions when choosing their own present actions.

We also note that the results remain largely the same with

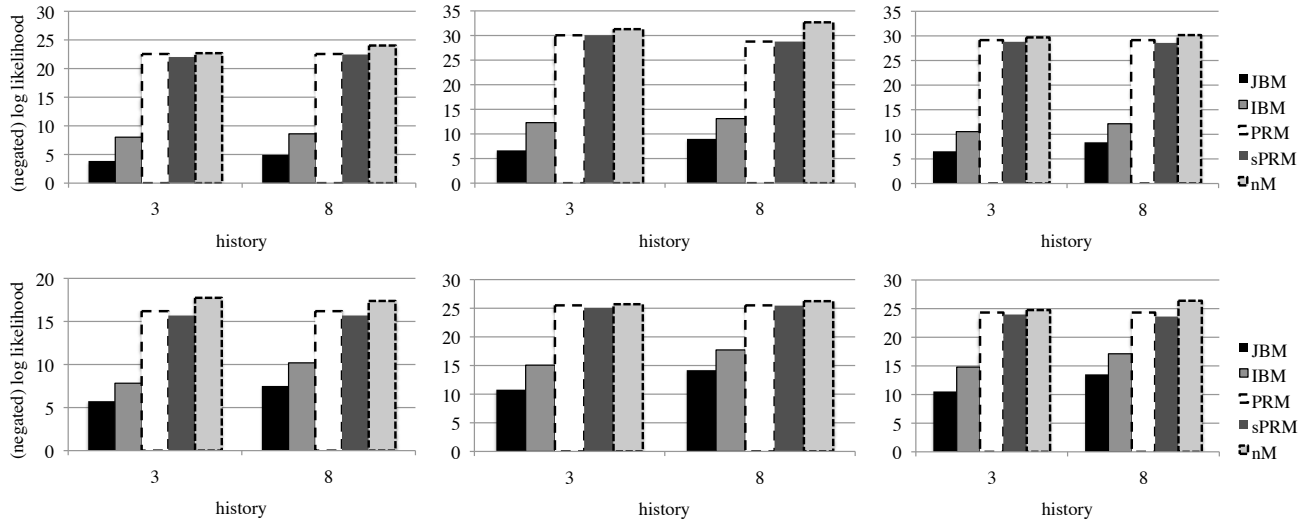


Figure 3: JBM provides better predictions than IBM, PRM, sPRM, and nM in three experiment sets with two different interval lengths $\delta = 0.5$ (top row) and $\delta = 1.5$ (bottom row): power22 (left), coER_2 (middle), and coPA_2 (right).

different discretization intervals δ. The large ratio between the two examined intervals implies a significant difference in the quantity and quality of discretized data, further highlighting the robustness of our learning method, as well as our empirical prediction results. These outcomes in general demonstrate JBM's ability to capture joint dynamic behavior, especially behavior correlations induced by limited historical information, as opposed to different IBMMs and a naive guessing model.

6.2 GRAPH ANALYSIS

Our first graph analysis focuses on the connections within and between nodes of different degrees in the learned graphs. We mainly focus on experiments with history length $h = 3$ and discretization interval $\delta = 0.5$. We partition the nodes into four classes, based on their degrees in the original experiment networks. Class 4 contains the most connected nodes, and class 1 the least. We narrow our focus on classes 1 and 4: in particular, we examine the average number of *inter edges*, which connect nodes within the same class, and *intra* edges, which link nodes in different classes. Note that intra and inter edges concern their end nodes' degree, while the characterization of intra-group and inter-group edges of experiment networks is based on the end nodes' vote preferences. Let us define E^-_M as the set of edges present in the original G, but excluded in the graph M, and E^+_M as the set of edges not present in G, but included in M.

We observe in Figure 4 that the distribution of edges within and between node classes of different connectivity in JBM diverges significantly from that in the original network for coER_2 and power22.[1] The degree constraint in the learning algorithm forces JBM to shed edges from class 4 nodes. In fact, the learned models end up with more intra edges in class 1 than class 4, whereas G has more intra edges in class 4 than in class 1.

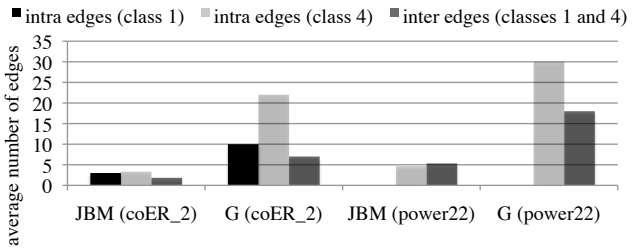


Figure 4: Distributions of edges within and between classes 1 and 4, for JBM and G in scenarios coER_2 and power22 ($h = 3$ and $\delta = 0.5$).

Our second analysis study examines different statistics and properties of E^+_{JBM}, E^-_{JBM}. In particular, we find that 40% to 60% of the edges in E^+_{JBM} connect nodes that share some neighbors in G, suggesting that JBM manages to identify nodes whose behavior correlates due to their observing common neighbors. Our investigation further focuses on *bridge* edges, which connect nodes that share no common neighbors in the original G. Let B denote the set of all bridges in G. Further, define $b^-_{\mathsf{M}} = \frac{|E^-_{\mathsf{M}} \cap B|}{|E^-_{\mathsf{M}}|}$ as the percentage of bridges among edges eliminated during the construction of model M, and $b_{\mathsf{G}} = \frac{|B|}{|G|}$ as the percentage of bridges in the original G. We observe that the percentage of eliminated bridges E^-_{JBM} is consistently below the percentage of bridges over all edges in G, as shown in Table 2, though only by small margins. This result may suggest the learning algorithm's slight preference in retaining bridges rather than edges of other types.

[1]We only include results for coER_2 and power22 in our graph analysis session, as coPA_2's outcomes appear similar to those of coER_2.

Table 2: Percentage of bridges in the set of eliminated edges by JBM and bridges in the set of all G's edges

	coER_2	power22
b^-_{JBM}	42%	30%
b_{G}	44%	31%

Table 3: JBM greatly diverges from G in terms of assortativity.

	coER_2	power22
JBM	-0.081	0.09
G	0.09	0.09

The last analysis concerns the learned graphs and original experiment networks' *assortativity* [Newman, 2003]. A graph G's assortativity coefficient $v_G \in [-1, 1]$ captures the tendency for nodes to attach to others that are similar or different in connectivity. Positive values of v_G indicate a correlation between nodes of similar degree, whereas negative values indicate relationships between nodes of different degrees.

The table shows that in scenario coER_2 JBM graphs exhibit negative assortativity, whereas the original network's v_{G} is positive. In other words, JBM graphs appear to contain sparsely connected hubs of highly connected nodes, which implies the learning method's ability to identify nodes of greater influence in the original networks. This observation is further supported by the significant difference in the proportion of intra edges among highly connected nodes between G and JBM in coER_2, as presented in Figure 4. Figure 4 indicates the same phenomenon for scenario power22, which, however, does not correspond to any divergence between G's and JBM's assortativity in the same scenario. We attribute this discrepancy to the existence of a strongly connected minority group in power22. Overall, the retention rate of G's edges within the minority group in the learned JBM (19.5%) is higher than that within class 4 (15.6%), which contains both minority and majority nodes. The structure learning algorithm appears to be able to identify the importance and influence of the minority group, a phenomenon also observed and reported by Kearns et al. [2009].

6.3 DISCUSSIONS OF CONSENSUS OUTCOMES

We also evaluate the models' capacity to predict the end state of a dynamic consensus experiment. As noted above, the original aim of modeling in these domains was to predict this final outcome. For a particular model M, we start an experiment run with agents choosing their preferred colors, and then proceed to draw samples from M for each time period until a consensus is reached or the number of time periods exceeds the time limit. We average over 500 runs for each experiment setting and model.

The convergence of PRM to consensus strongly correlates with observed experimental results in terms of the percentage of experiments reaching consensus, as shown in Figure 5. However, PRM's predictions on the color of consensus are out of line with the actual experiments for some experiment settings, rendering PRM ineffective in predicting end state results.[2]

We find that simulated runs drawn from JBM converge to a consensus at a relatively lower rates than PRM in the power22 setting where all lab experiments reach consensus, and at a significantly lower rates in other settings. As a result, JBM is not directly useful for predicting end state specifically. That the model's success in capturing transient dynamics fails to translate to outcome prediction is an interesting anomaly. It will be worth further investigating whether incorporating time dependence or other additional factors can remedy the discrepant effectiveness.

It is important to note that the input of our study has been transformed from ordinal-time to snapshot data [Cosley et al., 2010], which inherently results in a loss of information about the exact vote-update times. Moreover, the difference in the discretization interval δ does make a noticeable difference in the end-game results for both JBM and PRM, while the effect of δ in predicting transient dynamics is negligible. These observations suggest the need to experiment with constructing and evaluating asynchronous versions of this study's models.

We further observe that in settings coER_2 and coPA_2 where consensus appears less frequently, the vote majority tended to switch from one color to another before assuming its original color. As we suspect that the presence of "stubborn" players fueled this pattern, JBM could benefit from having an additional history summary function other than f that captures the level of "stubbornness" of an agent, gains greater importance as time passes, and thus effectively informs the potential function about future vote updates. We will also further incorporate in our models elements from different statistical models specifically developed for capturing agents' abilities to reach consensus on distributed networks [Masuda et al., 2010, Mossel and Schoenebeck, 2010]

7. CONCLUSIONS

We have introduced and empirically evaluated an algorithm for learning history-dependent graphical multiagent models of dynamic behavior given time series of agent actions. The empirical study demonstrates an ability to learn compact graphical representations capturing the dynamics of human-subject voting behavior on a network. In particular, we have shown that the learned joint behavior model JBM provides significantly better predictions of dynamic behavior than different individual behavior models, including the multiplicative model PRM and its variation sPRM, suggested by the original experimental analysis. This provides evidence that expressing joint behavior is important for dynamic modeling, even given partial history information for conditioning individual behavior. Our graph analysis further reveals characteristics of the learned JBM graphical structures, particu-

[2]Kearns et al. [2009] only examined the correlations between the number of seconds for human subjects to reach consensus during lab experiments and the number of simulation updates by a PRM-type model before reaching consensus, but not the color of consensus.

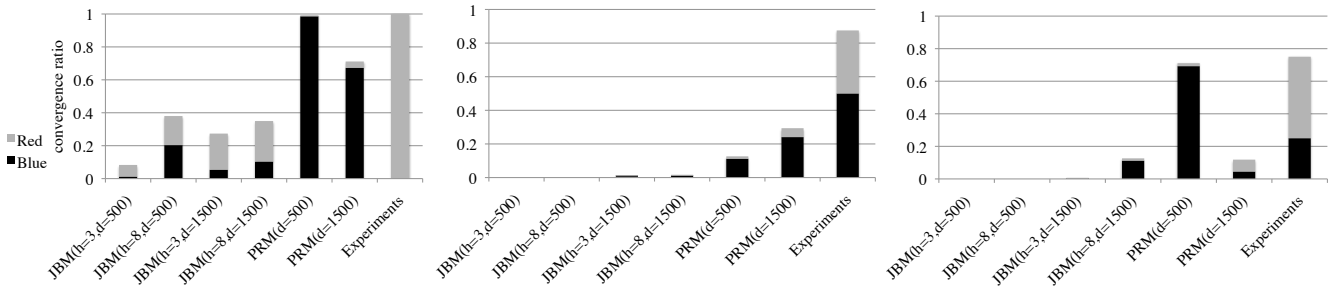


Figure 5: End-game results for three different experiment groups: power22 (left), coER_2 (middle), and coPA_2 (right).

larly in how they diverge from the original experiment networks.

In future work we plan to improve the learning algorithm for individual behavior models, which are currently constructed with a predetermined maximum degree constraint, by replacing this degree constraint with a cross-validation condition that can better help avoid overfitting. Another potential research study would examine and compare different approaches to learning hGMM structure in terms of efficacy and complexity. We would also like to build on this study's graph analysis a more systematic and comprehensive toolbox for dissecting graphical models of dynamic behavior and assessing model learning algorithms' contributions. Finally, we are interested in applying our modeling technique in studying similar problem domains where agents must coordinate their actions or make collective decisions while only communicating with their neighbors [Suri and Watts, 2011, Judd et al., 2010], as well as large network scenarios, such as social networks and internet protocols.

References

Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

Jonathan Yedidia Broadway, Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. Generalized belief propagation. In *Thirteenth Annual Conference on Advances in Neural Information Processing Systems*, pages 689–695, Denver, 2000.

Dan Cosley, Daniel Huttenlocher, Jon Kleinberg, Xiangyang Lan, and Siddharth Suri. Sequential influence models in social networks. In *Fourth International AAAI Conference on Weblogs and Social Media*, pages 26–33, Washington, DC, 2010.

Constantinos Daskalakis and Christos H. Papadimitriou. Computing pure Nash equilibria in graphical games via Markov random fields. In *Seventh ACM conference on Electronic Commerce*, pages 91–99, Ann Arbor, MI, 2006.

Quang Duong, Michael P. Wellman, and Satinder Singh. Knowledge combination in graphical multiagent models. In *Twenty-Fourth Conference on Uncertainty in Artificial Intelligence*, pages 153–160, Helsinki, 2008.

Quang Duong, Michael P. Wellman, Satinder Singh, and Yevgeniy Vorobeychik. History-dependent graphical multiagent models. In *Ninth International Conference on Autonomous Agents and Multiagent Systems*, Toronto, 2010.

P. Erdos and A. Renyi. On random graphs. i. *Publicationes Mathematicae*, 6(290-297):156, 1959.

Ya'akov Gal and Avi Pfeffer. Networks of influence diagrams: A formalism for representing agents' beliefs and decision-making processes. *Journal of Artificial Intelligence Research*, 33:109–147, 2008.

Albert Xin Jiang, Kevin Leyton-Brown, and Navin A. R. Bhat. Action-graph games. Technical Report UBC CS TR-2008-13, University of British Columbia, 2008.

Stephen Judd, Michael Kearns, and Yevegeiny Vorobeychik. Behavioral dynamics and influence in networked coloring and consensus. *Proceedings of the National Academy of Sciences*, 107(34):14978, 2010.

Sham Kakade, Michael Kearns, John Langford, and Luis Ortiz. Correlated equilibria in graphical games. In *Fourth ACM Conference on Electronic Commerce*, pages 42–47, San Jose, CA, 2003.

Michael Kearns and Jinsong Tan. Biased voting and the Democratic primary problem. In *Fourth International Workshop on Internet and Network Economics*, pages 639–652, Shanghai, 2008.

Michael Kearns, Michael L. Littman, and Satinder Singh. Graphical models for game theory. In *Seventeenth Conference on Uncertainty in Artificial Intelligence*, pages 253–260, Seattle, 2001.

Michael Kearns, Stephen Judd, Jinsong Tan, and Jennifer Wortman. Behavioral experiments on biased voting in networks. *Proceedings of the National Academy of Sciences*, 106(5):1347–52, 2009.

Daphne Koller and Brian Milch. Multi-agent influence diagrams for representing and solving games. *Games and Economic Behavior*, 45:181–221, 2003.

N. Masuda, N. Gibert, and S. Redner. Heterogeneous voter models. *Physical Review E*, 82(1):100–103, 2010.

Joris M. Mooij. libDAI: A free and open source C++ library for discrete approximate inference in graphical models. *Journal of Machine Learning Research*, 11:2169–2173, 2010.

Elchanan Mossel and Grant Schoenebeck. Reaching consensus on social networks. In *First Symposium on Innovations in Computer Science*, Beijing, China, 2010.

M. E. J. Newman. Mixing patterns in networks. *Physical Review E*, 67(2), 2003.

Siddharth Suri and Duncan J. Watts. Cooperation and contagion in web-based, networked public goods experiments. *PloS One*, 6(3):e16836, 2011.

Modeling Information Diffusion in Networks with Unobserved Links

Quang Duong Michael P. Wellman Satinder Singh
Computer Science and Engineering, University of Michigan
{qduong,wellman,baveja}@umich.edu

ABSTRACT

Modeling information diffusion in networks enables reasoning about the spread of ideas, news, influence, and diseases across a network of agents. Existing models generally assume a given network structure, in practice derived from observations of agent communication or other interactions. In many realistic settings, however, observing all connections is not feasible. We consider the problem of modeling information diffusion when the network is only partially observed, and investigate two approaches. The first learns a graphical model with the given network structure, which compensates for missing edges by capturing induced correlations among node states. The second attempts to learn the missing connections directly. Using data generated from a cascade model with different network structures, we empirically demonstrate that both methods improve over assuming the given network is fully observed, as well as a previously proposed structure-learning technique. We further find that the graphical model outperforms structure learning when given sufficient data.

1. MODELING INFORMATION DIFFUSION

Information diffusion can be seen at work in a wide range of scenarios, such as news propagation, political grass-root campaigning, product promotion, and technology adoption. Models of information diffusion on social networks, for instance, can inform viral marketing campaigns by offering predictions and analyses of product popularity and advertising effectiveness, given information about early buyers' behaviors [Leskovec et al., 2007]. They can also be employed to assess more abstract properties, such as the likelihood of information spreading from one node to another node separated by a given distance [Song et al., 2007, Choudhury et al., 2008, Bakshy et al., 2009].

Applying such models requires knowing the structure of the network, typically inferred based on evidence of affiliation, communication, or other observable traits of agent relationships. In practice, however, such evidence is generally incomplete and uncertain. Sophisticated learning methods may improve detection in such circumstances [Adar and Adamic, 2005, De Choudhury et al., 2010, Backstrom and Leskovec, 2011], but inevitably, real-world agents will have connections that are unobserved by specific third parties. For example, Internet social networks do not capture offline human interactions. Thus, any model of information diffusion must account for propagation of information along paths not included in the observed network.

In the now standard approach to modeling information diffusion on networks [Kleinberg, 2007], a node or agent is in one of a finite set of states (e.g., having particular information, adopting a technology, etc.) at a given time. In each discrete time period, each agent decides what state to adopt in the next period, as a probabilistic function of the states of its neighbors. Formally, let G be a network of n agents over a discrete time horizon T. We are interested in capturing the diffusion of a single bit of information, so at time t, each agent i can be in either of two states in the set $\mathcal{S}_i$: $s_i^t = 1$ indicates that agent i has been *infected*, and $s_i^t = -1$ otherwise. An undirected edge (i, j) in G represents a connection between agents i and j: conceptually, that i and j interact and may be aware of each other's state. For each node i, N_i denotes the neighborhood of i: $N_i = \{j \mid (i, j) \in G\} \cup \{i\}$. Let $s^t = s^t_{\{1,\ldots,n\}}$ be the state of all agents at time t.

In the version of information diffusion we study, infection *persists*: there exists no agent i and time t such that $s_i^t = 1$ and $s_i^{t+1} = -1$. We define a binary feature $a(s_i^t, s_i^{t-1}) = 1$ to indicate that i becomes infected at time t ($s_i^{t-1} = -1$ and $s_i^t = 1$), otherwise $a(s_i^t, s_i^{t-1}) = -1$. Let us define t_i^* as the time when i becomes infected, and $c(s^t_{N_i}) = |\{j \in N_i \mid s_j^t = 1\}|$, the count of agents in N_i who are infected at t. Similarly, let $\sigma(s^t_{N_i}, s^{t-1}_{N_i}) = |\{j \in N_i \mid a(s_j^t, s_j^{t-1}) = 1\}|$, the number of agents in N_i that *become* infected at time t.

The most popular model of infection is the *cascade model* [Kempe et al., 2003], in which the tendency to infect increases with the proportion of infected neighbors. Goldenberg et al. [2001] proposed a cascade model form:

$$\Pr(s_i^t = 1 \mid s^{t-1}_{N_i}) = 1 - (1 - \alpha)(1 - \beta)^{c(s^{t-1}_{N_i})}. \qquad (1)$$

In this model, $\beta \in [0, 1]$ represents the tendency of infection from interacting with one of its infected neighbors, and $\alpha \in [0, 1]$ reflects the possibility of node i getting information from sources other than its neighbors, or in other words

The 5th SNA-KDD Workshop '11 (SNA-KDD'11), August 21, 2011, San Diego CA USA . Copyright 2011 ACM 978-1-4503-0225-8...$5.00.

spontaneously becoming infected.[1]

Stonedahl et al. [2010] introduced an alternative cascade model, which we label **C**, that induces similar behavior based on the same intuitions, but with a simplified probability expression:

$$\Pr(s_i^t = 1 \mid s_{N_i}^{t-1}) = \alpha + \beta \frac{c(s_{N_i}^{t-1})}{|N_i|}. \quad (2)$$

Assuming independence of agent states given past states, the cascade models define a joint distribution of states at time t:

$$\Pr(s^t \mid s^{t-1}) = \prod_i \Pr(s_i^t \mid s_{N_i}^{t-1}). \quad (3)$$

Although the cascade models allow for positive probability of infection even if no known neighbors are infected, its accuracy suffers as the network structure is missing connections. Gomez-Rodriguez et al. [2010] were first to identify and tackle the problem of discovering underlying network structure given only diffusion history. They introduce an algorithm called `NetInf`, which identifies a network that optimizes an approximate measure of fit to observed infection times. In the present work, we also consider the structure-learning approach: a greedy algorithm of our own, named `MaxLInf`, as well as a version `NetInf'` of the previous algorithm modified to fit our problem setup. We further introduce a fundamentally different approach based on probabilistic graphical models, which can compensate for missing edges by capturing induced correlations of behavior among unconnected nodes.

2. PROBLEM SETUP

Let G be the observed network, and $S = \{s\}$ a set of diffusion instances or traces, each of length T, $s = \{s^0, \ldots, s^T\}$. We assume that the diffusion was generated by some process (in our empirical study, the cascade model) on a true underlying network $G^* \supseteq G$. Given G and S, we would like to make inferences about future diffusion events. We next provide an example of the problem and formally define our objectives.

2.1 Example

Consider the example four-node scenario shown in Figure 1, where the edge between nodes 1 and 3 is not observed. Suppose that in the true underlying model nodes can become infected only from interacting with their neighbors. In the example run, the spread started with node 1 at time $t = 1$ and reached nodes 4 and 3 at $t = 5$ and $t = 6$, respectively. The missing edge $(1, 3)$ may mislead us to interpret node 3's infection as spontaneous. Throughout Section 3, we provide more details about how each approach in this study compensates for the missing connection $(1, 3)$ in this particular example.

[1]The cascade model implies that influence is evenly distributed among a node's neighbors and all network nodes will eventually become infected. We adopt this model not because we necessarily desire these characteristics, but because the model is widely applied in the literature. Nothing about our approach is particularly geared to these properties.

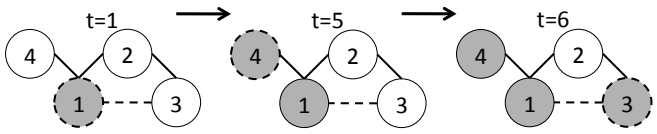


Figure 1: A four-node scenario with one missing edge (dashed). Infected nodes are shaded. Newly infected nodes at each time period are marked with dashed rims.

2.2 Evaluation

We capture the dynamics of information diffusion using the joint probability distribution of agent states, conditional on past states. Notationally, $\Pr^M(s^t \mid s^{t-1})$ is the joint distribution of agent states at time t induced by a particular model M. Given a set S of m traces, we can compute the data's average logarithmic likelihood induced by M,

$$L^M(S) = \frac{1}{m} \sum_{s \in S} \frac{1}{T} \sum_t \log \Pr{}^M(s^t \mid s^{t-1}). \quad (4)$$

We employ the above log-likelihood function for training models in the methods studied here, except for `NetInf'`. To learn model parameters, we search for settings that maximize the objective function (4) using gradient descent with early stopping and random restarts.

As in many information network studies, we are also interested in predicting the aggregate extent of future network infection, given some initial diffusion data. In particular, given a diffusion instance s, we sample diffusion traces of length $\hat{T}$ from the model M starting with the network state after the first period in s. These traces define a distribution over the fraction $c(s^{\hat{T}})/n$ of nodes that are infected at time $\hat{T}$, denoted as $Q^M(c(s^{\hat{T}})/n \mid s)$. We employ a version of the Kullback-Leibler (KL) divergence metric, the skew divergence [Lee, 1999], to measure the distance between the two discrete value distributions Q induced by the true model M^* and by the test model M, conditional on observations of the first round in s. We first specify the KL divergence as

$$d_{KL}(Q^{M^*}, Q^M \mid s) = \sum_{c=0}^{n} Q^{M^*}(c/n \mid s) \log \frac{Q^{M^*}(c/n \mid s)}{Q^M(c/n \mid s)},$$

and subsequently the average skew divergence given data S:

$$D(Q^{M^*}, Q^M \mid S) = \frac{1}{m} \sum_s d_{KL}(Q^M, \rho Q^{M^*} + (1-\rho) Q^M \mid s),$$

with ρ set at 0.99 to avoid the problem of undefined values in the KL divergence metric [Lee, 1999].

While the two measures above are designed to address the main problem of information diffusion, we can further measure the quality of the graphs learned by `MaxLInf` and `NetInf'` in simulated scenarios where access to the true underlying graph is available. In particular, we employ the following metrics on structural differences between the true underlying

graph $G^* = (V, E^*)$ and the learned structure $G' = (V, E')$:[2]

$$\delta_-(G^*, G') = \frac{|E^* \setminus E'|}{|E^*|}, \text{ and } \delta_+(G^*, G') = \frac{|E' \setminus E^*|}{|E'|}.$$

Here $\delta_-(G^*, G')$ represents the proportion of false negative edges in the true network G^*, or in other words, edges that remain missing in the learned graph G'. Similarly, $\delta_+(G^*, G')$ captures the proportion of false positive edges in the learned structure G'.

3. DEALING WITH PARTIALLY OBSERVED NETWORKS

We consider two broad approaches to learning diffusion models despite unobserved network edges. The first employs graphical models to capture probabilistic dependencies induced by the missing edges. The second attempts to recover the edges explicitly by learning network structure.

3.1 Graphical Multiagent Models

A *graphical multiagent model* (GMM) is a graphical model with graph elements: V, vertices representing the n agents, and E, edges capturing pairwise interactions between them [Duong et al., 2008]. We employ here a dynamic extension, the *history-dependent* graphical multiagent model (hGMM) [Duong et al., 2010], which captures agent interactions in dynamic scenarios by conditioning joint states on abstracted history. We further limit conditioning to the previous state, associating with each agent i a potential function $\pi_i(s^t_{N_i} \mid s^{t-1}_{N_i})$: $\prod_{j \in N_i} \mathcal{S}_j \to \mathbb{R}^+$. The potential of a local state configuration $s^t_{N_i}$ specifies its likelihood of being included in the global outcome, conditional on past state [Kakade et al., 2003, Daskalakis and Papadimitriou, 2006, Duong et al., 2010]:

$$\Pr(s^t \mid s^{t-1}) = \frac{\prod_i \pi_i(s^t_{N_i} \mid s^{t-1}_{N_i})}{Z}. \quad (5)$$

Unlike the cascade models (3), hGMMs do not assume conditional independence of agent states given history, but specify the joint behavior directly. That hGMMs compute the potentials of all state configurations of a neighborhood allows reasoning about behavior correlations between neighbors who appear disconnected in the input graphical structure, and thus helps to compensate for missing edges. We approximate the normalization factor Z using the *belief propagation* method [Yedidia et al., 2000], which has shown good results with reasonable time in sparse cyclic graphs. We in particular employ the libDAI approximate inference package [Mooij, 2010].

Figure 2 illustrates an example four-agent hGMM of the scenario described in Section 2.1. Note that the aforementioned potential function can be written in a more generalized form as $\pi_i(s^t_{N^u_i} \mid s^{t-1}_{N^d_i})$, which distinguishes two edge types in an hGMM, as depicted in Figure 2. Node i's undirected edges define a neighborhood N^u_i within a time slice, representing a configuration of these nodes states at a particular time t, $s^t_{N^u_i}$. Directed edges ending at i capture how past states of the neighborhood N^d_i influence i's present state. In contrast, a cascade model of the same scenario would omit the undirected edges, expressing the cascade functional form $\Pr(s^t_i \mid s^{t-1}_{N_i})$. For most of this paper, we assume that N^u_i, N^d_i, and the given N_i are the same for all i, and employ the potential function form $\pi_i(s^t_{N_i} \mid s^{t-1}_{N_i})$.

[2]A similar version of these metrics was also employed in evaluating the original `NetInf` [Gomez-Rodriguez et al., 2010].

Given data from the example scenario of Section 2.1, the cascade models would learn a high value for α, assuming spontaneous infections, even though in the true model, information spreads only from node to node. In contrast, hGMMs could use the potential function of node 2 to express correlations between nodes 1 and 3 to compensate for the missing edge $(1, 3)$. For instance, consider neighborhood $N_2 = (1, 2, 3)$ with history $s^5_{N_2} = (s^5_1, s^5_2, s^5_3) = (1, -1, -1)$, as depicted in Figure 1. Since the potentials $\pi(s^6_{N_2} = (1, 1, 1) \mid s^5_{N_2})$ and $\pi(s^6_{N_2} = (1, -1, 1) \mid s^5_{N_2})$ govern how node 1 infects node 3, assigning positive values to these potentials enables hGMMs to capture the interaction. Unlike $\pi(s^6_{N_2} \mid s^5_{N_2})$, the corresponding cascade-model construct $\Pr(s^6_2 \mid s^5_{N_2})$ cannot compensate in this way because it inherently assumes independence among nodes in N_2.

We introduce hGMMs for information diffusion that represent the network's joint probability distribution as a product of potentials (5) in two different forms.

3.1.1 Tabular hGMMs

The *tabular hGMM* **tabG** explicitly specifies the potential function of each neighborhood, $\pi(s^t_{N_i} \mid s^{t-1}_{N_i})$, as a function of five features:

$$c(s^{t-1}_{N_i}),\ \sigma(s^t_{N_i},\ s^{t-1}_{N_i}),\ |N_i|,\ s^{t-1}_i,\ s^t_i.$$

For learning such forms, we treat the potential value corresponding to each configuration of these features above as a parameter. The number of parameters thus grows polynomially with the largest neighborhood's size.

3.1.2 Parametric hGMMs

The *parametric hGMM* **paG** employs a functional form for potentials based on the cascade model formula (2). We define $g(s_y = 1 \mid s_Y)$ as a probability measure of any uninfected node y in the set of nodes Y becoming infected given the current state s_Y: $g(s_y = 1 \mid s_Y) = \alpha + \beta c(s_Y)/|Y|$. We overload $g(s_Y) = g(s_y = 1 \mid s_Y)$, as g is identical for all y. Let $N'_i = N_i \setminus \{i\}$ and $c = \sigma(s^t_{N'_i}, s^{t-1}_{N'_i})$. The potential function of neighborhood N_i is the product of three terms:

$$\pi(s^t_{N_i} \mid s^{t-1}_{N_i}) = \Pr(s^t_i \mid s^{t-1}_{N_i}; \alpha, \beta) g(s^{t-1}_{N'_i}; \alpha_1, \beta_1)^{|c - \gamma |N_i||} \left(1 - g(s^{t-1}_{N'_i}; \alpha_{-1}, \beta_{-1})\right)^{|N'_i| - c} \quad (6)$$

The first term of (6) represents the probability of node i's infection, given by the cascade model (2). The second term corresponds to nodes in N'_i that become infected at t. If we assumed independence of agent states, the exponent would simply be c, as the joint probability of c nodes from Y becoming infected is $g(s_Y)^c$. From our empirical study on **tabG**, we find that the distance of this count from a fraction of neighborhood size provides a good way to capture correlation among neighbor infections, hence the exponent $|c - \gamma |N_i||$ with $\gamma \in [0, \infty)$. Here γ indicates the degree of

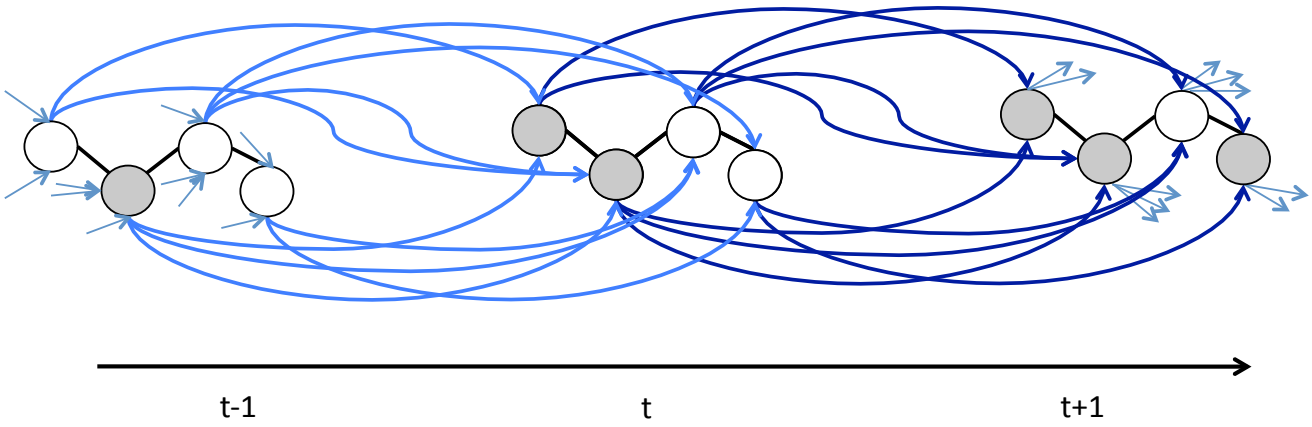


Figure 2: A history-dependent graphical multiagent model of four nodes. Undirected edges capture correlations among nodes of the same time point. Directed edges capture the conditioning of each node's state on its neighborhood's previous states: dark arrows connect nodes from t to $t+1$, and light arrows link nodes from $t-1$ to t.

correlation among agent states. Thus, complete independence of agent states results in $\gamma = 0$. The final term of (6) corresponds to nodes in N_i' not infected at t. Overall, the **paG** model employs seven parameters: α, β, α_1, β_1, α_{-1}, β_{-1}, and γ.

Our graphical multiagent model approach does not seek to discover missing edges, but takes advantage of the GMM's flexibility in modeling information diffusion on the original network.

3.2 Learning Graphical Structures

In the structure-learning approach, we attempt to discover unobserved connections using diffusion observation data. We consider two algorithms: one adapted from prior work under a different model, and the second a straightforward greedy learner introduced here.

3.2.1 `NetInf` and `NetInf′` Algorithms

Gomez-Rodriguez et al. [2010] employed the independent cascade model to capture information diffusion, and further assumed that every node becomes infected from exactly one other infected neighbor node. Given a graph structure G, let $\mathcal{T}$ be the set of all possible directed trees τ whose edges E_τ are a subset of E_G. A model M over G computes the likelihood of observing the transmissions in a diffusion instance s on each tree $\tau \in \mathcal{T}$ as follows. We first describe the computation of $\Pr(i; s, \tau)$: the probability of observing node i becoming infected in s on tree τ. For each node i such that there (uniquely) exists an edge $(j, i) \in \tau$, $\Pr(i; s, \tau)$ is defined as the probability of transmission from node j, infected at time t_j^*, to node i at time t_i^*, which is a function of $t_i^* - t_j^*$. `NetInf` employs the power-law and exponential waiting time models in specifying $\Pr(i; s, \tau)$, which would put it at a disadvantage when given data from this study's generative cascade model. Therefore, the modified version `NetInf′` replaces their definition with $\Pr(i; s, \tau) = \beta(1-\beta)^{(t_i^* - t_j^* - 1)}$ if $t_j^* < t_i^*$, and $\Pr(i; s, \tau) = 0$ otherwise. For each i that does not have an incoming edge in τ, the probability of observing node i's spontaneous infection is $\Pr(i; s, \tau) = \alpha$. Model M then interprets the probability of observing s on tree τ: $\Pr(s; \tau) = \prod_i \Pr(i; s, \tau)$. From this the algorithm approximates the probability of observing s on graph G as that of observing s on the maximum-likelihood tree τ_s: $\Pr(s \mid G) \approx \max_{\tau \in \mathcal{T}} \Pr(s; \tau)$.

Starting with some initial graph G, at each step, the `NetInf′` algorithm, as well as `NetInf`, adds an edge (i, j) that maximizes the log likelihood of data S: $L_{tree}(S \mid G \cup (i,j)) = \sum_{s \in S} \log(\Pr(s \mid G \cup (i,j))$. As L_{tree} is monotone in graph size, this method would produce the complete graph without a limit K on the number of edges that can be added. Note that both α and β here have the same meanings as in the cascade model (1). Since `NetInf′` does not learn parameters α and β, we provide it with the parameter values used in the generative cascade model of input data. We refer to the model learned by `NetInf′` as **netC**.

3.2.2 `MaxLInf` Algorithm

We introduce a similar greedy algorithm `MaxLInf`, described in pseudocode below. The result **maxC** is a cascade model, defined by (2). Unlike `NetInf′`, the `MaxLInf` algorithm learns the cascade model parameters α and β as well as the graphical structure G'. We employ the average log likelihood of diffusion observations in the input set S, defined by (4), as the algorithm's objective function.

Note that `MaxLInf` does not require any predetermined constraint on the model's complexity, since adding more edges does not necessarily increase the likelihood L of input diffusion observations.

3.2.3 Discussion

In the example of Section 2.1, these two learning algorithms may be able to discover the hidden edge $(1, 3)$, given sufficient observations. Since they are greedy, however, they may

Algorithm 1 `MaxLInf`

Input (G, S)
$G' \leftarrow G$
Learn (α, β) that maximizes $L^{\mathbf{maxC}}(S \mid G')$
repeat
  Find $(i,j) \notin G'$ maximizing $L^{\mathbf{maxC}}(S \mid G' \cup (i,j))$
  if $\Delta(i,j) = L^{\mathbf{maxC}}(S \mid G' \cup (i,j)) - L^{\mathbf{maxC}}(S \mid G') > 0$ **then**
    $G' \leftarrow G' \cup (i,j)$
    Learn (α, β) that maximizes $L^{\mathbf{maxC}}(S \mid G')$
  end if
until $\Delta(i,j) > 0$ or $\neg\exists (i,j) \notin G'$

also add spurious edges that distort their views of the true network. For example, given some limited data set in which the case where node 4 was infected before node 3 occurs disproportionally frequently, they may greedily add $(3,4)$ first to best explain the data, and may stop before adding $(1,3)$.

4. EMPIRICAL STUDY

Our empirical study employs graphs of modest size to focus on investigating the relative ability of structure learning and graphical models to deal with unobserved connections. We empirically evaluate models **paG**, **tabG**, and **maxC** against baseline models **C** and **netC** on simulated data generated from the true cascade $\mathbf{C}^*$, with various graphical structures and experiment settings. We examine their detailed prediction performances based on the log likelihood L, aggregate prediction performance captured in the skew divergence D, and graph difference measures δ_- and δ_+.

4.1 Data Generation

Our empirical study consists of two experiment groups: (A) stylized graphs of three to five nodes, and (B) larger graphs of size 30 and 100. In experiment group B, we employ the Erdos-Renyi model (ER) and the Barabasi-Albert preferential attachment model (PA) to generate random graphs G^* [Erdos and Renyi, 1959, Barabási and Albert, 1999]. The ER model randomly creates an edge between each pair of nodes with equal probability, independently of the other edges, while the PA model constructs graphs such that the more connected nodes are more likely to receive new connections. We generate data from the $\mathbf{C}^*$ model on the underlying graphs G^* with time horizon $T = 30$, using the α and β values that Stonedahl et al. [2010] learned from various real-world social networks at two different diffusion speeds: normal and fast (viral).

We uniformly randomly remove 50% of the edges from each G^* to generate the observable graph G, supplied as input to all the methods tested. Whereas Gomez-Rodriguez et al. [2010] developed and evaluated their original `NetInf` algorithm without graph input, we seed the modified version `NetInf'` with the partial structure G.

4.2 Experiment Settings

Each result data point is averaged over 10 trials. The group A trials use 200 diffusion instances for training and 500 for testing. In experiment group B, we vary the number of diffusion traces for training from 25 to 100, and use 200 instances for testing in each trial. For clarity, we present the negated log likelihood measure $-L^M(S)$ for a model M; lower values indicate better performance. To assess aggregate infection statistics for model M, we generate 200 diffusion traces of length $\hat{T} = 5$ from the initial state of each given test instance s. From these we construct the conditional distribution $Q^M(c(s^{\hat{T}})/n \mid s)$. Given the distributions Q^M, we calculate the aggregate prediction performance for model M, $D(Q^{\mathbf{C}^*}, Q^M \mid S)$. To establish statistical comparisons of two models M_1 and M_2, we further perform bootstrapped paired t-tests for both measures L and D. We distinguish three different outcomes in comparison tables: M_1 outperforms M_2 with $p < 0.05$ (black), M_2 outperforms M_1 with $p < 0.05$ (white), and neither (gray). We investigate two variants of `NetInf'`, set to induce graphs with 50% (model **netC**-small) and 100% (model **netC**-large) more edges than the given graph G. In addition to trials with partially observed networks (partial), we include some fully observed cases (full) for calibration and sanity checking.

4.3 Results

We present results on detailed prediction performance, measured by L, for the 5-node case in Figure 3. We used the insights gained from analyzing the potential function parameters for **tabG** in the three- and four-node cases to construct **paG**, but do not report results from these cases here.

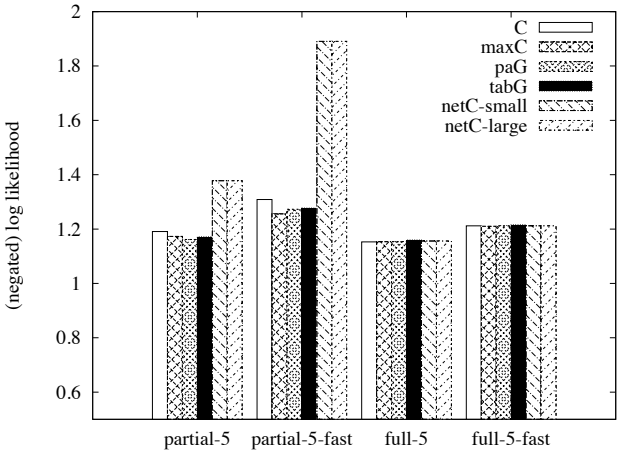


Figure 3: Detailed prediction performance for the 5-node graph (normal and fast spreading speeds).

We exclude results for D, δ_-, and δ_+, which provide little differentiation among the examined models due to the small network size. Models **maxC**, **tabG**, and **paG** provide better detailed prediction performance than the baseline models **C** and **netC**, when the input graph has missing edges. When the input graph is fully observed, **C** is the correct model, and as expected provides the best predictions. In addition, the hGMMs outperform **maxC** in the normal-spreading case but trail **maxC** in the fast-spreading scenario. In fact, we observe that **maxC** contains more correct edges (false negative $\delta_-(G^*, G') = 0.1$) in the fast spreading case than in the normal case ($\delta_-(G^*, G') = 0.25$). As there are more spontaneous infections and fewer inter-node transmissions in the normal case, `MaxLInf` may too quickly add edges that help to explain spontaneous infections even though these edges are not present in the underlying graph.

The statistical test results in Table 1 further cement confidence in the model comparisons above.

	partial-5	partial-5 -fast	full-5	full-5-fast
maxC vs **C**				
tabG vs **C**				
paG vs **C**				
netC-small vs **C**				
netC-large vs **C**				
paG vs **maxC**				

Table 1: Pairwise comparisons of detailed predictions for the 5-node graph. For M_1 vs M_2 black indicates M_1 outperforms M_2 with $p < 0.05$, white the reverse, and gray no significant difference found.

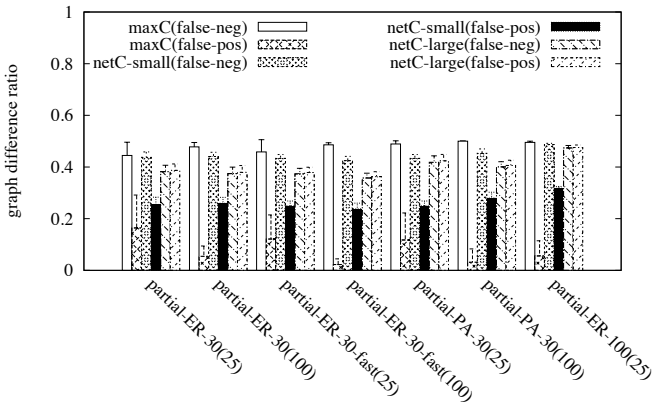


Figure 5: Graph difference measures in experiment group B for 30-node and 100-node graphs with different amounts of training data.

The results for larger graphs of 30 and 100 nodes in Figure 4 and Table 2 confirm that our methods in general provide better predictions than **C** when there are missing edges. Since **paG** is more complex than the cascade models, **paG**'s performance improves as the amount of training data increases. With sufficient data, **paG** emerges as the best performer, exceeding **C** and notably **maxC** in both ER and PA graphs of 30 and 100 nodes. Similarly, in fast diffusion cases where the number of time episodes in each spread is on average smaller, **paG**'s performance trails that of **maxC**. As for experiment group A, we verified in group B that **C** is the best model when the underlying graph is fully observed.

maxC outperforms the baseline **C** for ER graphs, but trails **C** for PA graphs. We speculate that missing edges in PA graphs may be harder to learn, particularly for nodes with few connections. Figure 5 suggests that as the amount of training data increases, `MaxLInf` becomes more conservative in adding edges, reflected by its relatively constant false negative measure δ_- and decreasing false positive δ_+.

We do not show most of the results for `NetInf'`, as the models produced by this algorithm perform relatively poorly by our measures, despite the advantage of being given the true cascade model's parameter values. This discrepancy is likely a consequence of `NetInf'`'s use of an objective function L_{tree} at variance from our use of $L^M(S)$ for evaluation. The respective objective functions are applicable for two different types of network influence data: ordinal-time and snapshot [Cosley et al., 2010]. The objective L_{tree} is suitable for ordinal-time data sets, which record the time of each individual node's change of state. The majority of data sets, however, store snapshots of the network state at various time periods, which align more directly with the objective $L^M(S)$. Future work could investigate the connection between these two objective functions, based on recent work exploring the correspondence between ordinal time and snapshot data [Cosley et al., 2010].

Even for ordinal data, L_{tree} is only an approximate measure of the likelihood of observed infection times, and thus does not exactly capture how information diffuses through connections. `NetInf'` also requires a predetermined constraint on the final graph's complexity. With respect to graph difference metrics, `NetInf'` is able to discover more missing edges than `MaxLInf`, resulting in a lower proportion of false negatives δ_- as depicted in Figure 4. However, `NetInf'` also adds significantly more spurious edges than `MaxLInf`, as reflected by δ_+. This problem worsens as `NetInf'` has more freedom to add new edges: **netC**-large contains many more spurious edges than **netC**-small, although **netC**-large is basically informed with the exact number of unobserved connections. Moreover, whereas **maxC** can compensate for the learned graph's inaccuracy by fitting its parameters to data, **netC** has to use the fixed parameters from the generative cascade model, since the objective function L_{tree} is maximized when $\alpha = \infty$ and $\beta = 1$.

Learned parameters of **paG** can provide additional insights about the underlying network. In particular, analyzing **paG**'s β_1 and β_{-1} may help to detect if the given network has unobserved edges, a functionality that cascade models do not provide. Intuitively, given a fully observed network, **paG** should use β solely in capturing information transmissions facilitated by the given network, and set β_1 and β_{-1} to near zero. When there exist missing edges, **paG** may assign higher values to β_1 and β_{-1} to capture how information diffuses among its seemingly unconnected neighbors. Let us define $b_1 = \beta_1/\beta$ and $b_{-1} = \beta_{-1}/\beta$. Table 3 shows that both b_1 and b_{-1} are notably higher in partially observed graphs than in fully observed structures, confirming this intuition. However, a more thorough examination using a wide range of measures is needed in order to draw a sufficiently confident connection between learned **paG** parameters and network structure.

5. DISCUSSION

We introduce two solutions to the problem of modeling information diffusion in networks with unobserved edges: learning an hGMM on the given network structure, and directly discovering the missing connections. We show that our approaches can improve prediction over existing methods in various settings of information diffusion with a considerable number of missing edges. With limited data, our structure-

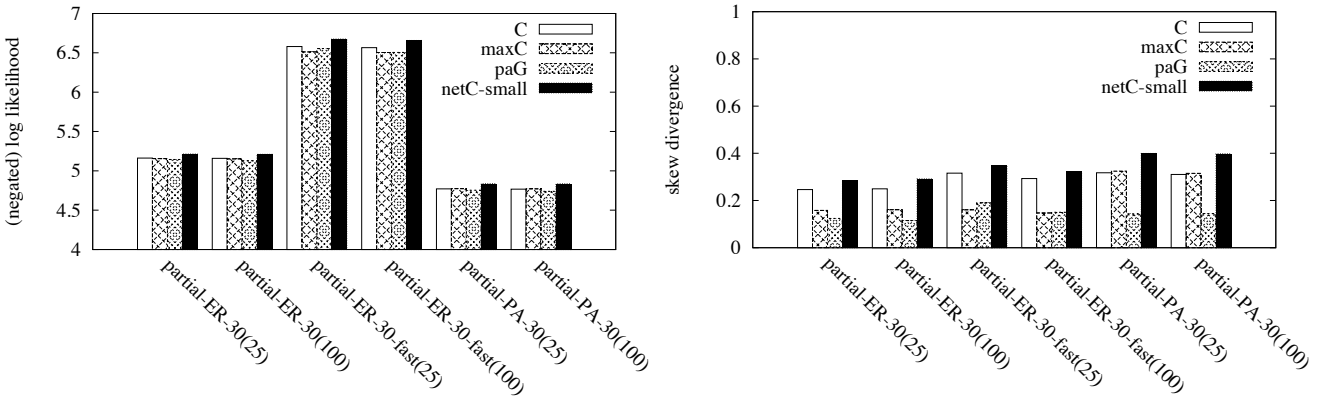


Figure 4: Detailed predictions (left) and aggregate predictions (right) in experiment group B for ER and PA 30-node graphs with different amounts of training data (numbers in parenthesis).

Detailed Predictions	**maxC vs C**	**paG vs C**	**paG vs maxC**
partial-ER-30(25)			
partial-ER-30(100)			
partial-ER-30-fast(25)			
partial-ER-30-fast(100)			
partial-ER-100(25)			
partial-PA-30(25)			
partial-PA-30(100)			

Aggregate Predictions	**maxC vs C**	**paG vs C**	**paG vs maxC**
partial-ER-30(25)			
partial-ER-30(100)			
partial-ER-30-fast(25)			
partial-ER-30-fast(100)			
partial-ER-100(25)			
partial-PA-30(25)			
partial-PA-30(100)			

Table 2: Pairwise comparisons of detailed predictions (left) and aggregate predictions (right) for experiment group B for ER and PA graphs of 30 and 100 nodes with different amounts of training data (numbers in parenthesis). For M_1 vs M_2 black indicates M_1 outperforms M_2 with $p < 0.05$, white the reverse, and gray no significant difference found.

	5 (200)	5-fast (200)	ER (25)	ER-fast (25)
b_1 partial	2.37	1.65	4.68	1.44
b_1 full	0.051	0.038	0.24	0.01
b_{-1} partial	3.87	3.15	2.68	1.722
b_{-1} full	0.02	0.01	0.03	0.015

Table 3: Parameter comparisons in paG in experiment groups A and B (data amounts in parentheses).

learning algorithm `MaxLInf` performs best in most scenarios, except for preferential attachment graphs. The hGMM approach produces superior predictions as more data is available, while dominating the other methods in experiments with PA structures. If structure learning could recover the true network reliably, that would clearly be preferred under our assumption that the actual data is generated by the cascade model. The inevitable imperfection of structure learning, however, opens the door to alternative model forms. By relaxing the conditional independence structure of the cascade model, our hGMMs can capture dependencies manifest in the data due to unobserved edges.

`NetInf` was originally designed for a different evaluation measure, which accounts for its relatively poor showing in our study. It remains possible that some ideas of that method could be incorporated in an algorithm that learns network structures with high predictive accuracy. Future work should explore that possibility as well as other approaches to improve `MaxLInf`, such as more effective interleaving of structure and parameter learning. Another promising approach is to combine structure learning with graphical models, capturing some unobserved edges explicitly and some implicitly.

The inference complexity of **paG** grows exponentially with respect to the largest neighborhood's size. Although this result enables **paG** to model medium-size systems, such as academic co-authorship networks [Leskovec et al., 2009] and voting network experiments [Kearns et al., 2009], handling large social networks on the order of Facebook or Twitter remains a challenge for future work. A potential solution may

arise from the potential function's generalized form: we can exploit the fact that the neighborhood size of $s^t_{N^u_i}$ dictates the inference complexity, while the role of $s^{t-1}_{N^d_i}$ in the model's complexity is negligible. In other words, we may choose to include nodes whose actions are most likely to be correlated in the construction of $s^t_{N^u_i}$, while keeping the neighborhood for $s^{t-1}_{N^d_i}$ intact. Another possible approach is to approximate inference using a mean field algorithm [Xing et al., 2003]. This approach has been shown to work reasonably well in comparable settings with large social networks [Shi et al., 2009], and therefore may improve the scalability of the hGMM framework.

Acknowledgments

This work was supported in part by the ARL Collaborative Technology Alliance on Network Science.

References

Eytan Adar and Lada A. Adamic. Tracking information epidemics in blogspace. In *IEEE/WIC/ACM International Conference on Web Intelligence*, pages 207–214, Compiegne, France, 2005.

Lars Backstrom and Jure Leskovec. Supervised random walks: Predicting and recommending links in social networks. In *Fourth ACM International Conference on Web Search and Data Mining*, pages 635–644, Hong Kong, 2011.

Eytan Bakshy, Brian Karrer, and Lada A. Adamic. Social influence and the diffusion of user-created content. In *Tenth ACM Conference on Electronic Commerce*, pages 325–334, Stanford, CA, 2009.

Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

Munmun De Choudhury, Hari Sundaram, Ajita John, and Dorée Seligmann. Dynamic prediction of communication flow using social context. In *Nineteenth ACM Conference on Hypertext and Hypermedia*, pages 49–54, Pittsburgh, 2008.

Dan Cosley, Daniel Huttenlocher, Jon Kleinberg, Xiangyang Lan, and Siddharth Suri. Sequential influence models in social networks. In *Fourth International AAAI Conference on Weblogs and Social Media*, pages 26–33, Washington, DC, 2010.

Constantinos Daskalakis and Christos H. Papadimitriou. Computing pure Nash equilibria in graphical games via Markov random fields. In *Seventh ACM Conference on Electronic Commerce*, pages 91–99, Ann Arbor, MI, 2006.

Munmun De Choudhury, Winter A. Mason, Jake M. Hofman, and Duncan J. Watts. Inferring relevant social networks from interpersonal communication. In *Nineteenth International Conference on World Wide Web*, pages 301–310, Raleigh, NC, 2010.

Quang Duong, Michael P. Wellman, and Satinder Singh. Knowledge combination in graphical multiagent models. In *Twenty-Fourth Conference on Uncertainty in Artificial Intelligence*, pages 153–160, Helsinki, 2008.

Quang Duong, Michael P. Wellman, Satinder Singh, and Yevgeniy Vorobeychik. History-dependent graphical multiagent models. In *Ninth International Conference on Autonomous Agents and Multi-Agent Systems*, pages 1215–1222, Toronto, 2010.

P. Erdos and A. Renyi. On random graphs. i. *Publicationes Mathematicae*, 6(290-297):156, 1959.

J. Goldenberg, B. Libai, and E. Muller. Talk of the network: A complex systems look at the underlying process of word-of-mouth. *Marketing Letters*, 12(3):211–223, 2001.

Manuel Gomez-Rodriguez, Jure Leskovec, and Andreas Krause. Inferring networks of diffusion and influence. In *Sixteenth ACM International Conference on Knowledge Discovery and Data Mining*, pages 1019–1028, Washington, DC, 2010.

Sham Kakade, Michael Kearns, John Langford, and Luis Ortiz. Correlated equilibria in graphical games. In *Fourth ACM Conference on Electronic Commerce*, pages 42–47, San Jose, CA, 2003.

Michael Kearns, Stephen Judd, Jinsong Tan, and Jennifer Wortman. Behavioral experiments on biased voting in networks. *Proceedings of the National Academy of Sciences*, 106(5):1347–52, 2009.

David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *Ninth ACM International Conference on Knowledge Discovery and Data Mining*, pages 137–146, Washington, 2003.

Jon Kleinberg. Cascading behavior in networks: Algorithmic and economic issues. In Noam Nisan, Tim Roughgarden, Éva Tardos, and Vijay V. Vazirani, editors, *Algorithmic Game Theory*, pages 613–632. Cambridge University Press, 2007.

Lillian Lee. Measures of distributional similarity. In *Thirty-seventh Annual Meeting of the Association for Computational Linguistics*, pages 25–32, College Park, MD, 1999.

Jure Leskovec, Lada Adamic, and Bernardo Huberman. The dynamics of viral marketing. *ACM Transactions on the Web*, 1(1):5–43, 2007.

Jure Leskovec, Kevin J. Lang, Anirban Dasgupta, and Michael W. Mahoney. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *Internet Mathematics*, 6(1):29–123, 2009.

Joris M. Mooij. libDAI: A free and open source C++ library for discrete approximate inference in graphical models. *Journal of Machine Learning Research*, 11:2169–2173, 2010.

Xiaolin Shi, Jun Zhu, Rui Cai, and Lei Zhang. User grouping behavior in online forums. In *Fifteenth ACM International Conference on Knowledge Discovery and Data Mining*, pages 777–786, Paris, France, 2009.

Xiaodan Song, Yun Chi, Koji Hino, and Belle L. Tseng. Information flow modeling based on diffusion rate for prediction and ranking. In *Sixteenth International World Wide Web Conference*, pages 191–200, Banff, 2007.

F. Stonedahl, W. Rand, and U. Wilensky. Evolving viral marketing strategies. In *Twelfth Annual Conference on Genetic and Evolutionary Computation*, pages 1195–1202, Portland, OR, 2010.

E.P. Xing, M.I. Jordan, and S. Russell. A generalized mean field algorithm for variational inference in exponential families. In *Nineteenth Annual Conference on Uncertainty in Artificial Intelligence*, Acapulco, Mexico, 2003.

Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. Generalized belief propagation. In *Thirteenth Annual Conference on Advances in Neural Information Processing Systems*, pages 689–695, Denver, 2000.

A Bi-Threshold Model of Complex Contagion and its Application to the Spread of Smoking Behavior

Chris Kuhlman, V. S. Anil Kumar, Madhav Marathe, Samarth Swarup, Gaurav Tuli
Network Dynamics and Simulation Science Lab, Virginia Bioinformatics Institute, Virginia Tech, Blacksburg, VA 24060.
{ckuhlman, akumar, mmarathe, swarup, gtuli}@vbi.vt.edu

S. S. Ravi, Daniel J. Rosenkrantz
Department of Computer Science, University at Albany, State University of New York, Albany, NY 12222.
{ravi, djr}@cs.albany.edu

ABSTRACT

We study the dynamics of a bi-threshold model of contagion, wherein each node can be in one of two states (0 or 1), and will only change state if a minimum number (specified by an up-threshold and a down-threshold at each node) of its neighbors are in the opposite state. This model applies to processes where peer pressure is a strong factor in behavior change in either direction, such as initiation and cessation of smoking among adolescents.

We investigate this model both theoretically and experimentally. On the theoretical side, we establish results which show significant differences between simple contagions (where all thresholds are 1) and complex contagions (where one or more thresholds exceed 1) with respect to the complexity of determining several global properties of the system. On the experimental side, we apply this model to the data about adolescent smoking behavior from the National Longitudinal Study of Adolescent Health (Add Health) to analyze network dynamics such as the rate of spread and outbreak size.

Categories and Subject Descriptors

J.4 [**Computer Applications**]: Social and Behavioral Sciences

General Terms

Algorithms, Experimentation, Theory

Keywords

Complex contagion, bi-threshold model, adolescent smoking, synchronous dynamical systems

1. INTRODUCTION

Many social phenomena, such as smoking behavior, spread of information, diseases and viral marketing, can be modeled as contagion processes [6, 9, 12, 18, 22, 24, 34], in which an individual's state or choice is influenced by her neighbors' choices, leading to cascading effects. These models are predominantly **progressive** models [23], where in a two-state $\{0, 1\}$ system, nodes can only transition from 0 to 1; not 1 to 0. One way to capture this influence is through thresholds (e.g., [6, 22]). Informally, if a sufficient (i.e., a threshold) number of one's neighbors behave in a particular way, the node will also adopt that behavior.

Many real-world issues are characterized by two-choice back-and-forth decisions where a node can change state back and forth between 0 and 1. Schelling refers to this as cyclic behavior (at the micro and macro levels), and states: "Numerous social phenomena display cyclic behavior ..." (p. 86 [28]). He goes on to present everyday examples such as whether pick-up volleyball games early in the fall semester at Harvard will continue through the semester or die (e.g., individuals regularly choosing to play or not play). In refering to repeated decisions as to whether people will cross a street against traffic lights, a threshold is used explicitly in Schelling's description: "At some point [after some have walked into the street], several appear to decide that the flow of pedestrians is large enough to be safe and they join it, enlarging it further and making it safe for a few who were still waiting and who now join in." (p. 92 [28]). He also describes how people may initially step out into the street, but will retreat if there is an insufficient number of followers. One can also look at public health concerns such as obesity, where an individual's back-and-forth decisions to diet or not are so commonplace that it has a name: "yo-yo dieting" [1]. Moreover, dieting decisions are peer-influenced [7]. The point is that back-and-forth threshold systems are prevalent, and as will be described, these systems are also applicable to smoking.

The focus of our paper is studying a model for smoking behavior in adolescents through social network analysis and agent-based simulation (ABS). According to the World Health Organization (WHO) [33], smoking is responsible for 10% of all adult deaths and is the leading cause of preventable deaths. According to The Centers for Disease Control and Prevention, direct health care costs of smoking and produc-

tivity losses from adverse health are $96 billion and $97 billion annually, respectively [29].

A large number of factors are involved in the risk for smoking initiation among adolescents, including place of residence [2], schools and peer networks [8, 31], parental and familial influences [16], media messages [32], cigarette prices and policies [26], socioeconomic factors [5], and biological and cognitive factors [10]. However, Hoffman et al. recently reviewed the literature on adolescent cigarette smoking and found that peer influence, typically measured as the number of friends who smoke, has been repeatedly found to be the strongest risk factor [21]. Other ways of measuring peer influence, such as embeddedness in friendships, friendship quality, and peer social status also confirm this basic picture [13]. Recently, Go et al. [17] examined the Add Health data and showed that peer influence is a factor in both initiation and cessation of smoking among adolescents. Since peer influence is the biggest factor in the spread of smoking behavior, our model is especially appropriate for its study.

Christakis and Fowler showed through analysis of the Framingham Heart Study data that people tend to both start and stop smoking in groups [8]. Simple independent cascade models are unlikely to be valid for smoking behavior, because they cannot ensure such a property. Complex contagion models, in which a node switches from state 0 to state 1 if the number of neighbors in state 1 exceeds a threshold, can exhibit such a property. However, these models are monotone, and cannot explain other effects such as cessation of smoking.

In this paper, we propose a novel bi-threshold model of complex contagion, where transitions at each node are governed by two threshold values. When a node is in state 0, it transitions to state 1 when the number of its neighbors in state 1 equals or exceeds its *up-threshold*, denoted t_{up}. On the other hand, when a node is in state 1, it transitions to state 0 when the number of its neighbors in state 0 equals or exceeds its *down-threshold*, denoted by t_{down}. We apply the bi-threshold model of complex contagion to data from the National Longitudinal Study of Adolescent Health (Add Health [20]) on the spread of smoking behavior through adolescent friendship networks. Our main contributions are:

1. The bi-threshold model captures a number of effects observed in smoking behavior, including changes in groups and non-monotonic behavior, as we discuss later. The dynamics of such models are very complex and sensitive to the threshold values and underlying networks, which we verify empirically as well.

2. We study the complexity of the problems of computing dynamical properties of bi-threshold systems, e.g., reachability and fixed points, and show that they are NP-complete or #P-complete, in general. For systems with up and down threshold values of 1, we characterize the dynamics completely if the underlying contact graph is undirected. However, the problems become harder for directed graphs.

3. We infer parameters for a bi-threshold model to fit the available data on smoking status in waves I and II of the Add Health survey. This survey only has smoking states for a subset of nodes, which allow us to infer either an up or down threshold for them. We use a regression analysis to infer the initial states and the remaining thresholds for all the nodes.

4. We experimentally characterize the dynamical properties of such systems, and find that they generally converge to a set of pseudo-stationary configurations, in which the number of nodes in state 1 (i.e., the smoking state) does not vary much.

5. We find that high out-degree nodes have a significant impact on the rate of prevalence or cessation of smoking. If a small fraction (about 3.5%) of the highest out-degree nodes become smokers, the fraction of nodes in state 1 is almost doubled. On the other hand, if the same fraction of nodes becomes non-smokers, the fraction of nodes in state 1 reduces to a third. This corroborates with the observation that peer-influence is one of the most significant determinants of smoking behavior [8, 21].

Thus, the bi-threshold model generalizes a number of earlier threshold-based models for contagion. It captures complex non-monotonic diffusion phenomena, such as smoking behavior, much more realistically than other contagion models. Formal validation of the applicability of this model in the context of smoking behavior is difficult because of lack of adequate longitudinal smoking behavior data. This is a general problem with social phenomena that occur on a timescale of years, because gathering a large enough sample over many years is an effortful enterprise. For instance, the Add Health data only has partial information of the smoking states of a subset of the population studied over waves I and II. Nonetheless, our results with the networks in the Add Health data show many of the qualitative features observed by empirical studies on smoking behavior (which are not based on networked processes). Our approach is a first step towards providing a formal framework to unify and explain the diverse results on smoking behavior, and can help in evaluating and designing policies for controlling its spread. Further, smoking is a prototypical example, and such a framework would be useful in other applications.

Organization. In Section 2, we formally describe the bi-threshold model, following which we describe related work (Section 3) and then prove several theoretical results about the bi-threshold model in Section 4. In Section 5, we describe the Add Health data and our methodology for building the bi-threshold model to fit these data. We numerically explore the behavior of the bi-threshold model on the five largest friendship networks from the data. Comparing data from wave I and wave II allows us to construct a model for estimating threshold values for each node in the networks. In Section 6, we perform ABS with these estimated threshold parameters to study the dynamics of this model.

2. MODEL DESCRIPTION

2.1 Definition of the Bi-threshold Model

We model the propagation of contagions over a social network using discrete dynamical systems (e.g. [3]). We begin

with a definition of the model. Let $\mathbb{B}$ denote the Boolean domain {0,1}. A **Synchronous Dynamical System** (SyDS) $\mathcal{S}$ over $\mathbb{B}$ is specified as a pair $\mathcal{S} = (G, \mathcal{F})$, where (a) $G(V, E)$, an undirected graph with n nodes, represents the underlying social network over which the contagion propagates, and (b) $\mathcal{F} = \{f_1, f_2, \ldots, f_n\}$ is a collection of functions in the system, with f_i denoting the **local transition function** that computes the next state of v_i, $1 \leq i \leq n$.

Each function f_i specifies the local interaction between node v_i and its neighbors in G. (We use the convention that a node is *not* a neighbor of itself.) To provide additional details regarding these functions, we note that each node of G has a state value from $\mathbb{B}$. The inputs to function f_i are the state of v_i and those of the neighbors of v_i in G; function f_i maps each combination of inputs to a value in $\mathbb{B}$. In this paper, function f_i at node v_i is a **bi-threshold function**, characterized by two non-negative integer values denoted by $t_{\text{up}}(v_i)$ and $t_{\text{down}}(v_i)$. A precise definition of the function f_i is as follows.

(a) If the state of v_i is 0, then f_i is 1 if at least $t_{\text{up}}(v_i)$ of the neighbors of v_i are in state 1; otherwise, the value of f_i is 0.

(b) If the state of v_i is 1, then f_i is 0 if at least $t_{\text{down}}(v_i)$ of the neighbors of v_i are in state 0; otherwise, $f_i = 1$.

Thus, $t_{\text{up}}(v_i)$, called the **up-threshold** of v_i, represents the minimum number of neighbors of v_i that must be in state 1 for v_i to change from 0 to 1. Likewise, $t_{\text{down}}(v_i)$, called the **down-threshold** of v_i, represents the minimum number of neighbors of v_i that must be in state 0 for v_i to change from 1 to 0. A SyDS in which each node has a bi-threshold transition function is called a **bi-threshold** SyDS, denoted by BT-SyDS.

A **configuration** $\mathcal{C}$ of a SyDS at any time is an n-vector $(s_1, s_2, \ldots, s_n)$, where $s_i \in \mathbb{B}$ is the value of the state of node v_i. A single SyDS transition from one configuration to another can be expressed by the following pseudocode.

for each node v_i **do in parallel**
(i) Compute the value of f_i. Let s_i' denote this value.
(ii) Update the state of v_i to s_i'.
end for

Thus, in a SyDS, nodes update their state *synchronously*. Other update disciplines (e.g. sequential updates) for discrete dynamical systems have also been considered in the literature [3].

If a SyDS has a transition from configuration $\mathcal{C}_1$ to configuration $\mathcal{C}_2$, we say that $\mathcal{C}_2$ is the **successor** of $\mathcal{C}_1$ and that $\mathcal{C}_1$ is a **predecessor** of $\mathcal{C}_2$. A configuration that has no predecessor is called a **garden of eden** (GE) configuration. A configuration $\mathcal{C}$ is called a **fixed point** if the successor of $\mathcal{C}$ is $\mathcal{C}$ itself.

One can also define the bi-threshold model where the underlying graph of the dynamical system is *directed*. This is useful in modeling diffusion processes where the the influence relation between pairs of nodes is not symmetric; for example, a node u may influence another node v, but v may not have any influence on u. This asymmetry holds in the context of peer influence which is known to play a major role in the smoking behavior of adolescents (e.g., see [21]). If u influences v, the underlying graph contains the directed edge (u, v). The threshold values are with respect to the in-neighbors of a node v, that is, the set of nodes which have a directed edge to v. Thus, if a node v has an up-threshold $t_{\text{up}}(v)$, then at least $t_{\text{up}}(v)$ of its in-neighbors must be state 1 for v to change from 0 to 1. A similar explanation holds for the down-threshold values.

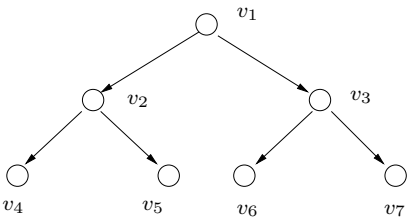


Figure 1: A directed BT-SyDS to illustrate an observed behavior.

Motivation: modeling smoking behavior. We now present a toy example to point out that the bi-threshold model with directed graphs can capture some observed behaviors in the context of smoking. In the literature, it is noted that "over time smokers were more likely to appear at the periphery" of the underlying social network [8]. To see how this behavior can occur under the bi-threshold model, consider the directed tree shown in Figure 1. Initially, the root node v_1 is in state 1 (corresponding to a smoker) and all the other nodes are in state 0 (corresponding to non-smokers). For each node, up-threshold is set to 1. For v_1, the down threshold is set to 0; for v_2 and v_3, the down threshold is set to 1, and for nodes v_4 through v_7, the down threshold is set to 2. It can be verified that during the first time step, v_1 changes to 0, v_2 and v_3 change to 1, while the other nodes remain at 0. During the second time step, v_1 remains at 0, v_2 and v_3 change to 0 and the other nodes change to 1. This is a *fixed point* of the system, and the nodes in state 1 correspond to leaves which are at the periphery of the graph. Also, observe that nodes v_2, v_3 simultaneously switch to state 1 in time step 2, and then switch to state 0 in time step 3. This captures the observation of Christakis and Fowler [8], that people tend to both start and stop smoking in groups.

2.2 Additional Definitions Related to the Model

For any SyDS $\mathcal{S}$, the **phase space** of $\mathcal{S}$ is a directed graph with one node for each possible configuration; there is a directed edge from the node representing configuration $\mathcal{C}$ to that representing configuration $\mathcal{C}'$ if and only if $\mathcal{C}'$ is the successor of $\mathcal{C}$. Since the domain is $\mathbb{B} = \{0,1\}$ and the underlying graph has n nodes, the number of nodes in the phase space is 2^n; thus, the size of the phase space is *exponential* in the size of the SyDS.

As defined above, the SyDS model is deterministic; that is, each configuration has a unique successor. Thus, the outdegree of each node in the phase space is 1. Each fixed point of a SyDS $\mathcal{S}$ is a self loop in the phase space of $\mathcal{S}$. Also, for any GE configuration, the corresponding node in the phase space has its indegree equal to zero.

A BT-SyDS in which $t_{\text{up}}(v) = t_{\text{down}}(v) = 1$ for each node v is called a **simple** BT-SyDS. If at least one of the threshold values in $\mathcal{S}$ is *greater than* 1, then $\mathcal{S}$ is referred to as a **complex** BT-SyDS.

If $t_{\text{up}}(v) = 0$ for some node v, then the state of v changes from 0 to 1 even when none of the neighbors of v is 1. We call such a node v an **uncontrolled up node**. Likewise, if $t_{\text{down}}(v) = 0$ for some node v, then node v will be referred to as an **uncontrolled down node**.

2.3 Computational Problems for BT-SyDSs

We study a number of different computational problems for BT-SyDSs. These problems model several questions regarding the global behavior of the underlying social network. Our results show a number of interesting differences between the complexities of these problems for simple and complex BT-SyDSs. Formal definitions of the problems studied in this paper are given below. In the literature, some of these problems have been considered for other dynamical system models (e.g. [3]).

Given a BT-SyDS $\mathcal{S}$, the **Fixed Point Existence** (FPE) problem asks whether $\mathcal{S}$ has a fixed point. The corresponding counting problem (i.e., finding the number of fixed points of $\mathcal{S}$) is denoted by #FPE.

Given a BT-SyDS $\mathcal{S}$ and a configuration $\mathcal{C}$, the **Predecessor Existence** (PRE) problem asks whether the configuration $\mathcal{C}$ has a predecessor. The corresponding counting problem (i.e., finding the number of predecessors) is denoted by #PRE.

Given a BT-SyDS $\mathcal{S}$ and two configurations $\mathcal{C}_1$ and $\mathcal{C}_2$, the **Configuration Reachability** (REACH) problem asks whether the system can reach $\mathcal{C}_2$ starting from $\mathcal{C}_1$.

Analytical results for the above problems are presented in Section 4.

3. RELATED WORK

Works on smoking behavior were presented in Section 1 and are not repeated. Here, we focus on contagion models and complexity results for contagion processes.

Different models have been used to study contagion processes, such as the independent cascade model, complex contagion [6] and its special case, the linear threshold model [22], and the linear influence model (LIM) [34]. Important questions in such applications include understanding steady state behavior, identifying influential individuals and the impact of network structure, and designing strategies to control the spread [12, 19, 22]. Yang et al., [34] use the linear influence model to fit the dynamics of information diffusion in Twitter. These models are so-called progressive models in which the only state transition allowed in a two-state $\{0, 1\}$ system is from state 0 to state 1.

Back-and-forth models also include the transition from state 1 to state 0. In the voter model [14], a node assumes the state of one randomly chosen neighbor, so that not all neighbors provide influence at each time, as does our model. In majority models (e.g., [11]), if one-half or more of a node's neighbors are in the opposite state, then a node transitions to that state. Our model is more general in that we can specify any minimum number of neighbors required to cause state transition, which may be more or less than one-half of nodes. We can also use relative thresholds, where a node's absolute threshold is normalized by its degree. A back-and-forth model is presented in [4]. However, that uniform mixing model assumes every node is connected to (and influenced by) all other nodes, so the graph forms a clique, and therefore that all information is globally known. In contrast, our system takes into account a population's connectivity, so that a node's interactions are local (confined to its neighborhood), which is more realistic in many cases. If global knowledge is required, we merely add a node and connect it to all others.

There are numerous results for computational complexity. In [22], the problem of finding the set of nodes of maximum size β that will result in the most nodes reached by a progressive diffusion process is shown to be NP-hard. A series of extensions is provided in papers up through [27]. See [25] for results on blocking diffusion. Results for the voter model and majority model are given in [14] and [11], respectively. We know of no complexity results for bithreshold models.

4. PHASE SPACE PROPERTIES OF BT-SyDSs: COMPLEXITY RESULTS

In this section, we present results on the complexity of testing various phase space properties for bi-threshold systems. The results are presented for the case of undirected graphs. By replacing each undirected edge $\{u, v\}$ by the pair of directed edges (u, v) and (v, u), it can be seen that the hardness results carry over to the directed case as well. We also discuss an interesting difference between the undirected and directed graph models with respect to phase space properties.

Due to space limitations, full proofs will appear in a complete version of this paper.

4.1 Fixed Point Existence and Counting

If a BT-SyDS does not have any uncontrolled up (down) nodes, then the configuration of all 0's (1's) is a fixed point. Therefore, the NP-hardness results given below for the FPE problem are for BT-SyDSs which contain nodes with up-threshold = 0 as well as those with down-threshold = 0.

THEOREM 4.1. *(i) The* FPE *and* #FPE *problems can be solved efficiently for BT-SyDSs with a maximum threshold of 1.*
(ii) The FPE *and* #FPE *problems are NP-complete and #P-complete respectively for BT-SyDSs where the maximum threshold is 2.*

Proof sketch: Part (i) is proven by considering each connected component (CC) of the underlying graph and showing that there are at most two candidate fixed points for the CC. Part (ii) is proven by a reduction from a restricted version of the Boolean Satisfiability problem (SAT), where each clause contains two or three literals and each literal appears in one or two clauses. (This restricted version of SAT is known be NP-complete [15].) ■

One may also consider variants of the FPE problem such as the following: Given a BT-SyDS $\mathcal{S}$ and an integer $q \leq n$, does $\mathcal{S}$ have a fixed point in which at most q nodes are in state 1? This variant, which we denoted by MIN-1-FPE, is motivated by the problem of determining whether a given social network has a stable configuration with only a small number of nodes in state 1. The following results hold for MIN-1-FPE.

THEOREM 4.2. *(i) The* MIN-1-FPE *problem can be solved efficiently for BT-SyDSs with a maximum threshold of 1.*
(ii) The MIN-1-FPE *problem is NP-complete for BT-SyDSs where the maximum threshold is 2.*

Proof sketch: Part (i) is proven by constructing a configuration $\mathcal{C}$ such that the only nodes which are in state 1 in $\mathcal{C}$ are those nodes which must be in state 1 in any fixed point of $\mathcal{S}$. The answer to the MIN-1-FPE instance is "yes" if and only if $\mathcal{C}$ is a fixed point and the number of nodes in state 1 in $\mathcal{C}$ is at most q. Part (ii) follows immediately from the NP-completeness of the FPE problems for BT-SyDSs in which the maximum threshold value is 2 (Part (ii) of Theorem 4.1) by setting $q = n$. ■

The above theorems provide a clear delineation between the complexities of FPE and #FPE problems for simple and complex contagions.

4.2 Predecessor Existence and Counting

THEOREM 4.3. *(i) The* PRE *problem can be solved efficiently for BT-SyDSs with a maximum threshold of 1.*
(ii) The #PRE *problem is #P-complete even when the maximum threshold is 1.*
(iii) The PRE *problem is NP-complete for BT-SyDSs even when all thresholds are 2.*

Proof (idea): Part (i) is proven by a careful analysis of the phase space of the system. (The proof involves several lemmas that capture how thresholds and state values of neighbors of a node v determine the state value of v in the predecessor configuration, if such a configuration exists.)

Part (ii) is shown by a parsimonious reduction from the problem of counting the number of satisfying assignments to 2CNF formulas which contain only positive literals. (The latter problem was shown to be #P-complete in [30].)

Part (iii) is shown by a reduction from a special version of SAT mentioned in the proof sketch for Theorem 4.1. ■

4.3 Reachability Problem

THEOREM 4.4. *The* REACH *problem can be solved efficiently for BT-SyDSs with a maximum threshold of 1.*

Proof (idea): This theorem is also proven by a careful analysis of the phase space of the system. In particular, the proof shows that BT-SyDSs in which the maximum threshold value is 1 either reach a fixed point or a 2-cycle in the phase space after a number of transitions that is bounded by the diameter of the underlying graph. ■

Like the FPE problem, one may also consider variants of the REACH problem. One such variant is the following: Given a BT-SyDS $\mathcal{S}$ an initial configuration $\mathcal{C}$ and an integer $q \leq n$, does the system reach a configuration in which at most q nodes are in state 1? If nodes in state 1 correspond to smokers, this question asks whether a social network will reach a configuration with a small number of smokers. This variant of reachability can also be solved efficiently for BT-SyDSs in which the maximum threshold value is 1 using the proof idea for Theorem 4.4 mentioned above.

Whether the REACH problem and its variant mentioned above can be solved efficiently for complex BT-SyDSs remains an open question. Below, we investigate it through simulation.

4.4 Undirected and Directed Graph Models

We now point out an interesting difference between the phase spaces of BT-SyDSs under undirected and directed graph models. As mentioned in the proof sketch for Theorem 4.4, for undirected graphs, when each threshold value is at most 1, every directed cycle in the phase space has length at most 2. This property doesn't hold when the underlying graph is directed. To see this, consider a BT-SyDS where the underlying graph is a simple directed cycle with $n \geq 3$ nodes. Let $V = \{v_1, v_2, \ldots, v_n\}$ denote the nodes and let $A = \{(v_1, v_2), (v_2, v_3), \ldots, (v_{n-1}, v_n), (v_n, v_1)\}$ be the edge set. Assume that the up and down threshold for each node is 1. In the initial configuration, let the state of v_1 be 1 and the states of the other nodes be 0. It can be verified that during successive time steps, the 1 value circulates among the nodes of the system, creating a cycle of length n in the phase space. One can construct other examples of BT-SyDSs with directed graphs such that their phase spaces have even longer cycles. Such examples point out that there are significant differences between the phase spaces of BT-SyDSs with undirected and directed graphs.

We now apply the bi-threshold model to smoking data from Add Health.

5. PARAMETER ESTIMATION

We focus on waves I and II of the Add Health survey [20]. Wave I data, collected in 1994-95, contain 85 adolescent friendship networks which were obtained by asking students at participating middle and high schools to nominate their friends. The networks contain 72589 distinct IDs. Of these, in-home interviews were conducted with 20746 individuals. For the present study, the variables we looked at were: whether the individual had smoked in the last 30 days, whether either of the individual's parents smoke, and the individual's age and gender. In wave II, follow-up interviews were conducted (in 1995-96) with 14739 of these individuals, and the same data were gathered again.

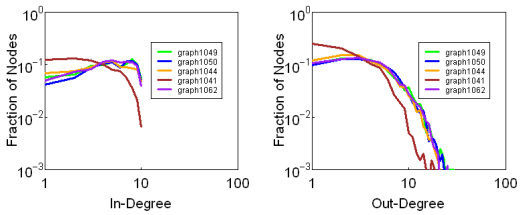


Figure 2: Frequency distributions of (a) indegree and (b) outdegree for five of the largest networks from the Add Health data set.

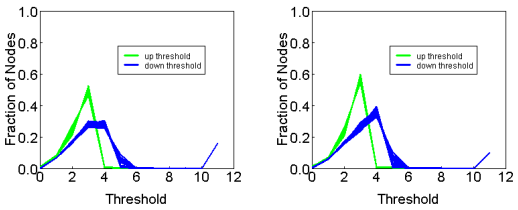
For the construction of the friendship networks, each person was asked to name up to five male and five female friends. We take these friends as influencers of the person; i.e., these friends influence the smoking behavior of the person. A directed edge (u, v) from node u to node v means that u influences v. Thus, a node's in-degree is the number of influencers for that node, and has a maximum value of 10. The maximum out-degree of any node was 36; i.e., some individuals influence 36 others. In-degree and out-degree distributions for the five largest networks, with numbers of nodes ranging from 2000 to 2600, are shown in Figure 2.

Since the variables of interest are not available for the individuals who were not interviewed at home, we first take some simple steps to fill in the missing data. From the $\sim$ 14000 nodes for which age is known, a histogram was generated. Ages were assigned to unknown nodes randomly from the resulting distribution. For nodes with unknown gender, male or female was assigned with 50% probability. Parents' smoking status (hereafter PS) is known for $\sim$17000 nodes; of these, 75% of nodes had PS value of 1 (i.e., at least one parent smokes). Nodes with unknown PS were assigned 1 with a 0.75 probability. This procedure was applied to the entire data set, rather than to data for individual networks, because for some networks the available data were meager.

We then performed a set of Poisson regressions, grouping nodes by age, and then using gender and PS to determine the probability of being a smoker. This was used to fill in initial (wave I) smoking states for nodes for which this variable was unknown. For each node, the initial state is decided based on $v = Poisson(\lambda)$, where $Poisson(\lambda)$ is the realization of a Poisson random number. If $v = 0$, the initial state is 0 (i.e., non-smoking) and if $v > 0$, the initial state is 1. Values for these fits for each age are given in Table 1. There the columns are the gender of the child and whether the parents smoke or not. This process of assigning age, gender, PS, and initial smoking state provides one collection of traits and initial state. To assess variability, this process was repeated 500 times. Hence, nodes with unknown traits or initial state were assigned different values across the 500 instances, but known values for nodes were always used. The initial smoking states are used directly to specify the initial configuration for each of 500 diffusion instances of simulations in the next section. Roughly two-thirds of nodes are initially in the non-smoking (0) state for each of the 500 instances. Both traits and initial state were also used to derive regressions for t_{up} and t_{down}, as described next. One threshold can be inferred for every node for which the

Age	F, PS = 0	M, PS = 0	F, PS = 1	M, PS = 1
13	0.08	0.08	0.2	0.2
14	0.12	0.14	0.3	0.33
15	0.23	0.23	0.47	0.47
16	0.24	0.25	0.47	0.5
17	0.27	0.32	0.53	0.63
18	0.31	0.39	0.52	0.65

Table 1: λ values by age, gender, and parents smoking status (PS) for deciding the initial smoking state of nodes in the networks. Values have been rounded to two places after the decimal point for presentation here.



(a) (b)

Figure 3: Twenty randomly chosen sets of up and down threshold distributions (out of the 500 total sets) for (a) network 1044 and (b) network 1049. Thresholds of 11 mean essentially an infinite threshold because the maximum indegree of any node is 10; such nodes are "pure influencers."

smoking states in wave I and wave II are known. The rules for determining thresholds are as follows. If a node is in state 0 in wave I and state 1 in wave II, then an up-transition has taken place. Hence, the up-threshold is at most the number γ of influencers in state 1 in wave I. The up-threshold is assigned uniformly at random from the interval $[0, \gamma]$. For nodes that are in state 1 in wave I and state 0 in wave II, the down-threshold is determined analogously. For nodes that are in state 0 in both wave I and wave II, the number of wave I influencers in state 1 is not sufficient to cause an up-transition, and hence the up-threshold for the node is at least $(\gamma + 1)$, and we use this value. Again, a similar argument is made for the down-threshold. Approximately 8100 thresholds were inferred in this manner.

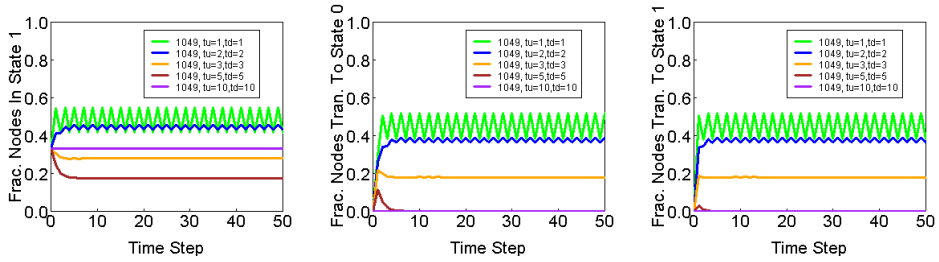
We now have for all nodes the values of the three traits and the initial smoking state, and for 8100 nodes, one threshold value. Linear regressions were performed to determine up-threshold as a function of the three traits and in-degree and out-degree for each of the 500 collections, and similarly for down-threshold, and these procedures yielded low variances. Up and down threshold distributions for 20 randomly chosen threshold assignments for two networks are given in Fig. 3.

6. SIMULATIONS AND RESULTS

We perform simulations to study multiple questions about the model. We use the largest five of the 85 networks from Add Health and the simulation parameters described in Section 5. We present results that elucidate the long-term dynamics of our model, and its sensitivity to different parameters. Then we conduct experiments where we freeze the highest out-degree nodes to study the impact of the thresholds of the hubs on the prevalence and cessation of smoking behavior in the network as a whole. We tie results to smoking behavior. Our main results are summarized below.

1. In our simulations we find that the system seems to converge to a "pseudo-stationary" state where the number of nodes in state 1 does not vary very much.

2. We find the results are very sensitive to the choices of thresholds. Increasing t_{up} and t_{down} uniformly (i.e., $t_{up} = t_{down} = t$) has interesting non-monotone effects. The fraction of nodes in state 1 decreases initially as t increases from 1 to 2, and then surprisingly *falls below* the fraction of 1-nodes in the initial configuration as t is further increased to 5. Note that this is



(a) (b) (c)

Figure 4: Dynamics for network 1049 for different $t_{up} = t_{down}$. (a) Fraction of nodes currently in state 1 as a function of time; (b) fraction of nodes transitioning to state 0; and (c) fraction of nodes transitioning to state 1. Figure (a) shows that the steady-state fraction of nodes in state 1 does not decrease monotonically to the initial fraction (at time 0), but rather as threshold increases to 3 and 5, the fractions of nodes in state 1 becomes less than the initial fraction.

counter to the behavior that would be observed if only the up-thresholds were increased, and thus the down-thresholds have an important role in this behavior.

3. We find that there can be large differences in the dynamics across different iterations, even if threshold values and proportion of initial smokers are held constant across iterations. The differences arise solely due to differences in initial conditions. These changes show up as subtle differences in average-case behavior, but show up as large differences when we plot the dynamics for all the individual iterations.

4. We find that nodes of high out-degree are highly influential in both prevalence and cessation of smoking rates. If a small fraction (3.5%) of the nodes with the highest out-degrees are permanent smokers (i.e., have states fixed at 1), the number of nodes in state 1 more than doubles. On the other hand, if this same set of nodes remain non-smokers (i.e., their states are fixed at 0), the fraction of nodes in state 1 becomes less than a third. This corroborates the observations that peer pressure has a significant impact on smoking [8, 21].

An **iteration** is a diffusion instance, where all nodes are assigned an initial state (either 0 or 1) and constant values for t_{up} and t_{down}. We only consider the first 50 time steps in the temporal evolution, where each step corresponds to one year, consistent with the one-year duration between wave I and wave II data sets in Add Health.

A **simulation** consists of a set of 100 or 500 iterations, each using a different set of initial state assignments. All the figures showing simulation results show curves which are calculated as point-wise averages from the set of iterations. Some of the figures (as will be noted) also show the dynamics of the individual iterations. Nodes whose initial states are known have the same initial state in all iterations; the remaining nodes are assigned a state according to their traits (age, gender, and whether their parents smoke), which can vary across iterations, as described in Sec. 5. Similarly, nodes for which a threshold can be inferred had the same inferred threshold over all iterations, but the other threshold for such nodes (either t_{up} or t_{down}), as well as both thresholds for other nodes, were assigned based on regressions that incorporate node traits. We discuss these experiments below.

1. Sensitivity to up and down threshold values. Nodes of networks were assigned homogeneous thresholds with $t_{up} = t_{down} = 1, 2, 3, 5$, and 10 for five simulations. Initial states were taken from the data sets. Average results in time over 100 iterations for network 1049 are provided in Figure 4. There is an initial transient phase, followed by a quasi-steady state phase where the behavior is approximately periodic. One would intuitively expect that as thresholds increase, the numbers of state transitions would decrease, and hence that the number of nodes in state 1 would monotonically approach that of the initial state configuration for all times. Figure 4(a) clearly shows that this is not the case. As threshold increases from 1 to 2, the numbers of nodes in state 1 decreases, but when thresholds increase to 3 and 5, the numbers of nodes in state 1 decrease below that of the initial configuration, before returning to the numbers of nodes in the initial configuration for threshold 10. Figures 4(b) and 4(c) show that the relative numbers of nodes transitioning to state 0 and state 1, respectively, are different for different thresholds in the transient regime, and these differences dictate whether the steady state fraction of nodes in state 1 is more or less than that of the initial configuration. These results are representative of the networks studied.

Note that the oscillations for $t_{up} = t_{down} = 1$ represent extreme behavior, perhaps applicable only to situations like street crossings in Section 1. However, it is interesting how increasing the thresholds by one attenuates the oscillations.

2. Variation across networks. Applying the heterogeneous thresholds and initial states of nodes as presented in Section 5 yields differences in behaviors across networks, exemplified by the results of Figure 5, which are average curves over 500 iterations. Network 1044 shows a decrease in numbers of nodes in state 1 over time, while other networks show an initial decrease, followed by an increase. The other two plots show that the numbers of nodes transitioning to state 0 are greater at the beginning of a simulation compared to the numbers changing to state 1, but that over time, the state transitions to 1 gradually increase. The plots also indicate that the numbers of nodes transitioning can fluctuate over 10 to 20 or more time steps (years) before settling to approximately steady state values. These swings are: (i) qualitatively similar to gradual changes observed in smoking, and (ii) illustrate that the model can produce "overshooting," which is important for social applications [4, 28].

3. Variance in long-term dynamics. Thus far, average behaviors have been emphasized to illustrate high-level trends.

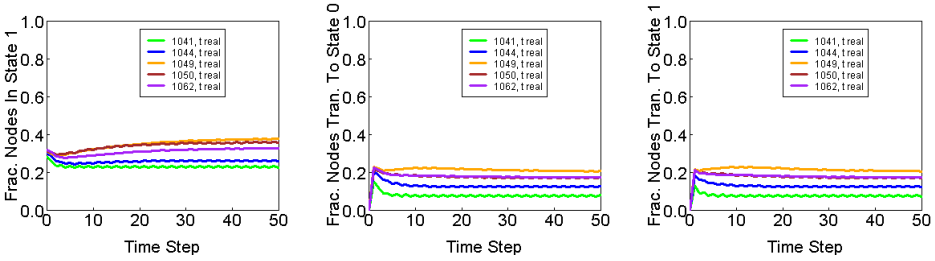


Figure 5: Dynamics for five networks for heterogeneous threshold values mined and inferred from observations. (a) Fraction of nodes currently in state 1; (b) fraction of nodes transitioning to state 0; and (c) fraction of nodes transitioning to state 1. These figures show that the average fractions of nodes in state 1 can either increase or decrease, or both, at early times, with transients out beyond time 20.

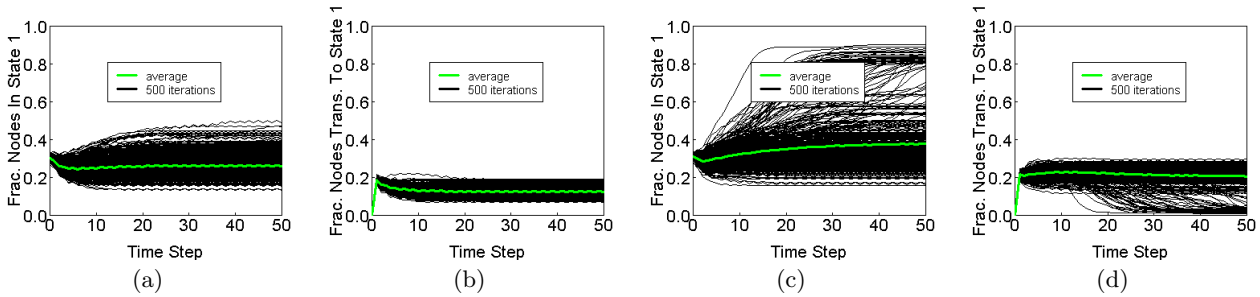
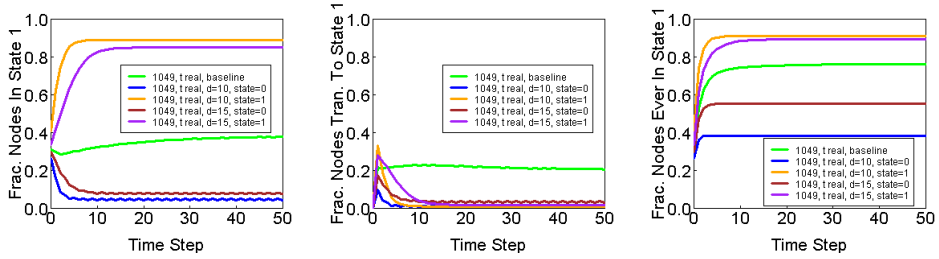


Figure 6: Dynamics for two networks showing 500 individual diffusion instances. (a) Fraction of nodes currently in state 1 for network 1044; (b) fraction of nodes transitioning to state 1 for network 1044; (c) fraction of nodes currently in state 1 for network 1049; (d) fraction of nodes transitioning to state 1 for network 1049. These results illustrate that focusing solely on average behavior hides ranges in behavior provided by individual diffusion instances.

However, average behaviors can hide variations in results across individual diffusion instances, and examples of this are depicted in Figure 6. The number of nodes in state 1 and the number of nodes transitioning to state 1 are provided for network 1044 in the first two plots, followed by corresponding plots for network 1049. Figures 6(a) and 6(c) show that differences in the average curves (in green) are driven primarily by diffusion instances for network 1049 that achieve 80% or more of nodes in state 1, and by diffusion instances for network 1044 that generate lesser fractions of nodes in state 1. Note that many instances between the two networks are within the same band. Figure 6(c) also indicates, through iterations whose curves are trending upwards at times between 40 to 50 steps, that transient behavior exists out to 50 years and beyond. Differences are also observed between Figures 6(b) and 6(d). The iterations in Figure 6(d) that tend to zero correspond to the iterations whose fraction of nodes in state 1 are 0.80 and above. There are no more nodes that can reach state 1, driving the number transitioning to zero. The plots for fractions of nodes transitioning to state 0 are very similar to those of Figures 6(b) and 6(d), so that the greater fractions in Figure 6(d) are offset by down transitions. Because the actual effects of smoking prevention policies correspond to one diffusion instance, policy makers must be aware that predicted average trends can hide complicated behaviors across iterations.

4. Relative influence of nodes in smoking prevalence and cessation. Correlation between the behavior of the popular students and of the group as a whole is well accepted, however causality can be argued in both directions. For example, Valente et al. [31] investigated the possibility that popular students are more susceptible to taking up smoking, by surveying middle-schoolers in southern California about their commitment not to smoke in the future. They found that popular students were statistically less likely to make the commitment. On the other hand, it can be argued that smoking becomes widespread in a network precisely if the popular students start smoking.

We present some simulations that show that the bi-threshold model exhibits a similar relationship between the states of the highest out-degree nodes and the fraction of smokers in the network. (The following results also control for the atmost q nodes in state 1, which is related to the variant of the REACH problem for complex BT-SyDSs in Section 4.3.) Specifically, we compare the effects of choosing a subset of nodes S according to two criteria: random and high out-degree based (i.e., consisting of the nodes with out-degree at least d, for different choices of d). In Figure 7, we show the effect of "freezing" the states of all nodes in S at value 1 (by setting their t_{down} to be very high values) for one of the networks we consider (refer to the curves labeled "state=1" in this figure); only the results for the choice of $d = 15$ and $d = 10$, corresponding to S having about 3.5% and 14.5% of the nodes, respectively, are shown here. We observe that freezing these top 3.5% high out-degree nodes results in a lot more nodes staying in state 1 (Figure 7(a)), compared to the baseline (which corresponds to the setting without any frozen nodes), and to random choice of a similar fraction of



(a) (b) (c)

Figure 7: For network 1049, (a) number of nodes currently in state 1, (b) number of nodes transitioning to state 1, and (c) cumulative number of nodes ever to reach state 1. The green curve is the baseline condition. The blue and brown curves fix the nodes that have out-degree at least 10 and 15, respectively, in state 0 and hence decrease the numbers of nodes in state 1 in the steady state and lower the fractions of nodes to ever reach state 1. The orange and purple curves, corresponding to fixing the same nodes in state 1, serve to increase the nodes in state 1. In fact, 3.5% of nodes fixed in state 1 (purple curve) is almost as effective as 14% of nodes fixed in state 1 (orange curve).

nodes (not shown here because of space constraints).

Similarly, the number of nodes which were ever in state 1 (which models individuals who might have ever smoked) is also much larger than the baseline (Figure 7(c)) and random. This suggests that if high out-degree individuals start smoking, it causes high prevalence of smoking. This raises a natural question: does this behavior also hold for smoking cessation? Indeed it does, as shown by the curves labeled "state=0" in Figure 7. Here, we freeze the states of the nodes in S to 0. Figure 7(a) shows that the number of nodes in state 1 at any time is much smaller than the baseline; further, the number of nodes ever in state 1 is also much smaller. As in the earlier scenario, we find that freezing 3.5% of the nodes to 0 is as effective as freezing 14.5% of the nodes at 0, with respect to the number of nodes in state 1 at any time. As mentioned earlier, the results for random choice make very minimal change in both scenarios, and are not shown here. These results are consistent with experimentally determined peer effects of smoking [8].

7. CONCLUSIONS

We present a novel model of complex contagion, the bi-threshold model, motivated by non-monotonic diffusion phenomena such as the spread of smoking behavior. Through theoretical and simulation results on networks from the Add Health survey, we find that this model captures a number of features of observed smoking behavior, such as the impact of peer-influence on smoking and its cessation. We also examine the computational complexity of determining fundamental dynamical properties of this model. The hardness results for absolute thresholds also hold for relative thresholds where thresholds are normalized by a node's in-degree; simulations can be run with relative thresholds as well. There are several directions for further research. Theoretically, it would be of interest to establish the complexity of the reachability problem for complex contagions, and to identify further differences in the structures of the phase spaces between undirected and directed graphs. From a practical perspective, a useful research direction is to identify and explore other contexts where the bi-threshold model (or a suitable generalization thereof) can be utilized.

Acknowledgments: We thank the reviewers, our external collaborators, and members of the Network Dynamics and Simulation Science Laboratory (NDSSL) for their suggestions and comments. We also acknowledge use of the Add Health data, administered by the University of North Carolina at Chapel Hill and funded by the Eunice Kennedy Shriver National Institute of Child Health and Human Development, among other agencies. We are especially indebted to Richard Beckman for his guidance and for developing the model for assigning initial smoking states. This work has been partially supported by NIH MIDAS project 2U01GM070694-7, NSF PetaApps Grant OCI-0904844, DTRA R&D Grant HDTRA1-0901-0017, DTRA CNIMS Grant HDTRA1-07-C- 0113, NSF Netse CNS-1011769, and NSF SDCI OCI-1032677.

8. REFERENCES

[1] R. Atkinson, W. Dietz, J. Foreyt, N. Goodwin, J. Hill, J. Hirsch, F. Pi-Sunyer, R. Weinsier, R. Wing, J. Hoofnagle, J. Everhart, V. Hubbard, and S. Yanovski. Weight Cycling. *Journal of the American Medical Association*, 272(15):1196–1202, 1994.

[2] J. R. Barnett. Does place of residence matter? Contextual effects and smoking in Christchurch. *The New Zealand Medical Journal*, 113(1120):433–435, 2000.

[3] C. L. Barrett, H. B. Hunt III, M. V. Marathe, S. S. Ravi, D. J. Rosenkrantz, and R. E. Stearns. Complexity of Reachability Problems for Finite Discrete Dynamical Systems. *J. Comput. Syst. Sci.*, 72(8):1317–1345, 2006.

[4] G. Bischi and U. Merlone. Global Dynamics in Binary Choice Models with Social Influence. *J. Math. Sociology*, 33:277–302, 2009.

[5] U. Broms, K. Silventoinen, E. Lahelma, M. Koskenvou, and J. Kaprio. Smoking cessation by socioeconomic status and marital status: The contributions of smoking behavior and family background. *Nicotine and Tobacco Research*, 6(3):447–455, 2004.

[6] D. Centola and M. Macy. Complex Contagions and the Weakness of Long Ties. *American J. Sociology*, 113(3):702–734, 2007.

[7] N. Christakis and J. Fowler. The Spread of Obesity in a Large Social Network Over 32 Years. *N. Engl. J.*

Med., pages 370–379, 2007.

[8] N. A. Christakis and J. H. Fowler. The collective dynamics of smoking in a large social network. *N. Engl. J. Med.*, 358(21):2249–2258, May 22 2008.

[9] D. Crandall, D. Cosley, D. Huttenlocher, J. Kleinberg, and S. Suri. Feedback effects between similarity and social influence in online communities. In *Proceedings of KDD*, Las Vegas, Nevada, USA, August 24-27 2008.

[10] I. T. Croghan, J. O. Ebbert, R. D. Hurt, J. T. Hays, L. C. Dale, N. Warner, and D. R. Schroeder. Gender differences among smokers receiving interventions for tobacco dependence in a medical setting. *Addictive Behaviors*, 34(1):61–67, Jan 2009.

[11] P. Dreyer and F. Roberts. Irreversible k-Threshold Processes: Graph-Theoretical Threshold Models of the Spread of Disease and Opinion. *Discr. Appl. Math.*, 157:1615–1627, 2009.

[12] D. Easley and J. Kleinberg. *Networks, Crowds and Markets: Reasoning About A Highly Connected World*. Cambridge University Press, New York, NY, 2010.

[13] S. T. Ennett, R. Faris, J. Hipp, V. A. Foshee, K. E. Bauman, A. Hussong, and L. Cai. Peer smoking, other peer attributes, and adolescent cigarette smoking: A social network analysis. *Prevention Science*, 9:88–98, 2008.

[14] E. Even-Dar and A. Shapira. A Note on Maximizing the Spread of Influence in Social Networks. In *WINE 2007, LNCS 4858*, pages 281–286, 2007.

[15] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. W. H. Freeman and Co., San Francisco, CA, 1979.

[16] S. E. Gilman, R. Rende, J. Boergers, D. B. Abrams, S. L. Buka, M. A. Clark, S. M. Colby, B. Hitsman, A. N. Kazura, L. P. Lipsitt, E. E. Lloyd-Richardson, M. L. Rogers, C. A. Stanton, L. R. Stroud, and R. S. Niaura. Parental smoking and adolescent smoking initiation: An intergenerational perspective on tobacco control. *Pediatrics*, 123:e274–e281, 2009.

[17] M.-H. Go, H. D. Green Jr., D. P. Kennedy, M. Pollard, and J. S. Tucker. Peer influence and selection effects on adolescent smoking. *Drug and Alcohol Dependence*, 109:239–242, 2010.

[18] M. Gomez-Rodriguez, J. Leskovec, and A. Krause. Inferring networks of diffusion and influence. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2010.

[19] Habiba, Y. Yu, T. Berger-Wolf, and J. Saia. Finding Spread Blockers in Dynamic Networks. In *Proc. SNA-KDD Workshop*, 2008.

[20] K. Harris. The National Longitudinal Study of Adolescent Health (Add Health), Waves I and II, 1994-1996; Wave III, 2001-2002 [machine-readable data file and documentation], 2008. Chapel Hill, NC: Carolina Population Center, University of North Carolina at Chapel Hill 2008.

[21] B. R. Hoffman, S. Sussman, J. B. Unger, and T. W. Valente. Peer influences on adolescent cigarette smoking: A theoretical review of the literature. *Substance Use and Misuse*, 41:103–155, 2006.

[22] D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the Spread of Influence Through a Social Network. In *Proc. ACM KDD*, pages 137–146, 2003.

[23] J. Kleinberg. Cascading Behavior in Networks: Algorithmic and Economic Issues. In *Algorithmic Game Theory*, chapter 24, pages 613–632. Cambridge University Press, NY, NY, 2007.

[24] G. Kossinets, J. Kleinberg, and D. Watts. The Structure of Information Pathways in a Social Communication Network. In *Proc. ACM KDD*, 2008.

[25] C. Kuhlman, V. Kumar, M. Marathe, S. Ravi, and D. Rosenkrantz. Finding Critical Nodes for Inhibiting Diffusion of Complex Contagions in Social Networks. In *Proc. ECML PKDD*, pages 111–127, 2010.

[26] L. Liang, F. Chaloupka, M. Nichter, and R. Clayton. Prices, policies and youth smoking, may 2001. *Addiction*, 98(Suppl 1):105–122, 2003.

[27] E. Mossel and S. Roch. On the Submodularity of Influence in Social Networks. In *Proc. ACM STOC*, pages 128–134, 2007.

[28] T. Schelling. *Micromotives and Macrobehavior*. W. W. Norton and Company, 1978.

[29] USA Today Newspaper. Do smokers cost society money?, 2009. 8 April 2009, `http://www.usatoday.com/news/health/2009-04-08-fda-tobacco-costs\_N.htm`.

[30] S. Vadhan. The Complexity of Counting in Sparse, Regular and Planar Graphs. *SIAM J. Comput.*, 31(2):398–427, 2001.

[31] T. W. Valente, J. B. Unger, and C. A. Johnson. Do popular students smoke? The association between popularity and smoking among middle school students. *Journal of Adolescent Health*, 37:323–329, 2005.

[32] M. Wakefield, B. Flay, M. Nichter, and G. Giovino. Role of media in influencing trajectories of youth smoking. *Addiction*, 98(Suppl 1):79–103, 2003.

[33] World Health Organization. Tobacco free initiative, 2010. `http://www.who.int/tobacco/health_priority`.

[34] J. Yang and J. Leskovec. Modeling information diffusion in implicit networks. In *Proceedings of the 10th IEEE ICDM*, Sydney, Australia, 2010.

Language-independent Bayesian sentiment mining of Twitter

Alex Davies
University of Cambridge
Cambridgeshire, United Kingdom
ad564@cam.ac.uk

Zoubin Ghahramani
University of Cambridge
Cambridgeshire, United Kingdom
zoubin@eng.cam.ac.uk

ABSTRACT

This paper outlines a new language-independent model for sentiment analysis of short, social-network statuses. We demonstrate this on data from Twitter, modelling happy vs sad sentiment, and show that in some circumstances this outperforms similar Naive Bayes models by more than 10%.

We also propose an extension to allow the modelling of different sentiment distributions in different geographic regions, while incorporating information from neighbouring regions.

We outline the considerations when creating a system analysing Twitter data and present a scalable system of data acquisition and prediction that can monitor the sentiment of tweets in real time.

Categories and Subject Descriptors

I.2.7 [**Artificial Intelligence**]: Natural language processing—*Language models, Text analysis*

General Terms

Theory

Keywords

Twitter, Sentiment analysis, Topic modelling, Geo mining

1. INTRODUCTION

People are increasingly posting public content on the internet that reveals their sentiments and opinions. By far the largest source of this content, in terms of numbers of users, is Twitter. Twitter, at the time of writing, has 200 million users, who generate over 65 million tweets per day. There is a vast amount of sentiment information contained in this data, but we need suitable techniques to effectively extract it.

Understanding the sentiment or opinions of people is a valuable resource. It is useful on both a macro scale, where we can evaluate aggregated sentiment to predict macro-scale human behaviour, and a micro scale, where individual's sentiments provide actionable information to personalize services and target particular users.

The 5th SNA-KDD Workshop '11 (SNA-KDD'11), August 21, 2011, San Diego CA USA . Copyright 2011 ACM 978-1-4503-0225-8...$5.00.

At the macro-scale, there has been work predicting a wide variety of trends based on sentiment information retrieved from Twitter. This has been in fields as varied as politics, marketing and finance. In [13], it was found that correlations as high as 80% could be found between political sentiment data on Twitter and traditional polling methods. Recent work has shown that Twitter sentiments are very strong predictors of movie box office performance [1] and even the closing value of the Dow Jones Index [5].

A useful feature of Twitter is that a large amount of its tweets are accompanied by geodata; information about where in the world the tweet was generated. This facilitates the creation of systems that can model differences in social network usage between different geographic areas. This data has been used to model language variation across the United States [8], as well as sentiment variation across the US by state and time of day.

2. TWITTER DATA, GEO MODELLING AND SENTIMENT ANALYSIS

Previously, sentiment analysis in Twitter has largely been concerned with English tweets. This has allowed the use of domain knowledge about English in models to substantially increase the effectiveness of sentiment prediction models. There are large lists of manually curated English words and their associated sentiments such as affective word lists ANEW [6]. Some methods also employ features obtained from Part-of-Speech taggers, though there are conflicting results as to their utility [16][2]. Both of these methods do not extend to other languages.

We approach the problem of modelling sentiment, while making no assumption on language. We propose a probabilistic model that learns a word distribution for each sentiment based on a set of key indicator words for each sentiment. To be language independent, throughout our examples we use emoticons as the key indicating words. Previous works have used emoticons as noisy labels for training sentiment classifiers. On this data, Naive Bayes tends to outperform other more sophisticated techniques, such as SVMs and CRFs[15], so we will use it as a baseline for comparison.

Secondly, we propose a simple method for modelling different sentiment distributions in different geographic regions, while incorporating information from neighbouring regions to improve estimates in areas of low tweet density.

3. SENTIMENT PREDICTION

Our model for sentiment prediction is as follows:

A set of tweets, T, are generated from a multinomial mixture model, where the hidden mixture component s is the sentiment and the multinomial $\vec{\theta}_s$ is the word distribution for sentiment s. $\vec{\theta}_s$ is a vector of length $|V|$, that sums to 1, where each element $\theta_{s,i}$ is the probability that word w_i is drawn on a given draw from sentiment s.

Our prior belief over each $\vec{\theta}_s$ is an asymmetric Dirichlet distribution $Dir(\vec{\alpha}_s)$. $\vec{\alpha}_s$ is based on a set of keywords for sentiment s, F_s, and will enforce that $\vec{\theta}_s$ is conceptually about sentiment s.

3.1 Likelihood

Our data consists of $|T|$ tweets $t_i = (w_{i,1}, ..., w_{i,n_i})$ where n_i is the number of words in tweet i. Each word is a member of the set V, which is the vocabulary of words we consider in our model. For each tweet t_i the probability of the tweet given a set of word distributions, one for each sentiment s, is

$$p(t|\vec{\theta}) = \sum_{s \in S} p(t|\vec{\theta}_s)p(s) \quad (1)$$

$$= \sum_{s \in S} \prod_{w \in t} p(w|\vec{\theta}_s)p(s) \quad (2)$$

$$= \sum_{s \in S} p(s) \prod_{w \in t} p(w|\vec{\theta}_s) \quad (3)$$

The multinomial mixture-model is a good language model for tweets, as due to their short size (max 140 characters) they very rarely exhibit multiple sentiments. This is the justification for choosing this model over a more expressive model such as Latent Dirichlet Allocation[4].

3.2 Prior

We would like our word distribution to reflect our chosen sentiments, rather than for arbitrary components that separate the data well. To do this we use an asymmetric dirichlet prior $Dir(\vec{\alpha}_s)$. By setting a high value of $\vec{\alpha}_{happy,i}$, where i is the index for the word ':)', we encode the belief that happy smilies are likely to occur in happy tweets. Similarly, by setting a low value of $\vec{\alpha}_{sad,i}$, we encode the belief that happy smilies are unlikely to occur in sad tweets.

Formally, we set $\vec{\alpha}_s = k\vec{1} + \vec{\delta}_s$. All vectors are of length $|V|$, the size of the vocabulary. $\vec{1}$ is a vector with all values equal to 1, k is a scalar constant and $\vec{\delta}_s$ is defined as follows:

$$\delta_{s,w} = \begin{cases} \delta_+ & \text{if } w \in F_s \\ \delta_- & \text{if } w \in F_{s'} \text{ where } s' \neq s \\ 0 & \text{Otherwise} \end{cases} \quad (4)$$

δ_- and δ_+ are tunable parameters that are the weights added to the prior on the values of the indicative words. $\delta_+ > 0$ adds mass to the indicative words for a sentiment, reflecting our belief that they are more likely, while $-k < \delta_- < 0$ removes mass from indicative words for other sentiments. The full form of the prior is thus:

$$p(\vec{\theta}_s) = Dir(k\vec{1} + \vec{\delta}_s) \quad (5)$$

Choice of δ_+

It is important to take some care in selecting the value of δ_+. The value of δ_+ shouldn't be set too high, as if the values of $\alpha_{s,i}$ for F_s are too high relative to the other values of $\vec{\alpha}_s$, our model would expect there to be multiple elements of F_s in a given tweet of sentiment s. It would then assign lower probabilities to tweets that have too few indicating words.

As a rule of thumb, it is best to set $\delta_+ < \frac{k|V|}{n|T_s|}$ $\forall T_s$, where n is the average number of words in a tweet.

3.3 Learning

Because of the latent variables s, exact inference in this model is not possible. However, we can perform Variational Bayes[9] to obtain an approximation to the posterior distribution $P(\vec{\theta}_s|\vec{\alpha}_s, T)$. Variational Bayes is a common method for efficiently finding an approximate distribution over model parameters in intractable Bayesian models. It finds this approximation by first assuming the distribution factorizes into a product of simpler distributions and iteratively optimizes each of the simpler distributions such that their product is closer to the true distribution we are trying to approximate.

The update equations for a Dirichlet-Multinomial model using the mean-field approximation are:

$$n_{s,w}^{k+1} = \langle n_{s,w}^k \rangle_{\vec{\theta}^k} \quad (6)$$

$$\theta_{s,w}^{k+1} = \frac{e^{\Psi(\alpha_{s,w} + n_{s,w}^{k+1})}}{e^{\Psi(\sum_{w' \in V} \alpha_{s,w'} + n_{s,w'}^{k+1})}} \quad (7)$$

Where $n_{s,w}^t$ is the number of times a word w appears in tweets with sentiment s at iteration k and Ψ is the digamma function $\Psi(x) = \frac{\Gamma'(x)}{\Gamma(x)}$. For the algorithm we also need a variable $\vec{\phi}^k$, where $\vec{\phi}_t^k$ is the sentiment of tweet t at iteration k.

The algorithm is executed as follows:

1. Both $\vec{\theta}^0$ and $\vec{\phi}^0$ are initialized randomly.
2. Update our estimates of the sentiment of each tweet $\vec{\phi}_t^{k+1}$, based on the current word distribution $\vec{\theta}^k$.
3. Count the number of times each word w occurs in a tweet with sentiment s and assign this value to $n_{s,w}^{k+1}$ (Equation 6).

4. Update our estimate of the word distributions $\vec{\theta}^{k+1}$, based on our new values of $\vec{n}^{k+1}$ (Equation 7).

5. Repeat steps 2-4 until convergence.

At completion, our posterior belief distribution over θ_s is $\text{Dir}(\vec{\alpha}'_s)$ where $\vec{\alpha}'_s = \vec{\alpha}_s + \vec{n}_s$.

3.4 Prediction

Once we have trained the model, the log probability of a tweet being generated given a particular sentiment can be computed as:

$$\log\left(p(t|\vec{\alpha}'_s)\right) = \sum_{w\in V}\sum_{i=0}^{n_{t,w}} \log(i+\alpha'_{s,w}) - \sum_{i=0}^{|t|}\log\left(i+\sum_{j=1}^{|V|}\alpha'_{s,j}\right) \tag{8}$$

Where $n_{t,w}$ is the number of occurrences of w in t. This is the log probability of an observation under a Dirichlet-Multinomial model. Using Bayes' rule and an empirical estimate of $p(s)$ this can be used to compute $p(s|t)$, which is our sentiment prediction.

$$p(s|t) = \frac{e^{\log(p(t|\vec{\alpha}'_s))}}{\sum_{s'\in S} e^{\log(p(t|\vec{\alpha}'_{s'}))}} p(s) \tag{9}$$

Importantly, this can be calculated efficiently and is suitable for prediction in an online setting.

4. MODELLING GEOGRAPHIC REGIONS

If we now want to model separate sentiment distributions for different geographic regions, a first option would be to merely consider tweets from each region independently. However, by doing this we are disregarding any potential information we can learn about regions' word distributions from neighbouring regions. Especially in the cases where a region has very few tweets, it makes sense to use data from neighbouring regions to improve our estimates of the word distributions. We propose a simple, tractable extension to separately model word distributions while maintaining comparable classification accuracy when data is sparse.

We consider a hierarchy of regions R. We have a root node, r_0, which contains all tweets. Below this, the child nodes of r_0 partition the tweets into disjoint subsets, with one belonging to each child node. These new sub-regions can in turn be subdivided into smaller subregions. For example, one possible hierarchy is the geo-political regions of the world. In this case, the root node is "The World". This has one child for each country of the world. Each country then has a child for each of its states/provinces. We define $N(r)$ as the neighbours of region r (those that share the same parent region), $P(r)$ as the parent of region r and T_r as the tweets contained in region r.

We train our original model independently for each region to obtain a distribution over words, $p(\vec{\theta}_{s,r}|T_r)\ \forall r$. This means we have separate word distributions for each region, parametrized by $\vec{\alpha}_{s,r}$. We now want to define a prior $p(\vec{\theta}_{s,r}|T_{\neg r})$, which is our belief distribution over $\vec{\theta}_s$ in region r, given tweets observed in other regions.

We design a prior that encodes a broad assumption that there will be similarities between close regions, that each region should have a limited effect on every other region and which is very simple to compute. We define this prior as follows:

$$P(\vec{\theta}_{s,r}|T_{\neg r}) = Dir(\vec{\beta}_{s,r}) \tag{10}$$

$$\vec{\beta}_{s,r} = \omega_0 \frac{1}{|N(r)|}\sum_{r'\in N(r)} \frac{f(\sum_{i=1}^{|V|}\alpha'_{s,r',i})}{\sum_{i=1}^{|V|}\alpha'_{s,r',i}}\vec{\alpha}'_{s,r'} + (1-\omega_0)\vec{\beta}_{s,P(r)} \tag{11}$$

$$f(x) = A\left(1 - e^{-x(\log(A)-(\log(A-1)))}\right) \tag{12}$$

$f(x)$ is a thresholding function which is monotonically increasing and asymptotes at A. This limits how much a given region can influence the prior of the region we are considering. w_0 is a tunable parameter that indicates how much more influence closer regions have than ones further away in the hierarchy.

Now we can train a new model that is the same as the model that considered the regions independently, only now we replace the priors with those $p(\vec{\theta}_{s,r}|T_{\neg r})$ that we just calculated.

5. DATA PIPELINE

Data acquisition is a non-trivial step, involving filtering of users, filtering of tweets, parsing of tweets and determining the region of tweets.

5.1 Obtaining data

Twitter provides two APIs to access information about tweets; the Search API and the Streaming API. When selecting an API to use, there are several important characteristics to consider.

APIs

The Search API is intended to provide the ability to perform specific, low throughput queries. One can refine by user, content, geographic location (defined by a GPS co-ordinate and radius) and date. Importantly, only tweets from the preceding 5 days can be searched and queries are limited to approximately 10 per minute at the time of writing.

The Streaming API is designed to allow access to a live stream of tweets, as they occur. One can refine by user, content and area (defined by a rectangular bounding box). However, the user/content filter and the geographic filter are performed with a logical "OR". This means that a search for

tweets "Charlie" in San Francisco will return all tweets from San Francisco and all tweets containing "Charlie". We are interested in predicting on the streaming API.

To collect a dataset of substantial size quickly we use the Streaming API, though the Search API is also used to compile testing datasets.

Data format
When querying either API, we are returned a list of status updates. Each status contains fields describing the tweet. The fields we are concerned with are shown in Table 1.

Some other information is returned, mainly relating to whether a tweet is in response to another tweet or user. Through another API call, more information can be requested about a particular user. However, the search API's rate limit is too low to query every user we see in the stream. Therefore for prediction on the stream, we cannot rely on having access to this extra data.

5.2 Filtering Tweets

Spam accounts
While there are millions of legitimate users on Twitter, there are also a large number of spam accounts, corporate accounts and bots (most notably weather bots), that have a very negative effect on standard language models. This is due both to the volume of tweets they generate, which is generally much higher than legitimate users, and the fact that many of the tweets are automatically generated or templated. This leads to many words artificially appearing together many times, which breaks the assumptions of language models based on co-occurrences such as ours.

Identification of spam accounts is a separate research question in itself [7][17][18]. However, as noted before, in the online setting we are severely limited in the number of features we have for a given user. We don't have access to their full feed or social graph. The only informative features are the number of followers that a user has, as well as the number that they are following. Research has shown that users with more than 1000 followers or who follow more than 1000 users have a drastically different usage pattern to normal users [11]. This may indicates that the account is spam, a celebrity, a corporation or other accounts we are not interested in. Therefore as a simple rule we filter out all these users.

Duplicate tweets
An important aspect of Twitter that breaks the assumption of most language models are retweets. Retweets are when a user rebroadcasts a tweet that was tweeted by another user. This is usually of the form: "RT @[Original User] [Original Text]". While the fact that a user has retweeted another tweet is likely to contain meaningful sentiment information, the fact that the words appear verbatim strongly breaks the iid assumption of common language models. While retweeting is an important part of Twitter culture, we choose simply to remove the retweets at sampling time.

5.3 Tweet Processing

Once we have retrieved our tweets from the API, we convert them from the string representation to a feature representation. For our representation we choose a bag-of-words model.

Tokenization
Tweets are not written like other text documents. The 140 character limit and informal setting result in tweets that heavily use slang, emoticons, spelling and punctuation that would not be found in traditional text documents. Because of this, we need to perform custom pre-processing and tokenization of the tweets. We use a tokenizer specifically designed for Twitter [14]. This much better captures the use of punctuation, emoticons and words on Twitter. Additional steps were to conflate more than three consecutive occurrences of the same letter down to two letters. There are very few meaningful terms that contain more than three of the same character in a row, but it is very common on Twitter to repeat letters for emphasis.

5.4 Determining location in geo-hierarchy

As already mentioned in the previous section, the geographic information may come as either the GPS co-ordinates that the tweets was tweeted from, or the name of a location with a bounding polygon for that region. When creating a system to model regional variation, we would like to use the geopolitical region of the tweet. Fortunately, there is an API provided by GeoNames.org that converts GPS co-ordinates to a code defined in ISO 3166-2. ISO 3166-2 are two region codes separated by a dash that are the country code and a sub region (state/province) code.

This service is also rate-limited such that we cannot query at the rate of the streaming API. However, if we receive a tweet's location information as a name and bounding region pair, we can calculate a representative GPS co-ordinate within the region, query the GeoNames API and cache the result against the location name. As a representative point we use the centroid. While this is not assured to be inside the bounding polygon, as the polygons can be non-convex, in practice this works for almost all points.

6. RESULTS

To evaluate our model we firstly compare the performance of our model without geo-information against Naive Bayes and then test the effect of including geo-information in our model. In our comparison against Naive Bayes we test using a sample of 50,000 tweets with a happy or sad emoticon. 60% of these contain a happy emoticon, while 40% contain a sad emoticon. No tweet contains both a happy and sad emoticon. In our geo model comparison, we test on a different sample of 20,000 tweets with happy or sad emoticon that also include geo information. For illustrative purposes, we also include an example list of high probability words for different regions and sentiments (Table 2) and maps (Figures 3,4) showing relative rates of happy and sad tweets for different regions of the world.

6.1 Comparison with Naive Bayes

Firstly, we test the accuracy of the proposed tweet model without geographic information. As a baseline comparison we use a Naive Bayes classifier trained on the same features.

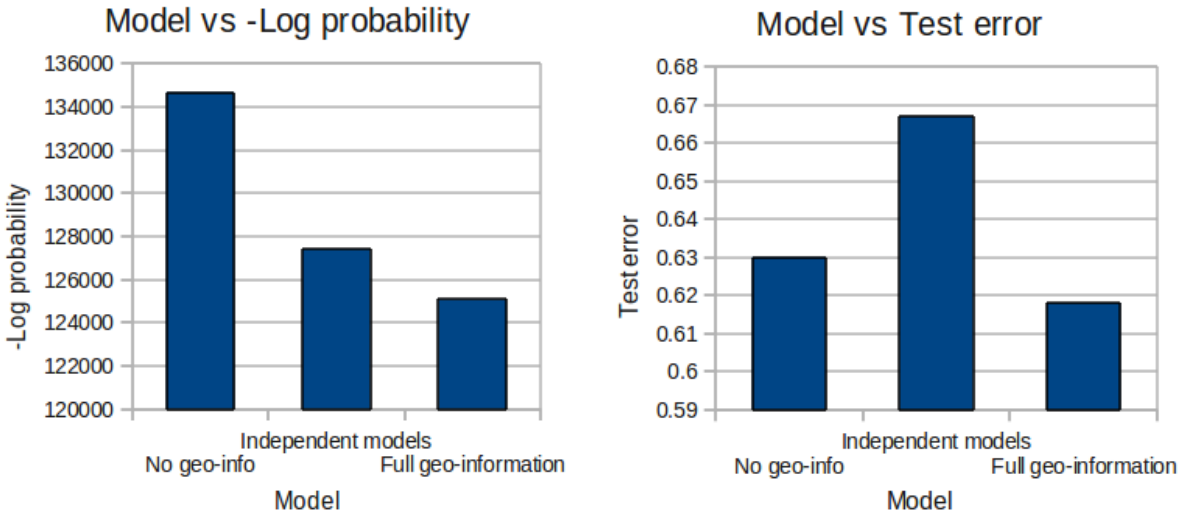
text	The body text of the tweet
id	The unique id of the tweet
author	The author of the tweet
retweeted	A boolean variable indicating if this is a retweet
coordinates	Co-ordinates the tweet originated from (if available)
followers	The number of followers the user has
following	The number of users that the user follows
place	Geo information, (place name, bounding box)
created at	Timestamp the tweet was created at

Table 1: Fields returned in Twitter API

(a) Effect of removing labels from training data

(b) Effect of altering percent train/test split

Figure 1: Comparison of our model with Naive Bayes



(a) -Log probability of test data under different models (smaller is better)

(b) Test error under different models

Figure 2: Comparison of our model without geo information, with independant partitioning and using neighbouring geo-information

Region	Sentiment	Top words
World	Happy	amore amour liebe happy :-) kasih amores bahagia love feliz :) makasih follback seguindo oooi good follow lovely xxx selamat hey s/o terima cheers thx
	Sad	sedih :-(sad triste :(dor saudade droga fome cade doendo aaa saudades garganta mimimi aff morreu odeio gripe perdi merda aaah :'(
UK	Happy	amour feliz kasih :-) happy love :) thanks birthday thank follow good welcome luck great nice morning amazing hello lovely xxx best hey awesome cheers
	Sad	:-(sad triste :(nooo booo </3 poor miss *cries* gutted afford :'(poorly hate dean ugh fml urghhh stressed headache *sigh* canada prayforkatie richards rip whyyy horrible revision *hugs*

Table 2: List of highest probability words for each sentiment at different regions

For the first test we take 50% of the data for training and from a fraction of these tweets, remove the emoticons. These represent emotive tweets that we would encounter on Twitter that do not have an emoticon. The models are trained on this data and then the classification accuracy is assessed on the remaining 50% of the data, with all emoticons removed. For the second test we remove emoticons from 90% of the training data, but alter the fraction of data used for training.

In Figure 1 we see that as less labelled training examples are supplied, our model further outperforms Naive Bayes. At very low label density this is an improvement in classification accuracy of over 10%. We see that while increased test data results in increased model performance in both cases, there is little difference between our model and Naive Bayes.

6.2 Effect of incorporating geo-information

Secondly, we look at the effect of incorporating geo-information into our model. The three models we test are our model without geo-information, independent instances of our model for each region and incorporating neighbouring region information as outlined in Section 4. We compare them based on classification error and the negative log probability of the test dataset under the model. Both of these are averaged over 10 different test splits. The negative log probability gives us an idea of how well the data is modelled, as a lower value of this means that the model was less "surprised" by the data.

We can see in Figure 2 that by training independent models for different regions, we lose classification accuracy, even though we are modelling the data better, as shown by the lower negative log probability. However, by incorporating neighbouring region information, we recover our classification accuracy and achieve an even further reduction in negative log probability.

7. CONCLUSIONS

In conclusion, we propose a probabilistic model for sentiment classification of short social network statuses. We show that this outperforms Naive Bayes on simple word features when there are additional unlabelled training examples. We also propose a simple extension for modelling distributions across multiple geographic regions, without sacrificing classification accuracy. Finally, using this extension, we have described an implemented a data pipeline and efficient algorithm for performing online geographic sentiment prediction of Twitter statuses.

8. REFERENCES

[1] S. Asur and B. Huberman. Predicting the future with social media. In *2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, pages 492–499. IEEE, 2010.

[2] L. Barbosa and J. Feng. Robust sentiment detection on Twitter from biased and noisy data. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 36–44. Association for Computational Linguistics, August 2010.

[3] M. Beal and Z. Ghahramani. The variational Bayesian EM algorithm for incomplete data: with application to scoring graphical model structures. *Bayesian statistics 7:*, 2003.

[4] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3(4-5):993–1022, May 2003.

[5] J. Bollen, H. Mao, and X.-j. Zeng. Twitter mood predicts the stock market. *Journal of Computational Science*, pages 1–8, 2011.

[6] M. M. Bradley and P. J. Lang. Affective norms for English words (ANEW): Stimuli, instruction manual and affective ratings. Technical report, The Center for Research in Psychophysiology, University of Florida.

[7] Z. Chu, S. Gianvecchio, H. Wang, and S. Jajodia. Who is tweeting on Twitter: human, bot, or cyborg? In *Proceedings of the 26th Annual Computer Security Applications Conference*, pages 21–30. ACM, 2010.

[8] J. Eisenstein, B. O'Connor, N. Smith, and E. Xing. A latent variable model for geographic lexical variation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1277–1287. Association for Computational Linguistics, 2010.

[9] Z. Ghahramani and M. Beal. Variational inference for Bayesian mixtures of factor analysers. *Advances in neural information processing systems*, 12:449–455, 2000.

[10] A. Go, R. Bhayani, and L. Huang. Twitter Sentiment Classification using Distant Supervision. Technical report, 2009.

[11] H. Kwak, C. Lee, H. Park, and S. Moon. What is Twitter, a Social Network or a News Media? In *WWW '10: Proceedings of the 19th international conference on World wide web*, pages 591—-600, 2010.

[12] T. Lake. Twitter Sentiment Analysis. Technical report, Western Michigan University, Apr. 2011.

[13] B. O'Connor, R. Balasubramanyan, B. Routledge, and

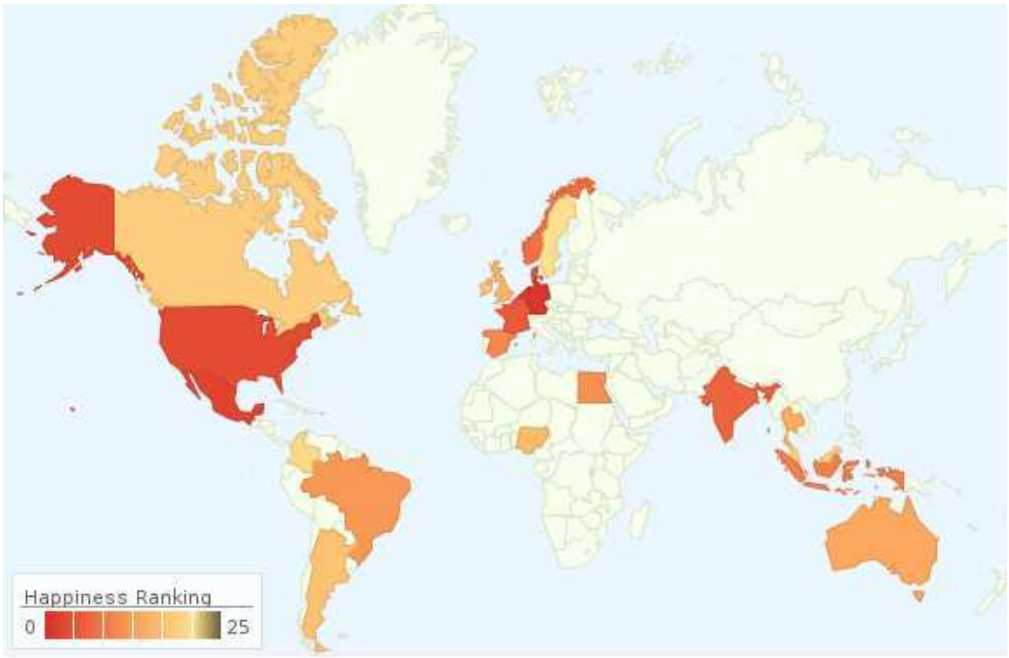


Figure 3: Map of the world with countries ordered by relative rate of happy to sad tweets

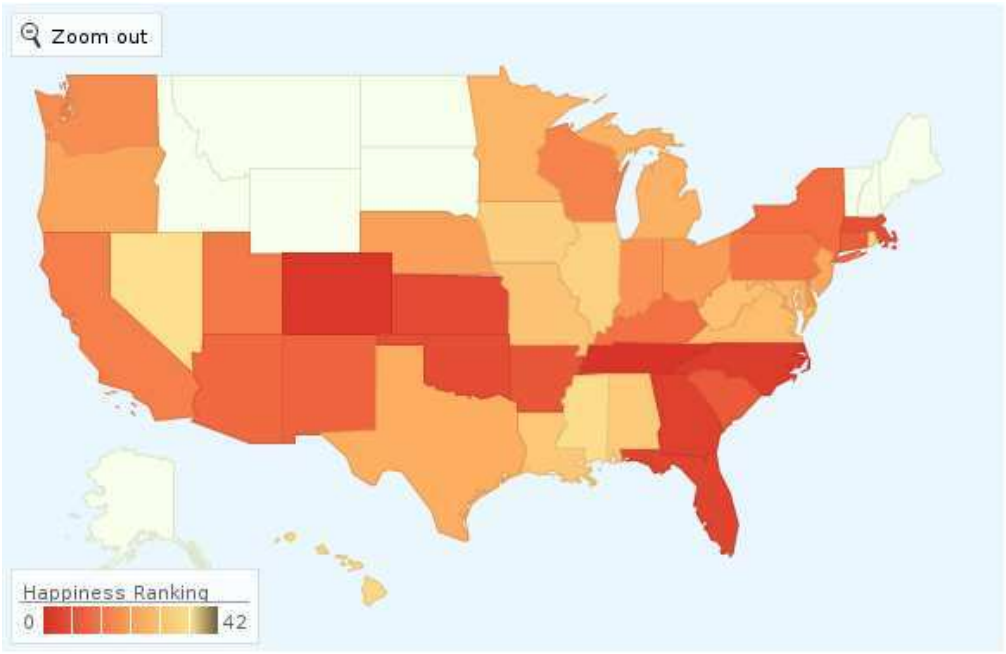


Figure 4: Map of the US with countries ordered by relative rate of happy to sad tweets

N. A. Smith. From tweets to polls: Linking text sentiment to public opinion time series. In *Fourth International AAAI Conference on Weblogs and Social Media*, May 2010.

[14] B. O'Connor, M. Krieger, and D. Ahn. TweetMotif: Exploratory search and topic summarization for twitter. *Proceedings of ICWSM*, pages 2–3, May 2010.

[15] A. Pak and P. Paroubek. Twitter as a corpus for sentiment analysis and opinion mining. *Proceedings of LREC 2010*, pages 1320–1326, 2010.

[16] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics, 2002.

[17] G. Stringhini, C. Kruegel, and G. Vigna. Detecting spammers on social networks. In *Proceedings of the 26th Annual Computer Security Applications Conference*, pages 1–9. ACM, 2010.

[18] A. Wang. Don't follow me: Spam detection in Twitter. In *Security and Cryptography (SECRYPT), Proceedings of the 2010 International Conference on*, pages 1–10. IEEE, 2010.

[19] T. Wilson, J. Wiebe, and P. Hoffmann. Recognizing Contextual Polarity: An Exploration of Features for Phrase-Level Sentiment Analysis. *Computational Linguistics*, 35(3):399–433, Sept. 2009.

An Algorithm and Analysis of Social Topologies from Email and Photo Tags

T. J. Purtell Diana MacLean Seng Keat Teh
Sudheendra Hangal Monica S. Lam Jeffrey Heer

Computer Science Department
Stanford University
Stanford, CA 94305
{tpurtell, malcdi, skteh, hangal, lam, jheer}@cs.stanford.edu

ABSTRACT

As peoples' participation in social media increases, online social identities accumulate contacts and data. We need a mechanism for creating a succinct but contextually rich representation of a person's "social landscape" that would facilitate activities such as browsing personal social media feeds, or sharing data with nuanced social groups.

We formulate the social topology extraction problem as the compression of a group-tagged data set in which each group has a significance value, into a set containing a smaller number of overlapping and nested groups that best represent the value of the initial data set. We present four variants of a greedy algorithm that constructs a user's social topology based on egocentric, group communication data. We analyze our algorithm variants on about 2,000 personal email accounts and 1,100 tagged Facebook photograph collections. We find that our algorithm variants produce different topologies suitable for different purposes.

We show that our algorithm can capture 80% of the input data set value with 20% and 42% of the number of input groups for email and photographs respectively. Using edit distance as an objective metric, we also show that our algorithm outperforms results generated by Newman's modularity-based clustering algorithm. We conclude that our algorithm is appropriately designed to find significant groups of friends from social contact data.

1. INTRODUCTION

While millions of users have accumulated large lists of "friends" in online social networks, managing these flat lists is challenging. A natural organizing principle is to assign friends to different categories that can then be used for targeted sharing and filtering of social content. However, existing tools such as Facebook friends lists or Gmail contact groups require users to create these lists from scratch, making this a tedious task that is not done by most users [20]. To address this problem, we present a novel algorithm for compressing a user's communication data into a compact set of relevant groups that may be useful for the aforementioned tasks.

The 5th SNA-KDD Workshop '11 (SNA-KDD'11), August 21, 2011, San Diego, CA USA.

Figure 1: An example of a social topology.

1.1 Social Topology

MacLean et al. have proposed the concept of a *social topology* – the structure and content of a person's social affiliations, comprising a set of overlapping and nested groups – as a first-class structure for facilitating social-based tasks such as data sharing or digital archive browsing [13]. We exploit the observation that a user's social topology is captured implicitly in routine communications, photographs, and others forms of personal data. In this paper, we present a novel algorithm for generating a social topology from a user's grouping data, assuming a constraint on the size of a social topology. We define group communications data as corpora in which items may be tagged with more than one social identity; for example, email, tagged photograph collections, and co-location data.

Figure 1 shows several defining properties of a social topology. First, any individual in the topology may appear in several groups. This models people who play several roles in the user's life, such as being both a colleague and a friend. Second, it may contain *manufactured* groups – group of people who never occur together in a single item in the original data set. Consider a university lab whose membership changes annually: a "core" group, such as a faculty team, might persist in the group over time, but never appear uniquely in a photograph collection. Third, social topology groups may

be *nested*. This captures specific subgroups within a supergroup, such as siblings within a family. Finally, a "group" may consist of just a single individual who is sufficiently important.

We formulate the problem of deriving a social topology as follows: given a data set d of group communications data, a value function v which measures the significance of a group with respect to d, and a budget of b groups, find b groups whose aggregate value is maximized. We derive these groups only from the data that is directly visible to the user, making these groups ego-centric.

1.2 Contributions

The contributions of this paper include:

- A greedy algorithm for constructing social topologies from group communications data. The algorithms make different trade-offs and can be tuned based on the target application. Our algorithm is incorporated in a Facebook application called GroupGenie[1].
- A validation and comparison of our algorithm using two data sets: a collection of 1,995 personal email archives containing over 24 million sent email messages and a set of 286,038 tagged photos from 1,099 Facebook users.
- An evaluation and comparison of social topologies constructed from these data sets. The evaluation includes a comparison with Newman's clustering algorithm using edit distances as an information-theoretic metric.

Source code for the algorithm is also publically available[2].

2. RELATED WORK

There is a substantial body of work in analysis of social data, both for global (e.g., [9, 11]) and ego-centric (e.g., [4, 7, 14]) networks. Below, we discuss and contrast prior work with our approach.

Clustering algorithms aim to elicit communities from a graph structure. Traditional algorithms based on hierarchical agglomerative clustering *partition* the input graph, disallowing node overlap between clusters [3, 17]. We find this approach unsuitable for our purposes, as one person can adopt several social roles simultaneously.

Palla et al. present an algorithm that discovers overlapping communities in global, unweighted networks [18]. Communities are generated in a bottom-up fashion from k-cliques. Taking a different approach, Banerjee et al. introduce "model based clustering", a probabilistic graphical model for inferring overlapping clustering [2]. Huberman et al. extract overlapping social clusters by running an edge betweenness clustering algorithm several times, starting from a network where an unweighted edge exists between 2 people if 5 or more messages were exchanged between them [23]. Lancichinetti et al. present another method for detecting overlapping and hierarchical structure in complex networks [10].

There are three major differences between our work and the above algorithms. First, these algorithms make the assumption that the global structure of the network is available. Second, many of them are evaluated on networks formed by publicly available information, while we evaluate our algorithm on personal data, where there may be different patterns of group formation. Third, the input model of the graph is reduced to edges between individuals, ignoring the fact that the input data was grouped in the first place.

Visualization and interface techniques such as ContactMap [24], Vizster [8] and LinkedIn InMaps [12] help users view and organize their social networks. Previously published work by MacLean et al. describes an algorithm to derive overlapping and hierarchical groups, and an interface to edit those groups [13]. This algorithm required the use of several parameter settings and was evaluated in a smaller study involving email data sets of 19 users; moreover it does not seamlessly handle individuals. In contrast, the work reported in this paper presents an algorithm with better accuracy, and has been evaluated on a larger scale on multiple data sets.

Association rule mining is a technique for finding related item sets in a corpus, given a specific seed [1]. Roth et al. present a group-finding algorithm for Gmail in which the goal is to complete the group as accurately as possible given an initial seed [19]. Like us, they assume that communications reflect *implicit* social structure, and use communication frequency as a proxy for tie strength. They develop an *interactions rank* metric that gives an ordering over unique recipient groups, allocating points according to communication frequency, recency, and direction. However, seed-based approaches are generally inadequate for the purposes of helping users construct a social topology; for example, Gmail users cannot access the set of probable groups or use them for other purposes. As a result, the algorithm does not directly create a summary of the input groups.

Graph summarization techniques are often applied to the problem of web graph compression. A small portion of graph summarization research focuses specifically on reducing the size and complexity of network data. Tian et al. present two approaches: SNAP, for lossless compression and k-SNAP, for lossy compression [22]. Taking an information-theoretic approach, Navlakha et al. employ the minimum description length (MDL) principle to produce a graph summary and a list of "corrections", allowing for both compression and perfect reconstruction of the input graph. The graph may be optionally reduced if lossy compression can be permitted [15]. The same authors employ this method to obtain rich but manageable summaries of protein interaction networks [16].

3. ALGORITHM

Our goal is to derive a user's social topology, consisting of potentially overlapping and nested groups of friends, from a corpus of a user's group communication. Our algorithm is parameterized to find the most significant *given* number of groups.

3.1 Problem Statement

We define a social topology to be a set of unique, potentially overlapping and nested groups, each of which has some *value*, and each of which is comprised of members drawn from the user's global set of friends. Permitting nested groups lends increased granularity to the topology, while permitting overlapping groups allows us to represent people who play multiple roles in the subject's life. Intuitively, the value of a group reflects the proportion of information that the user chooses to share with it, and we consider groups with a higher information share to be more important than others. We generate social topologies from a single user's *ego-centric*

[1]http://mobisocial.stanford.edu/groupgenie
[2]https://github.com/mobisocial/groupgenie-algo/

grouping data such as email records or tagged Facebook photographs.

Ego-centric group communication datasets already contain a natural social topology: the unique groups that occur together on items in the data set. Each of these groups can be assigned some appropriate valuation. For example for a user's collection of sent email, the natural social topology would be the unique recipient sets in the data set, and the value of each recipient set might be a function of its size and the number of messages on which it appears. Therefore our task of social topology construction is a task of *compression*, in which we want to reduce the natural social topology into a manageable size, while maximizing its value. We may need to combine groups in various ways, as well as drop groups from the topology altogether, if needed. Our problem formulation requires that each group in the original social topology be represented by at most one group in the compressed topology.

Depending on the objective, there are different trade-offs in generating a social topology. For example, we may wish to create a social topology that includes mostly *core* persons from different facets of our lives. Alternatively, we may wish to create a social topology containing as many related people as possible. In order to accommodate such diverse objectives, we introduce the notion of a *value function* that evaluates the value of each group in the generated social topology based on its mapping from the original one.

The social topology compression problem is thus defined as follows. Given

- a set of friends F,
- a natural social topology S consisting of unique groups $g \subset F$, where the value function $v_0(g)$ denotes the significance of the group g,
- a size b which is our *budget*, or number of groups required in the final topology,
- a value function $v(g, r)$, where g is the representative for a set of groups $r \subseteq S$.

find a social topology S' and a representation map R, mapping each $g \in S'$ to a non-overlapping set in S, such that $\sum_{g \in S'} v(g, R(g))$ is maximized.

3.2 The Sharing Value Metric

Our value function is based on a model of information sharing and over-sharing. Intuitively, if group g in the original social topology maps to group g' in the final social topology, the value is high if g' has the same members and low if g' has many additional members. The ratio between the number of common elements to the size of g determines the fraction of the positive contribution of g's value to g'. For each friend in g' and not in g, information from g is over-shared. Since different uses of social topologies may desire different over-sharing penalties, we allow the algorithm be parameterized with a penalty weighting function $w(f, g)$ that determines the penalty to be applied to each unit of over-sharing with friend f not in group g. We thus define a value function based on information sharing as

$$v(g', r) = \sum_{g \in r} \frac{v_0(g)}{|g|} \left(|g' \cap g| - \sum_{f \in (g' - g)} w(f, g) \right),$$

where $w(f, g)$ is the over-sharing penalty to be applied to friend f for group g.

One possible penalty weighting function is a simple constant, i.e.,

$$w(f, g) = C$$

If C is 0, there is no penalty for over-sharing; if C is 1, every person that a data item is overshared with costs as much as the value contributed by a person who was in the original group that the item was shared with.

A more sophisticated approach is to use a weighting function that depends on the relationship between the original group and the friends an item was over-shared with. Friends who are not in the original group g, but participate with members in group g in other groups, should have a lower sharing penalty. Let $P(\overline{f}|f')$ denote the conditional probability of not finding f in groups containing f'. Then we can define a function for the over-sharing penalty weight as

$$w(f, g) = \frac{1}{|g - \{f\}|} \sum_{f' \in (g - \{f\})} P(\overline{f}|f')$$

3.3 A Greedy Algorithm

We define a set of permissible actions, called *moves*, that may be taken on groups in a social topology. All moves reduce the social topology size by 1 and reduce the value of the topology according to its *error function*. Starting with the natural social topology, our algorithm greedily picks the move that maximizes the value of the resulting social topology until the topology is reduced to the desired size b.

We define the initial representation mapping R to simply map each group g to itself; if g is a group in the original topology, $v(g, \{g\}) = v_0(g)$. The moves and their error functions are defined below.

- DISCARD. Discard a group from the topology, thus losing the group's entire value.

$$E_{\text{DISCARD}}(g, r) = v(g, r)$$

- MERGE. Merge two groups to create a union that inherits the combined value, appropriately penalized to account for their membership mismatch. The over-sharing penalty built into the value metric ensures that the most closely related groups have the lowest error.

$$E_{\text{MERGE}}(g_1, r_1, g_2, r_2) = v(g_1, r_1) + v(g_2, r_2) - v(g_1 \cup g_2, r_1 \cup r_2)$$

- INTERSECT. Intersect two groups to capture the importance of a shared subset.

$$E_{\text{INTERSECT}}(g_1, r_1, g_2, r_2) = v(g_1, r_1) + v(g_2, r_2) - v(g_1 \cap g_2, r_1 \cup r_2)$$

- TRANSFER. Transfer the representation of a second group to the first group; the second group is discarded, but its value is partially transferred to the first, taking into account the over-sharing penalty.

$$E_{\text{TRANSFER}}(g_1, r_1, g_2, r_2) = v(g_1, r_1) + v(g_2, r_2) - v(g_1, r_1 \cup r_2)$$

3.4 An Approximate Algorithm

In the algorithm above, each group's value is defined in terms of the values of the original groups they represent. To simplify the algorithm, we adopt the model where each

member in a derived group contributes equally to the group's value. We can thus approximate the value of a derived group with a single quantity and compute the error term for each move based on the approximate value of its operands. The approximate value function is defined as

$$\overline{v}(g) = \begin{cases} v_0(g), & \text{if } g \in S, \\ \overline{v}(g_1) + \overline{v}(g_2) - E_m(g_1, g_2), & \text{if } g = m(g_1, g_2) \end{cases}$$

where $m(g_1, g_2)$ is the result of applying move m to g_1 and g_2. (Unary moves like discard are similarly defined.) Since in this model, the value of a group is considered uniformly distributed across all its members, the over-sharing error simply depends on the ratio of additional people getting the information to the size of the group. The error functions are thus analogously defined as:

$$E_{\text{DISCARD}}(g) = \overline{v}(g)$$

$$E_{\text{MERGE}}(g_1, g_2) = \sum_{f \in g_2 - g_1} \frac{\overline{v}(g_1)}{|g_1|} w(f, g_1 \cup g_2) + \sum_{f \in g_1 - g_2} \frac{\overline{v}(g_2)}{|g_2|} w(f, g_1 \cup g_2)$$

$$E_{\text{INTERSECT}}(g_1, g_2) = \sum_{f \in g_1 - g_2} \frac{\overline{v}(g_1)}{|g_1|} + \sum_{f \in g_2 - g_1} \frac{\overline{v}(g_2)}{|g_2|}$$

$$E_{\text{TRANSFER}}(g_1, g_2) = \sum_{f \in g_1 - g_2} \frac{\overline{v}(g_2)}{|g_2|} w(f, g_1 \cup g_2) + \sum_{f \in g_2 - g_1} \frac{\overline{v}(g_2)}{|g_2|}$$

The experimental results presented in this paper are based on this approximate algorithm, as summarized in Algorithm 1.

Algorithm 1 compressTopology(S)

Input Initial topology $S = \{g_i\}$,
a set of unique groups and values $\overline{v}(g_i)$.
Input A budget b, the size of the final topology.
Output S, the final topology

while $|S| > b$ **do**
  $g^* \leftarrow m(g_1, g_2)$ where m is the lowest loss move $\forall g \in S$
  $\overline{v}(g^*) \leftarrow \overline{v}(g_1) + \overline{v}(g_2) - E_m(g_1, g_2)$
  $S \leftarrow S + g^* - g_1 - g_2$
end while

4. EXPERIMENTAL EVALUATION

We have evaluated different versions of our algorithm on 2 types of datasets, one using personal email archives, and another using Facebook photo tags.

4.1 Email

Our email dataset is comprised of email headers from 1,995 users' personal email archives, totaling over 24 million sent email messages. The dataset, provided by Xobni Inc., was collected from a subset of users of their Xobni Cloud service. The data we received was fully anonymized; all personally-identifiable information had been removed. Most of these users connected to Xobni via Outlook, so we estimate that much of the email activity may be work related. Figure 2 outlines statistical properties of the corpus. We restrict our algorithm input to *sent* email only, noting that this is a more accurate signal for social importance, as sending an email incurs a cost on the user, whereas receiving one does not [13]. This also has the advantage of excluding spammers and advertisers. We see some startling anomalies in the data set, such as an individual who sent as many as 160,000 messages, and a message addressed to 2,826 recipients! Note that the majority of messages are sent to only one person.

	Messages	People	Groups	Group Size
Lower Q.	2038	329	373	1
Median	6640	738	1104	1
Upper Q.	14684	1422	2451	1
Max	159697	20813	24306	2825
Mean	11521	1109.5	1814.9	1.5
Std Dev	15205.1	1328.6	2231.4	2.3
Total	24228571	2213486	3816668	35781399

Figure 2: Summary of the 1,995-person email data set.

	Photos	People	Groups	Group Size
Lower Q.	31	28	19	1
Median	106	62	54	2
Upper Q.	325	130	142	3
Max	3062	594	1050	111
Mean	260.3	90.9	109.6	2.4
Std Dev	392.2	88.8	143.7	2.8
Total	286038	99910	120457	682126

Figure 3: Summary of the 1,099-person photo data set.

4.2 Tagged Photos

Just as emails capture co-occurrence of recipients on mails, tagged photographs capture physical co-occurrence of subjects. Given the fact that photo sharing is one of the most popular forms of online social activity, tagged photographs are an excellent source of social topology data. To evaluate our algorithm on tagged photos, we have developed GroupGenie, a Facebook application that allows Facebook users to infer their social topology from their tagged photo data.

GroupGenie users have found the social groupings suggested to them by our algorithm helpful for both data sharing and communication tasks, and for a certain degree of personal self-reflection. An informal pilot study of about 30 users aged 17-19 found that the groups suggested to them were good enough, with a few minor edits, to publish to their profile pages as Facebook Featured Friends [6]. Some found it useful to use their groups in Facebook Chat [5] to do group-wide chats.

At the time of this writing, 1,099 Facebook users have used GroupGenie. Most of these users discovered GroupGenie through friends, and from a press article about an earlier version of this work [21], suggesting strong interest among Facebook users in tools to help them create groups.

Figure 3 provides summary statistics of the tagged photograph corpus. Note that the owner of the Facebook account, if present, is excluded from the input groups. There

are distinct differences between this data set and the email data set in terms of group density. In particular, the average group size in a photograph is 2.4, compared to the average number of recipients on an email, which is 1.5. Moreover, more than half of the photographs are tagged with at least 2 people excluding the user; in contrast, a majority of emails involve only one other person excluding the user. On the other hand, tagged photograph collections are typically smaller than email collections, presumably due to the larger effort required to take, upload and annotate photographs.

5. ANALYSIS OF EMAIL DATASET

We run our experiment on four variants of our algorithm:

DISCARD. Considers only discard moves. This straw-man version simply reports the top b initial valued groups for a given budget b.

MERGE. Considers discards and merges, with a simple fixed penalty weight of 0.5.

COND-MERGE. Considers discards and merges, with a conditional probability metric for sharing penalty.

COND-ALL. Considers all moves, with a conditional probability metric for sharing penalty.

We define the initial value, or significance, of each input group g as $v_0(g) = min(|g|, sizeThreshold) \times msgCount(g)$. Intuitively, this captures the proportion of the corpus represented by g. The parameter $sizeThreshold$ prevents large groups, which are often once-off mailing lists, from being awarded excessively large initial values. Empirically, we set $sizeThreshold = 20$.

5.1 Algorithm Illustration

To provide insight into our algorithm, we first present its behavior on one user's data. As shown in Figure 4, all algorithm variants capture a significant fraction of the value with a small percentage of groups, with DISCARD, COND-MERGE, MERGE, and COND-ALL in increasing order of value captured for a given topology size. DISCARD allows no over-sharing; its sharing penalty is effectively ∞. COND-MERGE allows sharing mainly among those who are already sharing other messages. Next is MERGE, with a fixed penalty weight of 0.5, the algorithm is allowed to perform more merging.

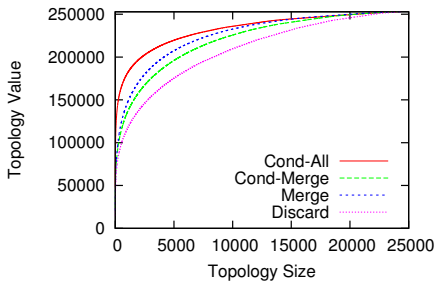


Figure 4: Social topologies for a representative data set.

COND-ALL has the highest compression ratio, though it actually discourages sharing in the final topology. Because the value of one group can be transferred to another with a sharing penalty, COND-ALL tends to identify the super individuals and groups that may play different roles in a user's interaction. Consider, for example, a secretary who is carbon-copied on all work-related emails. The secretary can amass a very large value as partial credit is transferred to him as low-frequency groups are dropped.

5.1.1 Aggregate Behavior

We glean additional insight about our algorithm's behavior by analyzing the frequency of the move types aggregated over the entire email data set. Figure 5 shows move frequency plotted against normalized algorithm progress.

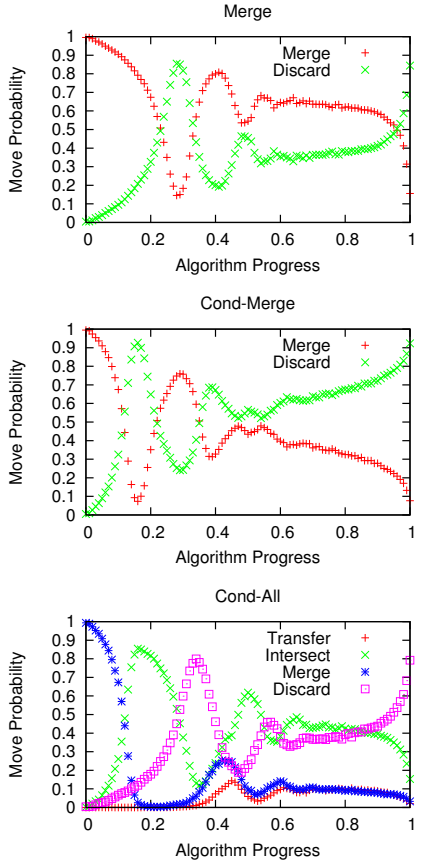


Figure 5: Algorithm behavior over the email corpus.

We see that in MERGE and COND-MERGE, there are distinctive alternating phases of merges and discards. The periodicity reduces over time with merges dominating at the beginning and discards dominating near the end. As the algorithm takes the move with the minimum value reduction, the periodicity results from the fact that there are many initial groups with values 1, 2, and so forth. Many discards of groups of value 1 kick in as the minimum drop in the algorithm reaches 1. Since the merged groups no longer have integral values, the choice between discards and merges become more irregular. Near the end of the algorithm, the remaining groups are distinct enough that merging them would incur a higher penalty than discarding them, thus we see many discards near the end. COND-MERGE is similar to MERGE, except that MERGE performs more unions since it has a lower sharing penalty.

COND-ALL has two more moves than MERGE: intersects

and transfers. Almost all the intersect moves occur between supersets and subsets. In such cases, intersects produce the same topology as discards of the larger group, but the smaller group now accumulates more value due to the transfer of value. Including this move favors the creation of smaller groups and helps identify the core people in each group. Similarly, transfer moves also create pressure to produce smaller groups, since values can be transferred from one group to another. Together, intersect and transfer moves reduce the number of merges.

5.2 Value Concentration

Figure 6 plots the median fraction of summary groups that capture a given fraction of value.

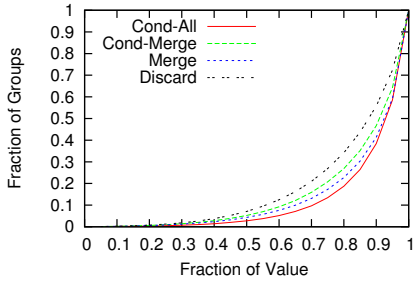


Figure 6: The values of social topologies obtained for the email corpus.

We see from the summary in Figure 7 that 50% of the value can be captured by 7% or less number of groups. If we are willing to tolerate some over-sharing, we can compress the social topology further. MERGE needs only 23% of the groups versus DISCARD's 34% to capture 80% of the value. COND-MERGE only supports merging of closely related friends, causing the need of slightly more groups. As discussed above, since COND-ALL allows the value of a group to be transferred to another, without having to include all members of the group, COND-ALL achieves the best value with the smallest number of groups. To reach 80%, COND-ALL needs a social topology whose size is less than 20% of the original.

	DISCARD	MERGE	COND-MERGE	COND-ALL
0.5	0.07	0.04	0.05	0.03
0.6	0.13	0.08	0.09	0.05
0.7	0.21	0.13	0.16	0.10
0.8	0.34	0.23	0.27	0.19

Figure 7: Fraction of groups needed to achieve a given fraction of the value.

5.3 Small Social Topologies

Since many users in our corpus have over 1,000 groups, even 10% of groups might be overwhelming for the user to review. How much value can be captured by a few tens of groups? Figure 8 shows the median of the values captured for the fixed size social topologies and Figure 9 tabulates the values for topologies with 10, 25, and 50 groups. We find that the top 10 groups capture 24-34% of the value and the top 50 groups capture 44-57%, depending on the algorithm variant used.

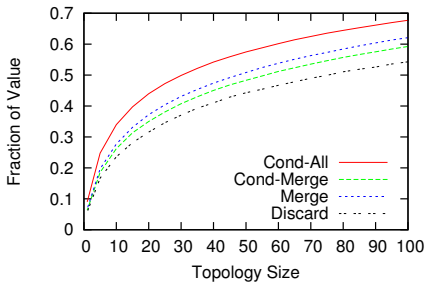


Figure 8: Values of small social topologies.

Our algorithm treats singletons the same as any other groups, allowing us to rank individuals uniformly against groups. However, certain applications may not need to be concerned with singletons. For example, a tool that helps users name groups only needs to show non-singleton groups, since individuals already have a name. We thus show the number of non-singletons in Figure 9 for reference. The majority of the top groups in email turn out, not surprisingly, to be singletons. As the allowance for over-sharing grows from COND-ALL, COND-MERGE, to MERGE, the number of non-singleton groups increases slightly. Thus for applications that work with only non-singletons, just 2-4 groups are needed to reach 24-34% of the value and 8-11 groups reach 35-47%.

	DISCARD	MERGE	COND-MERGE	COND-ALL
10	0.24 (2)	0.28 (4)	0.26 (3)	0.34 (3)
25	0.35 (8)	0.40 (11)	0.38 (10)	0.47 (8)
50	0.44 (21)	0.51 (25)	0.49 (24)	0.57 (18)

Figure 9: Values of social topologies with selected sizes. Non-singleton groups are shown in parentheses.

	DISCARD	MERGE	COND-MERGE	COND-ALL
Non-singleton	21	25	24	18
New groups	0	14	6	0
Group size	2.6	6.1	3.5	2.5
People	60	162	84	71
Roles/person	2.0	1.8	1.9	1.6

Figure 10: Properties of social topologies of size 50.

Which version of the algorithm should an application use? The different variants produce different topologies. Figure 10 shows additional properties of social topologies of size 50. It is clear from the figure that MERGE generates the largest social topology, followed by COND-MERGE. Perusal of the authors' own social topologies suggests that MERGE can create somewhat noisy groups consisting of people who are only peripherally related. Social topologies created by COND-MERGE are fairly coherent, and are the recommended choice for generating groups from social network data. On the other hand, COND-ALL is suitable for distilling key members of each group. We observe that no new groups are created for the COND-ALL case; since there is heavy traffic within the core groups, it is highly likely that there is at least one message sent to the entire core group.

5.4 Significant Groups

For applications without a fixed budget, it is useful to report to the user only the significant groups of his or her topology. Showing too many groups can bore the user, while showing too few might miss important groups. We can leverage our valuation framework to choose the appropriate number of groups to present. The average value of a group in the input data set serves as a baseline for the importance of a group. To identify groups that stand out above the average, we can simply display groups with value greater than one standard deviation above the average value of the input data. We can alter the algorithm to stop when the error for a move exceeds this threshold. As shown in Figure 11, the median of 11 groups was directly identifiable from the input data set (DISCARD); COND-ALL and COND-MERGE identify a median of 15 significant groups for the email data sets and MERGE identifies 14. These numbers appear to be quite reasonable in practice.

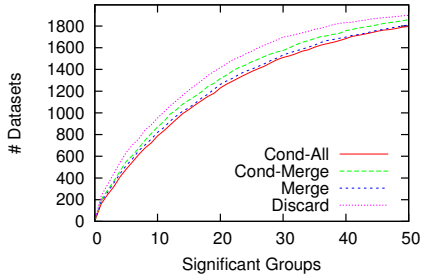


Figure 11: The cumulative distribution of the number of significant groups in the email corpus.

6. ANALYSIS OF PHOTOS

We analyzed the four variants of the greedy algorithm described in the previous section for Facebook photo tags. All plots shown represent the median observed in the data set.

6.1 Value Concentration

We observed the same overall trends with the photo data set as we saw in the previous section. In Figure 12, the fractional value curve climbs less steeply than in Figure 6, suggesting higher diversity in the photo data set compared to email. Figure 13 shows that COND-ALL requires only 15% of the groups to capture 50% of the value, and 42% to capture 80% of the value.

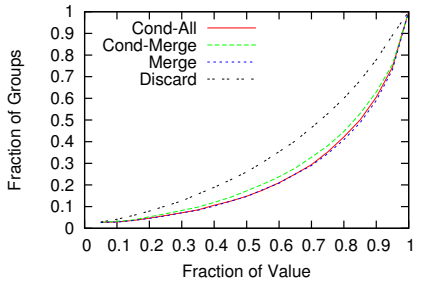


Figure 12: The values of social topologies obtained for the photo corpus.

	DISCARD	MERGE	COND-MERGE	COND-ALL
0.5	0.26	0.15	0.17	0.15
0.6	0.35	0.21	0.24	0.21
0.7	0.46	0.29	0.33	0.29
0.8	0.60	0.41	0.45	0.42

Figure 13: Fraction of groups needed to achieve given fraction of the value for photos.

One observed difference from email is that all variants other than DISCARD have almost identical curves. This suggests that the photo data set may be capturing tighter friendships since the trend of core friends tracked by the COND-ALL variant is similar to the MERGE variant which tends to create groups including more peripheral relationships.

From Figure 13, we see that DISCARD needs 26% of the groups to capture 50% of the value, whereas COND-ALL needs only 15%. That is, COND-ALL is better than DISCARD at compressing the social topology by a factor of 1.7. COND-ALL has compression improvement of 1.4 to 1.7 times for photos over DISCARD; it has an improvement of 1.8 to 2.6 for email.

6.2 Small Social Topologies

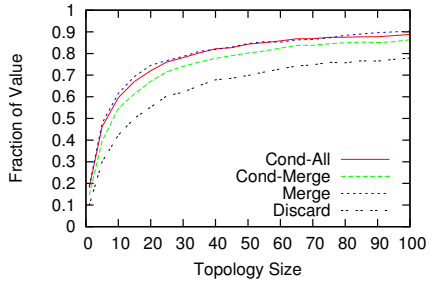


Figure 14: Values of small social topologies derived from photos.

If we wish to help users create Facebook friends lists to avoid over-sharing, it is important that we do not overwhelm them with too many groups. Even though a higher fraction of groups is needed than email, since photos are a smaller data set, the value is captured by a relatively small number of groups. Figure 14 shows the median of all values obtained for group sizes up to 100, and Figure 15 shows a few samples of the data. For example, with just 10 groups, 60% of value is captured by the COND-ALL algorithm, compared to 34% for the email data set. We see that the percentage of non-singleton groups is much higher, reflecting the fact that photo-taking is a gregarious activity, unlike email which often

	DISCARD	MERGE	COND-MERGE	COND-ALL
10	0.42 (8)	0.62 (9)	0.55 (8)	0.60 (7)
25	0.60 (21)	0.77 (21)	0.72 (21)	0.76 (18)
50	0.70 (42)	0.85 (41)	0.80 (42)	0.84 (37)

Figure 15: Values of photo-based social topologies with selected sizes. Non-singleton groups are shown in parentheses

involves correspondence with only one other person.

More characteristics of social topologies with 25 groups are shown in Figure 16. Note that the number of non-singleton groups included here are determined more by the data set than the algorithm. In this case, even the COND-ALL variant has a couple of new groups; it is harder to take a photo of a cohesive but broad group, whereas it is common to write at least one message to it. The median of the average group size is much higher across the board. MERGE still derives larger groups and includes more people, but not substantially more. The results show that people on average play about two roles, confirming the importance of our unique ability to find overlapping groups.

	DISCARD	MERGE	COND-MERGE	COND-ALL
Non-singletons	21	21	21	18
New groups	0	11	4	1
Group size	4.5	6.9	4.8	2.9
People	54	90	64	57
Roles/person	2.1	2.0	1.9	1.8

Figure 16: Properties of social topologies of size 25 from photos.

6.3 Significant Groups

Photo tagging data shows a distinctly lower number of significant groups than the email data set. In Figure 17 we see that the photo data set has a median of 7 significant groups.

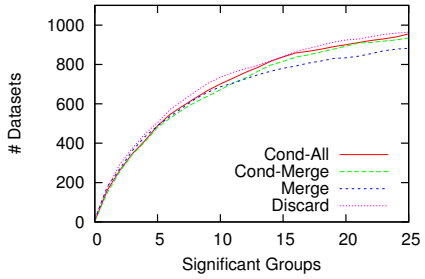


Figure 17: The cumulative distribution of the number of significant groups for the photo corpus.

7. EVALUATION BY EDIT DISTANCE

We now compare our algorithm variants with Newman's fast greedy clustering algorithm [17], which is a commonly used algorithm for discovering communities in social graphs. For this purpose, we used the implementation of Newman's algorithm in the igraph package of R. Unlike our algorithm, Newman's algorithm partitions the nodes in the graph into clusters via optimization of a modularity metric. As a neutral objective function, we select *edit distance*, an information-theoretic metric that is not a direct objective for either our algorithm or Newman's.

The *edit distance* between two words is defined as the minimum number of character alterations required to modify one of the words until it is equivalent to the second. We employ a modified version of edit distance for group communication data. The edit distance for a collection of communications C given a social topology S is

$$\text{EditDistance}(S, C) = \sum_{c \in C} \min_{s \in S} |c \cup s| - |c \cap s|$$

Intuitively, this metric captures the minimum number of insertions and deletions needed to specify the participants for each communication given a topology. The edit distance for each group is the number of members added and subtracted from its closest matching group in the topology. The sum of such edits defines the edit distance of a topology with respect to a set of groups. The largest possible edit distance is simply the sum of the sizes of the input groups.

We performed an experiment where we compute the edit distances for both the email and photo data sets using Newman's algorithm and the four variants of our algorithm. Each sent message and each tagged photo is treated as one unit of communication. We simply treat all the clusters generated by Newman's algorithm as the social topology for a user. As shown in Figure 18, the ratios of the minimum edit distance to the maximum edit distance (the total size of all the input groups) are similar for the two types of data, with the medians being 0.93 and 0.84 for email and photos, respectively.

	Group Size	# Groups	Edit Distance Ratio
Email	1	118	0.93
Photos	3	6	0.84

Figure 18: Median group parameters and edit distance ratios for Newman clustering.

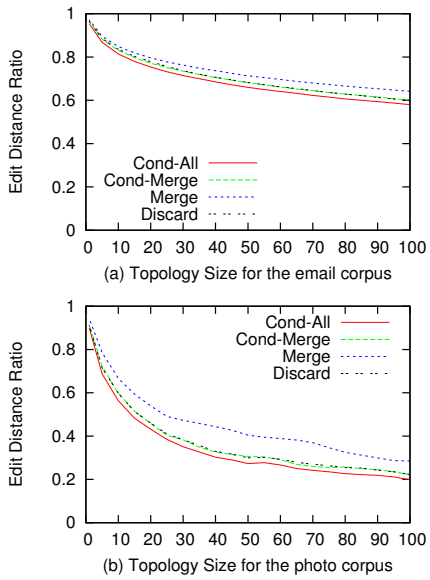


Figure 19: Comparison of the EditDistance metric across all 4 algorithm variants for the (a) email corpus and (b) photo corpus.

For our social topology algorithm, edit-distance ratios obtained is a function of the number of groups in the social topology. The medians of the edit-distance ratios are thus plotted in Figure 19. The results show that our algorithm outperforms Newman clustering in minimizing edit-distance

ratios. All variants of our algorithm beat the clustering algorithm with just 4 groups for email and 3 groups for photos. There is a significant difference in edit-distance ratios between the email and photo datasets. 10 groups generated by COND-ALL yield median ratios of 0.81 and 0.56 for emails and photos respectively; 25 groups yield ratios of 0.74 and 0.38.

Edit-distance ratios do not differ significantly between social topology algorithm variants, but we note that MERGE produces a worse edit-distance ratio than DISCARD. Given that MERGE uses a penalty weight of 0.5 for over-sharing whereas the penalty of a deletion for edit distances is 1, this makes sense. The goal of MERGE is find related people and not to optimize edit distance. Similarly, it is not expected for Newman's algorithm to produce small edit-distance ratios either. We performed this comparison mainly to illustrate how our algorithm is different from standard clustering algorithms. Our algorithm aims to identify the significant, possibly overlapping, groups where individuals may play multiple roles.

8. CONCLUSION

Unlike most other social network analysis algorithms that detect groups from global network data, our algorithm helps individuals automatically identify and use their social groups by analyzing their online social actions.

We formulated the social topology extraction problem as the compression of a natural social topology, where initial groups are labeled with their significance value, to a desired size according to a metric function that biases the composition of desired groups. We proposed a simple greedy algorithm derived from this value metric. Our algorithm can be used to produce the best representation of a social topology for a given size budget, though it can also automatically determine the number of significant groups a user has.

We have made publicly available two applications based on our algorithm to help users define friends groups and lists based on email and photo tags[3]. We are encouraged by the enthusiasm expressed by our users; the applications have been well received and it appears that the results are good enough to be interesting to many users. Our algorithm and source code are publicly available, and can be downloaded at the above URL.

We have performed an analysis of our algorithm over approximately 2,000 email archives and 1,100 photo collections, the latter collected by our Facebook application. We show that our algorithm is significantly different from the popular Newman's clustering algorithm for community detection. Using edit distances as an information-theoretic metric, we see that even a tiny topology consisting of 4 groups for email and 3 groups for Facebook produces significantly smaller edit distance ratios than Newman's algorithm. Our algorithm, with its ability to find nested and overlapping sets, is designed to find significant groups of friends from social data.

We found that both the email and photo corpus are highly amenable to compression, allowing our algorithm to produce social topologies that capture much of the value in the input set with a small percentage of groups. We show that the algorithm can capture 80% of the value with 20% and 42% of the groups for email and photos, respectively. More excitingly, we found that there are less than 15 significant groups in our email communications and 7 groups in photos for half of the population in our experiment. The results demonstrate the ability of our algorithm to distill out a small number of groups from thousands of emails and hundreds of photos. It also offers insight into people's social relationships as captured by their online activities.

[3]http://mobisocial.stanford.edu/groupgenie

9. ACKNOWLEDGMENTS

The authors would like to thank Peter Monaco from Xobni for providing the anonymized email data set. This research is supported in part by the NSF POMI (Programmable Open Mobile Internet) 2020 Expedition Grant 0832820, NSF grant CCF-0964173, Stanford Clean Slate Program, and Stanford MobiSocial Computing Laboratory.

10. REFERENCES

[1] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pages 207–216, 1993.

[2] A. Banerjee, C. Krumpelman, J. Ghosh, S. Basu, and R. Mooney. Model-Based Overlapping Clustering. In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 532–537, 2005.

[3] A. Clauset, M. Newman, and C. Moore. Finding Community Structure in Very Large Networks. *Physical Review E*, 70(6):66111, 2004.

[4] A. Culotta, R. Bekkerman, and A. Mccallum. Extracting Social Networks and Contact Information from Email and the Web. In *Proceedings of the First Conference on Email and Anti-Spam (CEAS)*, 2004.

[5] Facebook groups: How do I chat with a group? http://www.facebook.com/help/?faq=18808.

[6] Facebook featured friends: How do I feature specific friends on my profile? http://www.facebook.com/help/?faq=19417.

[7] E. Gilbert and K. Karahalios. Predicting Tie Strength with Social Media. In *CHI '09 Proceedings of the 27th International Conference on Human Factors in Computer Systems*, pages 211–220, 2009.

[8] J. Heer and D. Boyd. Vizster: Visualizing Online Social Networks. In *Proceedings of the IEEE Symposium on Information Visualisation (InfoVis 2005)*, pages 33–40, 2005.

[9] G. Kossinets and D. Watts. Empirical Analysis of an Evolving Social Network. *Science*, 311(5757):88, 2006.

[10] A. Lancichinetti, S. Fortunato, and J. Kertész. Detecting the Overlapping and Hierarchical Community Structure in Complex Networks. *New Journal of Physics*, 11:033015, 2009.

[11] J. Leskovec, K. Lang, A. Dasgupta, and M. Mahoney. Statistical Properties of Large Social and Information Networks. In *In Proceedings of the 17th International World Wide Web Conference (WWW 2008)*, pages 695–704, 2008.

[12] LinkedIn maps. http://inmaps.linkedinlabs.com/.

[13] D. Maclean, S. Hangal, S. K. Teh, M. S. Lam, and J. Heer. Groups Without Tears : Mining Social Topologies from Email. In *Proceedings of the 2011 International Conference on Intelligent User Interfaces*, pages 83–92, 2011.

[14] A. McCallum, X. Wang, and A. Corrada-Emmanuel. Topic and Role Discovery in Social Networks with Experiments on Enron and Academic Email. *Journal of Artificial Intelligence Research*, 30(1):249–272, 2007.

[15] S. Navlakha, R. Rastogi, and N. Shrivastava. Graph Summarization with Bounded Error. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, pages 419–432, 2008.

[16] S. Navlakha, M. Schatz, and C. Kingsford. Revealing Biological Modules via Graph Summarization. *Journal of Computational Biology*, 16(2):253–264, 2009.

[17] M. Newman. Fast Algorithm for Detecting Community Structure in Networks. *Physical Review E*, 69:066133:1–5, 2004.

[18] G. Palla, I. Derenyi, I. Farkas, and T. Vicsek. Uncovering the Overlapping Community Structure of Complex Networks in Nature and Society. *Nature*, 435(7043):814–818, June 2005.

[19] M. Roth, A. Ben-David, D. Deutscher, G. Flysher, I. Horn, A. Leichtberg, N. Leiser, Y. Matias, and R. Merom. Suggesting Friends Using the Implicit Social Graph. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 233–242, 2010.

[20] M. G. Siegler. Zuckerberg: "Guess What? Nobody Wants To Make Lists", August 2010. Retrieved from: http://techcrunch.com/2010/08/26/facebook-friend-lists/.

[21] T. Simonite. Facebook app reveals your social cliques, February 2011. http://www.technologyreview.com/communications/32394.

[22] Y. Tian, R. Hankins, and J. Patel. Efficient Aggregation for Graph Summarization. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, pages 567–580, 2008.

[23] J. Tyler, D. Wilkinson, and B. Huberman. E-mail as Spectroscopy: Automated Discovery of Community Structure within Organizations. *The Information Society*, 21(2):143–153, 2005.

[24] S. Whittaker, Q. Jones, B. A. Nardi, M. Creech, L. Terveen, E. Isaacs, and J. Hainsworth. ContactMap: Organizing Communication in a Social Desktop. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 11(4):445–471, 2004.

Compression of Web and Social Graphs supporting Neighbor and Community Queries

Cecilia Hernández
Dept. of Computer Science Univ. of Concepción
Dept. of Computer Science Univ. of Chile
chernand@dcc.uchile.cl

Gonzalo Navarro
Department of Computer Science
University of Chile
gnavarro@dcc.uchile.cl

ABSTRACT

Motivated by the needs of mining and advanced analysis of large Web graphs and social networks, we study graph patterns that simultaneously provide compression and query opportunities, so that the compressed representation provides efficient support for search and mining queries. We first analyze patterns used for Web graph compression while supporting neighbor queries. Our results show that composing edge-reducing patterns with other methods achieves new space/time tradeoffs, in particular breaking the smallest known space barrier for Web graphs when supporting neighbor queries. Second, we propose a novel graph compression method based on representing communities with compact data structures. These offer competitive support for neighbor queries, but excel especially at answering community queries. As far as we know, ours is the first graph compression method supporting such a wide range of community queries.

Categories and Subject Descriptors

H.2.8 [**Data Representation**]: Data Mining

General Terms

Algorithms, Experimentation, Theory

Keywords

Compression, Web Graphs, Social Networks, Compact Data Structures

1. INTRODUCTION

Much information on the structure, meaning and usage of the Web can be extracted by analyzing its graph. Web graphs are crucial for ranking algorithms, such as PageRank [7] and HITS [22], as well as for spam detection [3]. On the other hand, as never before, much information on social behavior is digitally available thanks to technologically supported social networks such as Facebook, Flickr, and many others. Current research on social networks includes detecting relevant communities, discovering important actors and understanding how the information flows in the network [21, 12, 27]. Web graphs and social networks are expanding in size. In 2008, Google reported more than 1 trillion unique URLs[1] on the Web, and Facebook reached 500 million active users in July 2010[2]. The scale of these networks has brought new challenges for analyzing and mining large graphs.

The research community has proposed compressed structures with direct access capabilities to facilitate mining and analysis of these large graphs. Compression rate is typically expressed as total number of bits per edge (bpe). The WebGraph framework [5, 4] is usually used as a reference on Web graph compression. Frequently, queries are reduced to the most basic one of listing the out-neighbors of a node, but in some cases in-neighbors are considered as well. Only recently, compression of social networks has been proposed, with out-neighbor [13] and with out/in-neighbor query support [26]. Even though out/in-neighbor queries may be used to build more complex operations such as community and outlier detection, we are not aware of any work studying the implementation and space/time requirements of such complex operations.

In this paper, we first analyze different graph patterns used for compressing Web graphs while supporting neighbor queries. We are particularly interested in analyzing whether the use of edge-reducing patterns [15, 11] might be combined with node ordering methods [5, 4, 2], and still take advantage of the similarity and locality found in graphs [5, 4, 2]. This combination has not been considered before. Our results show that it is possible to combine these methods to improve upon the best known results in space/time efficiency for compressing and supporting queries on Web graphs. In particular, we improve significantly upon the smallest spaces reported in the literature for Web graphs.

Our second contribution is a novel graph compression method based on finding communities and representing them with compact data structures. The resulting representation answers out/in-neighbor queries. However, the most interesting aspect is that it is very efficient at answering various community queries, which are useful for mining Web and social network graphs. The communities we consider are bicliques, which model node-centric communities based on complete mutuality [1] and have shown to be meaningful for mining [11]. Bicliques have already been used for Web graph

The 5th SNA-KDD Workshop '11 (SNA-KDD'11) August 21, 2011, San Diego CA USA.

[1] `http://googleblog.blogspot.com/2008/07/we-knew-web-was-big.html`
[2] `http://blog.facebook.com/blog.php?post=409753352130`

compression [11], but we introduce a different representation that permits querying those communities efficiently. Some of the queries we consider are: "enumerate the communities where entity X participates", "get the members that belong to a community Y", and "enumerate the communities with their sizes and densities". Querying over communities might be of great interest for discovering relevant actors based on positions and community sizes and densities where actors participate. Several measures consider prestige and centrality based on position and node degree [21, 27]. Unlike other compressed structures, our representation indexes internal graph patterns found in graphs (bicliques), enabling a wide set of query operations. We also index the rest of the graph (without bicliques), which facilitates its access and analysis.

2. RELATED WORK

The Web is one of the most studied graphs, and compressing it has been an active research area [10, 32, 5, 15, 11, 4]. Randall et al. [29] first proposed lexicographic ordering of URLs to exploit locality and similarity for compressing Web graphs. Later, Boldi and Vigna [5] proposed the WebGraph framework, one of the most competitive approaches in terms of space/time requirements for out-neighbor queries. This approach exploits power-law distributions, similarity and locality. They also developed ς codes, which are well suited for compressing power-law distributed data with small exponents [6]. More recently Boldi et al. [4] (BV) explored and evaluated other ordering methods, including Gray ordering, to improve their previous results.

Another recent compression scheme, by Apostolico and Drovandi [2], reorders the nodes based on a breadth-first (BFS) traversal of the graph, instead of the lexicographic order. It then encodes the outdegrees of the nodes in the order given by the BFS traversal, plus a list of the edges that cannot be deduced from the BFS tree. It achieves compression by dividing those lists into chunks and taking advantage of locality and similarity. We refer to this approach as AD.

Buehrer and Chellapilla [11] used a scalable pattern mining approach to provide compression of Web graphs. They used min-wise independent hashing [9] for clustering and identified directed complete bipartite graphs (i.e., bicliques) using a frequent itemset mining approach on each cluster. A biclique is formed by two sets of nodes, A and B, such that all the elements of A point to all the elements of B. For each biclique, they defined a virtual node that connects the two sets, replacing all the $|A| \cdot |B|$ links from A to B by $|A|$ links from A to the virtual node, plus $|B|$ links from the virtual node to B. Applying gap encoding to the resulting graph, they were able to improve the original compression of Boldi and Vigna [5]. We refer to this scheme as VNM (Virtual Node Miner). Recently, VNM has been used to improve running times for Web graph algorithms based on random walks, such as PageRank and HITS, achieving speedups proportional to the compression ratio [20]. Another edge-reducing approach was proposed by Claude and Navarro [15]. It is based on Re-Pair [25], a grammar-based compressor. Re-Pair repeatedly finds the most frequent pair of nodes in the concatenated sequence of all adjacency lists and replaces it with a new symbol.

All these methods provide efficient out-neighbor navigation, that is, retrieving the adjacency list of any node. Adding in-neighbor navigation (i.e., retrieving the nodes that point to a node) is usually performed by representing the transpose of the graph in addition to the graph itself. Boldi et al. [4] showed that Gray ordering outperforms others for the transpose of Web graphs. A proposal by Brisaboa et al. [8] supports out/in-neighbor navigation using a structure that represents the adjacency matrix taking advantage of its sparseness. A recent implementation improvement is available [24]; we refer to it as k2tree. Claude and Navarro [16] presented a compact Web graph representation that enrichs the output of Re-Pair compression with compact data structures (Re-Pair-GMR) [17] that yields in-neighbor queries as well.

Recent works on compressing social networks [13, 26] have exposed compression opportunities, although in less degree than in Web graphs. The approach by Chierichetti et al. [13] is based on the Webgraph framework [5], using shingling ordering (based on Jaccard coefficient) [9] and exploiting link reciprocity. Maserrat and Pei [26] achieve compression by defining a Eulerian data structure using multi-position linearization of directed graphs. Their approach supports out/in-neighbor queries in sublinear time.

In the context of communities, Saito et al. [30] presented a method for spam detection based on classical graph algorithms such as identification of weak and strong connected components, maximal clique enumeration and minimum cuts. Other techniques, based on graph algorithms, aim to extract small subgraphs or small communities for mining purposes [23]. The Web graph compression proposed by Buehrer and Chellapilla [11] use the notion of communities defined by bicliques. They provide community seed semantic evaluation showing four community patterns found during compression. However, none of these schemes supports community queries on the compressed structure.

3. COMPRESSING WEB GRAPHS

In this section we describe edge-reducing patterns (VNM [11] and Re-Pair [15]), node ordering methods (BV [6, 4] and AD [2]), and techniques that support out/in-neighbor queries such as k2tree [8]. We present experimental results on the effect of combining edge-reducing patterns with the other methods and discuss our results.

3.1 Edge-reducing compressors

The VNM [11] compressor is based on the idea of identifying bicliques and using virtual nodes as one level of indirection between the two sets. The use of virtual nodes reduces the number of edges. After reducing edges the authors apply gap and Huffman coding for compressing Web graphs. Identifying bicliques has two phases: clustering and mining. The clustering phase groups similar rows of the adjacency matrix. It uses the heuristic of grouping rows with high Jaccard coefficients [9]. For a graph $G = (V, E)$, they use k min-wise independent hash functions [9] to obtain a hash function matrix of size $k \cdot |V|$. Rows in the matrix are sorted lexicographically and then traversed by column, grouping rows with the same value. When the number of rows drops below a threshold, a new cluster is defined. The hash function matrix is only used for clustering.

The mining phase operates locally within each cluster, looking for vertices with common subsets of outlinks. More relevant communities are given by larger outlink sets (of size *pattern_size*) shared by larger vertex sets (of size *pattern_frequency*), which provide better compression. The

mining algorithm builds a trie on the adjacency lists of the vertices, and chooses long paths in the trie to find good sets.

The algorithm performs successive iterations, so that virtual nodes created during an iteration can participate in bicliques in a subsequent iteration.

Another edge-reducing compressor is based on Re-Pair [25], a grammar-compression algorithm consisting of repeatedly finding the most frequent pair of symbols in a sequence of integers and replacing it with a new symbol. Starting from an integer sequence S, the algorithm iterates over the following steps. (1) It identifies the most frequent pair ab in S. (2) It adds the rule $r \to ab$ to a dictionary R, where r is a new symbol that is not in S. (3) It replaces any occurrence of ab by r in S. This is repeated until the replacements do not improve compression. The output of the algorithm is the remaining sequence C and the dictionary R. C is formed with both terminal symbols (original symbols in S) and nonterminal symbols (introduced in step 2). In order to obtain the original symbols, nonterminals must be decompressed using the information stored in dictionary R.

Re-Pair has been used for compressing the Web graph adjacency lists, providing competitive results in terms of compression, and supporting fast out-neighbor queries [15].

3.2 Reordering nodes

The WebGraph framework [4] supports different Web page orderings (URL, lexicographic, Gray ordering, loose and strict host-by-host Gray ordering). With either ordering, each Web page is mapped to a unique integer identifier. WebGraph then exploits locality and similarity. Locality means that, usually, most of the links of a page point to nearby pages. Similarity means that many Web pages tend to have many outlinks in common, or in other words, that some adjacency lists are very similar to others.

WebGraph uses two compression parameters, *window_size* (w) and *maximum_reference_count* (m). The *window_size* corresponds to the number of previous rows in the adjacency matrix considered when compressing a row x for reference coding. The idea is to find the most similar previous row in that window. If that row exists, it is called a prototype. A larger *window_size* yields better and slower compression. On the other hand, the *maximum_reference_count* is the maximum allowed length of a reference chain. When compressing a row x of the adjacency matrix, the compression is done with respect to a previous prototype y, and row y could have been compressed differentially with respect to some other prototype z, generating a reference chain. Limiting the length of the reference chain retains fast decompression.

Each row is encoded using its prototype, if any, plus outlinks not in the prototype. WebGraph encodes consecutive outlinks using interval encoding and ς codes for integers [6].

Apostolico and Drovandi [2] proposed an ordering method based on breadth-first search. The method depends on the topological structure of graphs instead of the URLs. They use gap and run-length encoding and define a new integer encoding called π codes, claimed to be better suited than ς codes for power-law distributions with exponent close to 1.

The compression scheme works on chunks of *level* (l) nodes. Parameter *level* provides a tradeoff between compression performance and time to retrieve the adjacency list of a node. With $l = 8$ they achieve better compression and similar access times than WebGraph.

3.3 Compressors supporting out/in-neighbors

These compressors support out/in-neighbor queries on the same structure, avoiding the need of using the graph plus its transpose (as it is the case with the WebGraph, VNM, and Re-Pair compressors).

The k2tree scheme [8] represents the adjacency matrix by a k^2-ary tree of height $h = \lceil \log_k n \rceil$ (where n is the number of vertices). Simulating an MX-Quadtree decomposition [31], it divides the matrix into k^2 submatrices of size n^2/k^2. The tree representation, at each level, defines k^2 child nodes containing a bit "1" if there is at least a "1" on the submatrix it represents, and a "0" otherwise. Nodes represented with a bit "1" are recursively divided into k^2 nodes. At the last level, the leaf nodes contain the bits of the adjacency matrix. A recent improvement [24] uses statistical compression at the last level and retains fast access times.

Re-Pair GMR [16] uses the grammar-based compression technique Re-Pair, and defines compact data structures based on bitmaps and sequences with fast rank/select operations [14, 17]. These support in-neighbor queries by finding in the adjacency lists all the positions where the node, or a nonterminal expanding to it, is mentioned.

3.4 Composing methods

In this section we evaluate the impact of combining edge-reduction with other methods. We proceed in two stages: an edge-reduction stage yields a new graph, containing fewer edges and more nodes (including virtual nodes). Then a compression stage applies existing compression techniques on the edge-reduced graph.

For the edge-reducing stage we consider three alternatives. The first is VNM, which we implemented as described in its article [11], using C++ and STL, but we did not implement the compression of the resulting graph (the full method, including this last compression, will be called VNM_b). We added a compression parameter $maximum\_saving = pattern\_size \cdot pattern\_frequency$. This parameter defines the minimum biclique size we consider for edge reduction. Second, we implemented Re-Pair as just an edge-reducing method (i.e., rule $r \to ab$ is seen as the creation of a virtual node r with edges to a and b), which we call RP_o. Third, we consider VNMRP, which applies VNM and then RP_o.

For the compression stage we also considered three alternatives. First, we used the full Re-Pair compression scheme obtained from its authors [15] (RP_c). Second, we used WebGraph (BV), version 2.4.4 taken from `http://webgraph.dsi.unimi.it`. Third, we used AD version 0.2.1, without dependencies, taken from `http://www.dia.uniroma3.it/~drovandi/software.php`.

We executed our experiments on a Linux PC with 8 processors Intel Xeon at 2.4GHz, with 32 GB of RAM and 12 MB of cache, using sequential algorithms.

We used the following data sets, corresponding to real Web graphs and social networks; some statistics are given in Table 1. Web graphs are snapshots made available at `http://law.dsi.unimi.it` within the WebGraph project. As social network graphs, we use Facebook (undirected graph, from `http://socialnetworks.mpi-sws.org/data-wosn2009.html`) and Flickr (directed graph, from `http://socialnetworks.mpi-sws.org/data-www2009.html`) data sets used in social network research [33, 27]. The LiveJournal (directed graph) data set from the SNAP project (Stanford Network Analysis Package,

Data Set	Nodes	Edges
eu-2005	862,664	19,235,140
in-2004	7,414,866	194,109,311
uk-2002	18,520,486	298,113,762
ar-2005	22,744,080	639,999,458
it-2004	41,291,594	1,150,725,436
Facebook	45,622	817,090
Flickr	1,861,232	22,613,981
LiveJournal	4,847,571	68,993,773

Table 1: Some statistics of our test graphs. Here in-2004 corresponds to indochina-2004 and ar-2005 to arabic-2005.

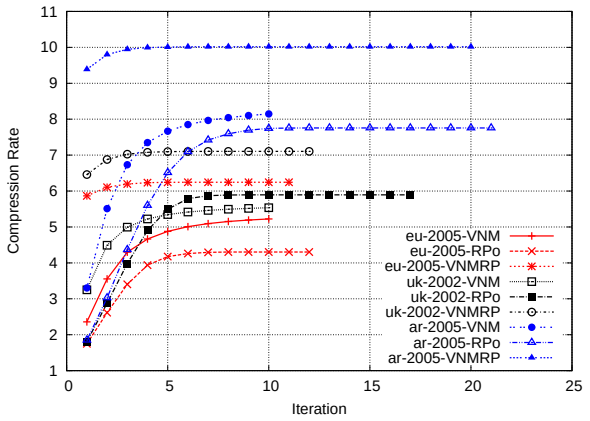


Figure 1: Compression rate based on number of edges.

`http://snap.stanford.edu/data/`).

The first experiment evaluates the compression rate defined as the total number of edges of the original graph divided by the edges remaining after compression (including from/to virtual nodes). Figure 1 shows the compression rate achieved by VNM, RP_o, and by VNMRP. The results suggest that combining both techniques provides better compression rates than applying either of them individually.

The second experiment measures the compression in terms of *bpe* and aiming at maximum compression (i.e., no structures for direct access to the graphs are maintained). We found that reducing the number of edges in the edge-reducing stage does not always improve the final compression figures, as shown in Table 2. For this table we combined edge-reduction using VNM with compression using BV and AD. We tuned VNM using parameter m. VNM* ($m = 1$) allows any virtual node pattern that reduces edges, thus minimizing the edges as much as possible in the edge-reducing stage. Table 2 shows that VNM* does not achieve the lowest possible bpe values, but these are achieved with $m = 30$ on `eu-2005`, and $m = 100$ on the other Web graphs (those are used in the row labeled VNM). The best compression is achieved with loose host-by-host Gray ordering of BV with $w = 8$ and $m = -1$ (for maximum compression). This suggests that there are sufficient regularities that BV can exploit after removing redundant edges, even if there are more nodes in the graph.

We also apply AD with $l = 100$ after VNM. Larger l did not yield more compression. In this case, it is best to run VNM reducing the edges as much as possible (VNM*). Once again, the combination works better than the individual techniques. In particular, VNM+BV and VNM*+AD *beat the best known space for compressing Web graphs.*

Name	eu-2005	in-2004	uk-2002	ar-2005	it-2004
BV	3.60	1.14	1.86	1.50	1.61
VNM^*+BV	2.61	1.29	2.01	1.64	1.52
VNM+BV	**2.24**	**1.04**	**1.65**	**1.35**	**1.28**
AD	2.89	1.08	1.90	1.67	1.55
VNM^*+AD	**2.13**	**1.00**	**1.68**	**1.36**	**1.32**

Table 2: Maximum compression comparison in bpe for combining VNM with BV and AD.

Name	eu-2005	in-2004	uk-2002	ar-2005	it-2004
VNM_b	2.90	-	1.95	1.81	1.67
RP_c	4.98	2.62	4.28	3.22	2.91
VNM*+RP_c	3.57	2.30	3.65	2.78	2.63
BV	4.49	1.88	2.82	2.26	2.37
VNM+BV	2.87	1.51	2.42	1.87	1.76
AD_8	3.68	1.61	2.64	2.28	2.17
RP_o+AD_8	3.15	1.71	2.67	2.01	2.26
VNMRP+AD_8	2.28	1.17	1.92	1.50	1.49
VNM*+AD_4	2.39	1.24	2.05	1.57	1.57
VNM*+AD_8	**2.26**	**1.12**	**1.87**	**1.47**	**1.45**

Table 3: Compression in bpe when allowing random access, for various methods. VNM_b are the bpe values reported by Buehrer and Chellapilla [11].

Table 3 shows the compression results when the graphs contain structures to support direct access. We include bare compression methods (VNM_b, RP_c, BV, and AD) and our best performing combinations. To enable direct access we use BV with $w = 8$ and $m = 3$, and AD with $l = 4$ and 8. Since both BV and AD exploit locality and similarity, the results suggest that the BFS ordering used in AD works better than the loose host-by-host Gray ordering used in BV. We achieve the best compression results using VNM*+AD_8.

Tables 4 and 5 give average times to retrieve adjacency lists of 20 million random nodes. In the first we do not include the time to recursively expand the virtual nodes and the node id mapping to obtain the original graph. Average times displayed in Table 4 are important considering recent research where using VNM allows speeding up Web graph algorithms based on random walk such as PageRank [20]. In Table 5 we include all the required operations to retrieve the original adjacency lists. We can see that adding the VNM* preprocessing using AD multiplies access times by 2–3, yet these are still within 20 microseconds per node, that is, below the microsecond per delivered edge. Using VNM with BV multiplies access times only by 2.

3.5 Bidirectional navigation

We additionally consider combining edge-reduction methods with techniques that support out/in-neighbor queries. We evaluate the following variants:

T1: Re-Pair GMR [16] on the original graph.
T2: k2tree [8] on the original graph.

Data set	AD		VNM*+AD		BV	VNM+BV
	$l = 4$	$l = 8$	$l = 4$	$l = 8$	$m = 3, w = 8$	
eu-2005	2.50	3.86	0.78	1.26	1.50	1.03
in-2004	2.02	2.92	0.75	1.12	1.41	1.12
uk-2002	1.75	2.61	0.83	1.14	1.34	1.19
ar-2005	2.54	3.65	0.80	1.22	1.58	1.34
it-2004	2.41	3.66	0.79	1.30	1.78	1.35

Table 4: Average time to retrieve an adjacency list, in microseconds, without any reordering nor expansion.

Data set	AD		VNM*+AD		BV	VNM+BV
	$l = 4$	$l = 8$	$l = 4$	$l = 8$	$m = 3,\ w = 8$	
eu-2005	4.36	6.63	10.9	19.93	2.18	4.71
in-2004	3.58	5.23	5.65	10.72	1.80	3.33
uk-2002	2.50	3.77	4.53	7.72	1.64	2.90
ar-2005	3.82	5.93	9.49	16.95	2.07	4.53
it-2004	3.70	5.93	7.87	14.23	2.14	4.18

Table 5: Average time to retrieve an adjacency list, in microseconds, considering full expansion and reordering.

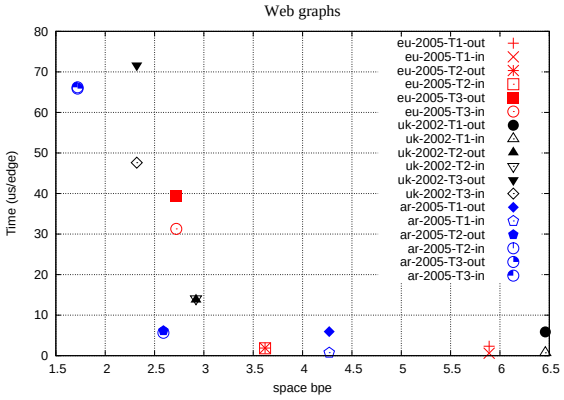


Figure 2: Space/time efficiency with out/in-neighbor queries with T1,T2, and T3 on Web graphs.

T3: VNM on the original graph and then k2tree.

The space/time requirements of these techniques, on Web graphs and social networks, are displayed in Figures 2 and 3, respectively. The combination of VNM and k2tree (T3) achieves by far *the best space efficiency on Web graphs when supporting bidirectional neighbors*. However, this comes at a significant price in access time. The combination, on the other hand, does not work well on social networks, where k2tree alone (T2) is unbeaten. This suggests that reducing the number of edges on social networks removes some degree of sparseness from the original adjacency matrices, which does not happen on Web graphs.

3.6 Discussion

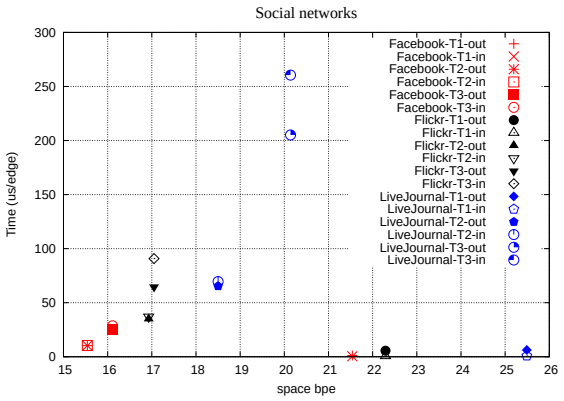


Figure 3: Space/time efficiency with out/in-neighbor queries with T1,T2, and T3 on social networks.

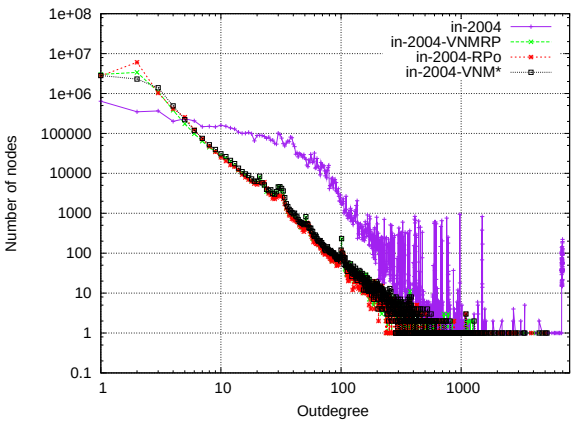


Figure 4: Node degree distribution of in-2004

Data set	Name	edges	bpe
eu-2005	RP_o	0.72	0.71
	VNMRP	0.98	0.99
in-2004	RP_o	0.79	0.65
	VNMRP	0.98	0.95
uk-2002	RP_o	0.74	0.70
	VNMRP	0.98	0.97
ar-2005	RP_o	0.75	0.73
	VNMRP	0.98	0.98
it-2004	RP_o	0.80	0.64
	VNMRP	0.98	0.97

Table 6: Ratios of edge reduction achieved by various methods versus VNM, and of bpe reduction using AD after them.

Our results show that edge reduction, combined with other methods, improves the state-of-the-art compression of Web graphs. Figure 4 shows the node degree distribution of the in-2004 original graph and after applying VNM, RP_o, and VNMRP. Very similar figures are obtained for the other graphs. We observe that reducing edges produces very clean power law distributions in the node degrees, regardless of the edge-reduction method used. However, this new shape does not seem to have, in general, an impact on the compression achieved with AD, beyond the mere reduction in the data size. Table 6 shows the reduction in number of nodes plus edges achieved by the edge-reduction methods as a fraction of VNM, and the corresponding reduction in bpe after applying AD. It can be seen that in most cases the reduction factors are very similar, that is, AD achieves the same reduction in space on the original and on the reduced graphs, and thus the overall space improvement over bare AD is a consequence of working on a smaller graph and not of the different node degree distribution.

Exceptions to this rule are in-2004 and it-2004, where AD obtains a significant further reduction after applying RP_o. This may be due to the fact that RP_o creates many virtual nodes of outdegree 2, which is possibly advantageous for a BFS ordering. In any case, this shows that further research is necessary to understand the interactions between edge-reduction and edge-reordering methods.

4. COMPRESSING COMMUNITIES

In this section we present a novel compressed data structure based on mining communities. We first provide some

basic notions of compact data structures we will need.

4.1 Compact data structures

A compact data structure provides the same abstraction as its classical counterpart, using little space and supporting interesting queries without having to expand the whole structure. Many compact data structures use as a basic tool a bitmap supporting rank/select/access query primitives. Operation $rank_B(b, i)$ on the bitmap $B[1, n]$ counts the number of times bit b appears in the prefix $B[1, i]$. Operation $select_B(b, i)$ returns the position of the i-th occurrence of bit b in B (and $n + 1$ if there are no i b's in B). Finally, operation $access_B(i)$ retrieves the value $B[i]$. A solution requiring $n + o(n)$ bits and providing constant time for rank/select/access queries was proposed by Clark [14] and a good implementation is available (RG) [18]. In later work, Rao et al. (RRR) [28] improved the required space to $nH_0(B) + o(n)$ bits. $H_0(B)$ corresponds to the zero-order entropy of bitmap B, $H_0(B) = \frac{n_0}{n} \log \frac{n}{n_0} + \frac{n_1}{n} \log \frac{n}{n_1}$, where B has n_0 zeros and n_1 ones.

The bitmap representations can be extended to compact data structures for sequences $S[1, n]$ over an alphabet Σ of size σ. The wavelet tree (WT) [19] supports rank/select/access queries in $O(\log \sigma)$ time. It uses bitmaps internally, and its total space is $n \log \sigma + o(n) \log \sigma$ bits if representing those bitmaps using RG, or $nH_0(S) + o(n) \log \sigma$ bits if using RRR, where $H_0(S) = \sum_{c \in \Sigma} \frac{n_c}{n} \log \frac{n}{n_c}$, being n_c the number of occurrences of c in S. Another sequence representation (GMR) [17] uses $n \log \sigma + n\, o(\log \sigma)$ bits, and supports *rank* and *access* in time $O(\log \log \sigma)$, and *select* in constant time.

4.2 Community-based compressed structure

We focus on detecting communities on Web graphs and social networks so that we can represent the original graph in terms of a set of *community graphs* and a *remaining graph*.

Definition 1. *Community.* We define a community as a complete (directed or undirected) bipartite graph, $H(S, C) = G(V = S \cup C, E = S \times C)$. Vertices $s \in S$ are called *sources* and vertices $c \in C$ are called *centers*. We define the community *size* as $|S| + |C|$.

Definition 2. *Community Density.* We define the density by considering the connections inside a community group [1]: $H(S, C) = G(V, E)$ is γ-*dense* if $\frac{|E|}{|V|(|V|-1)/2} \geq \gamma$.

Definition 3. *Directed (Undirected) Bipartite Partition, DBP (UBP).* Let $G(V, E)$ be directed (undirected). A bipartite partition of G consists of a class $\mathcal{H} = \bigcup H_r$ of bipartite graphs $H_r = H(S_r, C_r)$, and a *remaining* graph $\mathcal{R}(V_R, E_R)$, so that all H_r and $\mathcal{R}$ are edge-disjoint and $G = \mathcal{H} \cup \mathcal{R}$.

Definition 4. *Undirected plus Directed Bipartite Partition, UDBP.* Let $G(V, E)$ be a directed graph. We derive from G an undirected graph $G_u(V_u, E_u)$, containing an undirected edge per pair of reciprocal edges in G. Now consider the *UBP* of G_u into $\mathcal{H}_u$ and $\mathcal{R}_u(V_{R_u}, E_{R_u})$. Define $dup(E)$ as the set of directed edges formed by a pair of reciprocal edges per undirected edge in E. Then we call $G_d(V_d, E - dup(E_u - E_{R_u}))$ the remaining directed graph of G. Now consider the *DBP* of G_d into $\mathcal{H}_d$ and R_d. The *UDBP* of G is formed by $\mathcal{H}_u$, $\mathcal{H}_d$, and $\mathcal{R}_d$.

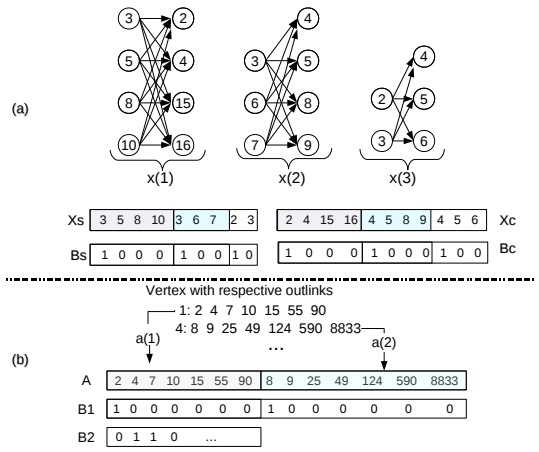


Figure 5: Compact representation of $\mathcal{H}$ (a) and $\mathcal{R}$ (b).

Note that *DBP* and *UBP* aim at representing a graph as a set of communities, regarded as directed or undirected bicliques. A large community $H(S, C)$ will allow us replacing $|S| \cdot |C|$ edges by just $|S| + |C|$ edges and a new node. *UDBP* is more sophisticated, and aims at exploiting *reciprocity* in directed graphs, that is, reciprocal edges. It first looks for reciprocal bicliques, and only then for directed bicliques. Whether *UDBP* is better or worse than plain *DBP* on a directed graph will depend on its degree of reciprocity.

Our construction proceeds in two phases. The first phase is community detection and extraction, and the second is building a compressed structure representation. We use the VNM scheme [11] for detecting and extracting communities. We apply VNM iteratively, extracting large communities first, using higher to lower *maximum_saving* values.

4.3 Compact representation of $\mathcal{H}$

Let $\mathcal{H} = \{H_1, \ldots, H_N\}$ be the communities found in either of the previous definitions. We represent $\mathcal{H}$ as two sequences of integers with corresponding bitmaps. Sequence X_s with bitmap B_s represent the sequence of sources of the communities and sequence X_c with bitmap B_c represent the respective centers. More precisely, we have $X_s = x_s(1)x_s(2)...x_s(r)...x_s(N)$, where $x_s(r) = s_1...s_k$ represents the set S_r of $H_r = (S_r, C_r)$, $s_i \in S_r$, $s_i < s_{i+1}$ for $1 \leq i < k$, and $B_s = 10^{|S_1|-1}...10^{|S_N|-1}1$. In a similar way, we have $X_c = x_c(1)x_c(2)...x_c(r)...x_c(N)$, where $x_c(r) = c_1...c_m$ represents the set C_r, $c_j \in C_r$, $c_j < c_{j+1}$ for $1 \leq i < m$, and $B_c = 10^{|C_1|-1}...10^{|C_N|-1}1$. Figure 5 (a) shows an example.

We represent integer sequences and bitmaps with compact data structures that support rank/select/access operations: we use WTs [19] for sequences and an uncompressed representation [14] for bitmaps, for a total space of $|X|(H_0(X) + 1) + o(|X| \log \sigma)$, where σ is the number of vertices in $\mathcal{H}$. Note that $|X|$ is the sum of the sizes of the communities in $\mathcal{H}$, which can be much less than the number of edges in the subgraph it represents.

We answer out/in-neighbor queries as follows. Their complexity is $O((|output| + 1) \log \sigma)$, which is essentially optimal up to a factor of $O(\log \sigma)$, where σ is the number of nodes in the graph.

out-neighbors (u)
$occur \leftarrow rank_{X_s}(u, |X_s|)$
for $i \leftarrow 1$ **to** $occur$ **do**
  $y \leftarrow select_{X_s}(u, i)$
  $p \leftarrow rank_{B_s}(1, y)$
  $s \leftarrow select_{B_c}(1, p)$
  $e \leftarrow select_{B_c}(1, p+1) - 1$
  for $j \leftarrow s$ **to** e **do**
    $out.add(access_{X_c}(j))$
  end for
end for

in-neighbors (u)
$occur \leftarrow rank_{X_c}(u, |X_c|)$
for $i \leftarrow 1$ **to** $occur$ **do**
  $y \leftarrow select_{X_c}(u, i)$
  $p \leftarrow rank_{B_c}(1, y)$
  $s \leftarrow select_{B_s}(1, p)$
  $e \leftarrow select_{B_s}(1, p+1) - 1$
  for $j \leftarrow s$ **to** e **do**
    $in.add(access_{X_s}(j))$
  end for
end for

We answer community queries in $\mathcal{H}$ as follows. Table 7 gives the time complexities achieved. Most of them are, again, optimal up to factor $O(\log \sigma)$. The exception is Q7, which can be costlier due to repeated results.

Q1 Get the *centers* of community x.

$start \leftarrow select_{B_c}(1, x)$
$end \leftarrow select_{B_c}(1, x+1) - 1$
for $i \leftarrow start$ **to** end **do**
  $centers.add(access_{X_c}(i))$
end for

Q2 Get the *sources* of community x.

$start \leftarrow select_{B_s}(1, x)$
$end \leftarrow select_{B_s}(1, x+1) - 1$
for $i \leftarrow start$ **to** end **do**
  $sources.add(access_{X_s}(i))$
end for

Q3 Get communities where u participates as a source.

$occur \leftarrow rank_{X_s}(u, |X_s|)$
for $i \leftarrow 1$ **to** $occur$ **do**
  $y \leftarrow select_{X_s}(u, i)$
  $p \leftarrow rank_{B_s}(1, y)$
  $comms.add(p)$
end for

Q4 Get communities where u participates as a center.

$occur \leftarrow rank_{X_c}(u, |X_c|)$
for $i \leftarrow 1$ **to** $occur$ **do**
  $y \leftarrow select_{X_c}(u, i)$
  $p \leftarrow rank_{B_c}(1, y)$
  $comms.add(p)$
end for

Q5 Get the number of communities where u participates as a source and as a center.

$ncs \leftarrow rank_{X_s}(u, |X_s|)$
$ncc \leftarrow rank_{X_c}(u, |X_c|)$

Q6 Enumerate the members of community x.

$ss \leftarrow select_{B_s}(1, x)$
$es \leftarrow select_{B_s}(1, x+1) - 1$
for $i \leftarrow ss$ **to** es **do**
  $members.add(access_{X_s}(i))$
end for
$sc \leftarrow select_{B_c}(1, x)$
$ec \leftarrow select_{B_c}(1, x+1) - 1$
for $i \leftarrow sc$ **to** ec **do**
  $members.add(access_{X_c}(i))$
end for

Q7 Enumerate the out-communities at distance 1 of community x.

$centers \leftarrow Q1(x)$
for c in $centers$ **do**
  $comms.add(Q3(c))$
end for

Q8 Enumerate all the communities with their sizes.

$nc \leftarrow rank_{B_s}(1, |B_s|)$
for $i \leftarrow 1$ **to** nc **do**
  $ss \leftarrow select_{B_s}(1, i+1) - select_{B_s}(1, i)$
  $sc \leftarrow select_{B_c}(1, i+1) - select_{B_c}(1, i)$
  $size\_list.add(ss + sc)$
end for

Q9 Enumerate all the communities with their densities.

$nc \leftarrow rank_{B_s}(1, |B_s|)$
for $i = 1$ **to** nc **do**
  $sources \leftarrow select_{B_s}(1, i+1) - select_{B_s}(1, i)$
  $centers \leftarrow select_{B_c}(1, i+1) - select_{B_c}(1, i)$
  $edges \leftarrow sources \cdot centers$
  $nodes \leftarrow sources + centers$
  $densities.add(\frac{edges}{nodes \cdot (nodes-1)/2})$
end for

4.4 Compact representation of $\mathcal{R}$

We define a sequence of integers A and two bitmaps $B1$ and $B2$ for representing $\mathcal{R}(V_R, E_R)$. Sequence A is defined as $A = a(1)...a(i)...a(N)$, where $|A| = |E_R|$, $a(i)$ is the i-th nonempty direct adjacency list of $\mathcal{R}$, and N is the total number of vertices with at least one edge in $\mathcal{R}$. Bitmap $B1$ is $10^{|a(1)|-1}...10^{|a(N)|-1}$, so $|B1| = |E_R|$. $B2$ is a bitmap such that $B2[i] = 1$ iff vertex i does not have out-neighbors and $|B2| = |V_R|$. Figure 5 (b) shows an example. The space using WTs is $|A|(H_0(A)+1)+|\sigma|+o(|A|\log\sigma)$, where $\sigma = |V_R|$. We answer neighbor queries on $\mathcal{R}$ as follows:

out-neighbors (u)
if $access_{B2}(u) = 0$ **then**
  $x \leftarrow rank_{B2}(0, u)$
  $start \leftarrow select_{B1}(1, x)$
  $end \leftarrow select_{B1}(1, x+1) - 1$
  for $i \leftarrow start$ **to** end **do**
    $out.add(access_A(i))$
  end for
end if

in-neighbors (u)
$occur \leftarrow rank_A(u, |A|)$
for $i \leftarrow 1$ **to** $occur$ **do**
  $y \leftarrow select_A(u, i)$
  $p \leftarrow rank_{B1}(1, y)$
  $in.add(select_{B2}(0, p))$
end for

Query	Time complexity
Q1/Q2	$O(\|output\| \cdot \log \sigma)$
Q3/Q4	$O((\|output\| + 1) \log \sigma)$
Q5	$O(\log \sigma)$
Q6	$O(\|output\| \log \sigma))$
Q7	$O(\|output\| \log \sigma) \ldots O(\|Q1\| \cdot \|output\| \log \sigma)$
Q8/Q9	$O(\|output\|)$

Table 7: Time complexity for community queries.

On directed graphs, in-neighbors and out-neighbors are found in time $O((|output|+1)\log\sigma)$. On undirected graphs we choose arbitrarily to represent each edge $\{u, v\}$ as (u, v) or (v, u). Consequently, finding the neighbors of a node requires carrying out both algorithms.

To carry out out/in-queries on the whole graph, we must query $\mathcal{H}$ and $\mathcal{R}$ (for *UBP* or *DBP* partitions) or $\mathcal{H}_u$, $\mathcal{H}_d$ and $\mathcal{R}_d$ (for *UDBP* partitions), and merge the results. Community queries are carried out only on $\mathcal{H}$ (or $\mathcal{H}_u$ and $\mathcal{H}_d$ for *UDBP*, then merging the results). Our pseudocodes on X and B addressed the directed case. Those for communities representing undirected graphs are very easy to derive, and left as an exercise.

5. EXPERIMENTAL EVALUATION

We evaluate our compact data structures supporting out/in-neighbor and community queries. We are interested in space/time requirements in terms of the community graph $\mathcal{H}$ (or $\mathcal{H}_u + \mathcal{H}_d$), the remaining graph $\mathcal{R}$, and the complete graph $G = \mathcal{H} \cup \mathcal{R}$.

5.1 Space/time evaluation

We compare space/time efficiency using the representations below. We refer as WT-N-b to representing sequence X with wavelet trees and bitmaps with RG [18], and as WT-N-r to using wavelet trees for X and bitmaps compressed with RRR [28]. N is the sampling parameter used for bitmap implementations (if left as a variable, it gives a space/time tradeoff). We will not give the results for using GMR [17] on $\mathcal{H}$ because the space achieved is not competitive.

T4 WT-N-b ($\mathcal{H}$) + Re-Pair GMR ($\mathcal{R}$)
T5 WT-N-r ($\mathcal{H}$) + Re-Pair GMR ($\mathcal{R}$)
T6 WT-N-b ($\mathcal{H}$) + k2tree ($\mathcal{R}$)
T7 WT-N-r ($\mathcal{H}$) + k2tree ($\mathcal{R}$)
T8 WT-N-b ($\mathcal{H}$) + WT-64-b ($\mathcal{R}$)
T9 WT-N-b ($\mathcal{H}$) + WT-64-r ($\mathcal{R}$)
T10 WT-N-r ($\mathcal{H}$) + WT-64-b ($\mathcal{R}$)
T11 WT-N-r ($\mathcal{H}$) + WT-64-r ($\mathcal{R}$)

We use the definitions of Section 4.2 and our Web graphs and social networks of Table 1. We use *UBP* to represent

Data Set	$\mathcal{H}_u$	$\mathcal{H}_d$	$\mathcal{R}$	%
Facebook	457,968	-	359,122	43.95
Flickr	8,009,958	7,697,030	6,906,992	30.50
LiveJournal	24,270,703	17,374,268	27,078,099	39.24
eu-2005	-	17,568,086	1,667,054	8.66
uk-2002	-	270,951,713	27,162,049	9.11
ar-2005	-	602,662,165	37,337,293	5.83

Table 8: Component sizes and % of edges not participating in communities.

the (undirected) Facebook graph, *DBP* on Web graphs, and *UDBP* for Flickr and LiveJournal graphs. Table 8 shows the number of edges for components $\mathcal{H}_u$ (= $\mathcal{H}$ on *UBP*), $\mathcal{H}_d$ (= $\mathcal{H}$ on *DBP*), and $\mathcal{R}$, and the percentage of $\mathcal{R}$ (the part not forming communities) with respect to the original graph. We also evaluated Web graphs with *UDBP*, and Flickr and LiveJournal with *UBP*, but these were less competitive.

Techniques T4–T11 support community queries on $\mathcal{H}$ and out/in-neighbor queries on $\mathcal{H}+\mathcal{R}$. We measure compression on G by computing $bpe = \frac{bits(\mathcal{H})+bits(\mathcal{R})}{edges(\mathcal{H})+edges(\mathcal{R})}$ and access time $query\_time = query\_time(\mathcal{H}) + query\_time(\mathcal{R})$. Table 9 shows the bpe for $\mathcal{H}_d$ representations of Web graphs, $\mathcal{H}_u$ representation of Facebook, and $\mathcal{H}_u$ plus $\mathcal{H}_d$ for Flickr and LiveJournal graphs. We compute $bpe(\mathcal{H}) = \frac{bits(\mathcal{H})}{edges(\mathcal{H})}$. When using *UDBP* we show $bpe(\mathcal{H}_u)$ and $bpe(\mathcal{H}_d)$ separately. We observe higher compression on Web graphs than on social networks and better results using compressed bitmaps (WT-r). We also show time efficiency for neighbor and different community queries in Table 10 (for *UDBP* the times for $\mathcal{H}_u$ and $\mathcal{H}_d$ must be added together). As it can be seen, *all community queries are supported within a few microseconds.*

Figure 6 shows the space and time on Web graphs and social networks, considering both partitions $\mathcal{H}$ (or $\mathcal{H}_u+\mathcal{H}_d$) and $\mathcal{R}$, and out/in-neighbor queries. On Web graphs, we include the results using k2tree (T2) and VNM + k2tree (T3), recall Figure 2. While in general T2 and T3 dominate the space/time tradeoff, variant T7 (WTs on $\mathcal{H}$ and k2tree on $\mathcal{R}$) is competitive in some cases. Nevertheless, we remind that T4–T11 additionally support community queries.

On social networks, on the other hand, we achieve better results using techniques T4–T11 than using techniques T1–T3 (we include T2, the best performing technique of Figure 3, in Figure 6). Variant T11 provides the least space, whereas variants T9 and T4 provide other relevant space/time tradeoffs. T2 is only mildly relevant on Facebook. There exist other proposals in the literature achieving less space [26], but these do not handle community queries.

5.2 Discussion

Figure 7 shows the histogram of community sizes found in each dataset, and Figure 8 shows the density-normalized distribution. We observe that community sizes on Web graphs are larger than on social networks, which explains the better compression of $\mathcal{H}$ for those graphs.

In Figure 8, we observe that the maximum community density (Def. 2) is 0.6. This is obtained when $|S| = 2$ and $|C| = 3$ or vice versa. We observe that social networks have more of those tiny communities. Second, it can be easily seen from Def. 2 that in a large community where $|S| = |C|$, the maximum density is 0.5. Figure 8 shows that the number of communities with density 0.5 is greater on Web graphs than on social networks. These density observations also explain

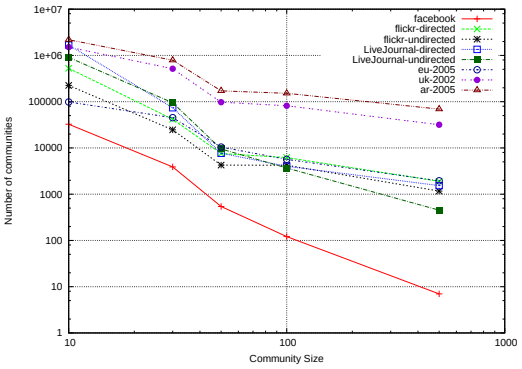


Figure 7: Community size histogram.

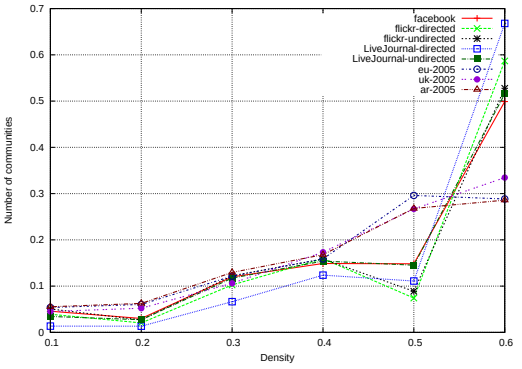


Figure 8: Community density normalized distribution.

why we obtain better compression of $\mathcal{H}$ for those graphs.

When compressing Web graphs with out-neighbor support, we found that it is possible to combine compression schemes that take advantage of different graph properties. Reducing edge redundancy works well with methods that use node ordering to exploit locality and similarity. This combination, however, does not work well on social networks. The reason is that on social networks community sizes are smaller, and then representing the virtual nodes adds a considerable overhead. In the context of compression techniques that support out/in-neighbor queries, the reduction of edges on Web graphs still allows k2tree to exploit sparseness of the adjacency matrix, but on social networks it removes part of the sparseness and degrades the compression.

Our biclique representation of $\mathcal{H}$, instead, does not represent those virtual nodes. This is why it works particularly well on the small communities arising in social networks.

6. CONCLUSIONS

This work has three contributions. First, it offers improved space/time tradeoffs for out-neighbor queries on Web graphs, including the best space requirements reported up to date. We achieve this by combining a technique that reduces the number of graph edges (VNM) [11] with techniques that reorder nodes to exploit ordering, locality, similarity and interval/integer encoding (AD [2] and WebGraph [5, 4]). Second, it achieves improved space/time tradeoffs and the best reported space for supporting out/in-neighbor queries on Web graphs. To achieve this result, we combine VNM with a technique that exploits the sparseness of the adjacency matrix (k2tree) [8]. Third, it proposes a novel

Compression	eu-2005	uk-2002	ar-2005	Facebook	Flickr Directed	Flickr Undirected	LiveJournal Directed	LiveJournal Undirected
WT-32-b	3.82	3.55	3.20	12.11	14.26	13.15	16.91	15.40
WT-32-r	2.77	2.52	2.12	11.72	13.35	12.09	16.15	15.00

Table 9: Compression (bpe) of $\mathcal{H}$ on Web graphs and social networks using compact data structures.

Queries	eu-2005	uk-2002	ar-2005	Facebook	Flickr Directed	Flickr Undirected	LiveJournal Directed	LiveJournal Undirected
Out WT-32-b	6.70	8.71	8.62	6.49	8.91	7.28	12.69	10.16
Out WT-32-r	9.66	12.02	12.03	9.17	12.94	10.45	18.22	14.39
In WT-32-b	6.84	8.62	8.68	8.11	10.15	10.00	13.01	13.25
In WT-32-r	9.44	11.51	11.89	11.50	14.88	14.25	18.88	18.93
Q3 WT-32-b	3.15	6.07	5.12	1.06	0.89	1.04	2.56	2.47
Q3 WT-32-r	3.53	6.87	6.05	1.28	0.98	1.09	2.77	2.59
Q4 WT-32-b	1.99	3.69	2.42	0.85	0.84	0.82	2.12	2.25
Q4 WT-32-r	2.39	4.40	3.04	0.90	0.94	0.96	2.59	2.71
Q5 WT-32-b	2.22	4.10	2.78	1.22	1.40	1.39	3.21	3.27
Q5 WT-32-r	2.61	4.86	3.43	1.31	1.61	1.61	3.68	3.80
Q6 WT-32-b	7.17	8.69	10.14	4.76	5.63	10.24	8.81	7.85
Q6 WT-32-r	8.83	11.79	13.18	7.14	9.01	10.20	11.75	10.46
Q7 WT-32-b	119.21	147.88	233.56	39.22	98.76	162.91	74.71	58.24
Q7 WT-32-r	163.02	199.47	303.11	53.48	114.33	189.35	90.70	78.95

Table 10: Times for community queries ($usecs$) on Web graphs and social networks representing $\mathcal{H}$ with compact data structures.

compressed structure that enables community and out/in-neighbor queries on Web graphs and social networks. This is the first compressed structure that supports community and out/in-neighbor queries.

7. ACKNOWLEDGEMENTS

This work is partially funded by Fondecyt Grant 1-110066.

8. REFERENCES

[1] C. Aggarwal and H. Wang. *Managing and Mining Graph Data*. Springer, 2010.

[2] A. Apostolico and G. Drovandi. Graph compression by bfs. *Algorithms*, 2(3):1031–1044, 2009.

[3] L. Becchetti, C. Castillo, D. Donato, R. A. Baeza-Yates, and S. Leonardi. Link analysis for Web spam detection. *ACM Transactions on the Web*, 2008.

[4] P. Boldi, M. Santini, and S. Vigna. Permuting Web graphs. In *WAW*, pages 116–126, 2009.

[5] P. Boldi and S. Vigna. The webgraph framework I: compression techniques. In *WWW*, pages 595–602, 2004.

[6] P. Boldi and S. Vigna. The Webgraph framework II: Codes for the World-Wide Web. In *DCC*, page 528, 2004.

[7] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks*, 30(1-7):107–117, 1998.

[8] N. Brisaboa, S. Ladra, and G. Navarro. K2-trees for compact Web graph representation. In *SPIRE*, LNCS 5721, pages 18–30, 2009.

[9] A. Z. Broder. Min-wise independent permutations: Theory and practice. In *ICALP*, page 808, 2000.

[10] A. Z. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. L. Wiener. Graph structure in the Web. *Computer Networks*, 33(1-6):309–320, 2000.

[11] G. Buehrer and K. Chellapilla. A scalable pattern mining approach to Web graph compression with communities. In *WSDM*, pages 95–106, 2008.

[12] M. Cha, A. Mislove, and P. K. Gummadi. A measurement-driven analysis of information propagation in the flickr social network. In *WWW*, pages 721–730, 2009.

[13] F. Chierichetti, R. Kumar, S. Lattanzi, M. Mitzenmacher, A. Panconesi, and P. Raghavan. On compressing social networks. In *KDD*, pages 219–228, 2009.

[14] D. Clark. *Compact Pat Trees*. PhD thesis, University of Waterloo, Canada, 1996.

[15] F. Claude and G. Navarro. A fast and compact Web graph representation. In *SPIRE*, pages 118–129, 2007.

[16] F. Claude and G. Navarro. Extended compact Web graph representations. In *Algorithms and Applications (Ukkonen Festschrift)*, LNCS 6060, pages 77–91, 2010.

[17] A. Golynski, J. I. Munro, and S. S. Rao. Rank/select operations on large alphabets: a tool for text indexing. In *SODA*, pages 368–373, 2006.

[18] R. González, S. Grabowski, V. Mäkinen, and G. Navarro. Practical implementation of rank and select queries. In *Posters WEA*, pages 27–38, 2005.

[19] R. Grossi, A. Gupta, and J. S. Vitter. High-order entropy-compressed text indexes. In *SODA*, pages 841–850, 2003.

[20] C. Karande, K. Chellapilla, and R. Andersen. Speeding up algorithms on compressed web graphs. In *WSDM*, pages 272–281, 2009.

[21] M. Katarzyna, K. Przemyslaw, and B. Piotr. User position measures in social networks. In *SNA-KDD*, pages 1–9, 2009.

[22] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.

[23] R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Trawling the Web for emerging cyber-communities. *Computer Networks*, 31(11-16):1481–1493, 1999.

[24] S. Ladra. *Algorithms and Compressed Data Structures for Information Retrieval*. PhD thesis, University of A

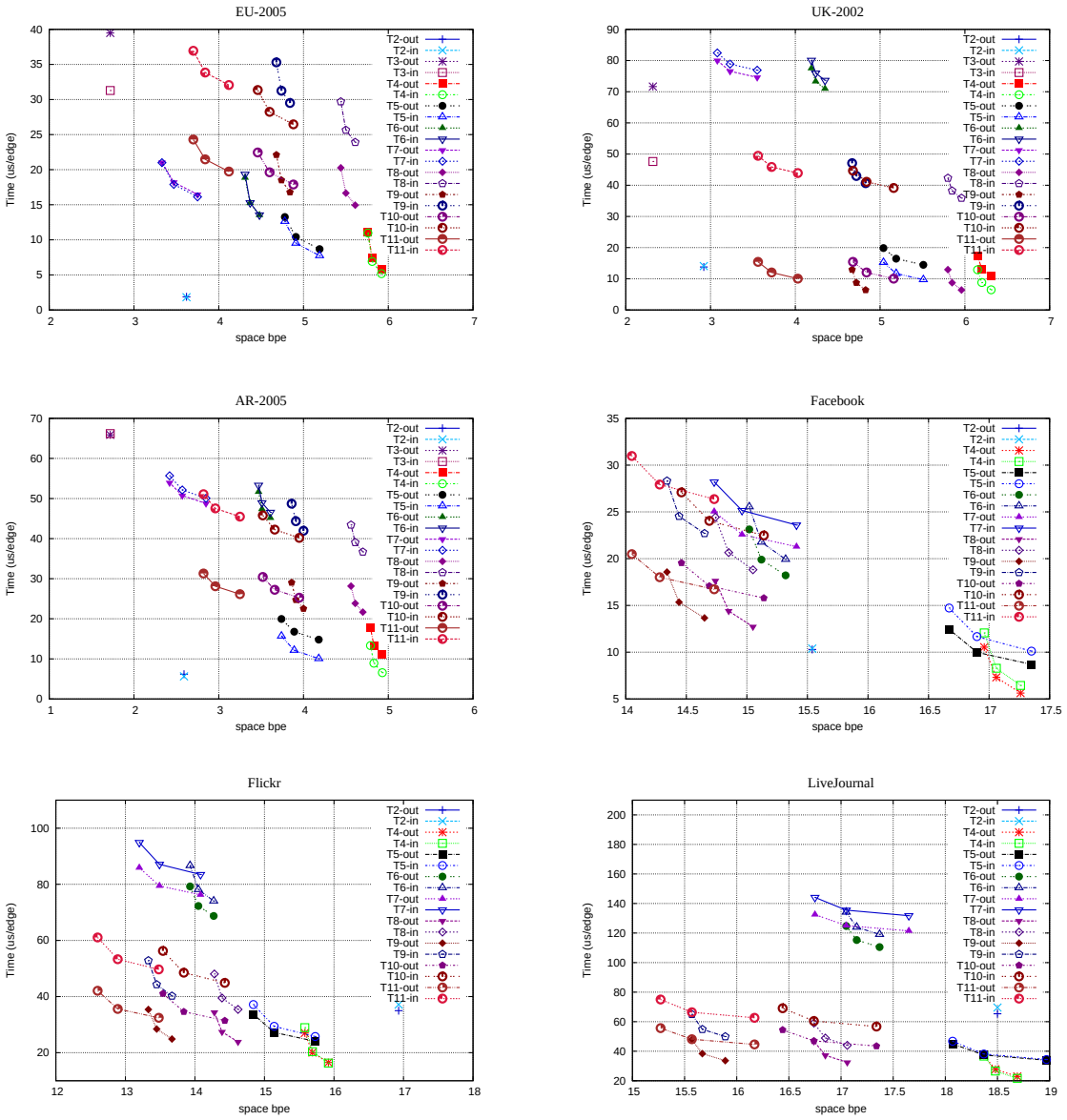


Figure 6: Space/time efficiency with out/in-neighbors queries on $\mathcal{H}+\mathcal{R}$ for representations T4–T11.

Coruña, Spain, 2011.

[25] N. J. Larsson and A. Moffat. Offline dictionary-based compression. In *DCC*, pages 296–305, 1999.

[26] H. Maserrat and J. Pei. Neighbor query friendly compression of social networks. In *KDD*, pages 533–542, 2010.

[27] A. Mislove, M. Marcon, P. K. Gummadi, P. Druschel, and B. Bhattacharjee. Measurement and analysis of online social networks. In *IMC*, pages 29–42, 2007.

[28] R. Raman, V. Raman, and S. S. Rao. Succinct indexable dictionaries with applications to encoding k-ary trees and multisets. In *SODA*, pages 233–242, 2002.

[29] K. H. Randall, R. Stata, J. L. Wiener, and R. Wickremesinghe. The link database: Fast access to graphs of the Web. In *DCC*, pages 122–131, 2002.

[30] H. Saito, M. Toyoda, M. Kitsuregawa, and K. Aihara. A large-scale study of link spam detection by graph algorithms (s). In *AIRWeb*, 2007.

[31] H. Samet. *Foundations of Multidimensional and Metric Data Structures*. Morgan Kaufmann, 2006.

[32] T. Suel and J. Yuan. Compressing the graph structure of the Web. In *DCC*, pages 213–222, 2001.

[33] B. Viswanath, A. Mislove, M. Cha, and K. P. Gummadi. On the evolution of user interaction in facebook. In *WOSN*, 2009.

Social Balance and Signed Network Formation Games

Mohammad Malekzadeh
Department of Computer Engineering
Sharif University of Technology
Tehran, Iran
momalekzadeh@ce.sharif.edu

MohammadAmin Fazli
Department of Computer Engineering
Sharif University of Technology
Tehran, Iran
fazli@ce.sharif.edu

Pooya Jalaly Khalilabadi
Department of Computer Engineering
Sharif University of Technology
Tehran, Iran
jalaly@ce.sharif.edu

Hamid R. Rabiee
Department of Computer Engineering
Sharif University of Technology
Tehran, Iran
rabiee@sharif.edu

MohammadAli Safari *
Department of Computer Engineering
Sharif University of Technology
Tehran, Iran
msafari@sharif.edu

ABSTRACT

This paper presents a game-theoretic approach that models the formation of signed networks which contain both *friendly* and *antagonistic* relations. In our model, nodes have a *pleasure* through their links to the others which is defined according to their triadic relations. They have a self centered goal based on the balance theory of signed networks: *each node tries individually to minimize its stress from undesirable relations with other nodes by maximizing the number of balanced triangles minus the number of unbalanced triangles she participates in.* Since nodes act selfishly and don't consider the social welfare, many theoretical problems about existence of the equilibrium, convergence and players' interaction are raised and verified in this paper. We prove the NP-hardness of computing best-response and give an approximation for it by using quadratic programming and rounding its solution. We show that there is a tight relation between players' best-responses. This result leads to a proof for convergence of the game. In addition, we report some experimental result on three online datasets. We show that as nodes play their best-response and time goes forward, the total pleasure of the network increases monotonically. Finally, we introduce a *smartness factor* for social media sites that helps people to measure the amount of deviation from best possible strategy and suggest a new model that is more adapted to these media. This measure can also be applied in the verification of all kinds of network formation games.

Categories and Subject Descriptors

H.5.3 [**Information Systems**]: Group and Organization Interfaces

General Terms

Human Factors, Measurement, Design

Keywords

Social Network Modeling, Theory of Social Balance, Signed Graphs, Network Formation Games

1. INTRODUCTION

Nowadays, one of the most important technologies for managing social relations are online social networks. Over the past years, analysis of the interaction between people in online social media has been taken into consideration. There exist a wide variety of relations on the online social networks; most of such relations have positive aspect, such as friendship, like, trust, and follow. Alongside these fine relations, there are always some negative relations such as antagonism, distrust, and unlike. The main focus of social network researches has been on the positive relations; however, some recent works investigate the interplay between positive and negative relations in social media sites [15, 8].

Social network analysis is usually based on the network theory concepts studied on the underlying network (graph). Such a network consists of a set of nodes along with a set of edges that connect them.

Our focus in this work is on *signed networks*. In signed networks, every edge has a sign which takes two values: pos-

*The research was partially supported by the Institute for Research in Fundamental Sciences under grant ♯: CS1389-4-09

The 5th SNA-KDD Workshop '11 (SNA-KDD'11), August 21, 2011, San Diego CA USA.

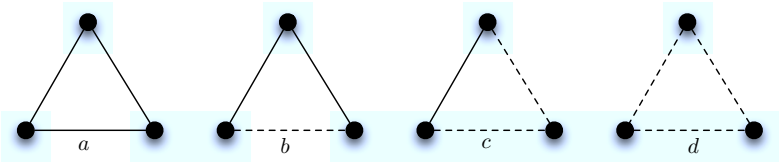


Figure 1: Possible undirected triad in a signed network: triad (a) and (c) are balanced and relatively stable, but triad (b) and (d) are unbalanced and susceptible to break apart. Full and dashed lines represent positive and negative relations respectively.

itive (+) for positive relations or negative (−) for negative relations.

One of the most important problems in the signed networks is the interplay between signs and their effects on the dynamics of the network. Structural balance theory is a basic framework to perceive such interactions between nodes' relations. In this work we present a robust evolutionary model that allows us to see the power of structural balance theory in signed networks.

Balance theory was formed by Heider [11] in 1946. Harary [10] developed some notion about this theory and, later on, Cartwright and Harary [4] extended the results to a graph-based concept. Structural balance theory describes attitudes of individuals to reduce cognitive dissonance among each other. When people set up dyadic relations that contain both positive and negative interactions, four different types of triad relations would be created (Figure 1). In conformity to this theory, we can classify these triangles in 2 classes: *balanced* and *unbalanced*.

A balanced triangle shows a relationship between three people without cognitive disturbance. Such as Figure 1-a that shows three people who are mutual friends and Figure 1-c shows two friends who have a mutual enemy. But in an unbalanced triangle we observe a psychological stress into the relationship. In the Figure 1-b we see a pair of enemies that have a common friend. In this situation these enemies feel pressure from their common friend to become friends; or else there is a pressure from one of the enemies on its friend against the other. In Figure 1-d we watch a challenging state. When three people are enemies, there is a chance for two of them to join together against the third person.

A usual variant of the structural balance theory is weak structural balance proposed by Davis [6]. This variant of the theory relaxes the assumption that "the enemy of my enemy is my friend" and considers a triangle like the one in Figure 1-d as a balanced triangle. Leskovec et al. [15] considered the online social media and counted the number of each triangle in corresponding datasets. They report that the number of triangles with three negative links appear much more than in the randomly generated graphs. These results confirm the validity of the weak balance theory of Davis in the online social media.

Many types of networks connect different individuals and the formation of such networks depends on the decisions of many participants. Network formation models have been widely used in order to characterize how network structures form and how they are affected by individuals' decisions [13]. Aumann and Myerson [2] presented the first network formation model describing the formation of a network in the context of cooperative games with communication structures. Later, several models in this context presented in both on-time and dynamic perspective [19, 12, 3, 5, 14, 9, 7]. Recently Arount van de Rijt [20] attempted to investigate the structure of signed graphs by this type of modeling. He presented a best-response model which has a node utility function that takes its maximum when the number of balanced triads that involve the node are maximized. He proved couple of statements about balanced and stochastically stable graphs that appear in the model. The main difference of this work from the aforementioned works is that he studied the evolution of sign changes while leaving the network structure fixed.

What we present here is a network formation game for studying the evolution of signed networks based on the structural balance theory. The players are nodes and their strategy is creating positive/negative/neutral edges to all other nodes. The edges are undirected and, once created, represent bilateral friendship/antagonism relationship among players, regardless of which node created it. The sum of these edges is the resulting signed graph. Each node has a self-centred goal: to minimize the stress of undesirable relations between itself and all other nodes. We formulate this goal in a best-response model by using the balance theory: each node attempts to maximize the number of balanced triangles minus the number of unbalaced ones. We show that computing the best-response in this model is NP-Hard and present a convex optimization method for its approximation. We also analyse the Nash equilibrium of the model and provide some relevant results.

1.1 Related works

There are many works considering structural balance theory in signed networks. Antal et al. [1] presented simple dynamical rule for resolving unbalanced triads and investigated the resulting evolution of the signed networks. They study the conditions and times under which their model reaches a balanced state. They also characterize a class of jammed states; states in which unbalanced triangles exist and the change of any link's sign increases the number of unbalanced triangles. They proved that jammed states turn out to be much more numerous than balanced states. Following Antal's work, Marvel et al. [17] define an energy landscape for signed networks and find out that the jammed states with higher energy are not only very rare, but they also have inherently greater structural complexity, as measured by their number of balanced cliques.

Ludwig and Abell [16] proposed a model that used balance theory to describe the development of social networks. Their model has two parameter, the first is a friendliness index that help to determine the probability of a link's sign to be positive. The second is a uniformly distributed threshold that indicates the quantity of unbalanced triangles that each

node tolerates.

Some works such as [15, 18] examine the validity of structural balance in online social media sites. Leskovec et al. show that aspects of balance theory hold more strongly on the links that are reciprocated (consisting of directed links in both directions between two nodes). Szell shows that a vast majority of changes in the signed networks are due to the creation of new positive and negative links, not due to switching of existing link from plus to minus or vice versa.

1.2 The Best-Response Model

We consider a repeated game with n players labeled by $1, 2, \cdots, n$. Each player i chooses a $(n-1)$-dimensional vector $L_i \in \{-1, 0, +1\}^{n-1}$ with components l_{ij} where $l_{ij} \in \{-1, 0, +1\}$ for each $j \in \{1, \cdots, N\} \setminus \{i\}$ as its strategy toward the others. At time t, by combining players' strategies $L = (L_1, L_2, , L_n)$ an undirected signed graph G_t with vertices $\{1, \cdots, n\}$ is formed. There is an edge between vertices i and j in G_t iff $l_{ij} \neq 0$. The sign of this edge is equal to the sign of l_{ij}. The pleasure received by player i under L is defined to be

$$p_i(L) = (\triangle_{b_i} - \bigtriangledown_{ub_i}) \tag{1}$$

where $\triangle_{b_i}$ is the number of balanced triangles that i participated in and $\bigtriangledown_{ub_i}$ is the number of unbalanced triangles. By choosing the best possible strategy, players attempt to maximize their pleasure in the network (best-response of player i). Player i is just permitted to play its best response at times $t = kn + i$ for $0 \leq k$. We say that a strategy profile L is a (pure) Nash equilibrium if for each player i, and for all L' that differ from L only in the i^{th} component, $p_i(L) \geq p_i(L')$. In our model computing the best-response for each player is not possible in polynomial time. A proof for this is given in section 2. So we should propose a method for its approximation. We will prove that the best response is an integral answer of a famous semidefinite programming.

2. GAME INTRACTABILITY

In this section, we provide a theoretic interpretation of the game. Let us start with introducing some notations and an easy yet supportive lemma:

- v-triangle: Is a triangle whose vertex set includes v.
- uv-triangle: Is a triangle whose vertex set includes both u and v.
- Δabc: Is a triangle whose vertex set is $\{a, b, c\}$.
- $\delta(A, B)$: The number of triangles whose vertex set include all vertices in A and at least one vertex from B.

LEMMA 2.1. *Assume u is playing its best response in G_t. Adding any set of edges to u with any sign does not decrease u's pleasure.*

PROOF. Let v be a vertex that u is not connected to. Connect u to v with positive sign. Let a be the number of balance uv-triangles and b be the number of unbalanced ones. So, the pleasure of u changes by $a - b$. If we change the sign of uv then the pleasure of u changes by $b - a$ (every balanced uv-triangles becomes unbalanced and vice versa). So, we must have $a = b$ or u is not playing its best response. We use the same argument for every other vertex that u is not connected to and obtain a graph in which u is connected to every other vertex. A side corollary of this result is that there exist a best response playing for player u that connects it to all other vertices. □

One important question while we are studying a game is the computational status of various solution concepts such as best response and Nash equilibirum. We prove that in our game, computing best-response for each player at each time is NP-hard.

THEOREM 2.2. *Given a strategy profile $L \in L_0 \times L_1 \times \cdots \times L_n$ and player $i \in \{1, .., N\}$, computing the best response of i is NP-hard.*

PROOF. The proof follows by reducing from the *Max-Cut* problem. In the *Max-Cut* problem, we are interested in partitioning the vertices of a given graph G into two sets A and B so as to maximize the number of edges between A and B. Let G be a graph that we want to find its maximum cut. Put a negative sign for every edge of the graph. Add a new vertex v to G. We prove that v's best response is, in fact, equivalent to finding a maximum cut in G. By Lemma 2.1, we find a best response playing by v in which v draws edge to every vertex of G. Now partition G's vertices into two subsets:

- $A = \{u | l(vu) = -1\}$
- $B = \{u | l(vu) = +1\}$

We prove that (A, B) is a maximum cut. We have:

$$\begin{aligned} BR(v) = \triangle_{b_v} - \bigtriangledown_{ub_v} &= E(A, B) - E(A, A) - E(B, B) \\ &= 2E(A, B) - E(G) \end{aligned}$$

where $E(X, Y)$ is the number of edges with one endpoint in X and one in Y and $E(G)$ is the number of edges in G. Since v is playing best response, $E(A, B)$ must be maximum over all possible cuts. □

If we allow the edge signs to be real values in the interval $[-1, 1]$ instead of integer values then we can compute the best response of every player in by quadratic programming. Assume we want to compute the best response of u. We first remove u from the network and model its best response by the following QCQP optimization problem.

$$\begin{aligned} &minimize - X_i A X_i^T \\ &subject\ \ to - 1 \leq x_{ij} \leq +1, \quad j = 1, \cdots, n-1 \end{aligned} \tag{2}$$

where n is the number of nodes in the network, the $(n-1) \times (n-1)$ matrix A is the weighted adjacency matrix of $G - \{i\}$, and x is the best-response of node i to the remainder of the network (Figure 2). We know that a triangle containing an even number of negative edges is defined as balanced while a triangle with an odd number of negative edges is unbalanced. Therefore, any integer solution to the above formulation computes the number of balanced triangle minus the number of unbalanced triangles that contain u.

We next consider the repeated game (in which one vertex is picked at each time and plays its best response). Our focus is the convergence of this game. First, we start with the following lemma:

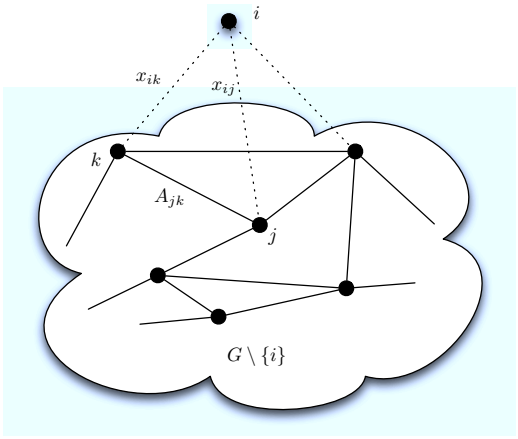


Figure 2: Node i's best-response computation. When node i decides to change her current strategy to achieve a better pleasure, she ignores her current relations and computes her best-response by QCQP optimization. x_{ik} and x_{ij} represent unknown relations that node i attempts to find according to the relations between nodes k and j (i.e. A_{jk}).

LEMMA 2.3. *Let u and v be two vertices that play their best responses at times t and $t+1$, respectively. Then we have*

- *If $l_{vu} = +1$, then we have: $l_{uk} = l_{vk}$, for all $k \neq u, v$ with $l_{uk} \neq 0$.*
- *If $l_{vu} = -1$, then we have: $l_{uk} = -l_{vk}$, for all $k \neq u, v$ with $l_{uk} \neq 0$.*

PROOF. W.l.o.g suppose that at time t, we have $l_{uv} = +1$. The case $l_{uv} = -1$ is treated similarly. We use u's best-response to build a strategy for v and prove that this strategy is strictly better than all other strategies for her. Define the following sets of vertices:

- X: The set of vertices which are connected to vertex u at time t.
- Y: The set of vertices which are connected to vertex v and have no edge to vertex u at time t.

Define $f(u)$ to be the pleasure of vertex u in graph $G_t[X \cup \{u\}]$ (the induced subgraph of G_t over the vertices of $X \cup \{u\}$). Let L_v be a strategy for v in which $l_{vk} = l_{uk}$ for each $k \in X$. At this time, the pleasure of v is equal to $f(u)+|X|$, because its edges to X have the same sign as u's edges to X and for every $k \in X$ we have one balanced uv-triangle, Δuvk. Next, at each step we choose a vertex k' from Y and set $l_{vk'}$ in such a way that it strictly increases the pleasure of v. We will prove that it is always possible.

At time t, there isn't any edge between k' and u. By Lemma 2.1 adding an edge between k' and u does not change u's pleasure. So the number of balanced uk'-triangles and unbalanced uk'-triangles are equal (If the edge uk' exists). This means that the number of uk'-triangles is even. This is true for every other $y \in Y$. Consider two of the vertices in Y, such as y_1 and y_2. We claim that there is no edge between these two vertices. Suppose that y_1y_2 is an edge of G_t. Again by using Lemma 2.1 we can show that the number of uy_i-triangles for $i = 1, 2$ is even. The contradiction happens here:

$$\delta(\{u\}, \{y_1, y_2\}) = \delta(\{u\}, \{y_1\}) + \delta(\{u\}, \{y_2\}) - \delta(\{u, y_1, y_2\}, \emptyset)$$

and we have $\delta(\{u, y_1, y_2\}, \emptyset) = 1$. This contradicts our assumption because left hand side of the above equality is even while the right hand side is odd. So there is no edge between any $y_1, y_2 \in Y$ and $G[Y]$ is independent set.

Now, consider vertices v and k'. The number of uk'-triangles in G_t by assuming existence of uk' is even (note that in this graph vk' is available and there is no edge between vertices of Y). So, the number of vk'-triangles in the graph G_{t+1} and in the absence of uk' is odd because $\Delta uvk'$ won't be enumerated. So we can choose a suitable sign for the edge vk' which makes the number of balanced vk'-triangles more than the number of unbalanced ones. The proof for the claim is now complete and the pleasure of vertex v is at least $f(x) + |X| + |Y|$ which means $BR(v) \geq f(x) + |X| + |Y|$.

Now, suppose that v chooses another strategy S. By using Lemma 2.1 we can assume that in S there is an edge from v to every vertex in X. Define X_1 to be the set of vertices which have edges to both u and v and their edges to these vertices have the same sign (In graph G_{t+1} constructed by adding S to G_t). Similarly define X_2 to be the set of vertices which have edges to both of these vertices but with different signs. Clearly we have, $X = X_1 \cup X_2$. Define X_1' and X_2' in a manner similar to X_1 and X_2 over the graph G_t. We have:

$$BR(v) = f(v) + |X_1| - |X_2|$$

$$BR(u) = f(u) + |X_1'| - |X_2'|$$

We proved that $BR(v) \geq f(u) + |X| + |Y|$, hence by extending this inequality we obtain:

$$f(v) + |X_1| - |X_2| \geq |X| + |Y| + f(u) \tag{3}$$

Now suppose that at time t vertex u uses the best response of vertex v at time $t+1$. Therefore:

$$BR(u) \geq f(v) + |X_1'| - |X_2'| - |X_2| - |Y|$$

From $BR(u) = f(u) + |X_1'| - |X_2'|$, thus:

$$f(u) \geq f(v) - |X_2| - |Y| \tag{4}$$

From Inequalities 3 and 4 we have:

$$|X_1| \geq |X| \Rightarrow |X_2| \leq 0$$

And the proof is complete. □

Lemma 2.3 reveals important information regarding the structure of the game equilibrium and its convergence time. We show that the equilibria of this repeated game are complete graphs and convergence happens after $O(n)$ steps.

THEOREM 2.4. *After one round the graph of the game will be complete.*

PROOF. Suppose that there is no edge between two vertices u and v, and u plays its best response first. Assume

Table 1: A method for making a signed graph undirected. This works because our datasets have very small fraction of reciprocated edges and in the balance theory it is reasonable to expect that the reciprocated edges have same signs. $'+'$, $'-'$, and $'\phi'$ represent positive,negative, and unknown relations between nodes u and v respectively.

k	$u \to v$	$u \leftarrow v$	$u \leftrightarrow v(undirected)$
0	$+$	$+$	$+$
1	$+$	ϕ	$+$
2	ϕ	$+$	$+$
3	$-$	$-$	$-$
4	$-$	ϕ	$-$
5	ϕ	$-$	$-$
6	$+$	$-$	ϕ
7	$-$	$+$	ϕ

that in v's turn, v does not draw an edge to u. So by using Lemma 2.1 we know that drawing an edge with positive sign between these two vertices will not change v's pleasure. This is also true for an edge with negative sign. Define X to be the set of u's neighbors after playing its best response. Now consider two different signing patters for the edges from v to X:

- E_1: For each $x \in X$, vx's sign is as same as the sign of ux.
- E_2: For each $x \in X$, vx's sign is invert of the sign of ux.

When we consider a positive edge between u and v, from Lemma 2.3 we know that E_1 signing pattern is strictly better than all other strategies. However, when we consider a negative edge between these vertices, E_2 is the best strategy for v. But as we justified earlier, these two strategies must have the same pleasure for v, but this implies a contradiction. □

THEOREM 2.5. *After one round ($O(n)$ steps) the game graph converges to an equilibrium.*

PROOF. Suppose that we are at $(n+1)$'th step and it is player one's turn to play its best response. W.l.o.g suppose that the edge between player 1 and player n has positive sign. We know that at the previous turn player n has played its best response. From the proof of the Theorem 2.4 we know that n has drawn edges to all vertices of the graph and its best response follows the properties mentioned in Lemma 2.3, i.e., all the (n, i) edges have the same sign as $(1, i)$ edges have. So this lemma also holds for player 1 and its current edges form its best response. So it has no desire for changing its outgoing edges. □

3. EXPERIMENTAL RESULTS

In this section we explore the above theoretical results experimentally on realistic datasets. For this, we consider a simulation scenario in which we start with an initial network and then in each step nodes iteratively change their strategy. Our simulations are conducted on both random networks and online social media datasets. The results are discussed in the following sections.

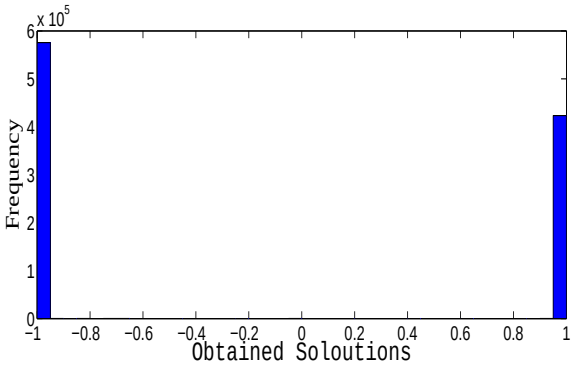


Figure 3: QCQP solution histogram. Frequency of the solutions in a subnetwork of Wikipedia dataset with 1000 nodes. Only very few of the solutions needs rounding and this result shows that our QCQP maximization method has a negligible noise.

As mentioned in the previous section, computing best response in each step is NP-hard (Theorem 2.2) and we need a method for its approximation. As we said, the relaxed problem in which the integral constraints are removed, can be solved by a QCQP (Equation 2). We solve this QCQP and round its solution (values near $\zeta \in \{-1, 0, +1\}$ are rounded to ζ). This would give a good approximation for the problem. Even though we round the obtained solution to the nearest integer value, this process has little effect on the accuracy of the optimized values achieved by relaxation vs. the truly optimal values.

Our experiments on some real subnetworks as well as random graphs show that almost all solutions to the QCQP problem are very close to $+1$ or -1, and there are very few different solutions. In Figure 3 the histogram of the players' solutions in a subnetwork of Wikipedia dataset with 1000 nodes is depicted.

For speeding up our simulation process, we use Lemma 2.3 whenever it can be applied.

3.1 Simulation Scenario

We consider three online social networks that have been used by Leskovec [15]: the trust network of the Epinions, the social network of the blog Slashdot, and the voting network of Wikipedia. While these datasets have hundreds of thousand nodes, for each one, we randomly extract an induced subnetwork that contains 1000 nodes. Moreover, these subnetworks are directed and we make them undirected by the procedure which is described in Table 1. We have two convincing reasons for this process of making undirected: (i) These datasets have very small fraction of reciprocated edges that have different signs(%0.0032 for Epinions, %0.0037 for Slashdot, and %0.0273 for Wikipeida), so the number of deleted edges is too small. (ii) In the balance theory (as opposed to status theory [15]) it is reasonable to expect that the reciprocated edges have same signs since the excessive edges should not have conflict with structural balance property of signed networks.

In addition to this real subnetworks, we use two randomly

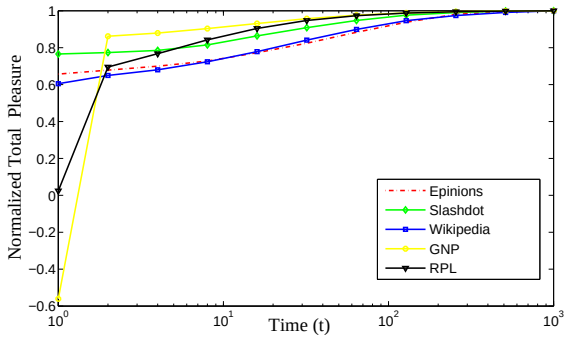


Figure 4: Normalized Total Pleasure of all nodes of the signed network in time units which are power of two. In each time step one node computed her best-response with QCQP maximization method and rounded its solution to values in $\{-1, 0, +1\}$. Each line represents temporal $\mathbb{P}$ for related real or random signed network.

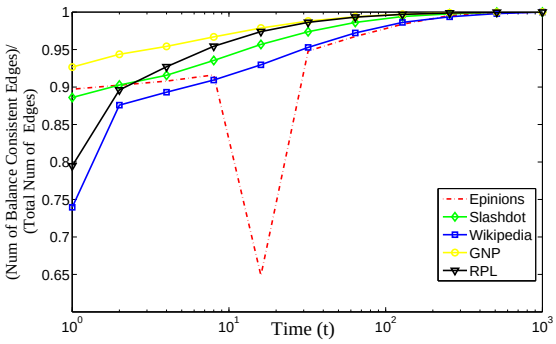


Figure 5: The fraction of edges that are consistent with global view of balance theory computed by Leskovec's method in [15]. After that each node played her best-response we enumerate the number of balance consistent edges with a maximization heuristic method. Each line represents temporal fraction of balance consistent edges for related real or random signed network.

signed networks which are available online[1]. One is a $G_{n,p}$ and the other is a random power law graph, both with 1000 nodes. we start by these networks as initial signed networks. At each time unit, a node is chosen and computes her best-response, i.e., connects some edges to other nodes and chooses their sign. This process is repeated until all nodes apply their best-responses.

Normalized Total Pleasure. We define *normalized total pleasure* ($\mathbb{P}$) as the pleasure of all nodes divided by total number of triangles in the underlying network:

$$\mathbb{P} = \frac{\sum_i p_i}{\sum_{i,j,k} \mid A_{ij} \mid \times \mid A_{jk} \mid \times \mid A_{ki} \mid} \tag{5}$$

In Figure 4, we calculate the normalized total pleasure in time units which are power of two. As we can see, independent of the initial value of $\mathbb{P}$, the normalized total pleasure of the network increases as time goes. One of the main observations of this experiment in random networks is that once the first player plays her best-response, many of the existing unbalanced triangles become balanced and, consequently, the normalized total pleasure of the underlying network substantially increases. Later on, this quantity increases at a decreasing rate. That is, whenever the first few changes in the structure of a random network are made based on the best-response model, the structure of random signed networks approaches to real-world signed networks. Another interesting note about Figure 4 is that in the real-world signed networks, unlike random networks, there is a high initial value for $\mathbb{P}$. The main reason for this difference is that real-world connections are formed according to structural balance theory which makes the pleasure received by each node in these networks high at the beginning.

Balanced Consistent Edges. One of the other properties of the signed networks, is the number of edges that are consistent with Cartwright-Harary theorem [4]. The Cartwright-Harary theorem suggests a global view of structural balance: the network can be divided into two mutually opposite sets of friends, i.e. every pair of nodes that are in the same set like each other, and every pair of nodes that are in the different sets are the enemies. In each time unit we calculate the number of edges that are consistent with this theorem by a maximization heuristic method that is also implemented and analysed by Leskovec [15]: we start by randomly partitioning the nodes into two sets. Then repeatedly pick a random node, and change the set it belongs to if that increases the number of positive edges with both ends in the same set, plus the number of negative edges with ends in opposite sets. Again we run the simulation and enumerate the number of balanced consistent edges in time units which are power of two. Figure 5 shows the temporal fraction of edges that are consistent with generalized balance theory. The existing spike in Epinions diagram is because of the heuristic basis of the method. As nodes change their relation based on best-response model, this quantity becomes large and after a few time units it gets sufficiently close to one.

Smartness Factor. One interesting way in analyzing nodes' behaviors is comparison between the current pleasure that the nodes have, and the new pleasure that they achieve if they play their best-response. Again, we randomly extract an induced subnetwork from each dataset that contains 1000 nodes. We randomly select 100 nodes in each subnetwork and calculate these two parameters: (i) $p_i(0)$ that is the pleasure of node i in its current state. (ii) $p_i(BR)$ that is the pleasure of node i if she uses her best-response to the remainder of the network. Our results show that the former has a wide variety in its value, but the latter has an almost constant value. Based on these two parameters a *smartness factor* is defined:

$$s_i = \frac{p_i(0)}{p_i(BR)} \tag{6}$$

This factor shows currently, how close is node i to her best-response. As shown in Figure 6, there is a big difference between these two values; this can be explained by the as-

[1]You can find this datasets at : http://snap.stanford.edu/na09/

sumption that nodes have limited computational power to compute their best response.

KORIP Model. As it was mentioned earlier, Szell [18] shows that a vast majority of changes in the signed networks are due to the creation of new positive and negative links and not because switching of existing link from plus to minus or vice versa. On the other hand, it is a natural assumption that when two people are friends/enemies, they remain as friends/enemies for a long time. According to these two experimental and theoretical approaches, we introduce an evolutionary model for signed social media that captures this property in a better way.

This model is similar to our best-response model except that when a node decides to play her best-response she Keeps her Old Relations but Improves her Pleasure (KORIP) by creating new connections that are suggested in the best-response. In other word, in this extended model when a node decides to play her best response, she considers two new restrictions: (i) she cannot change the sign of her old relations and (ii) she creates new links, based on her best response, only if these new links improve her pleasure in the network. The specification of the rest of the model remains unchanged.

We simulated the model in two $G_{n,p}$ and random power law graphs with 1000 nodes in which the difference between the number of positive and negative links in it was very small. By considering these graphs as initial signed networks, we let each node play her best response several times according to KORIP restrictions. Unlike the best-response model that reaches a fully balanced network after n times, the KORIP model stops in a an approximately balanced network. As depicted in Table 2, the characteristics of the above signed graphs gets close to the characteristic of the online social media sites after $2n$ times.

4. CONCLUSIONS

Human social networks are often a mixture of friendly and hostile relations that are usually modeled by signed networks. Structural balance theory is a basic framework to interpret the interactions between signs and their effects on the dynamics of the signed networks. This theory discusses every possible type of triadic relations that is formed among people in such networks. Triads that contain odd number of negative signs are said to be unbalanced because there are some pressure or stress on the people involved to change their sentiments. Studies on real-world signed networks (such as [18, 15]) show that these changes lead the network toward an approximately balanced state, where balanced triangles dominate unbalanced ones. We present a model of network formation games that capture the evolution of signed networks based on structural balance theory.

We investigate the theoretical aspects of the game. We prove that computing the best response for a player is NP-hard; we also obtain a good approximation for the best-response. We further prove that after one round the graph of the game will be complete and fully balanced. Also, we examine the best-response model on three online data sets and two types of random networks. We find that in the real-world signed networks, unlike random networks, there is a high initial value for total pleasure because of real-world connections formed according to structural balance theory. Furthermore, we conclude that in each signed network for new users there is an upper bound on the pleasure that they can achieve. Finally, we introduce a variant of the best-response model in which at each time step a user keeps her old relations and improves her pleasure by only creating new connections. We realized that this modified model can fit better in real signed networks modellings.

There are several directions for additional research. First, we aim to have more theoretical and experimental investigation on the KORIP model since its initial results show that it can be useful for modeling formation and evolution of the real social media networks. Second, we can pursue the signed network modeling based on Davis theory [6] in which a triangle with three negative sign is also considered as a balanced triangle. This relaxation is accepted and verified in various social media analysis and, therefore, has considerable motivation for further modeling. Finally, we suggest study of directed networks, where links among people need not be symmetric. In directed network we confront with status theory [15] which postulates that when person i makes a positive link to person j, then i is asserting that j has higher status.

5. ACKNOWLEDGMENTS

The authors are grateful to Mostafa Salehi and Mohsen Jamali for their helpful advices throughout the preparation of this paper.

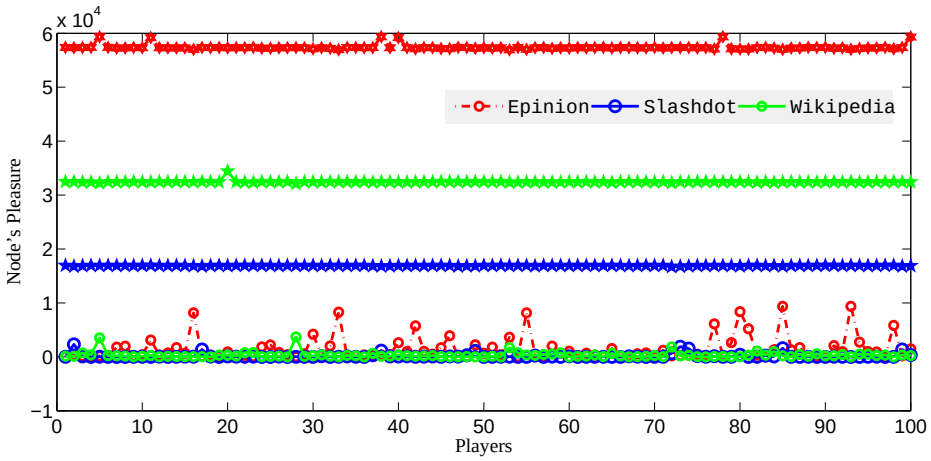


Figure 6: Nodes' pleasure, below lines show current pleasure and above lines show best-response pleasure

Table 2: Characteristic of initial and final signed graph according to KORIP model. T0, T1, T2 and T3 are columns which are used to show the number of triangles of types d, c, b and a of Figure 1 respectively.

	total pleasure	total triangles	balaced consistent edges	ubalaced consistent edges	T0	T1	T2	T3
initial GNP	109	334	8414	2526	41	119	85	103
KORIP GNP	142635186	144442865	945810	1960	226500	107963001	678228	35576025
initial RPL	1959	3218	24698	1947	182	1962	461	627
KORIP RPL	79249412	80559455	737350	1981	162684	59991528	492588	19912906

6. REFERENCES

[1] T. Antal, P. L. Krapivsky, and S. Redner. Dynamics of social balance on networks. *Phys. Rev. E*, 72(3):036121, Sep 2005.

[2] R. J. Aumann and R. B. Myerson. Endogenous formation of links between players and of coalitions: an application of the shapley value. *In: Roth, A. (ed.) The Shapley Value, Cambridge University Press*, pages 175–191, 1988.

[3] D. B. Stable networks. *Journal of Economic Theory*, 76:322–344(23), October 1997.

[4] D. Cartwright and F. Harary. Structure balance: A generalization of heider's theory. *Psychological Review*, 63(5)(277-293), September 1956.

[5] S. Currarini and M. Morelli. original papers : Network formation with sequential demands. *Review of Economic Design*, 5(3):229–249, 2000.

[6] J. A. Davis. Clustering and structural balance in graphs. *Human Relations*, 20(2)(181-187), 1967.

[7] B. Dutta, S. Ghosal, and D. Ray. Farsighted network formation. *Journal of Economic Theory*, 122(2):143 – 164, 2005.

[8] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins. Propagation of trust and distrust. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 403–412, New York, NY, USA, 2004. ACM.

[9] P. J. F. H, M. Wooders, and S. Kamat. Networks and farsighted stability. Technical report, 2003.

[10] F. Harary. On the notion of balance of a signed graph. *Michigan Math. Journal*, 2(2)(143-146), 1953.

[11] F. Heider. Attitudes and cognitive organization. *Journal of Psychology*, 21(2):107–112, 1946.

[12] M. Jackson and A. Wolinsky. A Strategic Model of Social and Economic Networks. *Journal of Economic Theory*, 71(1):44–74, Oct. 1996.

[13] M. O. Jackson. A survey of models of network formation: Stability and efficiency. Working Papers 1161, California Institute of Technology, Division of the Humanities and Social Sciences, 2003.

[14] M. O. Jackson and A. Watts. The evolution of social and economic networks. *Journal of Economic Theory*, 106(2):265 – 295, 2002.

[15] J. Leskovec, D. Huttenlocher, and J. Kleinberg. Signed networks in social media. In *CHI '10: Proceedings of the 28th international conference on Human factors in computing systems*, pages 1361–1370, New York, NY, USA, 2010. ACM.

[16] M. Ludwig and P. Abell. An evolutionary model of social networks. *The European Physical Journal B - Condensed Matter and Complex Systems*, 58:97–105, 2007. 10.1140/epjb/e2007-00200-x.

[17] S. A. Marvel, S. H. Strogatz, and J. M. Kleinberg. Energy landscape of social balance. *Phys. Rev. Lett.*, 103(19):198701, Nov 2009.

[18] S. Michael, L. Renaud, and T. Stefan. Multirelational organization of large-scale social networks in an online world. 2010.

[19] R. B. Myerson. *Game Theory: Analysis of Conflict*. Harvard University Press, Sept. 1997.

[20] A. van de Rijt. The micro-macro link for the theory of structural balance. *Journal of Mathematical Sociology*, 35:94–113(20), January 2011.

News Feed Filtering with Explanation Using Textual Concepts and Social Contacts

Jia-Yan Lin
Graduate Institute of Networking and Multimedia
National Taiwan University
r97944040@ntu.edu.tw

Jane Yung-jen Hsu
Computer Science and Information Engineering
National Taiwan University
yjhsu@ntu.edu.tw

Ting-yen Lee
Graduate Institute of Networking and Multimedia
National Taiwan University
r97944018@ntu.edu.tw

ABSTRACT

In this paper, we investigate whether summarization and social contact information help users receive shared content with higher interest. The ConceptNet, a semantic network of concepts, is used to summarize shared content, each of which is annotated with an explanation about its relevance to the summarized topics as well as the strength of social relation between its sender and receiver. The proposed idea is implemented as a Facebook Application. The results from our experiments show that, with summarization and explanations, shared contents get higher interest ratings from users. In contrast, ranking shared contents with relations between senders and receivers without summarization and explanations does not improve the interest ratings.

1. INTRODUCTION

For many years, activities in the virtual world on the web have become an indispensable part of people's lives. We purchase products, survey goods, gossip rumors or participate in discussion forum on the internet. These activities have become our daily routines.

Communications on the Internet are an important way to keep in touch with friends and families. People actively or passively share recent status, interesting news or moods with their friends.

Social network sharing has been studied in terms of user profiling, tagging behavior and trust system. They all focus on improving the performance of the prediction of user preference. However, most of them go beyond social contacts and try to get some information by consulting "experts". In the leisure time of surfing on the social networking site, people not only gain more knowledge but also try to get closed to friends. Ignoring the social contact will let users tend to miss some important interactions with acquaintances.

People use social networking websites to share and interact with people. But not all users in their contacts they confirmed whould be their friends, families or colleagues. They may be someone that has the same hobby or even someone they don't know and they don't care about. Most of the social network websites assume users may love to share and receive messages from their contacts. However, not always they would like to know everything from their contacts.

This study investigates that if the interest scores for a message will raise or not when we rank shared content based on contact information. Because the connection between contacts inferred by our system may be not intuitive to users, we provide a simple explanation for each highly ranked content. Experiment results show that under a topic, users have more interests in contents if the shared message is given a higher rank and attached an explanation by our system.

2. BACKGROUND AND RELATED WORK

For millions of users, social network sites such as MySpace[1], and Facebook[2] have become an integral part of their daily lives. Many users are "flooded" with information from status updates, news feeds, and blog posts, which created serious *information overload*.

2.1 Information Overload

There has been much research on reducing the information overload[15, 17, 1, 10, 3]. Collaborative filtering is a well-known and popular method adopted by many solutions. Statistic methods, e.g. probabilistic Latent Semantic Analysis (pLSA)[10] and Probabilistic Latent Semantic Indexing (PLSI)[3] are also common.

Another way to manage large set of data is to cluster or summarize data. Scatter/Gatter[2] was a system that scatters the collection into groups, and presents short summaries for each of them to users. TIPSTER SUMMAC[16] had users to judge the relevance of user's initial search query and the summaries of documents that came out as search result. Gong and Liu[7] proposed two methods to generate an overall sense summarization for document's contents. One is by relevance measure and the other is by latent semantic analysis.

The definition of "tag cloud"[18] is a visual presentations of a set of words, where each tag usually denotes a concept related to some objects. Therefore, tag clouds provide a summary or a semantic view of the concepts most important to represent an object[18]. PubCloud[11] summarize results returned by a search and allow the navigation from tag cloud to the results. Xexeo et al.[19] built summary tag cloud by measuring tf-idf for each word in the document collection and selects several top ranking terms.

[1] `http://www.myspace.com`
[2] `http://www.facebook.com`

2.2 Social Network

Lerman[14] reveals that users of the Digg, a social news aggregator site, are more interested in stories from their digg friends that stories recommended using a CF system. Guy[9] investigated the user preference on SONAR[8], estimated the similarity and familiarity between users. The result showed that users are more interested in items shared within familiarity networks than similarity ones.

Lampe, Ellison and Steinfield[12] draw a distinction between the use of Facebook for "social searching" and "social browsing". The former is to find out information about offline contacts, and the later is to develop new connections. A survey found that the primary use of Facebook was for "social searching". Lampe et all.[12] did user studies for student users and results showed they believed that their profiles portrayed them accurately and positively. Other research has shown that Facebook relationships tend to start offline and articulated online[6]. Many words[13, 5, 12] had results that approved communities in Facebook generally corresponed to existing offline network membership.

Demicco[4] investigated on users in IBM network and found out that graduated students who have used Facebook early on campuses would connect current coworkers and friends from school in the past. They also provide more completed profile information and engage in status messaging, wall conversations and photo sharing.

3. METHODOLOGY

We develop a novel way to improve user interest for contextual sharing messages in Facebook, where we call those messages "NewsFeeds". Given a set of content items, NewsFeeds, summarization and explanation serve as a guide to help the target user to grasp recent subjects.

We introduce natural language technique to deal with free text in sharing contents and in users' profiles. Based on ConceptNet, we anlayze the semantic relevance between contents and profiles. Similar contents are grouped together and one *textual topic* would be choosed for each group. Futhermore, several topics are extracted from senders' profile information for each sharing contents they sent.

Users are given a group of topics and each topic associates with a set of sharing contents. Sharing contents, also denoted as streams in this paper, are ranked by the estimated closeness between senders and recipients. Finally, explanations are generated for higher-ranked streams to give a reason why we recommend them.

3.1 Estimated User Closeness

In this section, a way to estimate how close two users might be will be given. The estimation depends on two factors: profile similarity and common friends.

$$\begin{aligned} \mathrm{EC}(u_u, u_v) &= \mathrm{PSim}(P(u), P(v)) * \lambda \\ &+ \mathrm{PRel}(u_u, u_v) * (1 - \lambda) \end{aligned} \quad (1)$$

The notation $\mathrm{EC}(u_u, u_v)$ indicates the estimated closeness between user u_u and u_v. It depends on the *profile similarity*, $\mathrm{PSim}(P(u), P(v))$, between their profiles $P(u)$ and $P(v)$, and also depends on the potential of relation factor, $\mathrm{PRel}(u_u, u_v)$, between u_u and u_v. Two factors will be described more details below.

3.1.1 Profile Similarity

Similarity is measured by content similarity between profiles, which usually can be formed by a list of fields. Some of the fields are required and some may be optional. Preformated fields are easily understood by system.

One the other hand, some fields, like the schools users have attended to, hometown and living places, could be free-formatted. However, many people expect these blanks are for some proper nouns so we can predict the complexity of analyse similarity between values for these kind of fields should be much lessen then dealing free text. Those data could roughly depict a user's background, denoted as "Background Information".

Segments for profile $P(u)$, denoted by $\mathrm{Segs}(u)$, is a set of nouns combined with geographical toponym, building names, school names, discipline titles, etc. $\mathrm{SBSegs}(v)$ is a social background description for u_v. $\mathrm{SBSegs}(v)$ represents an union of $\mathrm{Segs}(u)$ where u_u is a friend of u_v.

$$\begin{aligned} \mathrm{PSV}_v(P(u)) = \quad & [s_1, s_2, s_3, ..., s_n], s_i \in 0, 1, \\ & n = |\,\mathrm{SBSegs}(v)| \end{aligned} \quad (2)$$

$\mathrm{PSV}_v(P(u))$, the profile segment vector of $P(u)$, is a vector of n-tuple where element s_i describes the occurrence of $\mathrm{SBSegs}(v)$'s i'th term in $P(u)$.

$$\mathrm{PSim}(P(u), P(v)) = \cos(\mathrm{PSV}(P(u)), \mathrm{PSV}(P(v))) \quad (3)$$

The profile similarity between the profile of user u_u and that of user u_v, denoted by $\mathrm{PSim}(P(u), P(v))$, is given in equation (3). As shown, similarity is measured by term co-occurrence between profiles. No further natural language processing technique is implemented to deal with ambiguity, typo or synonym, etc. The reason is that segments in the background information should mostly be proper nouns and it's much easier to correct in the user input phase. Besides, proper nouns are domain specific and adjusting them is unsafe without user certification.

3.1.2 Potential Relation

Potential Relation is the connection that is established by intermediaries between two people, in this case, common friends. As we can see, more common friends and higher similarities between the friends and the users result in stronger social strength. Therefore, we measured the similarities between u_u and u_v by averaging the social strength of two users and their common friends.

$$\begin{aligned} &\mathrm{PRel}(u_u, u_v) = \\ &\frac{\sum_{i \in CF(u,v)} (\mathrm{PSim}(P(u), P(i)) * \mathrm{PSim}(P(i), P(v)))}{|CF(u,v)|^{0.5}} \end{aligned} \quad (4)$$

where $CF(u, v)$ denotes the common friends of u_u and u_v, and $|CF(u, v)|$ represents the total number of common frineds.

3.2 Stream Ranking

We rank streams based on how close the sender and the recipient we estimated. According to Section 3.1, we measure user closeness based on *profile similarity*, and the ranking is dependent on Function(1); the higher value, the higher rank; Function(1) contains two factors, profile similarity

and potential of relation, and weighted by coefficient λ. Depending on platform and data source, λ is chosen to balance the bias.

3.3 Natural Language Processing

To analyse the semantic meaning in shared contents and users' profiles, we split the context into peaces of tokens. Algorithms of context tokenization are extremely different in different languages. Although the languages of information in the social network platform are diverse, supporting all existing languages is not the goal of this study.

A feasible way to deal with free-text context with multiple language resources is to identify languages of paragraphs based on char encoding and apply tokenization to them separately. We use this trivial method because it's lightweight and can finish the detection right away. The latency is an important concern in our system since it will be online for service but not in a batch system.

3.3.1 Tokenization

Based on the background of user studies' participants, English and Chinese are two major languages being used in sharing content. We used two public tools to tokenize English or Chinese context separately.

To deal with English context, we used NLTK[3] , a well known Python toolkit, to tokenize context, and then we removed stop-words and inflectional ending for each word.

Yahoo! Developer Network, YDN, provide several api for developers to reach their open services. One of their services, CAS, provided by Taiwan branch, segments Chinese text and labels POS tag for each token. It only took one request to the CAS service to segment a bunch of Chinese paragraphs.

3.3.2 ConceptNet

Once the profiles or shared contents have been split into tokens, we can calculate the semantic relations between tokens by using Commonsense Knowledge Base. To extract the semantics meaning of content, ConceptNet is introduced. Since ConceptNet may be contributed by multiple projects, each of them supports part of knowledge or language. Combining several parts of data sources may complement the information.

In this work, the major languages of experiment participants and stream context are English and Chinese; so, the ConceptNet database is combined with two separated parts, one from English knowledge and the other from Chinese.

ConceptNet collects information between terms and represents its data in a form of a semantic network, which is hard to be deployed in the measurement. AnalogySpace is a way of representing ConceptNet in a multidimensional vector space. It uses dimensionality reduction to decrease the noise and make common sense knowledge information more feasible in computation.

3.4 Topic Extraction

Each stream can be described by multiple topics. Some topics are related with stream context in semantic aspect and some are about sender's background information. We extract keywords from each of streams and group streams according with semantic similarity. Among the streams in the same group, one keyword is chosen to be one of the topic for those streams. Furthermore, tokens from sender's profile content are also topics for streams.

[3]http://www.nltk.org/

3.4.1 Topics from Sender's Profile

Tokens in profile content for the sender of stream M_i^v, noted as Sender M_i^v, are topics for that stream. Profile data is, in some way, partial structured in lots of social networking sites, so we can predict the data type if we know what kind of information the fields are for.

In this work, fields about users education history, current place, place used to be and working corporations are used. Assumed that the data in such fields are lists of proper nouns, where each element could be a school name, a country, a place or a company. No natural language techniques will be applied since most typo can be avoided when users are typing. Thus, we leave it as a system engineer's duty to prevent error data in early stage. Those proper nouns from a sender's profile are *profile topic* for streams he/she shares.

3.4.2 Topics from Information Context

The topic for a stream, related to its content, is called *textual topic* and it is extracted from the union of keywords or *extended concepts* from streams within the same cluster. In the following, we will describe how to extract keywords, cluster streams, extend keywords to concepts and choose a topic for streams within a cluster.

3.4.3 Keyword Extraction and Concept Reformulation

Given a set of streams, M^v, received by u_v. Each stream content will be tokenized, removed from stop-words and applied further processing mentioned in Subsection 3.3. The remaining words were considered as *tokens*.

Keywords K_i^v is a subset of *tokens* of stream M_i^v that elements exist in conceptNet databse. The j'th element of K_i^v, $K_{i,j}^v$, can be presented in the form of vector in AnalogySpace. We compressed the space and each dimension stood for an *abstract concept*.

The semantic meaning of a stream is composed of its keywords and each can be formed by combination of *abstract concepts*. A semantic approximation of stream M_x^v, denoted by $\mathbf{M}_x^v$, is a linear combination of *keywords*. The coefficient of each *keyword* could be determined according to POS-Tagging, google count, TF-IDF, etc.

3.4.4 Stream Clustering

All streams M^v will be divided into groups, the amount of groups is limited by α and β, $\alpha < \beta$. Each group of streams, $C_c = \{M_i^v\} \subset M^v$, is mutual disjoint. The size of each group is also constrained by a soft threshold ρ. First, streams will be clustered by K-means into α sets based on $\mathbf{M}_i^v$ for each M_i^v.

Third, check if there are sets against the soft threshold ρ. If so, those sets will be further divided one by one, from the one with largest distance to the smallest, until the number of sets reaches the constraint β. After a set is split, new sub-sets are required to repeat from step two.

3.4.5 Extended Concept

To expand *keywords* to related and larger group, *extended concepts*, relations between terms in ConceptNet are explored. As mentioned in Subsection 3.3.2, terms in commonsense knowledge base are linked by several pre-defined *rela-*

tions. Toolkit can output terms in database that they are similar, in some aspects of relations, to a given word.

An objective relation, e.g the position or the type of a subject, between topics and text segments may be more reasonable for users than those implicit or emotional ones. Among *relations* defined in ConceptNets, we choose **IsA** and **AtLocation** as the weighted relations. Inferred from a specified keyword through weighted relations, terms with larger score are considered as *extended concepts*. For stream M_i^v, related *extended concepts* is denoted as Ex_i^v.

3.4.6 Choosing Topics

Given a cluster, C_c, the union of keywords K_v and their individual *extended concepts* are topic candidates. Distances between topic candidates to all streams within the same cluster could be assessed. *Textual Topic* for C_c is the candidate that with the minimum distance sum. If a stream M_i^v belongs to C_c, the *textual topic* for that cluster is also a *textual topic* for M_i^v.

3.5 Explanation Generation

Explanation differs by the way how users browse streams. Our system retrieve recent shared content items and produce ranked lists which may have several related topics respectively. Users have two ways to browse streams: first, list higher ordered recent items; second, list higher ordered streams described by a specified topic. Explanations will give users a reason that why a stream is ranked high or why a stream is related to the topic.

3.5.1 Explanation for Order

Streams are ranked by extimated closeness between senders and the recipient (see Function(1)), which has two factors, one is profile similarity(3) and the other is the strongness of Potential Relation (4). If a streams is ranked higher than others, it means that it's summed up value is larger than others'. The explanation will be given depends on which factor contributes more value.

3.5.2 Explanation for Topic Relevance

A stream are describes by two kinds of topic: *textual topics* or *profile topics*. If a stream is listed because a recipient click a *profile topic*, explanation will tell that the specified topic is written by the sender in profile. If the clicked one is a *textual topic*, it might be a keyword written in streams within the same cluster, or a *extended concept*, inferred from keywords.

If the topic is directly written in the context, the explanation will simply say so. If the topic is extracted from other stream or *extended concepts*, the explanation will tell that the context is semantically related to the topic. No information of how the inference be or how related they are will be depicted because too much detail in the explanation distracts users.

4. SYSTEM ENVIRONMENT

System utilized the above algorithms to serve as an application on Facebook[4] platform, which is called UrNewPage (pronounce: 'your new page'). In the following sections, we will briefly introduce the background of Facebook Development Platform and explain System model and UI design both in detail.

[4]`http://www.facebook.com`

4.1 Facebook Development Platform

Facebook is a well-known social networking website platform where various third-party applications have accesses to users' social information. Once user log-in to the Facebook, they are redirected to a page, so called "Wall Page", where they can browse recent shared streams or publish some new ones. An instance later, if they would, switch to another application built on the Facebook Development Platform. In addition to that, users have their own individual wall page along with contents posted by friends or themselves. When it comes to wall page, users can express personal point of view, while other people's comments almost immediately updated by the system. The refreshing action is able to accommodate more friends posting relevant information.

Official statistic page[5] showed that the site has had more than 400 million active users by February 2010, and more than 5 billion pieces of content shared each week. Each user has 130 friends on the site but only clicks the "Like" button on 9 pieces of content each month in average.

4.2 System Flow

This phenomenon means that Facebook users eager to get a better flow of information services along with endless sharing contents. First of all, UrNewPage, summarizes topics of shared contents to make users obtain latest information from senders. Secondly, clicking on topics will start from the non-relevant page. Furthermore, the ranking will refer to an interesting background or close friends. Next, in terms of style and template design, Facebook provides users seamless browsing experiences.

4.3 System Resource and Data

In the Facebook Platform, streams shared by friends could be retrieved by applications for the last 180 days. No clue that whether a stream is seen by a user attached. UrNewPage retrieves the latest N streams for each user once they enter the application. The presumption of these streams is not be browsed by the receiver. Their content are multimedia data, but only textual part will be extracted and used in computation.

When a user joins the application, his/her profile will be copied and every time he/she enter, his/her friendlist is updated. Every profile of joined users or their friends will be updated periodically.

4.4 System Constraint and User Demand

There are three dimensions, including "iframe", "fbml" and "connect", to be used as applications connecting to Facebook. As for "IFrame", it has as many multi-features as "connect" does, whereas the advantage lies in auto-switch new pop-out after activating the application. On the other hand, Low latency is a critical requirement for online service, because time-consuming computation needs to be simplified or moved from online. However, content of streams varies case by case and barely to pre-processing on online. Adopt Natural Language Processing technique suitable for online processing is needed.

4.5 User Interface and User Interaction in System

To achieve the goal of seamless experience for users, the layout of interface have to in accordance to the design rule of

[5]`http://www.facebook.com/press/info.php?statistics`

facebook. Nevertheless, as for an "iframe" application, there seems to be inevitable that advertising space on the right. A concession that two columns left; the left one contains self profile picture and tag-cloud, while the right one shows streams in users interest rows by rows.

Figure 1: Screenshot of UrNewPage

As in Figure 4.5, the right column filled by, at most ρ, top ranking streams from recent N publication, once user enter the main page and haven't click any tags. The left column contains a tag-cloud, in which each tag varies in color and size.

The Bigger the tag is, more streams related with. Since some tags are too long, especially those extracted from user profile, such as "united_ state" or "taiwan" in Figure 4.5, and line breaking is inevitable. A light shadow and a full-text appear when a cursor hover over a tag that let users see the entire tag. Clicked tags will be marked with darker background that hint which tags are read.

Once a user click a tag, related streams will list rows by rows in the right column. Streams are composed of sender's profile picture, message body, information about publishing application and comments are pending below if any. Message may be text, multi-media data or a mixture of them.

The pool of streams list for a tag should be complied with read. A threshold, ρ, is introduced as a soft upper-bound among a bundle of streams shown in page. If the number of related streams is more than ρ, only the top ρ ones will appear whereas others is trimmed off. In this work, ρ is set to 20.

5. EXPERIMENT ENVIRONMENT

In order to make the environment closer to the actual environment, the experimental platform takes advantage of UrNewPage system making users browse in the normal course of study. The screenshot of main page in Figure 5 and the instruction of experiment flow is written above the tagcloud in the upper-left place. And a rating guideline for each list of streams will attach over streams in the right column. It also explains which tag these streams are referred to or none of all. A set of star-rating, attached to a stream, stands for 1 to 5 point. Users are asked to rate each of stream based on the requirement of rating guidelines.

To evaluate this study effectively, it is necessary to embark on four various user studies. Each test conducted in experiment would be integrated in the same environment and once user entered the system will randomly assign the user a user study. The assigned study of 2nd and the 4th user would be analyzed with respects to rating scores without specific

Figure 2: Screenshot of UrNewPage, integrated with experiment

tag. Their content fill out the right column and was flush out if tag been clicked. The study of 1st and the 3rd user aims to examine the effectiveness of this work upon streams expanded by a tag. If a user clicked a tag, a list of stream showed in the right column. In the following, detail of user studies and involved rating guidelines are stated.

5.1 Rating Guidelines

The two proposed evaluation criteria are ready to name for Rating Guild line #1 and #2, respectively, which demonstrated in the version of English and Traditional Chinese. Text will be displayed below.

Interest Criteria "On a scale of ONE STAR to FIVE STARS with ONE STAR being "uninterested" and FIVE STARS being "highly-interested", how do you rank the following feeds according to your interests?"

Relevance Criteria "On a scale of ONE STAR to FIVE STARS with ONE STAR being "unrelated" and FIVE STARS being "highly related", how do you rank the following feeds according to your selected topics?"

5.2 User Study No.1

This test caters to analyzing the effectiveness of our proposed method to rank streams involving with a given tag. Baseline is ranking streams by publish time. Once users load into the main page, as seen in Figure 5, a set of tagcloud along with a list of streams are provided. They are asked to surf around and rate lists of streams they had seen, including the list just after page loaded or others triggered by clicking a tag. No restriction or requirement how many tags users have to click.

Except for the list without specific tag, the Rating Guideline for each list of streams may be #1 or #2. When each time slot used by a user in experiment, which rating guideline will be used was randomly determined and every list of rating will share the same. For list without specified tags, guideline must be the #1 since no target tag is related to.

There are two methods were ranked in terms of streams, one was proposed by this work, while the other was based on timestamp. As of rating guideline, ranking functions was determined by user access to apply the whole streams within the section. How streams ranked was hidden from users and how many methods or which method was been used wasn't explicitly reveal.

Whole streams extracted by system for each experiment target is the latest N, $N = 100$, ones. The length for each list of streams, expanded after clicking tags, is at least 1 and at most ρ, $\rho = 20$. The number of streams shown without tag referred is no more than 15. Some parameters are defined; let $\lambda = 1$(see equation 1), closeness between message sender and receiver are calculated directly from profile similarity.

5.3 User Study No.2

As of this test, the ranking effectiveness for larger set of streams without any tags specified is analyzed. This user study combined with the No.1 that the list of streams without specific tag may be replaced by another list for this test. Therefore, user study No.2 was taken with user study No.1 in the meanwhile.

Based on the previous test, our system extract the latest N, $N = 100$, streams. The top 50 streams ranked by our strategy or by baseline were shown to users. The index of each stream in different ranking functions was hidden from users.

Given parameter $\lambda = 0.5$, PRel between senders and the receiver is taken into account. It got chances that an online processing for PRel for the participant and friends had have done. Once the completeness of PRel was certified, the participant could take this test by chances.

5.4 User Study No.3

Adopted from user study No.1, this test has more valid explanation and more rating guidelines. The procedure was the same as the user study No.1 but right to each stream an explanation was attachment (see Figure 5). Rating guidelines were divided into 3 options that randomly selected from #1 and #2.

5.5 User Study No.4

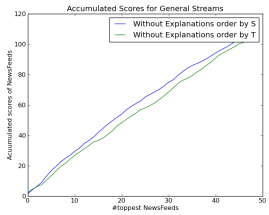
To further delve into the influence of explanation with respects to larger set of ranked streams, conditions almost the same as user study No.2, a list of randomly ordered N, $N = 100$, latest streams were extracted and the top 50 items ranked by our strategy shown to users. The difference of user study No.2 is that the top 25 were attached an explanation.

6. EXPERIMENT RESULT

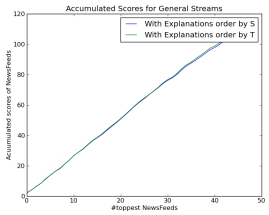
There are 20 unique participants in User Study #1 and 24 in User Study #2, #3 and #4. We received 150 rating lists in User Study #1, 3 in #2, 78 in #3 and 8 in #4 where for each user studies the total number of rated NewsFeeds were 315, 150, 530 and 400 correspondingly.

6.1 General Streams

In User study #2 and #4, participants were asked to rate NewsFeeds with explanations or without explanations based on their interest. The result is shown in Figure 6.1. Unfortunately, scores seem quite low for both methods that participant felt uninterested in what they received. In Figure 3(a),



(a) Rating for Interest in user study #2



(b) Rating for Interest in user study #4

Figure 3: "S" is an abbreviation of "our strategy" and "T" is of "timeline".

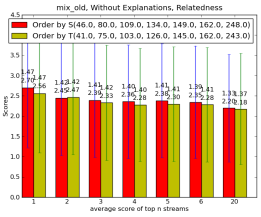
our method got bigger AUC (area under curve) than the baseline; however, result of two methods seem no different even if explanations were provided; see Figure 3(b).

A good reason for this phenomenon is that participants were flooded by posts where most of them were unimportant to him/her. Spammed by friends may be the most troublesome problem in surveying sharing contents on social networking sites.

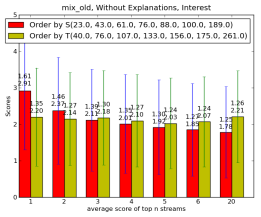
In the next section, the rating of grouped and summarized NewsFeeds will be analyzed and see if our ranking function and explanations help participants.

6.2 Tagged Streams

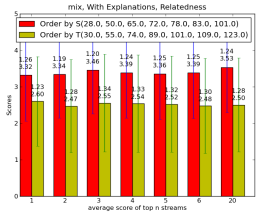
In this section, we analyze the scores of tagged NewsFeeds in User Study #1 and #3. Average and standard variation of scores of highest 1 to 20 posts in every list are calculated, see Figure 6.2.



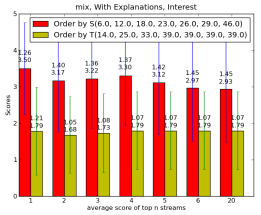
(a) Rating for Relevance in user study #1



(b) Rating for Interest in user study #1



(c) Rating for Relevance in user study #3



(d) Rating for Interest in user study #3

Figure 4: "S" is an abbreviation of "our strategy" and "T" is of "timeline". Standard variation and average of scores are denoted above each bar.

6.2.1 *Summary Relevance For Posts With and Without Explanation*

It makes sense if the Relevance scores of our strategy and of baseline seem no different, see Figure 4(a), and posts in both of user studies seem lack of relevance to clicked tag. It meets our expectation because the relevance between NewsFeeds and clicked tag is affected by the performance of summarization not the ranking function. However, participants agree that clicked tag and posts it triggered are more related when explanations were provided than without explanations.

Providing explanations of how posts and clicked tag are related, users tended be convinced. Even though explanations only provided intuitive information that clicked tag existed in context or senders' profiles or showed rough information that the post was related to others in the same group.

6.2.2 User Interest for Posts without Explanations

The results of user interest for streams sorted chronologically or sorted by our strategy without explanations, showed in the Figure 4(b), revealed that the majority of NewsFeed participants received is annoying. Our strategy doesn't influence participants how much interest they felt for all received posts according to the outcome that there are no difference between scores for the highest 20 posts (all posts) ranked by our method and by the creation time.

However, the slope of bars for scores "Order by S" told that the higher the score predicted by our strategy the higher the chance participants may be interested. Regardless of sharing content, prediction based on social contacts chose good NewsFeeds among those uninteresting ones. Compared with content analysis or user behavior history analysis, our ranking method takes much lesser pre-computing time and lighter computation load.

Most of services outside of data owner may be restricted in accessing data and that will limit the resources in analysis and lengthen the learning period for better performance. In such a case, services may take advantage in boosting performance by using contact information.

6.2.3 User Interest for Posts with Explanations

In Figure 4(d), our methods performed much better than baseline. Since we didn't ask participants to click tags based on a certain rule, we left users totally free to choose what and how much they want and when to end the process. It simulated the user behavior when they were just online, searched around and wondered what messages they received recently. Ratings for NewsFeeds without explanations and were ranked chronologically were quite low which revealed that the subset of NewsFeeds, those related to tags participants clicked, were still uninteresting. However, when we provided explanations, participants were influenced and mush more interested in content they received.

7. CONCLUDING REMARKS

Our work showed that the explanation and social contact based ranking promote information that are more interesting to users. Explanations also gave an information that convinced user the relatedness between summarization and contents. When we provided explanations and summarized topics to users, they felt more interested to those posts recommended by our methods. On the other way, our system had better performance on those contents with topic tagged than general contents. It's reasonable because when users have a way to narrow down the topic, they are more likely to get close to those contents that they will be interested in.

However, on the Facebook, users can grant permissions to any applications to send out posts on behalf of them, and those contents usually have the same pattern. Our summariztion tended to group those similar content under a same topic, and when user clicked on those topics, they saw similar posts on the page. Once users saw similar contents, usually sent by the same application, from a same sender and our system listed those posts together, participants in our experiments gave feedbacks that the contents shown on the page were duplicated.

Our system used a quite simple strategy to rank contents, showed a visualized summarization and gave explanations on summarizations and rankings. Experiment results gave a positive response to the improvement of recommending interesting Facebook shared contents to users.

Acknowledgments

This research was supported by a grant from the National Science Council of Taiwan, NSC 99-2221-E-002-139-MY3.

8. REFERENCES

[1] M. Balabanović and Y. Shoham. Fab: content-based, collaborative recommendation. *Commun. ACM*, 40(3):66–72, 1997.

[2] D. R. Cutting, D. R. Karger, J. O. Pedersen, and J. W. Tukey. Scatter/gather: a cluster-based approach to browsing large document collections. In *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '92, pages 318–329, New York, NY, USA, 1992. ACM.

[3] A. S. Das, M. Datar, A. Garg, and S. Rajaram. Google news personalization: scalable online collaborative filtering. In *Proceedings of the 16th international conference on World Wide Web*, WWW '07, pages 271–280, New York, NY, USA, 2007. ACM.

[4] J. M. DiMicco and D. R. Millen. Identity management: multiple presentations of self in facebook. In *GROUP '07: Proceedings of the 2007 international ACM conference on Supporting group work*, pages 383–386, New York, NY, USA, 2007. ACM.

[5] N. Ellison, C. Steinfield, and C. Lampe. Spatially bounded online social networks and social capital: the role of facebook. In *In Proceedings of the Annual Conference of the International Communication Association*, 2006.

[6] N. B. Ellison, C. Steinfield, and C. Lampe. The benefits of facebook f̈riends:s̈ocial capital and college students' use of online social network sites. *Journal of Computer-Mediated Communication*, 12(4), 2007.

[7] Y. Gong and X. Liu. Generic text summarization using relevance measure and latent semantic analysis. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 19–25, New York, NY, USA, 2001. ACM.

[8] I. Guy, M. Jacovi, E. Shahar, N. Meshulam, V. Soroka, and S. Farrell. Harvesting with sonar: the value of aggregating social network information. In *CHI '08: Proceeding of the twenty-sixth annual SIGCHI*

conference on Human factors in computing systems, pages 1017–1026, New York, NY, USA, 2008. ACM.

[9] I. Guy, N. Zwerdling, D. Carmel, I. Ronen, E. Uziel, S. Yogev, and S. Ofek-Koifman. Personalized recommendation of social software items based on social relations. In *RecSys '09: Proceedings of the third ACM conference on Recommender systems*, pages 53–60, New York, NY, USA, 2009. ACM.

[10] T. Hofmann. Latent semantic models for collaborative filtering. *ACM Trans. Inf. Syst.*, 22(1):89–115, 2004.

[11] B. Y.-L. Kuo, T. Hentrich, B. M. . Good, and M. D. Wilkinson. Tag clouds for summarizing web search results. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 1203–1204, New York, NY, USA, 2007. ACM.

[12] C. Lampe, N. Ellison, and C. Steinfield. A face(book) in the crowd: social searching vs. social browsing. In *CSCW '06: Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*, pages 167–170, New York, NY, USA, 2006. ACM.

[13] C. A. Lampe, N. Ellison, and C. Steinfield. A familiar face(book): profile elements as signals in an online social network. In *CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 435–444, New York, NY, USA, 2007. ACM.

[14] K. Lerman and A. Galstyan. Analysis of social voting patterns on digg. In *WOSP '08: Proceedings of the first workshop on Online social networks*, pages 7–12, New York, NY, USA, 2008. ACM.

[15] P. Maes. Agents that reduce work and information overload. *Commun. ACM*, 37(7):30–40, 1994.

[16] I. Mani, D. House, G. Klein, L. Hirschman, T. Firmin, and B. Sundheim. The tipster summac text summarization evaluation. In *Proceedings of the ninth conference on European chapter of the Association for Computational Linguistics*, pages 77–85, Morristown, NJ, USA, 1999. Association for Computational Linguistics.

[17] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *CSCW '94: Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186, New York, NY, USA, 1994. ACM.

[18] A. W. Rivadeneira, D. M. Gruen, M. J. Muller, and D. R. Millen. Getting our head in the clouds: toward evaluation studies of tagclouds. In *CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 995–998, New York, NY, USA, 2007. ACM.

[19] G. Xexeo, F. Morgado, and P. Fiuza. Differential tag clouds: Highlighting particular features in documents. In *WI-IAT '09: Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, pages 129–132, Washington, DC, USA, 2009. IEEE Computer Society.

Entropy-based Classification of 'Retweeting' Activity on Twitter *

Rumi Ghosh
USC Information Sciences Institute
Marina del Rey, CA 90292
{rumig}@isi.edu

Tawan Surachawala
USC Information Sciences Institute
Marina del Rey, CA 90292
{tawans}@isi.edu

Kristina Lerman
USC Information Sciences Institute
Marina del Rey, CA 90292
{lerman}@isi.edu

ABSTRACT

Twitter is used for a variety of reasons, including information dissemination, marketing, political organizing and to spread propaganda, spamming, promotion, conversations, and so on. Characterizing these activities and categorizing associated user generated content is a challenging task. We present a information-theoretic approach to classification of user activity on Twitter. We focus on tweets that contain embedded URLs and study their collective 'retweeting' dynamics. We identify two features, time-interval and user entropy, which we use to classify retweeting activity. We achieve good separation of different activities using just these two features and are able to categorize content based on the collective user response it generates. We have identified five distinct categories of retweeting activity on Twitter: automatic/robotic activity, newsworthy information dissemination, advertising and promotion, campaigns, and parasitic advertisement. In the course of our investigations, we have shown how Twitter can be exploited for promotional and spam-like activities. The content-independent, entropy-based activity classification method is computationally efficient, scalable and robust to sampling and missing data. It has many applications, including automatic spam-detection, trend identification, trust management, user-modeling, social search and content classification on online social media.

Keywords

Twitter, dynamics, classification, entropy

1. INTRODUCTION

Twitter has emerged as a critical factor in information dissemination, marketing [19], and influence discovery [9]. It has also become an important tool for mobilizing people, as witnessed by the events of the 2011 'Arab spring' [1, 5], and for crisis management, when it was used to reconnect Japanese earthquake victims with loved ones and to provide real time information during the subsequent nuclear disaster [13]. In the cultural arena, Twitter has developed into an effective mouthpiece for celebrities [3], spawning a generation of stars, like Justin Bieber, and starlets [2]. As a consequence, we have seen the rise of new social marketing strategies and sophisticated automated promotion campaigns. Information dissemination, advertising, propaganda campaigns, bot retweeting and spamming are some of the many diverse activities occurring on Twitter. Understanding these activities and their intent will lead to better tools for trend identification, spam detection, and improve user modeling and content analysis.

Through examples of retweeting activity, we illustrate the richness of Twitter dynamics in Section 2. Differentiating between these diverse activities on Twitter and classifying the short posts is a challenging problem. For example, a post that is retweeted multiple times by the same user may be categorized as spam. However, if the same message is of interest to and retweeted by many other users, it can be classified as a successful campaign or information dissemination. Such judgements are difficult to make based solely on content. The advent of bots and automatic tweeting services have added another dimension of complexity to the already difficult problem. How do we distinguish human activity from programmed or bot activity, campaigns designed to manipulate opinion from those that capture users' interest, and popular from unpopular content?

We propose a novel, quantitative approach to address these questions. In Section 3 we describe an *information-theoretic method* to characterize the dynamics of retweeting activity generated by some content on Twitter. While content- and language-independent, our method is nevertheless able to categorize content into multiple classes based on how Twitter users react to it: it can separate newsworthy stories from those that are not interesting, campaigns that are driven by humans from those driven by bots, successful marketing campaigns from unsuccessful ones. Previous work provided a binary (such as low-quality vs. high quality content) or tertiary classification of content based on analysis of content and structure [4] or user response to it [10]. However, the rich, heterogenous and complex activity on Twitter necessitates the need for a more detailed characterization.

When a user posts or 'tweets' a story, he exposes it to other Twitter users. We focus on tweets that contain URLs and

*The 4th SNA-KDD Workshop '11 (SNA-KDD'11), August 21, 2011, San Diego CA USA .

use these URLs as markers to trace the spread of information or content through the Twitter population. When a later tweet includes the same URL as an earlier one, we say that the new post *'retweets'* the content of the original tweet. We do not require the retweet to contain 'RT' string nor check that the user follows the author of the original tweet. Our retweets not only include traditional retweets from the original author's followers, but also conversations about the content associated with that URL and independent mentions of it. The collective user response to the tweet, what we call the *retweeting activity*, varies with the nature of content and users' interest in it, leading to characteristic dynamic patterns. For example, a popular news story will be retweeted by many different users (but only once by each user), whereas campaigns will get many retweets but from the same small group of users. Some retweets, however, could be automatically generated, and relying purely on frequency of retweets will mislead as to the popularity of content. The temporal signature of automated retweeting is drastically different from human response, allowing us to easily differentiate between them.

Given some content (URL), we characterize its retweeting dynamics by two distributions: distribution of the time interval between successive retweets and distribution of distinct users involved in retweeting. We use *entropy* to quantitatively characterize these distributions. We show that these two numeric features capture much of the complexity of user activity in Section 4. Using these features to classify activity on Twitter, we have been able to identify several different types of activity, including marketing campaigns, information dissemination, auto-tweeting and spam. In fact some of the profiles, that we have correctly identified as engaging in spam-like activities have been eventually suspended by Twitter. Our simple yet powerful approach can easily separate newsworthy content from promotional campaigns, independent of the language of the content, and provides an objective measure of the value of content to people.

2. DYNAMICS OF RETWEETING ACTIVITY

User's response to content posted on Twitter is encoded in the dynamics of retweeting of this content. Figure 1 shows the cumulative number of times nine different URLs were retweeted vs time. The figures show a wide variety of collective response to content. Figure 1(a) shows a characteristic response to newsworthy information: fast initial rise followed by a slow saturation in the number of retweets. Such response is typical of diffusion patterns of newsworthy information in online social networks [15, 16, 18]. A similar trend is also observed in the response to content (often photos) posted by major celebrities, as Fig 1(b).

Retweeting activity of posts made by starlets (without major following) is starkly different from that of stars. Figure 1(d) shows retweeting activity of a post by Young Dizzy, an aspiring artist and songwriter. Short bursts of intense activity are followed by long periods of inactivity. As we show later, this is one of the characteristics of automated tweeting, an increasingly popular feature on social media. In many of these cases, such automated retweets are generated by one or a small groups of users, pointing to attempts to manipulate the apparent popularity of content. Such automated methods to boost popularity are used not only by aspiring starlets, but also by dedicated fans of major stars, e.g., Justin Bieber as shown in Figure 1(e). In this case, fans are asked to register their Twitter accounts on a fan web site, which then automatically tweets posts about the star from their accounts. There are other example where users (or a small group of users) retweet the same message multiple times, often with the aid of some automated service, leading to a spam-like campaign. This is shown figures Figure 1(g) and Figure 1(h). One of these accounts EasyCash435 was eventually suspended by Twitter. Figure 1(i) shows similar characteristics of some content in Japanese. Note, that using only the retweet dynamics, without any knowledge of the content, we are able to deduce the spam-like advertisement campaign that this profile engages in. This is confirmed by analyzing content.

In addition to information dissemination, automated tweeting, promotional activities and advertisements, campaigns add to the diversity of Twitter dynamics. One of the successful campaigners in our sample was a Brazilian politician Marina Silva. Figure 1(c) traces the retweeting activity of a post made by her over a period of 4 days. Every day she posts the same link using the social media dashboard Hootsuite (www.hootsuite.com). The retweeting activity follows a news-like trace seen in (a) and (b). However, when the activity gradually slows down, she breathes new life into the campaign by retweeting the same URL, generating a new upsurge in interest (and retweeting). Contrast this with an not-so-popular animal rights campaign shown in Figure 1(f), where the same few users (as shown later) are repeatedly manually retweeting some content to raise its visibility.

3. ENTROPY-BASED ANALYSIS

Manual analysis of retweeting activity on Twitter is labor-intensive. Instead, in this section we describe a principled approach to categorize retweeting activity associated with some content.

Problem Statement. Given some user-generated content or tweet $c_j \in C$(where C is a set of tweets or content), our aim is to analyze the trace, $\mathcal{T}_j \in \mathcal{T}$ (where $\mathcal{T}$ is the collective activity on all content), of retweeting activity on it, to understand the content and associated dynamics. This trace, $\mathcal{T}_j$ can be represented by a sequence of tuples $((u_{j1}, t_{j1}),(u_{j2}, t_{j2}), \cdots, (u_{ji}, t_{ji}), \cdots, (u_{jK}, t_{jK}))$, where u_{ji} represents a user retweeting c_j at time t_{ji}. Given N such traces $\mathcal{T}_1, \cdots, \mathcal{T}_N \in \mathcal{T}$ and their corresponding tweets $c_1, \cdots, c_j, \cdots, c_N \in C$, how do we meaningfully characterize and categorize them?

3.1 Time Interval Distribution

The observations we made above about dynamics of retweeting can be succinctly captured by two distributions: *inter-tweet time interval* distribution and *distinct user* distribution. First, we consider the distribution of time intervals between successive retweets. These are shown in Figure 2 for the same URLs whose retweeting activity is shown in Figure 1. Humans are very heterogeneous; therefore, a signature of human activity is a broad distribution with time intervals of many different length that are all equally likely, as shown in Figure 2(a)-(c) and (f). Specifically, there is

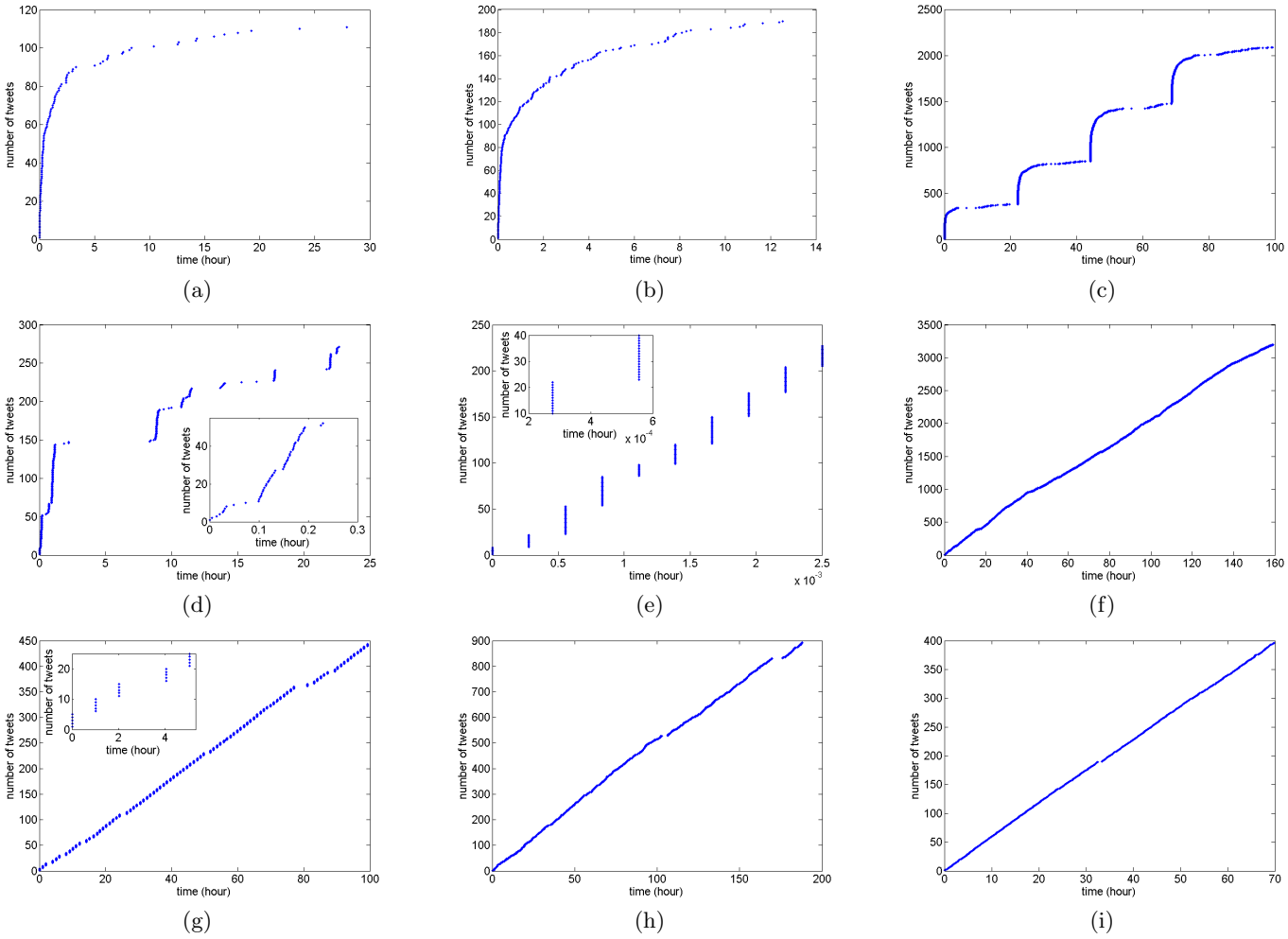


Figure 1: Evolution of retweeting activity for story posted by (a) a popular news website (nytimes) (b) popular celebrity (billgates) (c) politician (silva_marina) (d) an aspiring artist (youngdizzy) (e) post by a fan site (AnnieBieber) (f) animal rights campaign (nokillanimalist) (g) advertisement using social media (onstrategy) (h) advertisement from an account eventually suspended by Twitter(EasyCash435) (i) advertisement by a Japanese user (nitokono). Insets in (d), (e) and (g) show automatic retweeting, with multiple retweets made within a short time period either by the same or different users.

a lot of activity initially associated with newsworthy content, which gradually decreases with time, resulting in many short intervals and some long ones, as shown in Figure 2(a)–(b). Automated retweeting results in tweets at regular time intervals, which will lead to an isolated peak or peaks in the distribution (as in Figure 2(i)), or bursty behavior with many zero second intervals (as seen in Figure 2(e) and (g)).

We measure the regularity or predictability of the temporal trace of tweets using *entropy*. Let ΔT represent the time interval between two consecutive retweets in a trace $\mathcal{T}_j$, with possible values $\{\Delta t_1, \Delta t_2, \cdots, \Delta t_i, \cdots, \Delta t_{n_T}\}$. If there are $n_{\Delta t_i}$ time intervals of length Δt_i, then $p_{\Delta T}(\Delta t_i)$ denotes the probability of observing a time interval Δt_i:

$$p_{\Delta T}(\Delta t_i) = \frac{n_{\Delta t_i}}{\sum_{k=1}^{n_T} n_{\Delta t_k}} \tag{1}$$

The entropy $H_{\Delta T}$ of the distribution of time intervals is:

$$H_{\Delta T}(\mathcal{T}_j) = -\sum_{i=1}^{n_T} p_{\Delta T}(\Delta t_i) log(p_{\Delta T}(\Delta t_i)) \tag{2}$$

Automatic retweeting with a regular pattern has a lower time interval entropy, and is therefore, more predictable, than human retweeting, which is more broadly distributed and less predictable.

3.2 User Distribution

In addition to time interval, we also measure the distribution of the number of times distinct users retweet some URL. Figure 3 shows the number of retweets made by each user involved in the tweeting activity shown in Figure 1. Newsworthy content is usually retweeted once by each user who participates in the tweeting activity, as shown in Figure 3(a)–(c). Spam-like activity and campaigns, on the other hand,

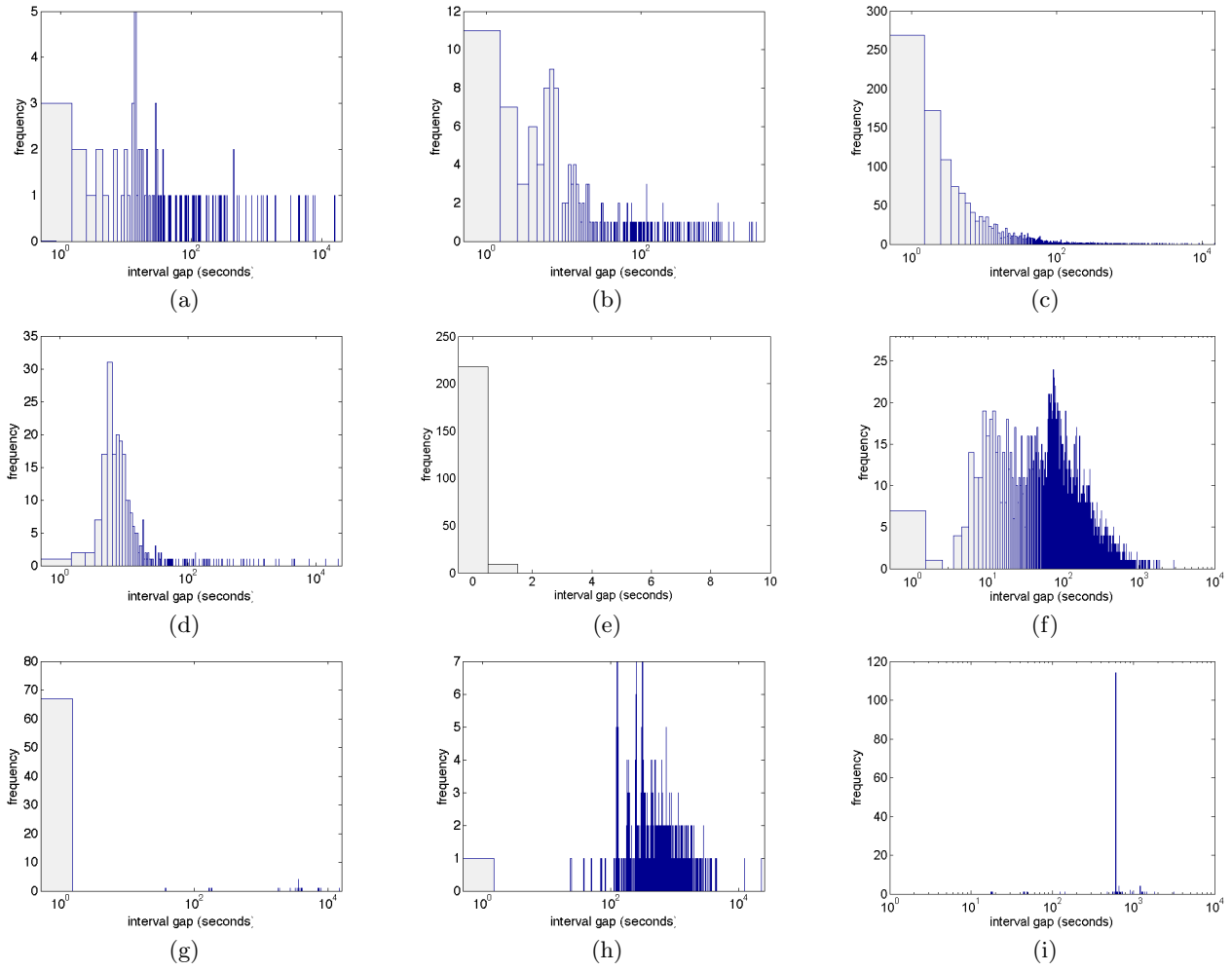


Figure 2: The distribution of the inter-arrival gaps for the retweeting activities shown in Figure 1 (a)nytimes (b)billgates (c) silva_marina (d) youngdizzy1 (e) AnnieBieber (f) nokillanimalist (g) onstrategy (h) EasyCash435 (i) nitokono.

result when an individual (Figure 3(g)–(i)) or a small group (Figure 3(f)) repeatedly retweet the same post. The higher the retweeting, the greater the manipulation effort.

The campaign shown in Figure 1(c) is successful, since there are many distinct users who participate in it, as shown in Figure 3(c). However, there are some dedicated campaigners, including *silva_marina* herself, who retweet the same message multiple times. Also the distribution of inter-arrival times in Figure 2(c) is similar to that of Figure 2 (a) and (b), indicating human activity. A campaign probably not as successful as that by *silva_marina* is one by *nokillanimalist* (Figure 1(f)), which has very few participating users in it. The distribution of the inter-arrival times in Figure 2(f) is also comparable to Figure 2(a)–(c), with a large number of nonzero inter-arrival times and the frequency of shorter inter-arrival gaps being larger than that longer ones indicating human activity. However, the distribution of the number of retweets by distinct users shows a stark contrast. In fact it shows that there are only three dedicated users generating over 3000 retweets. Similarly in case of of the retweeting activity shown in Figure 1(h), there are only two users engaged in spreading spam-like advertisements (Figure 3(h)). These two users together account for around 900 retweets. Spam-like characteristics are also observed in the advertisements, whose retweeting activity is shown in Figure 1(g) and 1(i) which have one (Figure 3(g)) and two users (Figure 3(i)) generating a bulk of the content. However on looking into the temporal distribution more closely, we observe that in case of Figure 1(g), almost two-thirds of the retweets occur almost consecutively (time interval gap is zero seconds), indicating a possible autotweeting activity. Figure 1(i) too, shows some kind of probable scheduled or automated tweeting activity with around 28% of the tweets having an exact interval gap of 604 seconds. Possible autotweeting is also indicated in the promotional activity shown in Figure 1(e). Although a large number of users participate in this activity as shown by Figure 3(e), almost all the retweets are generated simultaneously as seen in Figure 2(e).

We use entropy to measure the breadth of user distribution. Let random variable F represent a distinct user in a trace

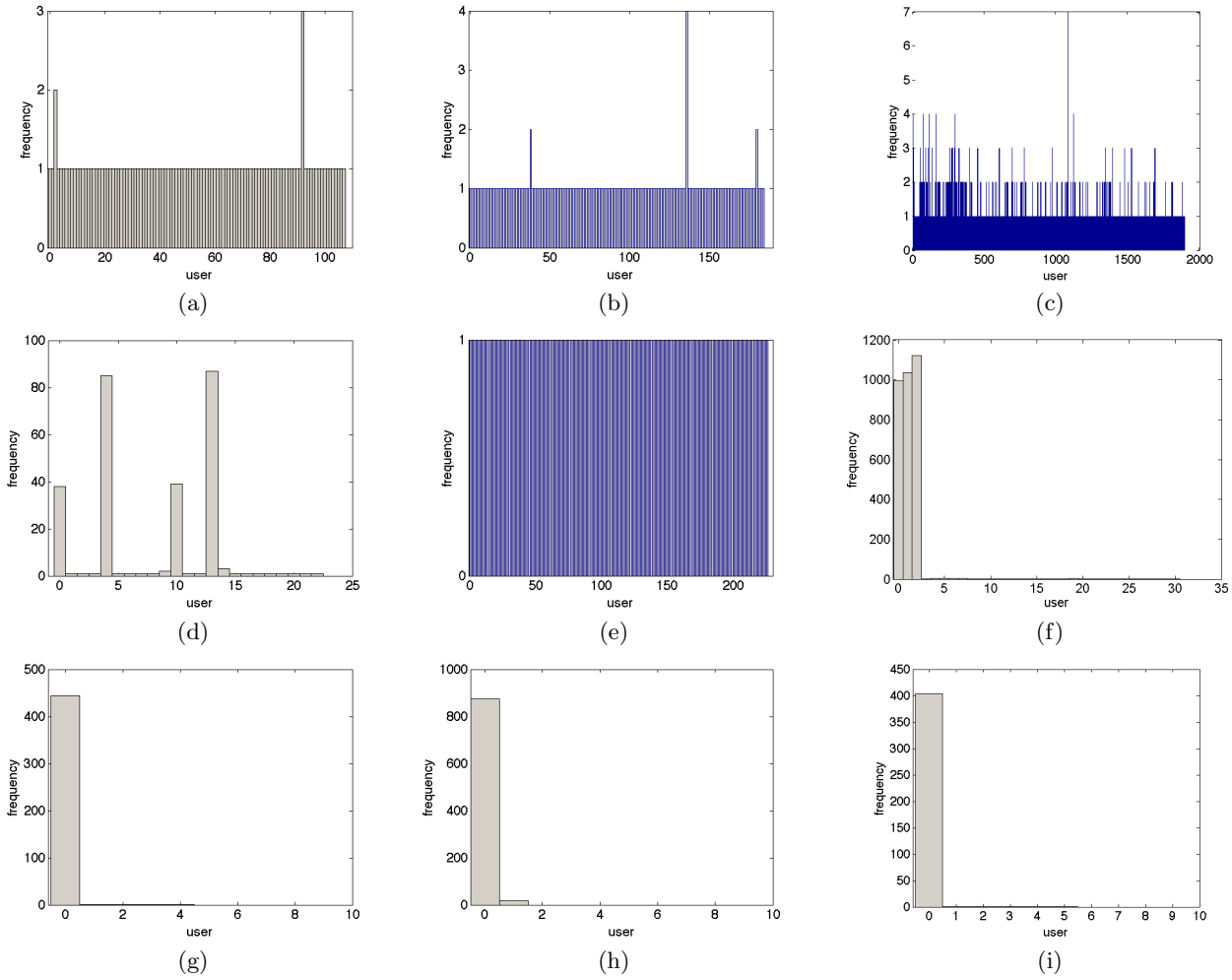


Figure 3: The number of retweets by distinct users. Each user is marked by a unique user id for the retweeting activities shown in Figure 1 (a)nytimes (b)billgates (c) silva_marina (d) youngdizzy1 (e) AnnieBieber(f) nokillanimalist (g) onstrategy (h) EasyCash435 (i) nitokono.

$\mathcal{T}_j$, with possible values $\{f_1, f_2, \cdots, f_i, \cdots, f_{n_F}\}$. Let there be n_{f_i} retweets from user f_i in the trace $\mathcal{T}_j$. If p_F denotes the probability mass function of F, such that $p_F(f_i)$ gives the probability of a retweet being generated by user f_i, then

$$p_F(f_i) = \frac{n_{f_i}}{\sum_{k=1}^{n_F} n_{f_k}} \tag{3}$$

The user entropy H_F is given by:

$$H_F(\mathcal{T}_j) = -\sum_{i=1}^{n_F} p_F(f_i) log(p_F(f_i)) \tag{4}$$

As clear from the Equation 4, in spam-like activity a small number of users are responsible for large number of tweets, which leads to a lower entropy than retweeting activity of newsworthy content. On the other hand, automated retweeting coming from many distinct users (as seen in Figure 3(e)) indicates that users' accounts may have been compromised.

3.3 Classification

The time interval and user entropies $H_{\Delta T}(\mathcal{T}_j)$ and $H_F(\mathcal{T}_j))$ can used to categorize retweeting activity of any content. This classification helps us not only identify the different dynamic activities occurring on Twitter, but also provides valuable insight into the nature of the associated content. The linear runtime complexity of entropy calculation and the presence of scalable methods of clustering [7] ensures that this entropy-based approach can be easily applied to very large data sets.

4. VALIDATION

Twitter's Gardenhose streaming API provides access to a portion of real time user activity, roughly 20%-30% of all user activity. We used this API to collect tweets for a period of three weeks in the fall of 2010. We focused specifically on tweets that included a URL (usually shortened by a service such as bit.ly) in the body of the message. Using regular expression based rules like '$(https?|ftp|gopher|telnet|file|notes|ms-help) : ((//)|(\\\\\\\\\))+[\\w\\d : \#@\%/; \$() ?$

\\+\\− = \\\\\\.&]∗', we extracted these URL embedded tweets from this corpus. In order to ensure that we had the complete retweeting history of each URL, we used Twitter's search API to retrieve all activity for that URL.

Data collection process resulted in 3,424,033 tweets which mentioned 70,343 distinct shortened URLs. There were 815,614 users in our data sample. We study the retweeting activity of URLs posted by users who posted at least two popular URLs. By popular, we mean URLs that were retweeted at least 100 times. There were 687 such distinct URLs.

We apply entropy based approach to study the retweeting dynamics of these URLs. We show that entropy-based analysis gives a good characterization of different types of activities observed in collective retweeting of these URLs.

4.1 Manual Annotation

We manually examined the content of each URL (using Google translate on foreign language pages) to annotate the activity along following categories:

News If the URL belongs to the twitter profile of a news organization, we label the retweeting activity as following news.

Blogs If the URL links to the blog or webpage maintained by an individual, we classify the retweeting activity as following blogs or celebrity.

Campaigns If the URL belongs to an individual or an organization with a discernible agenda (politics, animal rights issues), we classify the retweeting activity as a campaign.

Advertisements and promotions If the URL links to an advertisement or promotion, we classify the retweeting activity as such. This includes instances where users post the same link repeatedly, leading to spam-like content generation, and the promotional activities of aspiring starlets.

Parasitic ads This is a form of parasitic advertisement in which users participate unwittingly. This happens when a user logs into a website or web service, and then that service tweets a message in user's name telling his followers about it. For example, when a user visits sites such as Tinychat [1] or Twitcam [2], a message is posted to the user's Twitter account "join me on tinychat..."

Automated/robotic activity Retweeting that is mainly generated through Twitterfeed [3] or similar services is classifies as automatic activity. Note that automated activity could be associated with any type of content, but since it has its own unique characteristics, different from all the aforementioned activities, we included it as a separate class. This can be easily identified by looking at the source of the tweet, which will identify *twitterfeed* (or a similar service) as the originator.

[1] tinychat.com
[2] twitcam.com
[3] www.twitterfeed.com

We found that users respond to news stories and blog posts in identical manner, making them difficult to distinguish. Generally, the type of information contained in these two sources is also very similar. Therefore, for classification purposes, we put them in the same category of *newsworthy* content.

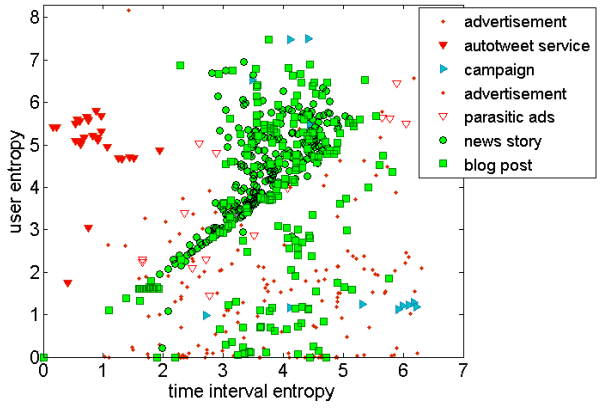


Figure 4: Manually annotated URLs shown in the entropy plane.

Figure 4 shows the retweeting activity of URLs in our data sample as measured by the time interval and user entropy. The bulk of the URLs belong to news or blog category. They are also characterized by medium to high user and time interval entropies, indicating newsworthy content. Blog posts or websites of major celebrities represent more popular content and are located in the upper section of the plot. Blog posts from starlets without major following are located in the lower section of the plot. Though these posts have similar numbers of retweets, lower user entropy means that the starlets, or their dedicated followers, generate some of the retweeting activity. The automatic retweeting cluster is isolated. This contains URLs like one whose activity is shown in Figure 1(e), but also several news stories, most notably from the online technology magazine TechCrunch. This is because some Twitter users employ Twitterfeed to automatically tweet stories that are posted on TechCrunch. This helps users appear to be more active on Twitter than they really are. The uninteresting stories are for the most part retweeted automatically, and not manually by other people. Therefore, they have low time interval entropy, but high user entropy, since many different Twitter accounts are used.

Advertisements are mostly located in the lower half of the figure, although successful advertisements that capture public interest are indistinguishable from newsworthy content. Unsuccessful campaigns that are driven by a few dedicated zealots are in their own cluster with high time interval and low user entropy, but successful campaigns are also indistinguishable from newsworthy content.

4.2 Classification

The distribution of distinct time intervals and users involved in the retweeting activity gives a good characterization of the retweeting activity. As explained in Section 3, temporal and user entropy are used to quantify these distributions. Temporal entropy is maximum when the time intervals between

Table 1: F-Measure (F) and ROC area for 10-fold cross validation experiments using SVM and k-NN classification

		ads & promotion	auto-tweet	campaign	news & blog	parasitic ads
k-NN	F	0.686	0.96	0.5	0.89	0.105
	ROC	0.807	0.959	0.678	0.837	0.644
SVM	F	0.719	0.939	0.526	0.897	0
	ROC	0.833	0.973	0.685	0.875	0.718

any two successive retweets is different. User entropy is maximum when each user retweets the message only once. Next, using temporal and user entropies as features, we classify the retweeting activity represented by a trace $T_j \in \mathcal{T}$. We perform both unsupervised and supervised classification. The data is manually labelled to train the supervised classifier and to evaluate the performance of the classification techniques. We used Weka software library [4] for off-the-shelf implementation of EM (expectation maximization [11]),k-NN (k-nearest neighbors) and SVM(support vector machines [6]) classification.

4.2.1 Supervised Classification

We used Support Vector Machine with radial basis function (RBF) kernel and k-NN algorithm with three nearest neighbors and Euclidean distance function to classify the data. Table 1 reports results of 10-fold cross validation in each model was trained on 90% of the labeled data and tested on the remaining 10%. The F-scores of both algorithms are relatively high, showing that they have well separated instances into different classes.

4.2.2 Unsupervised Classification

We use Expectation Maximization (EM) algorithm to automatically cluster points. EM uses Gaussian mixture model and can decide how many clusters to create by cross validation. The number of clusters determined automatically by this method was nine. Figure 5(a) shows the resulting clusters, and the confusion matrix is shown in Table 2. If the number of clusters were predefined to be 5, the resulting confusion matrix is shown in Table 3, and discovered clusters are shown in Figure 5(b).

4.2.3 Observations

Broadly speaking, we identify five classes of retweeting activity and associated content on Twitter.

Automatic/Robotic Activity. As we can see from the results, almost all methods classify automatic or robotic retweeting (auto-tweet) with high accuracy. While some of such activity, in our data set is related to technology news stories, and their user entropy is similar to that of other news stories, such activity has a much lower time interval entropy than other news stories.

Two primary kind of automated services that we identified are auto-tweeting services and tweet-scheduling services.

[4]www.cs.waikato.ac.nz/ml/weka

There are two categories of auto-tweeting activities. The first arises when an individual subscribes to an automatic service that tweets messages on the user's profile on his behalf. One such automatic service is Twitterfeed [5], through which the user can subscribe to a blog or news website (any service with an RSS feed). Twitter users employ this service to automatically retweet stories posted on technology news sites Mashable and TechCrunch. This leads to *individual auto-tweets* observed from the profile of that user.

However, this auto-tweeting feature is also being used for promotional and perhaps phishing activities. For example, a fan site (http://bieberinsanityblog.blogspot.com/) for Justin Bieber asks fans to provide their Twitter account information. The site is powered by Twitterfeed, and then auto-tweets Justin Bieber news from the profiles of registered fans, resulting in *collective auto-tweeting*.

Services like Tweet-u-later [6] and Hootsuite can be used to schedule tweeting activities. These websites can be used for spamming. Registering a collection of profiles to these websites and scheduling the a tweet to posted repeatedly, enables spammers to post the same message multiple times.

Since our method can easily differentiate human activity from bot or automated activity, we are able to identify marketing companies which engage automated services to increase their visibility on Twitter. Such services include OperationWeb (http://www.operationweb.com/) and TweetMaster (http://tweetmaster.tk/), which claim that they "will tweet your ad or message on my Twitter accounts that add up to over 170k* followers 2-6 times per day for 30 days."

Most of these services use bots or automated services to push up the perceived visibility of the advertisements. To increase visibility they need a large number of profiles. To gain access to a large number of profiles, such services ask users to register, set their own prices for tweets and feature the sponsored tweets in their profile. In this way these services create a win-win situation, helping companies to promote their product and users to make money by featuring sponsored messages on their profiles.

Newsworthy information. This class comprises of mostly news and blogs and some successful campaigns. Newsworthy information is characterized by comparable (usually high) user and temporal entropy. Since people, not bots, are involved in disseminating such content, we call this "human response to information." Both supervised and unsupervised clustering algorithms able to separate news and blogs i.e information sharing by humans, from the rest of retweeting activity with good accuracy (Tables 1, 3 and 2). However, EM algorithm with five classes breaks this class into smaller clusters (cluster0, cluster3 and cluster4). This is indeed a meaningful subdivision based on popularity, with content in cluster3 being the most popular, content in cluster0 being normal content and content in cluster4 having low popularity. When EM is allowed to automatically adjust the number of clusters, the popular clusters found by the earlier

[5]www.twitterfeed.com

[6]http://www.tweet-u-later.com/

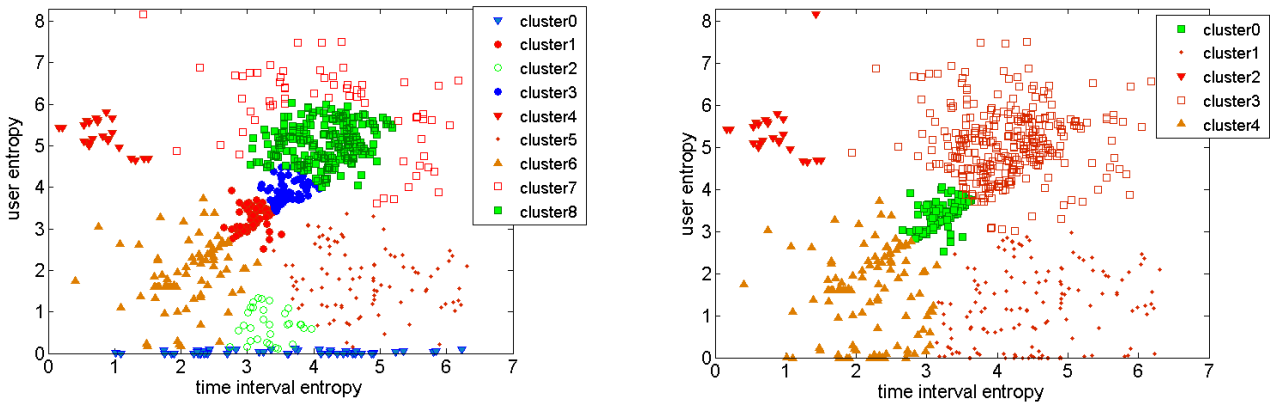


Figure 5: Unsupervised clustering of the data points using EM: (a) when EM automatically finds the best number of clusters, and (b) when the number of clusters is constrained by be five.

Table 2: Confusion matrix with manually annotated data and clusters automatically detected by EM algorithm

	advertisement & promotion	auto-tweet	campaign	news	blogs	parasitic advertisement
cluster0	45	0	0	0	8	0
cluster1	7	0	0	41	13	1
cluster2	17	0	0	0	14	0
cluster3	0	0	0	53	10	1
cluster4	0	23	0	0	0	0
cluster5	53	0	7	2	34	0
cluster6	36	2	1	27	19	6
cluster7	10	1	3	14	30	6
cluster8	11	0	2	130	60	0

algorithm gets subdivided into two more classes giving five clusters of human response to information (cluster1, cluster3, cluster6, cluster7 and cluster8 in Figure 5(b)). Compared to hand-labeled dataset (Figure 4) and from the confusion matrix in Table 2, we observe that cluster7 comprises predominantly of popular blogs, cluster8 comprises mostly of popular news, cluster1 and cluster3 comprise of normal human response to information and cluster6 shows human response to unpopular information.

Advertisements and Promotions. Advertisements and promotions are distinguished by low user entropy and low to high temporal entropy. Supervised clustering is able to accurately detect advertisements and promotions (Table 1). Most spam-like advertisements fall in this section. These are unwanted advertisements which are never retweeted by any user besides the originator of the advertisement. EM algorithm with five classes also identifies a group comprising predominantly of advertisements. However, EM algorithm with automatic class detection, divides this group further into three classes: cluster0 comprising mostly of spam-like activity with very low user entropy (≈ 0), cluster2 containing advertisements with low user and medium time entropy and cluster5 comprising of campaign-like promotions and advertisements with low user entropy and medium to high temporal entropy.

Campaigns. Campaigns are identified by low user entropy and very high temporal entropy. There are very few campaigns in the hand-labeled dataset. Even then, supervised algorithms are able to classify campaigns with a fair degree of accuracy (cf. Table 1). However, unsupervised algorithm merges campaigns with advertisements and promotions. Due to considerable overlap of characteristics of campaigns with advertisements or promotions, to distinguish a campaign from an advertisement is difficult, even for manual annotators. Note, that when a campaign is very successful like the one by *silva_marina*, Figure 1 (c), information that the campaigner intends to propagate spreads through the online social media. The retweeting activity in this case becomes similar to human response to information.

Parasitic Advertisements. None of the methods were able to identify parasitic advertisements very accurately. One possible reason may be their parasitic nature, where they

Table 3: Confusion matrix with manually annotated data and clusters detected by EM algorithm when number of clusters is predefined to be 5

	advertisement & promotion	auto-tweet	campaign	news & blogs	parasitic advertisements
cluster0	7	0	0	82	1
cluster1	85	0	7	49	0
cluster2	1	23	0	0	0
cluster3	22	1	5	272	7
cluster4	64	2	1	52	6

do not have a distinct characteristic feature of their own, but adopt the characteristics of the hosting user profile.

5. RELATED WORK

There has been some work to define temporal variation on online social media. In [21], the authors enumerate the different approximate shapes of temporal distribution of content in Twitter. But unlike us, they are not able to associate semantic meaning to the clusters they observe.

Previous work has tried to estimate the quality or interestingness of content [4, 10]. However, quality or interestingness is a subjective measure and is biased by the perspective of the user. For instance, what would be high quality information or interesting to a campaigner might be junk to a news aggregator. Therefore there is the need for an objective quantitative measure of user-generated content. Our entropy-based approach for classifying user activity and content addresses this need. While the method described in [10] is similar in spirit to ours, it can discover only three classes of activity. Heterogeneous activity on Twitter requires more than three classes.

Most of the existing spam detection [17] and trust management systems [8] are based on content and structure but do not look at collective dynamics. Besides, they usually require additional constraints like labelled up-to-date annotation of resources, access to content and cooperation of search engine. Satisfiability of so many constraints is difficult especially when one takes the diversity and astronomical size of online social media into account. Our method on the other hand, while having no such constraints, may be able to detect spams with an accuracy close to humans.

There has been some work done on spam detection on Twitter. Grier et. al [12] analyzed the features of spam on Twitter. However, they detect spam using three blacklisting services. Similarly, one of the methods employed to remove spam on Twitter is using Clean Tweets[7] [14]. Clean tweets filter tweets from users who are less than a day (or any duration specified) old and tweets that mention three (or any number specified) trending topics. However, it would be unable to detect spammers who auto-tweet or posts spam-like tweets at regular intervals (like EasyCash435 or onstrategy, Figure 1 (g) and (h)), which our approach can easily detect. Also, since URL shortening services such as http://bit.ly are often used on Twitter, users cannot guess which references are pointed at, which in turn is an attractive feature for spammers. However, since our categorization method is content independent, we can easily identify such spams using this method. Yardi et al. [22] state "Twitter spam varies in style and tone; some approaches are well-worn and transparent and others are deceptively sophisticated and adaptable." Using this method, we can capture the characteristics of spam, whether, it is generated by a auto-tweet service, a malicious advertiser or a passionate campaigner.

[7]http://www.seoq.com/blvdstatus/clean-tweets.html

Automated email spamming has been studied by [20]. They have identified the activity of botnets generating e-mail spam as being 'bursty' (inferred from the duration of activity) and 'specific' (pertaining to a random generated URL matching the signature). In this study of Twitter, we identify automated activity by a set pattern of retweeting (indicated by much lower time-interval entropy compared to user entropy).

Note, that this approach is based on observed collective response to content. By reposting some content, users give an implicit feedback to that content. Therefore unlike[4, 8] this method can even be applied in systems, where users do not explicitly rate other users.

We can automatically detect newsworthy, information-rich content and separate it from other user-generated content, based on user-response. We have showed that this method can further categorize content within this class into blogs or celebrity websites and news. [19] study the flow of information between these sub-categories.

6. CONCLUSION AND FUTURE WORK

We characterize dynamics of retweeting activity of some content on Twitter by the entropy of the user and time interval distributions, and show that these two features alone are able to separate user activity into different meaningful classes. The method is computationally efficient and scalable, content and language independent and is robust to missing data. Entropy-based classification can be used for spam detection, trend identification, trust management, user modeling, understanding intent and detecting suspicious activity on online social media. We have identified five categories of retweeting activity on Twitter: newsworthy information dissemination, advertisements and promotions, campaigns, automatic or robotic activity and parasitic advertisements. We have observed that human response to news, blogs and celebrity posts is very similar. The novel entropy-based classification method not only enables us to characterize user activity, but it also helps us to understand user-generated content and separate popular content from

normal or unpopular content.

Future work includes analyzing the effect of adding more features to the categorization of user activity and content. This study focusses on URL embedded tweets and uses URL as markers to trace and characterize the activity associated with the tweets. Ongoing work includes extending this analysis beyond URL-embedded tweets and using other markers to trace user activity on tweets. Such markers may include retweets containing 'RT' string, hash-tags, free-text snippets (keywords, domain terminology, people names) and so on. We also plan to apply this analysis on larger datasets and other online social media.

In the course of our study we have observed the gradual emergence of sophisticated spamming and the birth of an alternate industry to manipulate content on Twitter like promotional activities to improve the perceived popularity of stars. Kwak et al. [14] had asked – What is Twitter, a Social Network or a News Media? Our analysis of Twitter shows that it is not only a social network but much more – the diversity of twitter activity is a reflection of complexity of collective user response on online social media.

7. REFERENCES

[1] The face of egypt's social networking revolution. In *http://www.cbsnews.com/stories/2011/02/12/eveningnews/main20031662.shtml*, 2011.

[2] Lady gaga a bigger twitter star than justin bieber- 10 million fans say so. In *http://sanfrancisco.ibtimes.com/articles/147005/20110517/lady-gaga-a-bigger-twitter-star-justin-beiber-10-million-fans-say.htm*, 2011.

[3] Social networking sites used by celebrities- the twitter revolution. In *http://www.twittingsound.com/social-networking-sites-used-by-celebrities-the-twitter-revolution.html*, 2011.

[4] E. Agichtein, C. Castillo, D. Donato, A. Gionis, and G. Mishne. Finding high-quality content in social media. In *Proceedings of the international conference on Web search and web data mining*, WSDM '08, pages 183–194, New York, NY, USA, 2008. ACM.

[5] P. Beaumont. Can social networking overthrow a government? In *http://www.smh.com.au/technology/technology-news/can-social-networking-overthrow-a-government-20110225-1b7u6.html*, 2011.

[6] B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, COLT '92, pages 144–152, New York, NY, USA, 1992. ACM.

[7] P. S. Bradley, C. A. Reina, and U. M. Fayyad. Clustering Very Large Databases Using EM Mixture Models. *Pattern Recognition, International Conference on*, 2:2076+, 2000.

[8] J. Caverlee, L. Liu, and S. Webb. Socialtrust: tamper-resilient trust establishment in online communities. In *JCDL '08: Proceedings of the 8th ACM/IEEE-CS joint conference on Digital libraries*, pages 104–114, New York, NY, USA, 2008. ACM.

[9] M. Cha, H. Haddadiy, F. Benevenutoz, and K. P. Gummadi. Measuring User Influence in Twitter: The Million Follower Fallacy. In *Proceedings of 4th International Conference on Weblogs and Social Media (ICWSM)*, 2010.

[10] R. Crane and D. Sornette. Viral, quality, and junk videos on youtube: Separating content from noise in an information-rich environment. In Proceedings of the AAAI Symposium on Social Information Processing, 2008.

[11] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Royal statistical Society B*, 39:1–38, 1977.

[12] C. Grier, K. Thomas, V. Paxson, and M. Zhang. @spam: the underground on 140 characters or less. In *Proceedings of the 17th ACM conference on Computer and communications security*, CCS '10, pages 27–37, New York, NY, USA, 2010. ACM.

[13] S. Kessler. Social media plays vital role in reconnecting japan quake victims with loved ones. In *http://mashable.com/2011/03/14/internet-intact-japan/*, 2011.

[14] H. Kwak, C. Lee, H. Park, and S. Moon. What is Twitter, a social network or a news media? In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 591–600, New York, NY, USA, 2010. ACM.

[15] K. Lerman. Social information processing in social news aggregation. *IEEE Internet Computing: special issue on Social Search*, 11(6):16–28, 2007.

[16] K. Lerman and R. Ghosh. Information contagion: an empirical study of the spread of news on digg and twitter social networks. In *Proceedings of 4th International Conference on Weblogs and Social Media(ICWSM)*, 2010.

[17] B. Markines, C. Cattuto, and F. Menczer. Social spam detection. In *Proceedings of the 5th International Workshop on Adversarial Information Retrieval on the Web*, AIRWeb '09, pages 41–48, New York, NY, USA, 2009. ACM.

[18] F. Wu and B. A. Huberman. Novelty and collective attention. In *In PNAS*, volume 104(45):17599–17601, 2007.

[19] S. Wu, J. M. Hofman, W. A. Mason, and D. J. Watts. Who Says What to Whom on Twitter. In *Proceedings of World Wide Web Conference (WWW '11)*, 2011.

[20] Y. Xie, F. Yu, K. Achan, R. Panigrahy, G. Hulten, and I. Osipkov. Spamming botnets: signatures and characteristics. *SIGCOMM Comput. Commun. Rev.*, 38(4):171–182, Aug. 2008.

[21] J. Yang and J. Leskovec. Patterns of temporal variation in online media. In *Proceedings of the fourth ACM international conference on Web search and data mining*, WSDM '11, pages 177–186, New York, NY, USA, 2011. ACM.

[22] S. Yardi, D. Romero, G. Schoenebeck, and D. Boyd. Detecting spam in a Twitter network. *First Monday*, 15(1), Jan. 2010.