The Pennsylvania State University
The Graduate School

MINING SOCIAL DOCUMENTS AND NETWORKS

A Thesis in
Computer Science and Engineering
by
Ding Zhou

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

May 2008

The thesis of Ding Zhou was reviewed and approved* by the following:

C. Lee Giles
David Reese Professor of Information Sciences and Technology
Thesis Co-Advisor
Co-Chair of Committee

Hongyuan Zha
Professor of Computer Science
Thesis Co-Advisor
Co-Chair of Committee

Jia Li
Associate Professor of Statistics

Wang-chien Lee
Associate Professor of Computer Science and Engineering

David J. Miller
Associate Professor of Electrical Engineering

Mahmut Kandemir
Associate Professor of Computer Science and Engineering
Director of Graduate Affairs in Computer Science and Engineering

*Signatures are on file in the Graduate School.

Abstract

The Web has connected millions of users by various communication tools for social purposes. Daily, huge amount of social data are being created through fingertips, driven by various of social actions that involve a wide range of user-produced content (e.g. emails or collaborative documentations). Often being a part of many contemporary Web applications, *user social networks* are gaining increasing attention from both industry and academia as they seem to have become a promising vehicle for delivering better user experience. Accordingly, computational social network analysis has become an important topic in user data mining.

Despite a long history of structural social network analysis and recent interests in user behavior analysis, little research has addressed *social contents* in social networks and *heterogeneous networks* in user behavior. In fact, *social content* and *heterogeneity* are two key elements in contemporary online social networks that offer great benefits: *social contents* provide more semantic information; meanwhile *heterogeneous social networks* allow diversified perception of users. Motivated by these considerations, this dissertation seeks to improve traditional computational social network analysis by covering analysis of not only (1) social networks of users; but also (2) social content composed by users, and (3) social actions among users. A series of new methods are presented for knowledge discovery in social documents and social networks, with a special focus on modeling social content and machine learning of heterogeneous networks. In particular, this study first proposes new probabilistic content models for user generated social documents and annotations and investigates the connection between social content and social actions. A set of new techniques for computational analysis of heterogeneous social networks constructed by various social actions constitutes the conclusion. The methods proposed in this dissertation have been applied to a wide range of applications including ranking, community discovery, information retrieval, and

document recommendations. For large scale real world data sets, this research shows significant experimental improvements over currently applied methods.

Table of Contents

List of Figures

List of Tables

Acknowledgments

The efforts and support of many individuals have made this a pleasant and memorable journey. I owe an enormous debt of gratitude to my co-advisors, C. Lee Giles and Hongyuan Zha, who have showed me the beauty of science, offered me precious opportunities for collaborations, and supplied sustenance, literally and metaphorically, through long Pennsylvania winters. I am very grateful to my committee members and collaborators, Jia Li and Wang-chien Lee, for offering many insightful suggestions and friendly accessibility throughout my graduate study. I would like to thank David J. Miller for his detailed comments on this dissertation. This dissertation has also received contributions from outside the university, for which I owe my special thanks to Gordon Sun who mentored my first industrial job. I thank Chris H.Q. Ding, Doug Rohde, Belle L.Tseng for giving me so much valuable advice from industry. I feel especially thankful to my colleagues from The Pennsylvania State University, Yahoo!, Google, NEC Labs America, and Lawrence Berkeley National Laboratory, with whom I have enjoyed collaboration.

I am deeply indebted to my parents for bringing me into this world and providing me with a sound education. Those early days when I indulged in hand-making toy transformer robots from postcards have become a primitive drive for my passion for science and love for engineering.

Chapter 1

Introduction

Social network analysis (SNA) is the study of mapping and measuring relationships among a set of social actors in social networks, in which each node denotes a social actor and each edge represents a certain relationship between two social actors. The fundamental questions in SNA are: *Why are social actors connected? How they are connected in networks?* And, *what inference can be derived from their connections?* Traditional social network analysis has had wide application in social and behavior sciences, as well as in economics and industrial engineering [76]. Much of the interest in SNA arises from its appealing focus on social relations and the patterns and implications of these relations. SNA has had application for many domains including viral marketing [60], customer value evaluation [15], and measurement of social influences of actors in a network [78] for customer relations management.

SNA, while an established field in sociology [76], has recently gained popularity in the computer science community with the emerging pervasiveness of the Web. The new challenges for this traditional field are mainly driven by the increasing availability of various kinds of social content, such as emails [64], blogs [25], mes-

sage boards [46], and heterogeneous types of social behavior of users, such as social annotations [22], collaborations [37], and citations [20]. As a result, recent SNA has a stronger computational emphasis and seeks to leverage diversified information resources and heterogeneous user behavior. In most related literature, the definition of a social network is:

Definition 1. *A Social Network (SN) is a homogeneous graph, in which each node denotes an individual, also known as a Social Actor (SA), and each edge represents a certain relationship between two SA's, also known as a Social Tie (ST).*

Typical social network instances encountered everyday, include the SNs of authors, Web bloggers or email users. The social ties between two SA's are recognizable in a variety of ways depending on the application's settings. For example, social actions such as the collaboration between authors can be seen as one social tie between those individuals. However, a single type of relationship in SNs is not enough to capture a real world association. Very often, a variety of existing social ties, which correspond to different types of social actions among SA's, gives rise to multiple semantics associated with each social tie. Furthermore, social actions usually involve heterogeneous objects that were missing in the traditional definition of an SN. Accordingly, where the notion of heterogeneous social networks comes into focus, this dissertation generalizes the definition of SNs to:

Definition 2. *A heterogeneous social network is a heterogeneous graph which the nodes denote different types of objects consisting of social actors and others; the edges denote different types of relations among social actors or between social actors and other objects.*

In practice, a heterogeneous social network can be constructed by defining various relationships. One of the most natural ways to define a relationship among

social actors (or between social actors and other objects), the concern of this dissertation, is by social actions. For example, collaboration among authors can be a social action among social actors; a comment by a user on a blog is, perhaps, a social action involving the blog document. Formally, the definition of social actions is:

Definition 3. *Social actions refer to any action that takes into account the actions and reactions of social actors in a social network.*

Notably, a social action can be between social actors or between social actors and other objects. Different types of social actions offer different semantics on the edges of a social network; different kinds of objects involved in a social action represent the heterogeneous nodes in a social network.

Probably the most common types of objects involved in social actions, especially on the Web, are documents. Many social actions, for the purpose of information exchange, are usually associated with text documents, including emails, academic papers, or annotations, all of which, for the purposes of this study, are social documents:

Definition 4. *A social document is a text file involving a set of social actors in a social network for the purpose of exchanging information or soliciting future social ties.*

Traditionally, researchers focused on analysis of homogeneous social networks. However, social networks in practice are heterogeneous and content rich, partially due to the fact that users can connect using various types of social actions. Most traditional SNA methods, which are structural and homogeneous approaches, are not capable of handling such content-rich and heterogeneous SNs, which is the

primary concern of this dissertation. In fact, the content of social documents and the semantics of social actions embrace valuable information about development of user networks and interests. Thus, mining such social documents and heterogeneous social networks to interpret and understand real-world social networks is an important direction for computational SNA.

1.1 Related Work on Content Analysis

This dissertation has two categories of related work: (1) document content analysis and (2) network analysis. A variety of statistical approaches have been proposed for document content analysis, among which the most popular methods include the unigram model [55], latent semantic analysis (e.g. LSI [10], pLSA [30]), and generative models (e.g. LDA [3]), mentioned earlier. Despite the wide range of choices for content analysis, few of them consider the social networks to which these documents belong. This dissertation is one of the first to propose modeling social documents with social networks.

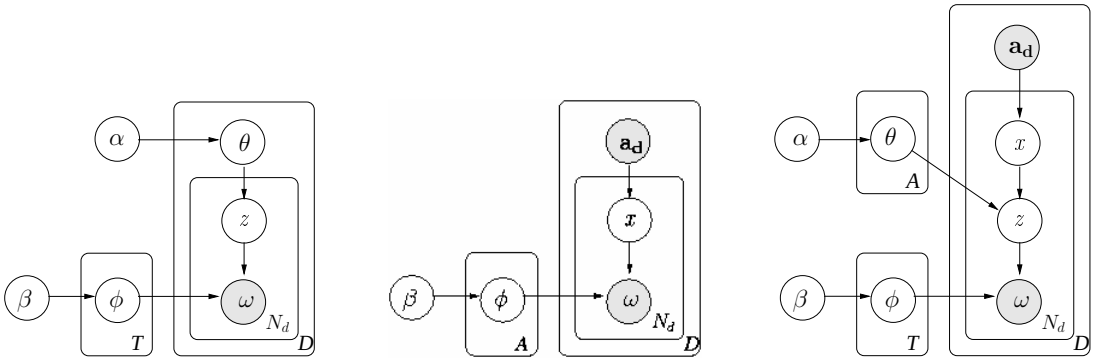
As an overview for the study an initial introduction includes one or two representative examples in each category of document content analysis: (1) The unigram models each document with a multinomial distribution and the words in the document are independently drawn from the multinomial distribution [55]. This assumes that each document in the collection has a distinct topic and develops a mixture of unigrams. The mixture of unigrams constructs models for each document by considering the words in a document as generated from the conditional probability distribution over topics. (2) Latent semantic indexing (LSI) [10] uses a space to implicitly capture a large portion of the information documents contain. Text analysis can be performed on the latent semantic space where the document

similarities are preserved. In order to measure similarity between documents, the dot products of the corresponding vectors of documents in the latent space can be used. Similar to the approach of LSI, the probability latent semantic analysis (pLSA) [30], has each document generated by the activation of multiple topics, and each topic is modeled as multinomial distributions over words, which relaxes the mixture of unigram models (the unigram model considers each document to be generated from only one topic). However, the pLSA model uses a distribution indexed by training documents, which means the number of parameters being estimated in a pLSA model must grow linearly with the number of training documents. This suggests that pLSA could be prone to overfitting in many practical applications. In addition, pLSA does not support generalization of models to unknown documents. (3) Another recent advance in document content analysis is generative models of documents, such as the Latent Dirichlet Allocation (LDA) [3] model. LDA addresses the overfitting of pLSA by using the Dirichlet distribution to model the distribution of topics for each document. Each word is considered sampled from a multinomial distribution over words specific to this topic. As an alternative, the LDA model is a well-defined generative model and generalizes easily to new documents without overfitting.

The methods this study developed for social content analysis relates closely to the methods for content analysis through generative models. Three related representative generative models for documents considered in this research are: Topic-Word model, Author-Word model and Author-Topic model. The heart of generative model-based methods is to simulate the generation of a document using probabilistic models [3, 24, 62, 48, 68]. Several factors arising from producing a document, either observable (e.g. author [48]) or latent (e.g. topic [24, 3]), are modeled as variables in the generative Bayesian network and have been shown to

work well for document content characterization.

With a given set of documents, D, each consisting of a sequence of words, $\mathbf{w_d}$, of size, N_d, the generation of each word, $w_{di} \in \mathbf{w_d}$, for a specific document, d, can be modeled from the perspective of either author or topic, or the combination of both. Fig. 1.1 illustrates the three possibilities using plate notations. If ω denotes a specific word observed in document, d, and if A represents the number of authors and T represents the prescribed number of topics, then $\mathbf{a_d}$ is the observed set of authors for d. To clarify, the latent variables are light-colored while the observed ones are shadowed. Figure 1.1(a) models documents as generated by a mixture of topics [3]. The prior distributions of topics and words follow Dirichlet, parameterized respectively by α and β. Each topic is a probabilistic multinomial distribution over words. If ϕ denotes the topic's distributions over words, and θ denotes the document's distribution over topics[1].



(a) Topic-Word (LDA) (b) Author-Word (c) Author-Topic

Figure 1.1. Three Bayesian network models for document content generation

In the Topic-Word model, a document is as a mixture of topics, with each topic corresponding to a multinomial distribution over the vocabulary. The existence of observed word, ω, in document, d, is accepted as being drawn from the word

[1] then, usually, the ϕ is represented using $T \times V$ matrix, where T and V are the number of topics and the size of the vocabulary. Similar is θ modeled as a $D \times T$ matrix.

distribution, ϕ_z, which is specific to topic, z. Similarly, topic, z, was drawn from the conditional topic distribution, θ_d, which is represented using a row in the matrix, θ[2].

The Author-Word model, similar to the Topic-Word model, prioritizes the author's interest as the origin of a word [48]. In Figure 1.1(b), $\mathbf{a_d}$ is the author's set that composes document, d. Each word in d is represents a choice from the author-specific word distribution. In this Author-Word model, the author responsible for a certain word is a random choice from $\mathbf{a_d}$.

In the Author-Topic model [68], another influential research option follows the same line of reasoning, combines the Topic-Word and Author-Word models, and regards the generation of a document as affected by both factors in a hierarchical manner. Figure 1.1(c) presents the hierarchical Bayesian structure. According to the Author-Topic model in Figure 1.1(c), for each observed word, ω, in document, d, an author, x, is uniformly sampled from the corresponding author group, $\mathbf{a_d}$. Then with the probability distribution of topics conditioned on x, $\theta_{x\cdot}$, a topic, z, is generated. Finally, word-by-word, the z produces ω as observed in document, d. The Author-Topic model has been shown to perform well for document content characterization because it involves two essential factors in producing a general document: the author and the topic. Modeling both factors as variables in the Bayesian network provides the model with the capacity to group into semantic topics the words used in a document collection. Based on the posterior probability obtained after establishing the network, a document can be denoted as a mixture of topic distributions, and each author's word choice preference and involvement in topics can be discovered.

[2]The Topic-Word model was first introduced as Latent Dirichlet Allocation (LDA), and for consistency this research uses the alternative name.

The estimation of the Bayesian network in the aforementioned models typically relies on the observed pairs of author and words in documents. Each word, treated as an instance, generates the probabilistic hierarchy in the models. Some layers in the Bayesian hierarchy are observed, such as authors and words. Other hidden layers, such as topics, require estimation.

1.2 Related Work on Network Analysis

While content analysis of documents introduced earlier has provided powerful tools for discovering topics in documents, the task of understanding how these topics form has never been easy. This difficulty motivated consideration of social actions behind the content. Chapter 4 presents the research [87] that is among the first to combine content analysis and network analysis as opposed to studies concerned with discovering patterns from document content alone [75]. Apparently, the patterns in social content can be explained to a great extent using social actions. Accordingly, this dissertation follows social content analysis with a strong focus on mining heterogeneous social networks constructed from social actions.

Two main objectives of mining heterogeneous social networks in this dissertation are establishing ranking and discovering communities. The following material concerns the introduction of ranking social actors and their clustering by graph partitioning.

1.2.1 Ranking Social Actors

The problem of ranking scientists and their work naturally belongs to at least two different fields: sociology [76] and bibliometrics [70]. For example, metrics of academic impact have been publication intensity or citation counts, impact factors

of journals, and the Hirsch index [29]. An important step in bibliometrics was a paper by Garfield [19] in the early 1970s, which discussed the methods for ranking journals by Impact Factor. Within a few years, Gabriel Pinski and Francis Narin proposed several improvements [56]. Most importantly, they recognized that citations from a more prestigious journal should be given greater weight [56]. They introduced a recursively defined weight for each journal. In particular, incoming citations from more authoritative journals, according to the weights computed during the previous iteration, contributed more weight to the journal being cited. Pinski and Narin stated the ranking problem as an eigenvalue problem as applied to 103 journals in physics. However, their approach did not attract sufficient attention, and simpler measures have remained in use. One advantage of bibliometric approaches is that they are simple and easy to understand. However, metric-based approaches exploit none of the semantics in topics. Because bibliometric approaches require manual calculation, they are not suitable for large-scale semantic social networks.

Another family of methods for ranking social actors has a basis in network topology and the relative position of the vertices on the network, which infers actors' social positions by measuring the centrality.

Probably the most famous algorithm for ranking networked entities is the PageRank [5] which defines the actor network as a weighted, directed graph, where $G = (V, E)$ with E are the edge weights and V are the actor vertices. Normalization of the edge weights allows construction of a Markov random walk, described by a square matrix, $P \in \mathbb{R}^{|V|\times|V|}$, where $i, j \in V$ and $p_{i,j} = P(j|i)$ denotes the conditional probability of the transition from vertex, i, to vertex, j. Assuming an ideal ranking exists and that the ranking scores are contained in $\mathbf{x} \in \mathbb{R}^{|V|\times 1}$, the PageRank paradigm [5, 38] suggests that the ranking score, x_i, for vertex, i, is the

weighted combination of all edges pointing to i, i.e.:

$$\mathbf{x} = P^T \mathbf{x}. \tag{1.1}$$

Thus, $\mathbf{x}$ is the principal eigenvector of the transpose of the Markov transition matrix, or P^T. In standard PageRank [5], the Markov transition matrix P is:

$$p_{i,j} = \begin{cases} \alpha \frac{\mathbb{I}((i,j)\in E)}{d_i} + (1-\alpha)\frac{1}{|V|} & \text{if } d_i \neq 0; \\ \frac{1}{|V|} & \text{otherwise.} \end{cases} \tag{1.2}$$

where $\mathbb{I}(x)$ is an indicator function; d_i is the out-degree of vertex i, and α is the parameter that balances the graph weights and the effect of randomness.

Since P uniquely determines the ranking scores, recent study has focused on the design of the Markov transition matrix [1, 71, 78]. This work argued that the arbitrary design of P in standard PageRank requires a strong domain-based hypothesis and might fail to fit certain ranking applications. In particular, advocates argue for a machine learning-based design of P because of its flexibility and readily accomplished generalization. Transition matrix learning [71], or network flow modeling, formulates the problem as a constrained entropy maximization of P. The entropy is maximized for generalization. The matrix, P, is required to satisfy several constraints for being a Markov process as well as a network flow, including $\forall v \in V, \quad \sum_j p_{v,j} = 1$ and $\sum_i p_{i,v} = \sum_j p_{v,j}$. Following the same entropy framework, introduced constraints [1] capture the *vertex-wise* preferences among vertices. If denoting the *vertex-wise preference* for v over u as $u \prec_a v$, then *vertex-wise preference* constraints are:

$$\forall u \prec_a v, \quad \sum_i p_{i,u} \leq \sum_i p_{i,v}, \tag{1.3}$$

where the sum of flow into vertex, u, is smaller than that to v. However, a practical problem with Eq. 1.3 is the availability of such *vertex-wise preferences*. For example, for the focus on actor ranking, a vertex-wise preference between two actors is a ranking of one actor over the other, which can not be readily derived from various social network information sources, such as social interactions, social document contents, and personal acknowledgments.

1.2.2 Discovering Communities of Social Actors

In addition to the ranking of heterogeneous social networks, the current study also concerns discovering communities among social actors.

Well known graph-theoretic methods include spectral graph partitioning [58, 13], hierarchical community discovery [79], and clustering[3] based on random walks [28]. Spectral graph partitioning is a classic spectral method based on the Laplacian of the graph adjacency matrix [58, 13], with a characteristic focus on the design of cost functions for partitioning graphs. Hierarchical community discovery seeks to merge the vertices and edges based on the "closeness" between vertices measured by distances on graphs, such as the length of the shortest paths or the diffusion distance [79]. Finally, random walk-based clustering described in [28] applies random walks to the graphs iteratively. By doing so, the edge weight between two vertices is modified based on the probabilities that the random walk will circle back to one of the vertices through the other.

Spectral graph partitioning is a classic spectral method for partitioning graphs and discovering communities on graphs [58, 13]. This method has been applied to various domains including image segmentations [65] and text analysis [82, 14, 12, 18, 36]. The principal aim of spectral graph partitioning is to minimize the

[3]Here, the term "clustering" and "community discovery" are used unless otherwise noted.

cost of cutting graphs as a function of the Laplacian of the graph adjacency matrix. The partitioning embeds a graph into a low-dimensional subspace subject to the minimal partitioning cost imposed by the graph adjacency matrix. After embedding the graph into the subspace, the clustering can be performed via an additional light-weight clustering algorithm (such as k-means) or by recursively searching for the binary cutting points [82] on the subspace axes. One traditional cost function uses the sum of weights on the edges between clusters [58]; however, this simple approach can create bias towards unbalanced cutting points. Recent work proposes variants to the cost function, including ratio cut, normalized cut, and others (a survey can be found in [13]). The most popular cost function for partitioning graphs is the Normalized Cut (NCut) [65]. The NCut cost function was originally applied to partitioning homogeneous or bipartite graphs [65, 82, 12]. Due to growing interest in analyzing correlated heterogeneous graphs, recent work generalizes NCut to star-structured tri-partite graphs and a solution has been proposed based on semi-definite programming [18]. Another recent work introduces prior knowledge into the cost function so partitioning will inflict minimal violation of prior knowledge as well [36].

1.3 Contributions

This dissertation seeks to combine the analysis of social document content with that of heterogeneous social actions. The study pays special attention to social documents and annotations in its analysis of social content. Of consideration are traditional social networks as well as heterogeneous social networks defined by various social actions. The inferences from social actions in heterogeneous social networks, leveraged for application, include ranking, community discovery,

information retrieval, and document recommendations.

The first part of this dissertation is analysis of social content. The wide availability of social content on the Web has its motivation, in part, form the semantic nature of that medium [2], which aims to render Web resources understandable to both humans and machines. A stream of Web applications with rich content, generated by users, have emerged and include Web blogs [39], social annotations (a.k.a social bookmarking) [22, 67, 80], and Web social networks [90]. This research addresses two important topics in content-based social network analysis: discovery of latent communities [39, 90] and analysis of social annotations [22].

(1) For community discovery, most recent research exploits the topology properties of social networks [52, 8]. However, discovering a community simply based on network topology sometimes becomes problematic due to a lack of consideration for semantics and an unreliability in the construction of networks (see Chapter 2). The methods this study proposes adequately combines document content and networks to allow for a greater number of semantic meanings to associate with the communities discovered.

(2) For social annotations, many popular applications exist including those of *delicious* (del.icio.us) and *flickr* (flickr.com), but the analysis of such social annotation data is still in its infancy. Much of the work focused on the study of the data properties, the analysis of usage patterns of tagging systems [22], and the discovery of hidden semantics in tags [80]. The objective of analyzing social annotations in the current research, however, is to leverage social annotations for improving user experience *information retrieval* (IR), This concept, while natural, is a barely explored area. An objective is to advance the value of previous investigation by combining the models of

social annotations with the models of language: the methods used in IR (see Chapter 3).

The second part of this study analyzes heterogeneous social networks. Interest in researching heterogeneous social networks arise naturally from the availability of heterogeneous actions and interactions by and among social actors. Throughout, this study construct models for heterogeneous actions among social actors and the interactions between social actors and other items such as heterogeneous networks. Interestingly, the evolution of social document content can be explained using the social interactions in such networks (see Chapter 4), which bridge the gap between social content and social actions. The later part of this research has particular interest in the problems of ranking, community discovery, and embedding of such networks. Traditional research in social network analysis focused on a single network (or homogeneous networks). For example, community discovery from social networks has employed methods which include spectral graph partitioning [58, 13], hierarchical community discovery [79], and clustering by random walks [28], all on a single network. Ranking methods for networked entities (such as PageRank [5] and HITS [38]) presume the existence of only a single network and only single types of vertices and edges. However, these methods are unable to deal with heterogeneous social networks with which this dissertation is concerned. Thus, the proposal is for new methods to address the analysis of heterogeneous social networks, in particular, the analysis of heterogeneous social networks from several aspects:

(1) A new framework unifies heterogeneous social actions based on network flow modeling, where the implicit preferences from various social actions parameterize the network flow (see Chapter 5). The network flow then ranks

social actors. The new learning-based ranking framework outperforms traditional methods and demonstrates the merits of combining heterogeneous social actions into a single framework.

(2) Ranking social actors can be further improved by leveraging, not only the relationship among actors, but also leveraging their relationships with other items, and resulting in a proposed new co-ranking method (see Chapter 6).

(3) After approaching the ranking problem, this research suggests a, new, effective and efficient method for partitioning temporal heterogeneous graphs for community discovery. (see Chapter 7).

(4) An approach to the more general problem of measuring graph node similarities combines multiple graphs (see Chapter 8), thus, showing that in real-world cases, a single graph is usually insufficient to depict the similarities among vertices due to sparsity and noise. The proposed graph embedding methods address three general types of graphs and propose different factorization techniques tailored to the unique characteristics of each graph type. Based on the obtained graph embedding, a new recommended framework is developed using semi-supervised learning on graphs.

1.4 Summary and Dissertation Organization

The availability of rich social content and semantic-rich social actions on the Web and the lack of appropriate computational approaches for such data are the primary reasons for development of much of this study's methodology.

Figure 1.2 illustrates the remainder of this study's organization. Following Chapter 1, the connection between the two parts of the study becomes clear. (1)

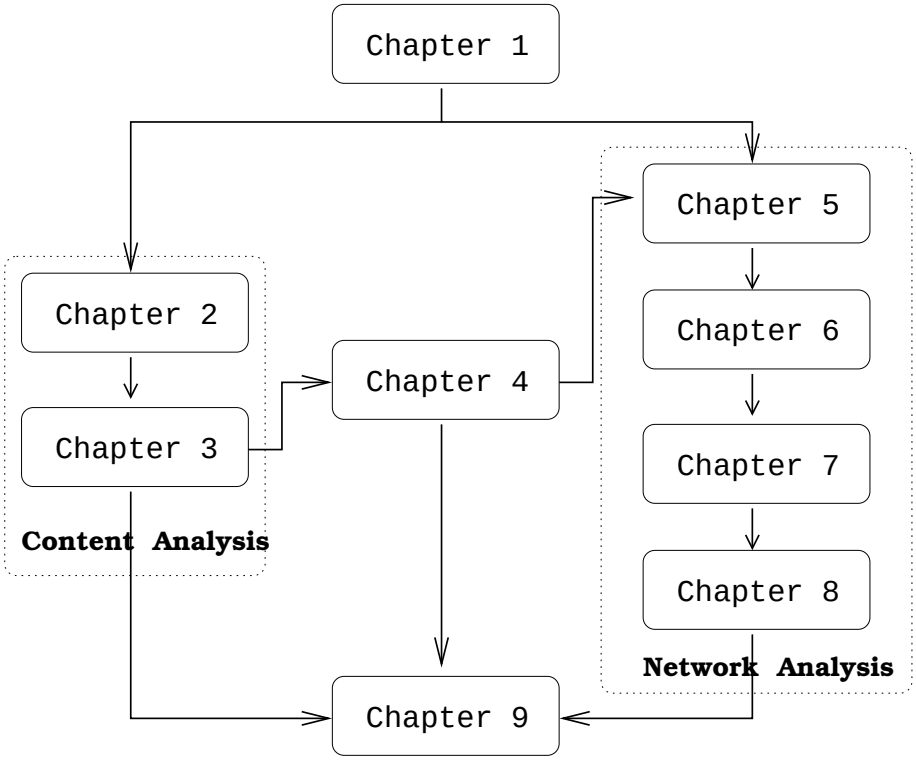


Figure 1.2. Dissertation Organization.

In Part 1 of the content analysis: Chapter 2 proposes new probabilistic models for social document content. Chapter 3 introduces another modeling method for social annotations and applies the model for information retrieval. (2) For the connection between Parts 1 and 2, Chapter 4 presents a new way for explaining the content evolution of social documents using latent social networks among authors. (3) For the network analysis in Part 2, Chapter 5 initiates focus on heterogeneous social networks and provides a uniform learning-based framework for modeling various social actions by network flows. Chapter 6 extends the research on ranking to heterogeneous networks. Chapter 7 introduces new methods for community discovery in heterogeneous social networks and presents a new technique for considering the temporal aspect while performing clustering on temporal data. Chapter 8 studies the general problem of measuring vertex similarities on multiple connected graphs

and applies the new method for recommending documents housed in digital libraries. Finally, conclusions are drawn in Chapter 9.

Chapter 2

Probabilistic Models for Social Documents

2.1 Community Discovery by Content Analysis

An important characteristic of all social networks (SN) is the community graph structure: how social actors gather into groups such that they are intra-group close and inter-group loose [53]. An illustration of a simple two-community SN is sketched in Fig. 2.1. Here each node represents a social actor in the SN and different node shapes represent different communities. Two nodes share an edge if and only if a relationship exists between them according to social definitions such as their role or participation in the social network.

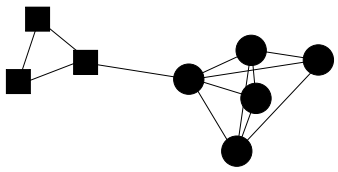


Figure 2.1. A social network with two communities.

Discovering community structures from general networks is of obvious interest.

For the extraction of community structures from email corpora [72, 8], the social network is usually constructed measuring the intensity of contacts between email users. In this setting, every email user is a social actor, modeled as a node in the SN. An edge between two nodes indicates that the existing email communication between them is higher than certain frequency threshold.

However, discovering a community simply based purely on communication intensity becomes problematic in some scenarios. (1) Consider a spammer in an email system who sends out a large number of messages. There will be edges between every user and the spammer, in theory presenting a problem to all community discovery methods which are topology based. (2) Aside from the possible bias in network topology due to unwanted communication, existing methods also suffer from the lack of semantic interpretation. Given a group of email users discovered as a community, a natural question is why these users form a community? Pure graphical methods based on network topology, without the consideration of semantics, fall short in answering to such questions.

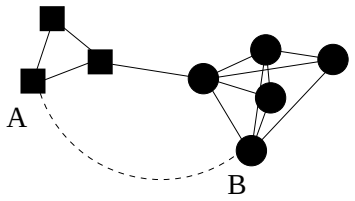


Figure 2.2. Semantic relationships and hidden communities.

Consider other ways a community can be established, e.g. Fig. 2.2. From the preset communication intensity, person A and person B belong to two different communities, denoted by squares and circles, based on a simple graph partitioning. However, ignoring the document semantics in their communications, their common interests (denoted by the dashed line) are not considered in traditional community discovery.

Much communication in SNs usually occurs by exchanging documents, such as emails, instant messages or posts on message boards [46]. Such content rich documents naturally serve as an indicator of the innate semantics in the communication among an SN. Consider an information scenario where all communications rely on email. Such email documents usually reflect nearly every aspect of and reasons for this communication. We define such a document carrier of communication as a *communication document*. In this chapter [90], we examine the inner community property within SNs by analyzing the semantically rich information, such as emails or documents. We approach the problem of community detection using a generative Bayesian network that models the generation of communication in an SN. As suggested in established social science theory [76], we consider the formation of communities as resulting from the similarity among social actors. The generative models we propose introduce such similarity as a hidden layer in the probabilistic model. Our main contribution is resolving the SN communication modeling problem into the modeling of generation of the *communication documents*, based on whose features the social actors associate with each other. Modeling communication based on *communication document* takes into consideration the semantic information of the document as well as the interactions among social actors. Many features of the SN can be revealed from the parameterized models such as the leader-follower relation [47]. Using such models, we can avoid the effect of meaningless *communication documents*, such as those generated by a network spammer, in producing communities. As a parallel study in social network with the sociological approaches, our method advances existing algorithms by not exclusively relying the intensity of contacts. We test our method on the newly disclosed email corpora benchmark – the Enron email dataset and compare with an existing method.

2.2 Community-User-Topic Models

Our definition for a *semantic community* in a social network is:

Definition 5. *A semantic community in a social network includes users with similar communication interests and topics that are associated with their communications.*

We study the community structure of an SN by modeling the *communication documents* among its social actors and the format of *communication documents* we model is email because emails embody valuable information regarding shared knowledge and the SN infrastructure [72].

Our Community-User-Topic (CUT) model[1] builds on the Author-Topic model. However, the modeling of a *communication document* includes more factors than the combination of authors and topics.

Serving as an information carrier for communication, a *communication document* is usually generated to share some information within a group of individuals. But unlike publication documents such as technical reports, journal papers, etc., the *communication documents* are inaccessible for people who are not in the recipient list. The issue of a *communication document* indicates the activities of and is also conditioned on the community structure within an SN. Therefore we consider the community as an extra latent variable in the Bayesian network in addition to the author and topic variables. By doing so, we guarantee that the issue of a *communication document* is purposeful in terms of the existing communities. As a result, the communities in an SN can be revealed and also semantically explainable.

[1] In order to fit our model literally to the social network built on email communication, we change the name "Author" to "User". An alternative name of our model is *Community-Author-Topic* Model: *CAT*.

We will use generative Bayesian networks to simulate the generation of emails in SNs. Differing in weighting the impact of a community on users and topics, two versions of CUT are proposed.

2.2.1 CUT_1: Modeling community with users

Given the impact of community in the generation of communication, the first step is to determine the interrelationships among this latent variable, the email users and the topics, i.e. the structure of the Bayesian network.

We first consider an SN community as no more than a group of users. This is a notion similar to that assumed in a topology-based method. For a specific topology-based graph partitioning algorithm such as *Modularity* [53], the connection between two users can be simply weighted by the frequency of their communications. In our first model CUT_1, we treat each community as a multinomial distribution over users. Each user u is associated with a conditional probability $P(u|c)$ which measures the degree that u belongs to community c. The goal is therefore to find out the conditional probability of a user given each community. Then users can be tagged with a set of topics, each of which is a distribution over words. A community discovered by CUT_1 is typically in the structure as shown in Fig. 2.7.

Fig. 2.3 presents the hierarchy of the Bayesian network for CUT_1. Let us use the same notations in Author-Topic model: α and β parameterizing the prior Dirichlet for topics and words. Let ψ denote the multinomial distribution over users for each community c, each marginal of which is a Dirichlet parameterized by γ. Let the prior probabilities for c be uniform. Let C, U, T denote the number of community, users and topics.

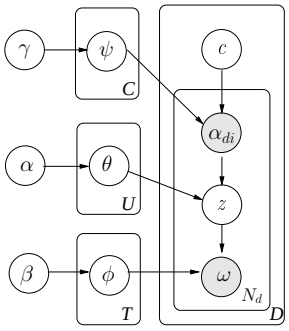


Figure 2.3. Modeling community with users

Typically, an email message d is generated by four steps: (1) there is a need for a community c to issue an act of communication by sending an email d; (2) a user u is chosen from c as observed in the recipient list in d; (3) u presents to read d since a topic z is concerned, which is drawn from the conditional probability on u over topics; (4) given topic z, a word ω is created in d. By iterating the same procedure, an email message d is composed word by word.

Note that the u is not necessarily the composer of the message in our models. This differs from existing literatures which assume α as the author of document. The assumption is that a user is concerned with any word in a *communication document* as long as the user is on the recipient list.

To compute $P(c, u, z|\omega)$, the posterior probability of assigning each word ω to a certain community c, user u and topic z, consider the joint distribution of all variables in the model:

$$\begin{aligned} P(c, u, z, \omega) =& P(\omega|z)P(c, u, z) \\ =& P(\omega|z)P(z|u)P(c, u) \\ =& P(\omega|z)P(z|u)P(u|c)P(c) \end{aligned} \tag{2.1}$$

Theoretically, the conditional probability $P(c, u, z|\omega)$ can be computed using the joint distribution $P(c, u, z, \omega)$.

A possible side-effect of CUT[1], which considers a community c solely as a multinomial distribution over users, is it relaxes the community's impact on the generated topics. Intrinsically, a community forms because its users communicate frequently and in addition they share common topics in discussions as well. In CUT_1 where community only generates users and the topics are generated conditioned on users, the relaxation is propagated, leading to a loose connection between community and topic. We will see in the experiments that the communities discovered by CUT_1 is similar to the topology-based algorithm Modularity proposed in [53].

2.2.2 CUT_2: Modeling community with topics

In contrast to CUT_1, our second model introduces the notion that an SN community consists of a set of topics, which are of concern to respective user groups.

As illustrated in Fig. 2.4, each word ω observed in email d is finally chosen from the multinomial distribution of a user α_{di}, which is from the recipient list of d. Before that, α_{di} is sampled from another multinomial of topic z and z is drawn from community c's distribution over topics.

Analogously, the products of CUT_2 are a set of conditional probability $P(z|c)$ that determines which of the topics are most likely to be discussed in community c. Given a topic group that c associates for each topic z, the users who refer to z can be discovered by measuring $P(u|z)$.

CUT_2 differs from CUT_1 in emphasizing the relation between community and topic. In CUT_2, semantics play a more important role in the discovery of com-

munities. Similar to CUT_1, the side-effect of advancing topic z in the generative process might lead to loose ties between community and users. An obvious phenomena of using CUT_2 is that some users are grouped to the same community when they share common topics even if they correspond rarely, leading to the different scenarios for which the CUT models are most appropriate. For CUT_1, users often tend to be grouped to the same communities while CUT_2 accentuates the topic similarities between users even if their communication seem less frequent.

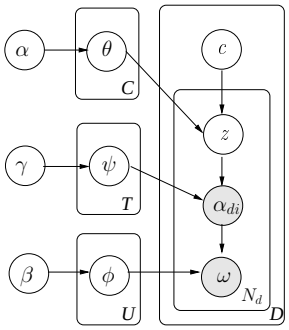


Figure 2.4. Modeling community with topics

Derived from Fig. 2.4, define in CUT^2 the joint distribution of community c, user u, topic t and word ω:

$$P(c, u, z, \omega) = P(\omega|u)P(u|z)P(z|c)P(c) \tag{2.2}$$

Let us see how these models can be used to discover the communities that consist of users and topics. Consider the conditional probability $P(c, u, z|\omega)$, a word ω associates three variables: community, user and topic. Our interpretation of the semantic meaning of $P(c, u, z|\omega)$ is the probability that word ω is generated by user u under topic z, in community c.

Unfortunately, this conditional probability cannot be computed directly. To

get $P(c, u, z|\omega)$,we have:

$$P(c, u, z|\omega) = \frac{P(c, u, z, \omega)}{\Sigma_{c,u,z} P(c, u, z, \omega)} \tag{2.3}$$

Consider the denominator in Eq. 2.3, summing over all c, u and z makes the computation impractical in terms of efficiency. In addition, as shown in [24], the summing doesn't factorize, which makes the manipulation of denominator difficult. In the following section, we will show how an approximate approach of Gibbs sampling will provide solutions to such problems. A faster algorithm EnF-Gibbs sampling will also be introduced.

2.3 The Algorithm

2.3.1 Gibbs sampling

Gibbs sampling is an algorithm to approximate the joint distribution of multiple variables by drawing a sequence of samples. It is a Markov chain Monte Carlo algorithm that usually applies when posterior probability is easier to evaluate. Gibbs sampling was first introduced to estimate the Topic-Word model in [24]. In Gibbs sampling, a Markov chain is formed, the transition between successive states of which is simulated by repeatedly drawing a topic for each observed word from its conditional probability on all other variables. In the Author-Topic model, the algorithm goes over all documents word by word. For each word ω_i, the topic z_i and the author x_i responsible for this word are assigned based on the posterior probability conditioned on all other variables: $P(z_i, x_i|\omega_i, \mathbf{z_{-i}}, \mathbf{x_{-i}}, \mathbf{w_{-i}}, \mathbf{a_d})$. z_i and x_i denote the topic and author assigned to ω_i, while $\mathbf{z_{-i}}$ and $\mathbf{x_{-i}}$ are all other

assignments of topic and author excluding current instance. $\mathbf{w_{-i}}$ represents other observed words in the document set and $\mathbf{a_d}$ is the observed author set for this document.

A key issue in using Gibbs sampling for distribution approximation is the evaluation of conditional posterior probability. In Author-Topic model, given T topics and V words, $P(z_i, x_i|\omega_i, \mathbf{z_{-i}}, \mathbf{x_{-i}}, \mathbf{w_{-i}}, \mathbf{a_d})$ is estimated by:

$$P(z_i = j, x_i = k|\omega_i = m, \mathbf{z_{-i}}, \mathbf{x_{-i}}, \mathbf{w_{-i}}, \mathbf{a_d}) \propto \tag{2.4}$$

$$P(\omega_i = m|x_i = k)P(x_i = k|z_i = j) \propto \tag{2.5}$$

$$\frac{C^{WT}_{mj} + \beta}{\Sigma_{m'} C^{WT}_{m'j} + V\beta} \frac{C^{AT}_{kj} + \alpha}{\Sigma_{j'} C^{AT}_{kj'} + T\alpha} \tag{2.6}$$

where $m' \neq m$ and $j' \neq j$, α and β are prior parameters for word and topic Dirichlet, C^{WT}_{mj} represents the number of times that word $\omega_i = m$ is assigned to topic $z_i = j$, C^{AT}_{kj} represents the number of times that author $x_i = k$ is assigned to topic j.

The transformation from Eq. 2.4 to Eq. 2.5 drops the variables, $\mathbf{z_{-i}}$, $\mathbf{x_{-i}}$, $\mathbf{w_{-i}}$, $\mathbf{a_d}$, because each instance of ω_i is assumed independent of the other words in a message.

By applying the Gibbs sampling, we can discover the semantic communities by using the CUT models. Consider the conditional probability $P(c, u, z|\omega)$, where three variables in the model, community, user[2] and topic, are associated by a word ω. The semantic meaning of $P(c, u, z|\omega)$ is the probability that ω belongs to user u under topic z, in community c. By estimation of $P(c, u, z|\omega)$, we can label a community with semantic tags (topics) in addition to the affiliated users. The

[2]Note we denote user with u in our models instead of x as in previous work.

problem of semantic community discovery is thus reduced to the estimation of $P(c, u, z|\omega)$.

---

(1) /* Initialization */
(2) for each word ω_i in each d
(3) assign ω_i to random community, topic and user;
(4) /* Markov chain convergence */
(5) $i \leftarrow 0$;
(6) $I \leftarrow$ desired number of iterations;
(7) while $i < I$
(8) for each ω_i in each email d
(9) get current assignment of ω_i: c, t, u;
(10) decrement assignments of ω_i for c, t, u;
(11) estimate $P(c_i, u_i, z_i|\omega_i)$, $u \in \alpha_{\mathbf{d}}$;
(12) sample c_p, u_q, z_r based on $P(c_i, u_i, z_i|\omega_i)$;
(13) increment assignment counts $\tau(c_p, u_q, z_r, \omega_i)$;
(14) $i++$;

Figure 2.5. Gibbs sampling for CUT models

The framework of Gibbs sampling is illustrated in Fig. 2.5. Given the set of users U, set of email documents D, the number of desired topic $|T|$, number of desired community $|C|$ are input, the algorithm starts with randomly assigning words to a community, user and topic. A Markov chain is constructed to converge to the target distribution. In each trial of this Monte Carlo simulation, a block of $(community, user, topic)$ is assigned to the observed word ω_i. After a number of states in the chain, the joint distribution $P(c, u, z|\omega)$ approximates the targeted distribution.

To adapt Gibbs sampling for CUT models, the key step is estimation of $P(c_i, u_i, z_i|w_i)$. For the two CUT models, we describe the estimation methods respectively.

Let $P(c_i = p, u_i = q, z_i = r|\omega_i = m, \mathbf{z_{-i}}, \mathbf{x_{-i}}, \mathbf{w_{-i}})$ be the probability that ω_i is generated by community p, user q on topic r, which is conditioned on all

the assignments of words excluding the current observation of ω_i. $\mathbf{z_{-i}}$, $\mathbf{x_{-i}}$ and $\mathbf{w_{-i}}$ represent all the assignments of topic, user and word not including current assignment of word ω_i.

In CUT_1, combining Eq. 2.1 and Eq. 2.3, assuming uniform prior probabilities on community c, we can compute $P(c = p, u = q, z = r|\omega_i = m, \mathbf{z_{-i}}, \mathbf{x_{-i}}, \mathbf{w_{-i}})$ for CUT_1 by:

$$\begin{aligned} P(c_i = p, u_i = q, z_i = r|\omega_i = m, \mathbf{z_{-i}}, \mathbf{x_{-i}}, \mathbf{w_{-i}}) \propto \\ P(\omega_i = m|z_i = r)P(z_i = r|u_i = q)P(u_i = q|c_i = p) \propto \\ \frac{C^{WT}_{mr} + \beta}{\Sigma_{m'} C^{WT}_{m'r} + V\beta} \frac{C^{TU}_{rq} + \alpha}{\Sigma_{r'} C^{TU}_{r'q} + T\alpha} \frac{C^{UC}_{qp} + \gamma}{\Sigma_{q'} C^{UC}_{q'p} + U\gamma} \end{aligned} \tag{2.7}$$

where $P(\omega_i = m|z = r)$, $P(z_i = r|u_i = q)$ and $P(u_i = q|c_i = p)$ are estimated via:

$$P(\omega_i = m|z_i = r) \propto \frac{C^{WT}_{mr} + \beta}{\Sigma_{m'} C^{WT}_{m'r} + V\beta} \tag{2.8}$$

$$P(z_i = r|u_i = q) \propto \frac{C^{TU}_{rq} + \alpha}{\Sigma_{r'} C^{TU}_{r'q} + T\alpha} \tag{2.9}$$

$$P(u_i = q|c_i = p) \propto \frac{C^{UC}_{qp} + \gamma}{\Sigma_{q'} C^{UC}_{q'p} + U\gamma}. \tag{2.10}$$

In the equations above, C^{WT}_{mr} is the number of times that word $\omega_i = m$ is assigned to topic $z_i = r$, not including the current instance. C^{TU}_{rq} is the number of times that topic $z = r$ is associated with user $u = q$ and C^{UC}_{qp} is the number of times that user $u = q$ belongs to community $c = p$, both not including the current instance. C is the number of communities in the social network given as an argument.

The computation for Eq. 2.8 requires keeping a $W \times T$ matrix C^{WT}, each entry C_{ij}^{WT} of which records the number of times that word i is assigned to topic j. Similarly, a $T \times U$ matrix C^{TU} and a $U \times C$ matrix C^{UC} are needed for computation in Eq. 2.9 and Eq. 2.10.

Similarly, $P(c_i = p, u_i = q, z_i = r | \omega_i = m, \mathbf{z_{-i}}, \mathbf{x_{-i}}, \mathbf{w_{-i}})$ is estimated based on the Bayesian structure in CUT_2:

$$P(c = p, u = q, z = r | \omega_i = m, \mathbf{z_{-i}}, \mathbf{x_{-i}}, \mathbf{w_{-i}}) \propto \frac{C_{mq}^{WU} + \beta}{\Sigma_{m'} C_{m'q}^{WU} + V\beta} \frac{C_{qr}^{UT} + \gamma}{\Sigma_{q'} C_{q'r}^{UT} + U\gamma} \frac{C_{rp}^{TC} + \alpha}{\Sigma_{r'} C_{r'p}^{TC} + T\alpha} \tag{2.11}$$

Hence the computation of CUT_2 demands the storage of three 2-D matrices: C^{WU}, C^{UT} and C^{TC}.

With the set of matrices obtained after successive states in the Markov chain, the semantic communities can be discovered and tagged with semantic labels. For example, in CUT_1, the users belonging to each community c can be discovered by maximizing $P(u|c)$ in C^{UC}. Then the topics that these users concern are similarly obtained from C^{TU} and explanation for each topic can be retrieved from C^{WT}.

2.3.2 Gibbs Sampling with entropy filtering

In this section, we further develop Gibbs sampling to improve computational efficiency and performance.

Consider two problems with Gibbs sampling illustrated in Fig. 2.5: (1) efficiency: Gibbs sampling has been known to suffer from high computational complexity. Given a textual corpus with $N = 10^6$ words. Let there be $U = 150$

users, $C = 10$ communities and $T = 20$ topics. An $I = 1000$ iteration Gibbs sampling has the worst time complexity $O(I * N * (U * C * T))$, which in this case is about $3 * 10^{13}$ computations. (2) performance: unless performed explicitly before Gibbs sampling, the algorithm may yield poor performance by including many non-descriptive words. For Gibbs sampling, some common words like 'the', 'you', 'and' must be cleaned before Gibbs sampling. However, the EnF-Gibbs sampling saves such overhead by automatically removing the non-informative words based on entropy measure.

---

(1) /* Initialization */
(2) assign each ω_i in each d to random topic, user and community;
(3) /* Markov chain convergence */
(4) $i \leftarrow 0$; $TrashCan \leftarrow \phi$;
(5) $I \leftarrow$ desired number of iterations;
(6) while $i < I$
(7) for each observed ω_i
(8) if $i < A$ /* in early iterations */
(9) get current assignment of ω_i: c, t, u;
(10) decrement assignments of ω_i for c, t, u;
(11) estimate $P(c_i, u_i, z_i|\omega_i)$, $u \in \alpha_{\mathbf{d}}$;
(12) sample c_p, u_q, z_r based on $P(c_i, u_i, z_i|\omega_i)$;
(13) increment assignment counts $\tau(c_p, u_q, z_r, \omega_i)$;
(14) else if $\omega_i \notin TrashCan$ /* removing non-informative words */
(15) if Entropy$(\omega_i) \leq \theta$
(16) get current assignment of ω_i: c, t, u;
(17) decrement assignments of ω_i for c, t, u;
(18) estimate $P(c_i, u_i, z_i|\omega_i)$, $u \in \alpha_{\mathbf{d}}$;
(19) sample c_p, u_q, z_r based on $P(c_i, u_i, z_i|\omega_i)$;
(20) increment assignment counts $\tau(c_p, u_q, z_r, \omega_i)$;
(21) else
(22) $TrashCan \leftarrow TrashCan \cup \{\omega_i\}$;
(23) $i++$;

Figure 2.6. EnF-Gibbs sampling

Fig. 2.6 illustrates the EnF-Gibbs sampling algorithm we propose. We incor-

porate the idea of entropy filtering into Gibbs sampling. During the interactions of EnF-Gibbs sampling, the algorithm keeps in $TrashCan$ an index of words that are not informative. After A times of iterations, we start to ignore the words that are either already in the $TrashCan$ or are non-informative. In Step 15 of Fig. 2.6, we quantify the informativeness of a word ω_i by the entropy of this word over another variable. For example, in CUT_1 where C^{WT} keeps the numbers of times ω_i is assigned to all topics, we calculate the entropy on the ith row of the matrix.

2.4 Experiments on Emails

We present experimental results of our models with the Enron email corpus. Enron email dataset was made public by the Federal Energy Regulatory Commission during its investigations and subsequently made available [64].

2.4.1 Semantic community representation

We processed the Enron email dataset by removing the common stop words. Each employee in Enron is identified by an email address. For brevity, we use only the email ids without organization suffixes hereafter.

In all of our experiments, we fixed the number of communities C at 6 and the number of topics T at 20 [3]. The smoothing hyper-parameters α, β and γ were set at $5/T$, 0.01 and 0.1 respectively. We ran 1000 iterations for both our Gibbs sampling and EnF-Gibbs sampling with the MySql database support. Because the quality of results produced by Gibbs sampling and our EnF-Gibbs sampling are very close, we simply present the results of EnF-Gibbs sampling hereafter.

[3] In this Chapter, for the sake of simplicity, we do not seek to automatically look for the number of communities nor the number of topics. In Chapter 3, we will use a heuristic using model perplexity for determining the number of latent topics.

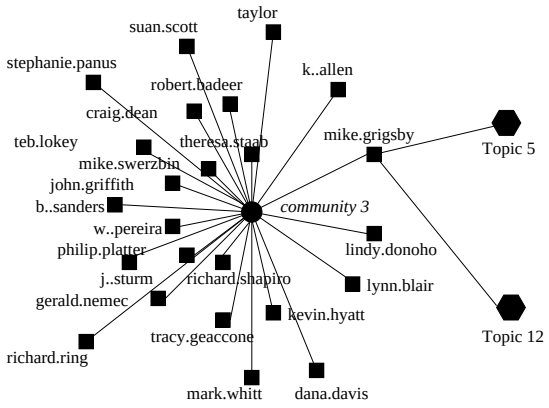


Figure 2.7. A Community Discovered by CUT_1

The ontology for both models are illustrated in Fig. 2.7 and Fig. 2.10. In both figures, we denote user, topic and community by square, hexagon and dot respectively. In CUT_1 results, a community connects a group of users and each user is associated with a set of topics. In Fig. 2.7, we present all the users and two topics of one user for a discovered community. By merging all the topics for the desired users of a community, we can tag a community with topic labels.

Topic 3	**Topic 5**	**Topic 12**	**Topic 14**
rate	dynegy	budget	contract
cash	gas	plan	monitor
balance	transmission	chart	litigation
number	energy	deal	agreement
price	transco	project	trade
analysis	calpx	report	cpuc
database	power	group	pressure
deals	california	meeting	utility
letter	reliant	draft	materials
fax	electric	discussion	citizen

Table 1: Topics Discovered by CUT_1

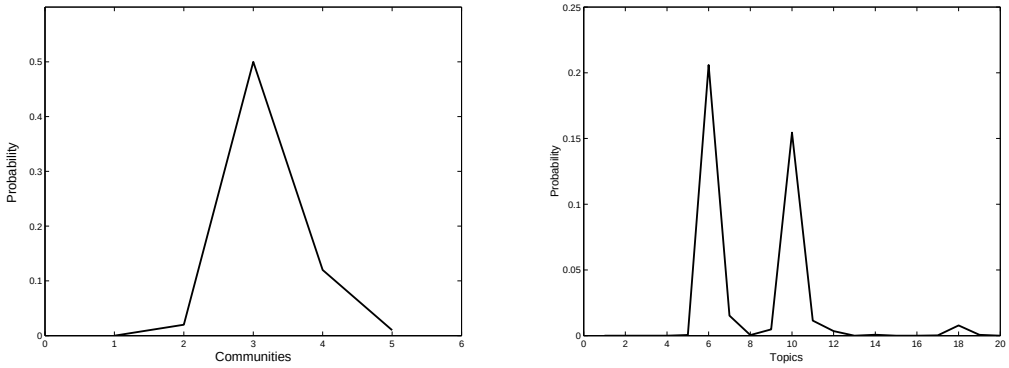
Fig. 2.7 shows that user mike.grigsby is one of the users in community 3. Two of the topics that is mostly concerned with mike.grigsby are topic 5 and topic 12. Table 1 shows explanations for some of the topics discovered for this community. We obtain the word description for a topic by choosing 10 from the top 20 words that maximize $P(w|z)$. We only choose 10 words out of 20 because there exist some names with large conditional probability on a topic that for privacy concern we do not want to disclose.

abbreviations	**organizations**
dynegy	An electricity, natural gas provider
transco	A gas transportation company
calpx	California Power Exchange Corp.
cpuc	California Public Utilities Commission
ferc	Federal Energy Regulatory Commission
epsa	Electric Power Supply Association
naruc	National Association of Regulatory Utility Commissioners

Table 2: Abbreviations

We can see from Table 1 that words with similar semantics are nicely grouped to the same topics. For better understanding of some abbreviate names popularly used in Enron emails, we list the abbreviations with corresponding complete names in Table 2.

For a single user, Fig. 2.8 illustrates its probability distribution over communities and topics as learned from the CUT_1 model. We can see the multinomial distribution we assumed was nicely discovered in both figures. The distribution over topics for all users are presented in Fig. 2.9. From Fig. 2.9, we can see some Enron employees are highly active to be involved in certain topics while some are



(a) Over communities (b) Over topics

Figure 2.8. Communities/topics of an employee

relatively inactive, varying in heights of peaks over users.

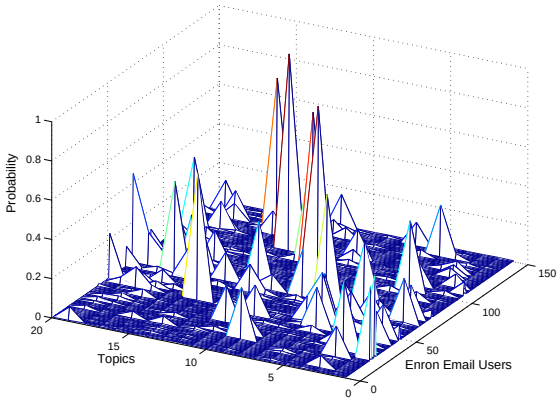


Figure 2.9. Distribution over topics for all users

Fig. 2.10 illustrates a community discovered by CUT_2. According to the figure, Topic 8 belongs to the semantic community and this topic concerns a set of users, which includes rick.buy whose frequently used words are more or less related to business and risk. Surprisingly enough, we found the words our CUT_2 learned to describe such users were very appropriate after we checked the original positions of these employees in Enron. For the four users presented in Table 3, d..steffes was the vice president of Enron in charge of government affairs; cara.semperger was a senior analyst; mike.grigsby was a marketing manager and rick.buy was the chief

risk management officer.

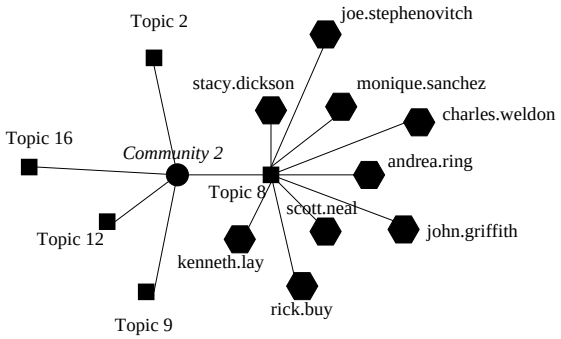


Figure 2.10. A Community Discovered by CUT_2

d..steffes	**cara.s**	**mike.grigsby**	**rick.buy**
power	number	file	corp
transmission	cash	trader	loss
epsa	ferc	report	risk
ferc	database	price	activity
generator	peak	customer	validation
government	deal	meeting	off
california	bilat	market	business
cpuc	caps	sources	possible
electric	points	position	increase
naruc	analysis	project	natural

Table 3: Distribution over words of some users

2.4.2 Semantic community discovery quality

We evaluate the quality of discovered communities against the topology-based algorithm in [8], a hierarchical agglomeration algorithm for community structure detection. The algorithm is based on Modularity, which is a measurement of whether a division of a network is a good one, in the sense that there are many

edges within communities and only a few between them. We employ the clustering comparison method in [59] to measure the similarity between our communities and the clusters of users produced by [8].

Given N data objects, the similarity between two clustering results λ is defined[4]:

$$\lambda = \frac{N_{00} + N_{11}}{N(N-1)/2}$$

where N_{00} denotes the count of object pairs that are in different clusters for both clustering and N_{11} is the count of pair that are in the same cluster.

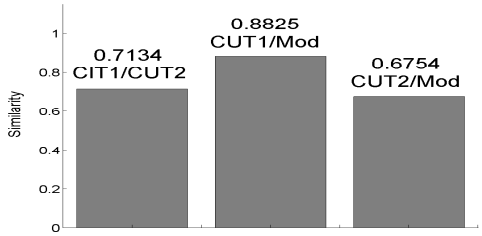


Figure 2.11. Community similarity comparisons

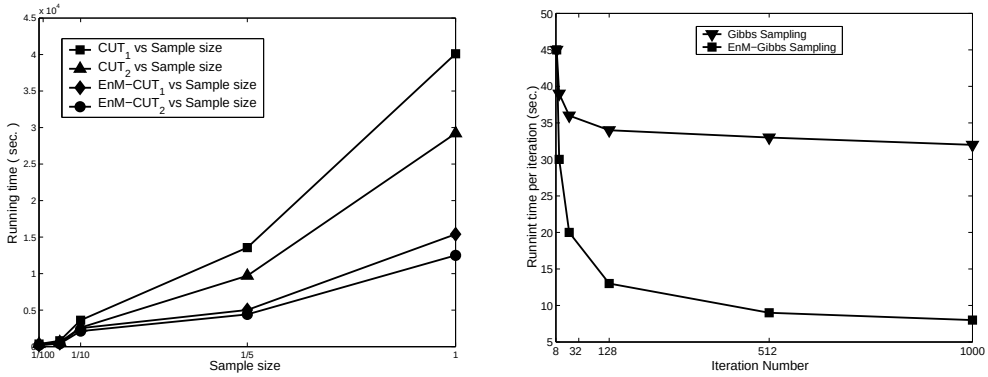
The similarities between three CUT models and Modularity are illustrated in Fig. 2.11. We can see that as we expected the similarity between CUT_1 and *Modularity* is large while that between CUT_2 and *Modularity* is small. This is because the CUT_1 is more similar to *Modularity* than CUT_2 by defining a community as no more than a group of users.

We also test the similarity among topics(users) for the users(topics) which are discovered as a community by CUT_1 (CUT_2). Typically the topics(users) associated with the users(topics) in a community represent high similarities. For example, in Fig. 2.7, Topic 5 and Topic 12 that concern mike.grigsby are both

[4]Another recent work on comparing clusterings is defined introduced in [92]. But for our problem where cluster labels are categorical, both clustering comparison perform similarly as suggested.

contained in the topic set of lindy.donoho, who is the community companion of user mike.grigsby.

2.4.3 Computational complexity and EnF-Gibbs sampling



(a) Time vs. sample size (b) Time vs. iterations

Figure 2.12. Computational complexity

We evaluate the computational complexity of Gibbs sampling and EnF-Gibbs sampling for our models. We measure the computational complexity based on (1) total running time and (2) iteration-wise running time. For overall running time we sampled different scales of subsets of messages from Enron email corpus. For the iteration-wise evaluation, we ran both Gibbs sampling and EnF-Gibbs sampling on complete dataset.

In Fig. 2.12(a), the running time of both sampling algorithms on two models are illustrated. We can see that generally learning CUT_2 is more efficient than CUT_1. It is a reasonable result considering the matrices for CUT_1 are larger in scales than CUT_2. Also entropy filtering in Gibbs sampling leads to 4 to 5 times speedup overall.

The step-wise running time comparison between Gibbs sampling and EnF-Gibbs sampling is shown in Fig. 2.12(b). We perform the entropy filtering removal

after 8 iterations in the Markov chain. We can see the EnF-Gibbs sampling well outperforms Gibbs sampling in efficiency. Our experimental results also show that the quality of EnF-Gibbs sampling and Gibbs sampling are almost the same.

2.5 Summary

We present two versions of Community-User-Topic models for semantic community discovery in social networks. Our models combine the generative probabilistic modeling with community detection. To simulate the generative models, we introduce EnF-Gibbs sampling which extends Gibbs sampling based on entropy filtering. Experiments have shown that our method effectively tags communities with topic semantics with better efficiency than Gibbs sampling.

Chapter 3

Probabilistic Models for Social Annotations

3.1 Social Annotation and Information Retrieval

The social annotating is a form of *folksonomy*, which by definition refers to Internet-based methods consisting of collaboratively generated, open-ended text labels that categorize content such as Web pages, online photographs, and Web links. Many popular Web services rely on folksonomy including those of *delicious* (del.icio.us) and *flickr* (flickr.com). Incorporating social annotations with document content is a natural idea, especially for IR applications. Consider the IR methods based on language modeling, for example [57, 41], we may simply treat the terms in annotation tags the same as those in document content, consider them as additional terms of the documents, and then follow the existing IR approaches. The pitfalls here, however, come in several aspects: (1) A tag term is generated differently than a document content term. A tag, upon its generation by a user, represents an abstract of the document from *one* perspective of *one* user; (2) The

differences in domain expertise of users should be taken into consideration when incorporating user tags. Some users in certain domains might be more trustworthy than others. Some users for various reasons may just give totally bogus tags. Although it remains an open problem to discover domain expertise of users, such peer differences are believed to be important [33] for IR in an information society; (3) The improvement for IR will be limited without considering the semantics of the tag terms. Usually the number of tag terms is much smaller than the number of terms in a document being tagged. Therefore using the tag terms the same way as the document terms will leads to the same difficulties in traditional language modeling-based IR, such as the lack of smoothness and the sparsity of observations.

In this chapter, we develop a framework that combines the modeling of social annotations with the expansion of traditional language modeling-based IR using user domain expertise [93]. Firstly, we seek to discover topics in the content and annotations of documents and categorize the users by domains. We propose a probabilistic generative model for the generation of document content as well as the associated tags. Secondly, we follow an IR framework based on risk minimization proposed earlier [41]. The framework is based on Bayesian decision theory focusing on improving language models for queries and documents. We then study several ways for expanding the language models where the user domain interests and expertise and the background collection language models are incorporated. In particular, we apply linear smoothing between the original term-level language models and the new topic-level language models. The newly proposed framework benefits from the consideration of the differences between document content terms and tag terms in the modeling process. User domain expertise can be readily included in the retrieval framework by the proposed ways of language model expansion. The smoothing of the original term-level language model with the topic-level

language models addresses the issues raised by the sparsity of observations.

The main contributions of our work presented in this chapter include (1) a general and a simplified probabilistic generative model for the generation of document content as well as the associated social annotations; (2) a new way for categorizing users by domains based on social annotations. The user domain expertise, evaluated by activity frequency, are considered to weigh the user interests; (3) the study of several ways for combining term-level language models with those topic-level models obtained from topics in documents and users.

3.2 Modeling Social Annotations

We propose a probabilistic generative model for social annotations. The model specifies the generation process of the terms in document content as well as the associated user tags. The motivation for modeling the social annotations with document content is to obtain a simultaneous topical analysis of the terms, documents, as well as the users. As we will discuss later, the topical analysis of terms (or the clustering of them by topics) essentially provides the basis for expanding query and document language models. In addition, the topical analysis of users, which categorizes the users by domains, enables the input of domain expertise of users in addition to the tags generated by them.

3.2.1 Generative models for annotations

We start by modeling the generation of words in documents and annotations. Intuitively, the content of documents and annotations are generated by two similar but correlated approaches. We illustrate our understanding of the generation

process in plate notation in Fig. 3.1. On the document side (left-hand side), for an arbitrary word ω in document d, a topic z is first drawn, then conditioned on this topic, ω is drawn; Repeating this process for N_d times, which is the number of words in d, d is generated. The whole collection repeats the same process for D times [1]; On the annotation side (right-hand side), each word in the annotation is generated similarly. First, an observed user a decides to make annotation on a particular document, then the user picks a topic z to describe the d, followed by the generation of ω. The generation of z by user, however, depends not only on the user but also the topic of d. Note the dependency of user topics on document topics can be seen as a mapping between two conceptions. Generally speaking, due to likely differences in perception of document content and annotation content, there are different number of topics on both sides, T_d and T_a. The two topic sets can be different but are usually very similar.

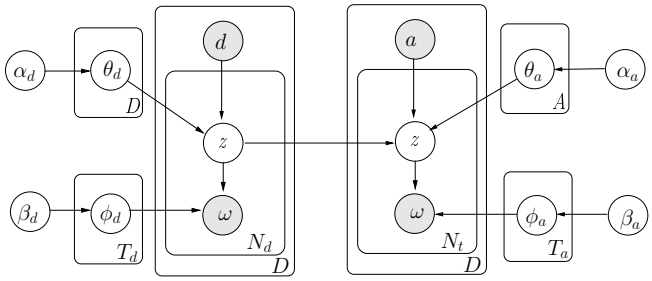


Figure 3.1. The general generative model for content of documents and annotations in plate notation. T_d (T_a) is the number of topics in documents (annotations); N_d (N_t) is the number of content words (or tag words) in document d; A and D are the number of users and documents; θ_d, θ_a ϕ_d, and ϕ_a are Dirichlet priors parameterized respectively by, α_d, α_a, β_d, and β_a. Shaded circles denote the observed variables and the blank circles denote the hidden ones. Rectangles denote the repetition of models with the same structure but different parameters, where the lower-right symbol indicates the number of repetitions.

Inspired by related work on topic analysis [3, 62, 68], we make assumptions

[1]Note the document side of the general annotation model is essentially the LDA model proposed in [3]. But the right side takes into consideration the generation of annotations as dependent on the document content generation.

about the probability structures of the generative model in Fig. 3.1. First, we assume all the conditional probabilities follow multinomial distribution. For example, each topic is a multinomial distribution over words where for the conditional probability of each word is fixed. Second, we assume that the prior distribution for topics and words follow Dirichlet (θ_d,ϕ_d for documents and θ_a,ϕ_a for annotations), which are conjugate priors for multinomial, respectively parameterized by α_d,β_d and α_a, β_a.

Given the great number of latent variables and parameters, the generative model, illustrated in Fig. 3.1, is not quite scalable in practice. The probability distributions we would have to estimate include: (1) $D + A$ multinomial distributions for documents over topics; (2) $T_d + T_a$ multinomial distributions for topics over words; (3) $T_d \times T_a$ conditional probabilities to capture the correlation of the topics in documents and the topics in annotations. In addition, there are many parameters that adds difficulty in tuning in practice (α_d, β_d, α_a, β_a, T_d, and T_a). Therefore, we will simplify this general annotation model with some relaxations in assumptions, arriving at a scalable model with easy training algorithms available.

In order to reduce the general model to a one scalable with fewer parameters, we make several compromises in assumptions. First, we assume the topics in documents and annotations are the same. This assumes that the taggers conceptually agree with the original document authors without variation of information in their understanding. Second, we assume that documents and users have the same structure of prior distributions which are only parameterized differently. Although arguably the users and documents might have different types of distributions over topics, we make the assumption here for the sake of simplicity. These assumptions lead to a simplified generative model for annotations. As illustrated in Fig. 3.2, we have a single topic-word distribution ϕ with parameter β; a single source-topic

distribution with extended dimension (here the source can be a document or a tagger). Now we have much fewer distributions to estimate, making the modeling more scalable in practice.

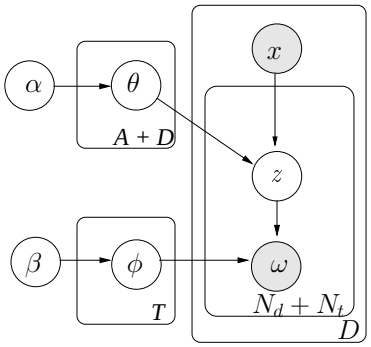


Figure 3.2. The User-Content-Annotation (UCA) Model in plate notation. T, A, and D are the number of topics, users, and documents. N_d and N_t denote the number of terms in the document and the number of terms in the tag. ϕ is the topic-word distribution with parameter β; θ is the source-topic distribution with parameter α.

Let us name the the model in Fig. 3.2 as the user-content-annotation (UCA) model. The UCA model describes the generation of words in document content and in the tags in similar but different processes. For document content, each observed term ω in document d is generated from the source x (each document d maps one-to-one to a source x). Then from the conditional probability distribution on x, a topic z is drawn. Given the topic z, ω is finally generated from the conditional probability distribution on the topic z. For document tags, similarly, each observed tag word ω for document d is generated by user x. Specific to this user, there is a conditional probability distribution of topics, from which a topic z is then chosen. This hidden variable of topic again finally generates ω in the tag.

According to the model structure, we have the joint probability conditional joint probability of θ, ϕ, x, z, ω, given the parameters α, β, as below:

$$P(\theta, \phi, x, z, w|\alpha, \beta) = \tag{3.1}$$

$$P(w|z, \phi)P(\phi|\beta)P(z|x, \theta)P(\theta|\alpha)P(x); \tag{3.2}$$

For inferences of words, we can calculate the conditional probability given a word as:

$$P(\theta, \phi, x, z, |\omega, \alpha, \beta) = \frac{P(\theta, \phi, x, z, \omega|\alpha, \beta)}{\sum_x \sum_z P(\theta, \phi, x, z, \omega|\alpha, \beta)}. \tag{3.3}$$

Again, similar to related work, we assume the prior distribution of topics and terms follow Dirichlet distributions parameterized respectively by α and β. Let T be the number of topics (input as a parameter); A is the number of users; D is the number of documents; N_d and N_t respectively denote the number of terms in the document and the number of terms in the tag. Each topic is a probabilistic multinomial distribution over terms, denoted by ϕ; Each user (or source) is a probabilistic multinomial distribution over topics, denoted by θ. As illustrated in Fig. 3.2, there are $A + D$ distributions of topics, each of which corresponds to an observed user or source. There are T distributions of words, each corresponds to an unobserved topic. For each document, the generation process repeats for $N_d + N_t$ times where N_d of the iterations correspond to the terms in the document content and N_t corresponds to the terms in the tags. The above again repeats for D times for all documents.

3.2.2 Model training

The UCA model includes two sets of unknown parameters, the source-topic distributions θ, and the topic-word distributions ϕ, corresponding to the assignments of individual words to topics z and source x. We use an effective parameter estimation method, Gibbs sampling [61], for training the model, which is gaining popularity in topic analysis recently [24, 90]. Instead of estimating the parameters directly, we evaluate the posterior distributions.

The algorithm keeps track of the number of times that a term is assigned to a topic C_{zw}^{TW} and the number of times that a topic is assigned to the user or source $C_{xz}^{(A+D)T}$. Here C^{TW} denotes a $T \times W$ matrix and $C^{(A+D)T}$ denotes a $(A+D) \times T$ matrix, where x, z, ω are the indices of the sources (document or user), topics, and words. We repeat the Gibbs sampling until the perplexity score [2] measured on distributions converges. The algorithm below illustrates the Gibbs sampling algorithm for model training.

It can be seen from the algorithm that the key issue here is the evaluation of the posterior conditional probabilities, i.e. $P(z|w)$, $P(d|z)$, $P(x|z)$, which leads to the evaluation of $P(d|w)$ or $P(x|w)$. Let us again consider the joint probabilities $P(x, z|w)$, $P(d, z|w)$. These posterior conditional probabilities can be expressed as the product of several conditional probabilities on the edges of the Bayesian network. In particular, for documents, we have:

$$P(d, z|\omega) \quad \propto \quad \frac{C_{\omega z}^{WT} + \beta}{\sum_k C_{kz}^{WT} + V\beta} \frac{C_{dt}^{(A+D)T} + \alpha}{\sum_k C_{dk}^{(A+D)T} + T\alpha}, \tag{3.4}$$

[2] The measurement of perplexity will be introduced in Sec. 3.2.3.

Algorithm 1 Training User-Content-Annotation Model

Given a sequence of triplets $\langle x, d, \omega \rangle$, where d is the document id; ω is the word id; x = nil if ω is a content word; x = user id if ω is a tag word.
Given ϵ as the threshold for determining convergence.
Initialize C^{TW}, $C^{(A+D)T}$ with random positive values.
repeat
  for all $\langle x, d, \omega \rangle$ **do**
    t = z(ω) // get the current topic assignment
    $C^{TW}_{tw} \leftarrow C^{TW}_{tw} - 1$ //decrement count
    if $x ==$ nil **then**
      // ω is a document word
      $C^{(A+D)T}_{dt} \leftarrow C^{(A+D)T}_{dt} - 1$ // decrement count
      // compute $P(t)$ below
      for all z = 1, ..., T **do**
        $P(z) \leftarrow P(d, z|\omega) = P(d|z)P(z|\omega)$
      end for
      sample to obtain t using $P(t)$
      $C^{(A+D)T}_{dt} \leftarrow C^{(A+D)T}_{dt} + 1$ // increment count
    else
      // ω is a tag word
      $C^{(A+D)T}_{xt} \leftarrow C^{(A+D)T}_{xt} - 1$ // decrement count
      // compute $P(t)$ below
      for all z = 1, ..., T **do**
        $P(z) \leftarrow P(x, z|\omega) = P(x|z)P(z|\omega)$
      end for
      sample to obtain t using $P(t)$
      $C^{(A+D)T}_{xt} \leftarrow C^{(A+D)T}_{xt} + 1$ // increment count
    end if
    $C^{TW}_{tw} \leftarrow C^{TW}_{tw} + 1$
  end for
  measure the perplexity on a held-out sample;
  measure the perplexity change in δ;
until $\delta \leq \epsilon$

and for users, we have:

$$P(x, z|\omega) \quad \propto \quad \frac{C^{WT}_{\omega t} + \beta}{\sum_k C^{WT}_{kz} + V\beta} \frac{C^{(A+D)T}_{xt} + \alpha}{\sum_k C^{(A+D)T}_{xk} + T\alpha}. \tag{3.5}$$

Here the unit conditional probabilities in fact are Bayesian estimation of the pos-

teriors: $P(d|z)$, $P(x|z)$ and $P(z|w)$:

$$P(d|z) = \frac{C_{dz}^{(A+D)T} + \alpha}{\sum_k C_{dk}^{(A+D)T} + T\alpha}, \tag{3.6}$$

$$P(x|z) = \frac{C_{xt}^{(A+D)T} + \alpha}{\sum_k C_{xk}^{(A+D)T} + T\alpha}, \tag{3.7}$$

$$P(z|\omega) = \frac{C_{\omega t}^{WT} + \beta}{\sum_k C_{kt}^{WT} + V\beta}. \tag{3.8}$$

Accordingly, for implementation, we need to keep track of $\sum_k C_{dk}^{(A+D)T}$, $\sum_k C_{xk}^{(A+D)T}$ and $\sum_k C_{kt}^{WT}$ in addition to $C_{dt}^{(A+D)T}$, $C_{xt}^{(A+D)T}$ and C_{tw}^{TW}. It is easy to implement these counting using several hash tables. In practice, we set α and β to be $50/T$ and 0.05 respectively.

3.2.3 Number of topics

The remaining question is how to select the number of topics. We resort to the perplexity measure, which is a standard measure for estimating the performance of a probabilistic model. The perplexity of a set of term-source test pairs, $< \mathbf{w}_d, \mathbf{x}_d >$, for all $d \in D_{test}$ documents, is defined as the exponential of the negative normalized predictive log-likelihood using the trained model:

$$\text{perplexity}(D_{test}) = \exp[-\frac{\sum_{d=1}^{D} \ln P(\mathbf{w}_d|\mathbf{x}_d)}{\sum_{d=1}^{D} |\{\mathbf{w_d}, \mathbf{x_d}\}|}]. \tag{3.9}$$

Here the probability of a set of term-source pairs on a particular document is obtained by a straightforward calculation:

$$P(\mathbf{w}_d|\mathbf{x}_d) = \prod_{(w_d,x_d)\in\{\mathbf{w_d},\mathbf{x_d}\}} P(w_d|x_d) \tag{3.10}$$

where the probability of an individual term-source pair $P(w_d|x_d)$ is evaluated using the model hierarchy:

$$P(w_d|x_d) = \sum_{t=1}^{T} P(w_d|t)P(t|x_d). \tag{3.11}$$

Note that the better generalization performance of a model is indicated by a lower perplexity score over a held-out document set. We run the Gibbs sampling using perplexity score as the termination criterion; the topic number is determined by using the smallest T that leads to the near maximum perplexity. Similar approach is also used in previous work [3, 62].

3.3 Information Retrieval based on Risk Minimization

In the language modeling (LM) approach to information retrieval (IR), queries and documents are modeled respectively by a probabilistic LM. Let θ_Q denote the parameters of a query model, and let θ_D denote the parameters of a document model. The LM-based IR involves two independent phases: In one case, the generation of a query is viewed as a probabilistic process associated with a certain user. This user first selects the query model θ_Q then picks a query $\mathbf{q}$ from the query model θ_Q with probability $P(\mathbf{q}|\theta_Q)$; In the other case, the document generation has been carried out. First the document language model θ_D is chosen and then the d is generated word by word with probability $P(\mathbf{d}|\theta_D)$. The task of an IR system is to determine the probability of a document being relevant to the query given their LMs are respectively estimated.

Here we work within a risk minimization framework for IR proposed earlier [41].

Suppose the relevance is a binary random variable $R \in \{0, 1\}$. Consider the task of a retrieval system as the problem of returning a list of documents to the issued query $\mathbf{q}$. In the general framework of Bayesian decision theory, to each action, there is an associated loss, which, in our case, is the loss for returning a particular document to the user. Assume that the loss function only depends on θ_Q, θ_D, and R_i, the expected risk of returning $\mathbf{d_i}$ is:

$$R(\mathbf{d_i}; \mathbf{q}) = \sum_{R \in \{0,1\}} \int_{\Theta_Q} \int_{\Theta_D} L(\theta_Q, \theta_D, R) \times P(\theta_Q|\mathbf{q}) P(\theta_D|\mathbf{d_i}) P(R|\theta_Q, \theta_D) d\theta_D d\theta_Q \tag{3.12}$$

where $L(\theta_Q, \theta_D, R)$ is the loss function, $P(\theta_Q|\mathbf{q})$ is the probability of the query model being parameterized by θ_Q given the query $\mathbf{q}$, $P(\theta_D|\mathbf{d_i})$ is the probability of the document model being parameterized by θ_D given the document $\mathbf{d_i}$, and $P(R|\theta_Q, \theta_D)$ is the probability of relevance of R given the parameter sets are θ_Q and θ_D.

Following earlier work [41], we make the assumption that the loss function only depends on θ_Q and θ_D and is proportional to the distance between θ_Q and θ_D. Let there be a distance function between two language models named as Δ. We have:

$$L(\theta_Q, \theta_D, R) \quad \propto \quad \Delta(\theta_Q, \theta_D) \tag{3.13}$$

The expected risk for returning $\mathbf{d_i}$ to $\mathbf{q}$ is thus:

$$R(\mathbf{d_i}; \mathbf{q}) \propto \int_{\Theta_Q} \int_{\Theta_D} \Delta(\theta_Q, \theta_D) P(\theta_Q|\mathbf{q}) P(\theta_D|\mathbf{d_i}) d\theta_D d\theta_Q. \tag{3.14}$$

Note here $P(\theta_Q|\mathbf{q})$ depends on the input $\mathbf{q}$ only and is the same for all candidate documents $\mathbf{d_i}$. Rather than explicitly computing the risk in the integral format, we can use the point estimate with the posterior θ_D and θ_D:

$$R(\mathbf{d_i};\mathbf{q}) \quad \propto \quad \Delta(\widehat{\theta}_q, \widehat{\theta}_{d_i}).P(\theta_D|\mathbf{d_i}) \tag{3.15}$$

where $\widehat{\theta}_q$ and $\widehat{\theta}_{d_i}$ can be obtained using maximum likelihood estimation observing the words in query and documents.

Further assuming that $P(\theta_D|\mathbf{d_i})$ is the same for all $\mathbf{d_i}$, the risk minimization framework finally becomes a measurement of the distance between two LMs: $\widehat{\theta}_q$ and $\widehat{\theta}_{d_i}$. As in other related work, we can employ the Kullback-Leibler divergence to measure Δ, yielding

$$R(\mathbf{d_i};\mathbf{q}) \quad \propto \quad \Delta(\widehat{\theta}_q, \widehat{\theta}_{d_i}) \;=\; \sum_w P(w|\widehat{\theta}_q) \log \frac{P(w|\widehat{\theta}_q)}{P(w|\widehat{\theta}_{d_i})}. \tag{3.16}$$

where w is a word in either language model $\widehat{\theta}_q$ or $\widehat{\theta}_{d_i}$.

According to Eq. 3.16, the setup of the risk minimization framework has made the measurement of relevance depend only on the LMs of the query and the document, i.e. the posterior parameters $\widehat{\theta}_q$ and $\widehat{\theta}_{d_i}$. This chapter proposes a refinement of the query and document LMs using the LMs obtained from social annotations.

3.4 Language Model Expansion using Social Annotations

Define our goal now to be improving the LMs of query and documents, say $\widehat{\theta}_q \rightarrow \theta'_q$ and $\widehat{\theta}_{d_i} \rightarrow \theta'_{d_i}$. Here the $\widehat{\theta}_q \rightarrow \theta'_q$ is also known as *query expansion* [40] and the $\widehat{\theta}_{d_i} \rightarrow \theta'_{d_i}$ is also known as *document expansion* [69].

There are several ways for LM expansion. In this chapter we focus on the linear interpolation [35] (a.k.a linear smoothing) for combining two LMs. Define an operator $\oplus\lambda$ for linear smoothing where $a \oplus_\lambda b \equiv \lambda a + (1-\lambda)b$, assuming a, b are both normalized to the same scale. When applied to combining two LMs, θ_1 and θ_2, we define that $\theta_1 \oplus_\lambda \theta_2 \quad \equiv$:

$$\forall v \in \theta_1 \cup \theta_2, \quad P(v|\theta_1 \oplus_\lambda \theta_2) = \quad \lambda P(v|\theta_1) + (1-\lambda)P(v|\theta_2) \tag{3.17}$$

where the v here can be a word, a phrase, or simply a token that denotes special meaning (e.g. a topic). In the case when $v \notin \theta_1$, $P(v|\theta_1 \oplus_\lambda \theta_2) = (1-\lambda)P(v|\theta_2)$. Similarly, $P(v|\theta_1 \oplus_\lambda \theta_2) = \lambda P(v|\theta_1)$ when $v \notin \theta_2$. So far, we have seen the original LMs can be easily improved once we find the new LM to apply the above operator.

Suppose the LMs we want to improve are already estimated. In the following, we give three types of additional LMs we can estimate based on the previous topical analysis of annotations and content.

3.4.1 Word-level annotation language model

The annotation LM we give is an ad-hoc improvement. For each document d, let $\tau(d)$ be the set of words in its tags, each having the frequency of being used

for d. We are able to estimate a LM, say L_w^d, from the observations of $\tau(d)$ for all d's. It easily follows that L_w^d can be combined with $\widehat{\theta}_{d_i}$ using Eq. 3.17. We now focus on the simple case of unigram LM, in which each word is assumed to occur depending on the latent probability distribution only regardless of the surrounding words.

3.4.2 Topic-level query language models

Recall in the standard framework, $\widehat{\theta}_q$ is just the empirical distribution of the query $q = \langle w_1, ...w_k \rangle$. This original word-level query model has been shown to underperform [41, 40]. In our approach, we consider each topic discovered as a token in the LM. These tokens will later match the topics discovered for the documents to determine their relevance.

First, we estimate the conditional probability that a query word ω belongs to the topic t, say $P(t|w)$. Over all topics, we have a vector $\mathbf{v_{t|w}} = \langle P(t_1|w), ..., P(t_T|w) \rangle$. After normalization, $\mathbf{v_{t|w}}$ becomes the probability distribution over topics, or rather, a topic-level LM. Second, we merge the multiple topic distributions for each query word into a single topic distribution. Let the desired topic-level query LM be L_t^q. In the unigram case. L_t^q is also a vector of T dimension where each element denotes the probability of a particular topic. Formally, we have:

$$L_t^q = \sum_{w \in q} \delta_w \mathbf{v_{t|w}}. \tag{3.18}$$

where δ_w is the normalized weight for the word ω, and $L_t^q(i)$ denotes the probability of topic i under this model. Note the setting of δ_w allows us to have $\sum_{i \in L_t^q} L_t^q(i) = 1$. Again, using $\oplus_\lambda$, we combine the models at different levels.

3.4.3 Topic-level document language models

Now let us focus on the document LMs. It is easy to see that each document already has a probability distribution over topics discovered from the proposed modeling, denoted by a vector $\mathbf{v_{t|d}} = \langle P(t_1|d), ..., P(t_T|d)\rangle$. Consider this vector as a LM where each topic is a unit. We use $\oplus_\lambda$ to combine this topic-level LM with the original document LM.

Then how to leverage the user information in annotations?. Again, note that the probabilistic model in Sec. 3.2 also outputs the topic distribution for users. Denote the distribution by a T dimensional vector $\mathbf{u_{t|x}} = \langle P(t_1|x), ..., P(t_T|x)\rangle$. Here each element $P(t_i|x)$ denotes the probability of a user x belonging to the topic t_i. Let the document d be tagged by a set of users, say $U(d)$. We combine the multiple LMs of users in $U(d)$. In particular, the desired model L_t^d is generated in addition to and will be combined with the original topic-level LM of document: $\mathbf{v_{t|d}}$. Let the trust or importance of user x be δ_x. The L_t^d is obtained as:

$$L_t^d = \delta_d \mathbf{v_{t|d}} + \sum_{x \in U(d)} \delta_x \mathbf{u_{t|x}}, \tag{3.19}$$

where $\delta_d + \sum_{x \in U(d)} \delta_x = 1$. The δ_d accounts for the emphasis we place on the original discovery of topics for d, and $\forall x \in U(d)$, δ_x determines the trust we place on each user x. Now we have successfully incorporated the topical analysis of documents and users into the original LM-based IR. User domain differences are also considered. How to evaluate user importance is out of the scope of this chapter.

3.5 Experiments on *delicious* Data

A data sample is collected from del.icio.us using the method similar to [80]. We crawled the del.icio.us Web-site starting with a set of popular URL's in Jan. 2006. Then we followed the URL collection of users who have tagged these URL's, arriving at a new set of URL's. By iteratively repeating the above process, we ended up with a collection of 84,961 URL's tagged from May, 1995 to Apr., 2006. There are 9070 users along with 62,007 tag words. Then we crawled the URL's to collect document content. There are 34,530 URL's in the collection which are still valid and have textual content, including 747,935 content words. The activity of users seems to follow a power-law distribution.

3.5.1 Model perplexity

We first perform the training of the proposed model using the algorithm introduced above. For different settings of the desired topic number, we test the perplexity of the trained model on a held-out sample dataset. Over iterations, the perplexity scores always decreases dramatically after the first several iterations and then soon converges to a stable level. We show a plot of perplexities on five different settings of T in Fig. 3.5.1. Here the training set is a 1% random sample of the data available. We are able to see that the larger setting of topic number leads to a lower perplexity score from the start, indicating a better prediction performance. This is because the increased number of topics (before a certain point) reduces the uncertainty in training. For the same reason, the larger setting of topics also leads to a smaller perplexity value in the first several iterations, followed by a sharper drop in perplexity. From the figure, we can see that empirically the algorithm converges within 20 iterations for a relative small sample. For the full dataset, we

repeat the Gibbs sampling for 100 iterations.

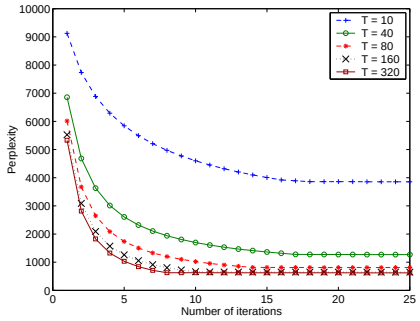


Figure 3.3. The perplexities over the iterations in training for five different settings of topic number. The training set is a 1% random sample of the available data. The perplexity is tested on a held-out sample whose size is proportional to size of the training set.

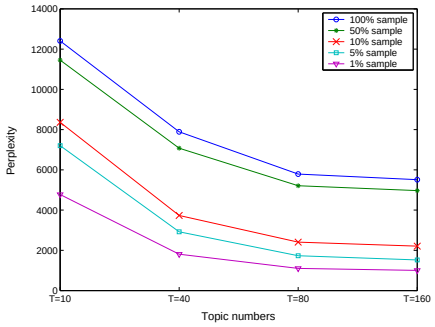


Figure 3.4. The perplexities over different settings of topic numbers, for $T = 10, 40, 80, 160$. Different sample sizes are tested yielding similar curves indicating a minimum optimal topic number of 80 on the collected data. The perplexity is tested on a held-out sample whose size is proportional to size of the training set.

The second set of experiments carried out seeks to determine the best number of topics in the setting. Using the perplexity measure defined in Eq. 3.9 - Eq. 3.11. We perform the experiments by setting different number of topics in training on various sizes of samples from the available data. Generally, the perplexity score first decreases and then remains stable after T is at certain size. We prefer the smallest T that yields a convergence since the greater T requires larger computation. In Fig. 3.5.1, we show the perplexity scores over different T for various sample sizes. It

is clear that the perplexity decreases much slower from after $T = 80$. Accordingly, we choose the desired topic number to be 80 in the following experiments.

3.5.2 Information retrieval quality

Now let us evaluate the IR quality of various language modeling (LM) approaches. The methods we compare are:

- **Word-level LM on content (W-QD)**: Query LM is trained on the original query and the document LM is trained on the original document content.
- **Word-level LM on content and annotations (W-QDA)**: The query LM is trained on the original query and the document LM is trained on both document content and annotations.
- **Word-level LM + LDA on content and annotations (WT-LDA)**: We run LDA on document plus annotations by treating annotations as additional words, without consideration of user differences. The topic-level LM is combined with W-QDA using the parameter λ_1.
- **Word-level LM + Topic-level LM (WT-QDA)**: We run the proposed topic analysis model on the documents and annotations, obtaining topic information of documents and users. Then, the topic-level LM is combined with the word-level LM W-QDA, using the parameter λ_1.
- **Word-level LM + Topic-level LM on document and users (WT-QDAU)**: User domain interests are considered here. First, the word-level LM and topic-level LM and their combination are trained using WT-QDA. Second, the document LM is combined with the mixture of topics on users

who tag the document, using the parameter λ_2. Note here the users are treated the same in the first step.

- **Word-level LM + Topic-level LM on document, and users with differentiation (WT-QDAU$^+$)**: During the training of the WT-QDAU is obtained using the parameter λ_2, the weights on users are set different. For simplicity, we use the number of annotations a user has made for the user-specific trust weights.

In addition, we implement the EM-based retrieval method proposed in a related work [80], which is defined as:

- **EM-based information retrieval (EM-IR)**: As proposed in [80], the URL's and users are first clustered using the EM algorithm. Then the probability of seeing certain words for a URL is estimated. Those probabilities are used for retrieval.

For evaluation, we generate 40 queries with lengths varying from one to five words. The words are chosen from tag and document content. Then for each query, we use the above six approaches for document retrieval. The quality of retrieval is evaluated on the top 10 documents using the Discounted Cumulated Gain (DCG) metric [34]. In particular, two human judges are invited to provide feedback on the composite set of URL's which occur in any of the top 10 retrieval results, yielding the DCG_{10} scores. Judgments are carried out independently based on their experience of the relevance quality. Numerical judgment scores of $0, 1, 2$, and 3 are collected to reflect the judges' opinion on the relevance of documents, which respectively imply the sentiment of *poor*, $fair$, *good*, and $perfect$. In general, the judges represent high agreement on the ranking quality. The average judge scores are used for computing the DCG.

In Table 3.1, we illustrate the DCG_{10} scores for the six approaches: W-QD, EM, W-QDA, WT-LDA, WT-QDA, WT-QDAU, and WT-QDAU$^+$. We can see that both the EM-based IR and the newly proposed approaches outperform the traditional LM-based IR. We read Table 3.1 from several aspects:

First, we take a look at the improvement according to the use of tags. The EM-based IR proposed in related work [80] increased the DCG scores by 11.5% over traditional LM-based IR (W-QD); The method that uses annotations as additional words improved the DCG by 18.3% (W-QDA over W-QD), which demonstrates that the use of annotation can dramatically improve IR quality.

Second, we examine the improvement based on topical analysis on both document content and annotations. The basic use of the topic information (WT-LDA) further improves the use of annotations (W-QDA) by 2.7%. The topic analysis based on the new generative model, compared with WT-LDA, achieves a gain of 1.3%. It is worthwhile to mention that the LDA-based topic analysis improves a very recent related work [80] (EM-IR) by 9.1%.

Third, we test the improvement by incorporating tagger interests. As illustrated in Table 3.1, WT-QDAU outperforms pure topic-based IR by 1.1%, showing the importance of user interests.

Fourth, we show the improvement by considering the differences of users while incorporating user interests. The WT-QDAU$^+$ adds another 1.3% in DCG over WT-QDAU. This shows that due to the different user expertise, the quality of tags can be different and thus should be taken into consideration.

Overall, the top performance of our proposed model (WT-QDAU$^+$) improved the traditional LM-based IR model by 26%, compared with the the 11.5% improvements by the EM-based approach in [80].

W-QD	**EM-IR**	**W-QDA**	**WT-LDA**
7.6192	8.4945	9.0167	9.2602
WT-QDA	**WT-QDAU**	**WT-QDAU$^+$**	
9.3820	9.4938	9.6167	

Table 3.1. The DCG_{10} scores of six compared approaches: W-QD, EM-IR, W-QDA, WT-LDA, WT-QDA, WT-QDAU, WT-QDAU$^+$.

3.6 Summary

This chapter presents a framework that combines the modeling information retrieval on the documents associated with social annotations. A new probabilistic generative model is proposed for the generation of document content as well as the associated social annotations. A new way for discovering user domains is presented based on social annotations; Several ways are studied for combining language models from tags with those from the documents; An exploration is carried out for evaluating user expertise based on activity intensities; Experimental evaluation on real-world datasets demonstrates effectiveness of the proposed model and the improvements over traditional IR approach based on language modeling.

Chapter 4

Topic Dynamics and Social Actions

4.1 Topic Dynamics in Social Documents

While there are a rich set of choices regarding temporal topic discovery in sets of temporally related documents [51, 49], our concern is when and where these topics evolve and how the topics relate, if any, dependencies with each other. In Fig. 4.1, for example, we illustrate the probability of appearance in documents in CiteSeer of four research topics discovered using Latent Dirichlet Allocation [3], which is similar to previous topic trend discovery [62].

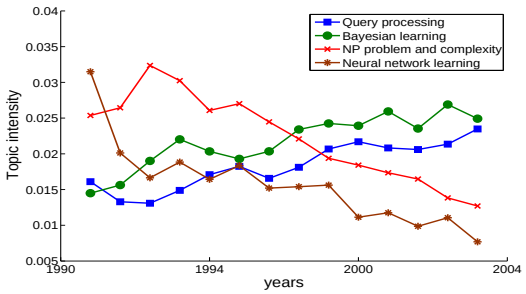


Figure 4.1. Probability of four research topics over 14 years in CiteSeer.

Some topics in Fig. 4.1 have been growing dramatically in popularity while other topics seem to be less popular, at least according to the CiteSeer database. A

more interesting question is whether a newly emergent topic is truly new or rather a variation of an old topic? We address this by revealing the dependencies [1] among the discovered topics. With the temporal dependencies among topics available, one can survey a research area from a genealogical graph of related topics instead of perusing the citation graphs.

In order to interpret and understand the changes of topic dynamics in documents, we resort to discovering the *social reasons* of why a topic evolves and relates dependencies with others. We hypothesize that one topic evolves into another topic when the corresponding social actors interact with other actors with different topics in the latent social network. Consider an actor a_u associating a topic t_i at time k. For some reason, this actor meets and establishes a social tie with actor a_v who is mostly associated with a new topic t_j and they start to work on the new topic with a higher probability. At a later time $2k$, we observe that a_u is more likely to be concerned with t_j rather than the previous t_i. In return, t_j has received a higher popularity than t_i. When such a switch of topics in actors is statistically significant, we will observe an aggregate transition tendency from t_i to t_j in the topic dynamics, yielding the marginal probabilities of t_i and t_j moving towards different directions over time, as illustrated in Fig. 4.1. Such a trend is defined as transition between topics. The dependency of t_i on t_j is measured by the probability of transition from t_j to t_i. Abstracted from the above example, our goal seeks to estimate the dependencies between topics using social interactions.

Here we attempt to bridge the dynamics of topics in documents with the latent social network [87]. In particular, we hypothesize the changes of topics in docu-

[1] More precisely, here we consider topics as observed labels of documents that are generated from a Markov process over time. In particular, we assume these topic labels correspond to instances from the state space in a Markov process. A "dependency" that we seek to discover is in fact the transition probability from one state to another.

ments in a social network as a Markov chain process which is parameterized by estimating the social interactions among individuals conditioned on topics. The primary assumption is that topics in *social documents* evolve due to the development of the latent social network. Our contributions are: (1) a model of the topic dynamics in *social documents* which connect the temporal topic dependency with the latent social interactions; (2) a novel method to estimate the Markov transition matrix of topics based on social interactions of different order; (3) the use of the properties of finite state Markov process as the basis for discovering hierarchical clustering of topics, where each cluster is a Markov metastable state; (4) a new topic-dependent metric for ranking social actors based on their social impact. We test this metric by applying it to CiteSeer authors.

4.2 Topic Transition & Social Interactions

We give a formal definition of the problem we address here. Denote the *social document* stream as a matrix $\mathcal{DW} \in \mathbb{R}^{D \times W}$, where D is the number of documents and W the number of words. Define the matrix $\mathcal{DA} \in \mathbb{R}^{D \times A} = \{\lambda_{i,j}\}^{D \times A}$ denoting the creators of these documents, where A is the number of social actors and $\lambda_{i,j} = \mathbf{1}(d_i, a_j)$, an indicator function of whether document d_i is composed by actor a_j. Note that one document may have several actors. (For our experiments actors will be denoted as authors.)

Using the summarization tools (LDA [3]) we transform $\mathcal{DW}$ into $\mathcal{DT} \in \mathbb{R}^{D \times T}$, where T is the number of pre-specified topics. We assume that $\mathcal{DT}$ is normalized by row such that each document is a distribution over topics.

Using the matrix $\mathcal{DA}$, a collaboration matrix $\mathcal{A}$ is obtained by setting $\{\alpha_{i,j}\}^{A \times A} = \mathcal{A} = (\mathcal{DA})^t DA$, where $\alpha_{i,j}$ denotes the number of collaborations between so-

cial actors a_i and a_j if $i \neq j$ and the number of composed documents by a_i if $i = j$. Let the author set be Λ. Using matrix $\mathcal{DT}$ and $\mathcal{DA}$, we obtain a set $\mathcal{Q} = \{\langle \mathbf{a}, \mathbf{t} \rangle | \mathbf{a} \subseteq \mathbf{\Lambda}, \mathbf{t} \in \mathbb{R}^{\mathbf{1} \times \mathbf{T}}\}$, where $\mathbf{a}$ is the set of authors on a document and $\mathbf{t}$ is the distribution over topic specifying this document. Here each element q_i in $\mathcal{Q}$ denotes an observation of a document.

Now the problem becomes, given a set $\mathcal{Q} = \{\langle \mathbf{a}, \mathbf{t} \rangle | \mathbf{a} \subseteq \mathbf{\Lambda}, \mathbf{t} \in \mathbb{R}^{\mathbf{1} \times \mathbf{T}}\}$ and $\mathcal{A} \in \mathbb{R}^{A \times A}$ that can be calculated from $\mathcal{Q}$, find a Markov transition matrix $\Gamma \in \mathbb{R}^{T \times T}$ that captures the dependencies among the discovered topics, i.e. determine a function Ψ such that $\Gamma = \Psi(\mathcal{Q}, \mathcal{A})$.

In our setting, where topics are those discovered from *social documents*, we propose a measurement method that accentuates the social interactions in the latent SN in order to estimate the topic transitions. The function Ψ determines the measurement of pair-wise dependencies between topics.

Namely, we limit our search for Ψ to consider only the social interactions mediating the evolution of topics. The assumption is supported by the intuition that topics created by close *social actors* in a SN sense[76] represent greater dependencies than those created randomly. For example, a topic t_a is more likely to be dependent on t_b if the social actors found in t_a are tightly connected to those social actors found in t_b. The idea here is similar to but different from that of *collaborative filtering* [50] in that now heterogeneous social ties are taken into consideration.

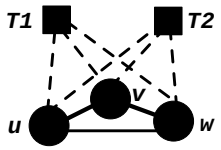


Figure 4.2. Different dependency networks among two sets of variables: *topics (squares)* and *social actors (circles)*.

The estimation of Markov topic transition matrix Γ breaks down to a set of

estimation tasks each in the form of $P(t_i|t_j)$, which denotes the probability that topic t_j transits to t_i in the Markov chain process.

In order to estimate $P(t_i|t_j)$ using social ties in a SN properly, we first set up the probability independence among two sets of variables: topics and social actors. Let Fig. 4.2 illustrate our assumptions of the dependencies of variables. The topics are assumed to be of no direct dependency between each other. The social actors are assumed to be pair-wise dependent. For two social actors with no relationships, we can consider their dependency as zero. In Fig. 4.2, we show that three actors u,v and w are socially connected (solid lines). The two topics associated with them, T_1 and T_2 respectively, are linked (dashed lines) to all actors.

Following the above, consider the joint distribution $P(t_i, t_j)$ resulting from the interactions in the latent SN. In particular, we consider the social interaction bounded by order two, i.e. $P(t_i, t_j)$ is constrained by single self and pairs of social interactions only, respectively denoted by $P(t_i, t_j)^{(1)}$ and $P(t_i, t_j)^{(2)}$. This can be denoted by:

$$P(t_i, t_j) = \gamma P(t_i, t_j)^{(1)} + (1 - \gamma) P(t_i, t_j)^{(2)} \tag{4.1}$$

$$= \gamma \sum_{1 \le u \le A} P(t_i, a_u, t_j) + (1 - \gamma) \sum_{1 \le u,v \le A} P(t_i, a_u, a_v, t_j) \tag{4.2}$$

where a_u and a_v are social actors in the underlying SN. γ is a smoothing parameter that weighs the importance of 1st-order social interactions. Eq. 4.2 assumes independence when estimating $P(t_i, a_u, t_j)$ and $P(t_i, a_u, a_v, t_j)$.

Note the assumption above regarding the order of social interaction can be relaxed to deal with higher order. We leave it to the readers to generalize Eq. 4.2 in high order case.

4.2.1 Multiple orders of social interactions

Multiple types of social ties can be considered as a basis for determining the estimation of the topic transition probability. In this subsection, we provide a solution to the estimation problem based on social interactions, one typical social tie in a SN, with different orders. Denote the measurements based on 1st-order and 2nd-order social interactions respectively by $P(t_i, t_j)^{(1)}$ and $P(t_i, t_j)^{(2)}$. We focus on deriving $P(t_i, a_u, t_j)$ and $P(t_i, a_u, a_v, t_j)$ estimation formulas from our social interaction considerations.

First, we consider the estimation of $P(t_i, a_u, t_j)$ as a 1st-order social interaction. We illustrate the 1st-order probability dependence between topics and social actors in Fig. 4.3. The social actor u is present in both topics T_1 and T_2.

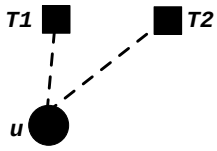


Figure 4.3. 1st-order probability dependence between topics and a social actor.

We can estimate $P(t_i, a_u, t_j)$ by Eq. 4.3:

$$P(t_i, a_u, t_j) = P(t_i|a_u, t_j)P(a_u|t_j)P(t_j) \tag{4.3}$$

We derive Eq. 4.3 using the chain rule for a joint probability. Based on the assumption of the conditional independence between t_i and t_j on a_u as illustrated in Fig. 4.3, we obtain the joint probability $P(t_i, a_u, t_j)$ as a chain of probabilities:

$$P(t_i, a_u, t_j) = P(t_i|a_u)P(a_u|t_j)P(t_j) \tag{4.4}$$

The intuition behind the 1st-order social interaction is that a new topic may be initiated by the same actor who is present in an older topic but without collaboration with any other social actors.

Second, we discuss the estimation of $P(t_i, a_u, a_v, t_j)$ considering the 2nd-order social interaction (a dyad in SN notation [76]). The 2nd-order probability dependency between topics and social actors is presented in Fig. 4.4. Here we introduce the pair-wise interaction in the latent SN as the motivation for the evolution of topics.

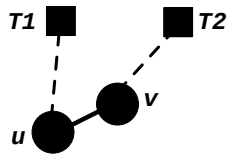


Figure 4.4. 2nd-order probability dependence between topics and social actors.

Again, consider the joint distribution $P(t_i, t_j)$ as being constrained by the relationship between two social actors a_u and a_v to the 2nd-order, as measured by $P(t_i, a_u, a_v, t_j)$. The constraint is captured in Eq. 4.1 by $P(t_i, t_j)^{(2)}$. These 2nd-order SN interaction constraints can be seen as the sum of the joint probabilities $P(t_i, a_u, a_v, t_j)$, which is represented as:

$$P(t_i, t_j)^{(2)} = \sum_{1 \leq u, v \leq A, u \neq v} P(t_i, a_u, a_v, t_j). \tag{4.5}$$

We factorize the joint probability $P(t_i, a_u, a_v, t_j)$ in Eq. 4.6 to Eq. 4.8 using chain rule:

$$P(t_i, a_u, a_v, t_j) = P(t_i | a_u, a_v, t_j) P(a_u, a_v, t_j) \tag{4.6}$$

$$= P(t_i | a_u, a_v, t_j) P(a_u, a_v | t_j) P(t_j) \tag{4.7}$$

$$= P(t_i|a_u, a_v, t_j)P(a_u|a_v, t_j)P(a_v|t_j)P(t_j) \tag{4.8}$$

Based on the independence assumption in Fig. 4.4, we arrive at a new form of $P(t_i, a_u, a_v, t_j)$ as:

$$P(t_i, a_u, a_v, t_j) = P(t_i|a_u)P(a_u|a_v)P(a_v|t_j)P(t_j) \tag{4.9}$$

where $P(a_u|a_v)$ can be seen as the conditional probability that social actor a_u interacts with another social actors a_v.

Note that the idea of relating the evolution of topics in SNs with the various orders of social interactions naturally coincides with the assumption that "collaborations bring about new topics".

The assumptions we made about the independence networks in 1st- and 2nd-order social interactions help the derivations of Eq. 4.4 and Eq. 4.9. In traditional topic discovery methods where social factors are not considered (e.g. LDA), topics are assumed to be unconditionally independent from each other. Thus we see the assumptions of our approach as weaker and relaxed in conditions.

4.2.2 Markov transition

With the derivation for the joint probability of two topics with 1st- and 2nd-order social interactions, the estimation for Markov transition matrix, Γ, becomes straightforward. In particular, we define $\Gamma \in \mathbb{R}^{T\times T}$, where each element $\Gamma_{i,j}$ quantifies the transition probability from t_j to t_i. Then, $\Gamma_{i,j}$ is the conditional probability $P(t_i|t_j)$:

$$\Gamma_{i,j} = P(t_i|t_j), \quad \text{where } \Gamma \in \mathbb{R}^{T\times T}. \tag{4.10}$$

where the transition probably is a directional estimate such that $\Gamma_{i,j}$ does not necessarily equal to $\Gamma_{j,i}$. We assume Γ will be normalized by row such that the row elements sum to one.

Next we revisit the estimation of the joint probability $P(t_i, t_j)$ in Eq. 4.1 for the estimation of $P(t_i|t_j)$. Using Bayes rule, we have $P(t_i|t_j) = \frac{P(t_i,t_j)}{P(t_j)}$. Substituting this into Eq. 4.1, we obtain $P(t_i|t_j) = \gamma P(t_i, t_j)^{(1)} + (1-\gamma)\frac{P(t_i,t_j)^{(2)}}{P(t_j)}$. According to Eq. 4.3 and Eq. 4.8, we rewrite Eq. 4.2 as:

$$P(t_i|t_j) = \frac{\gamma \sum_u P(t_i, a_u, t_j) + (1-\gamma) \sum_{u,v} P(t_i, a_u, a_v, t_j)}{P(t_j)} \quad (4.11)$$

$$= \gamma \sum_{1 \le u \le A} P(t_i|a_u)P(a_u|t_j) + (1-\gamma) \sum_{1 \le u,v \le A, u \ne v} P(t_i|a_u)P(a_u|a_v)P(a_v|t_j) \quad (4.12)$$

So far we have given analytical formulas for $P(t_i|t_j)$ which are required for deriving the Markov transition matrix Γ. In practice, we estimate the required $P(t_i|a_u)$, $P(a_u|t_i)$ and $P(a_u|a_v)$ using the Maximum Likelihood Estimation (MLE). For details, refer to [87].

4.3 Markov Metastable State Discovery

Now we have topic and topic-topic dependencies respectively estimated as the system states and the stochastic transition probability of a Markov chain. We will explore other topic discovery using well established methods in Markov analysis [63]. This section describes the discovery of metastable states [11] in a Markov chain as an approach to identifying hierarchical clustering of topics.

Consider a Markov chain with its transition matrix P, state set S with the marginal distribution of S as π. Let $A \subseteq S$, $B \subseteq S$ be two subsets of S. Then the

transition probability from B to A with respect to π is defined as the conditional probability from B to A:

$$\omega_\pi(A|B) = \frac{\sum_{a\in A, b\in B} \pi_a p_{a|b}}{\sum_{b\in B} \pi_b} \tag{4.13}$$

where a,b are dummy variables denoting the states in S.

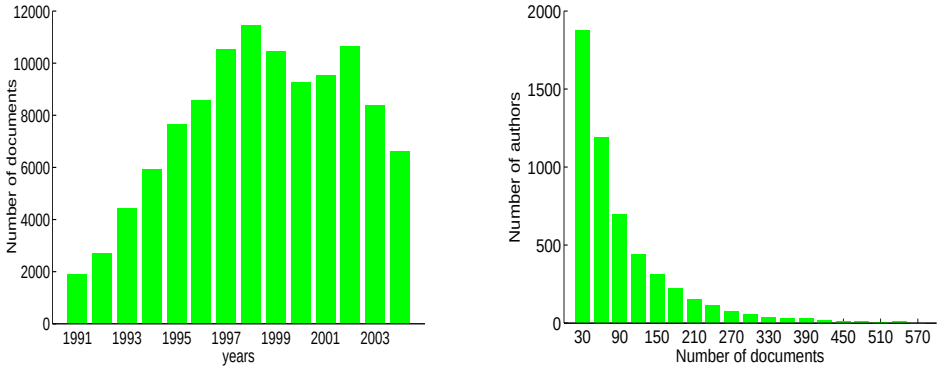
Let A_1,..., A_K be disjoint K subsets of S. We define a new $K \times K$ transition matrix $W = \{\omega_\pi(A_i|A_j)\}_{ij}$ as described above. Thus we arrive at another Markov chain with dimensionality reduced to K in which each state now is an aggregate of the unit states from the previous state space.

Markov chains are called *nearly uncoupled* if its state space can be decomposed into several disjoint subsets **A** such that $\omega_\pi(A_i|A_j) \approx 1$ for $i = j$ and $\omega_\pi(A_i|A_j) \approx 0$ for $i \neq j$. Each aggregate in a *nearly uncoupled* Markov chain M is called a *metastable state* of M. In our setting, a metastable state in Γ is a cluster of topics. Recursively discovering the metastable states[11], we may obtain a hierarchical clustering of topics that capture their taxonomy. Identification of the metastable states in a Markov chain has been studied extensively [14, 11]. In numerical analysis, the identification can be viewed as a process which seeks the matrix permutation such that the transition matrix is as block diagonal as possible; a method [14] we also use.

4.4 Experiments on CiteSeer

For experiments, we use data from CiteSeer [20], a popular online search engine and digital library which currently has a collection of over 739,135 academic documents in Computer Sciences, most of which were obtained by web crawling and

the others by author submissions. The documents have 418,809 distinct authors after name disambiguation. Each document is tagged with a time-stamp giving the parsed time of the first crawled date.



(a) Number of documents versus year acquired

(b) Authors associated with number of documents.

Figure 4.5. Statistics of the sample CiteSeer.

We associate each document with the list of disambiguated authors [27]. Then we construct a co-authorship graph where two nodes share an edge if they ever co-authored a document. Next we perform breadth-first-search search on the co-authorship graph from several predefined well known author seeds until the graph is completely connected or there are no new nodes. For seeds selection, we choose two researchers with a large number of publications in CiteSeer, Michael Jordan and Jiawei Han, from statistical learning and data mining and database respectively. The constructed subgraph of authors is further pruned by eliminating the authors with less than 50 publications in CiteSeer over the last fourteen years. We end up with a sampling of CiteSeer containing 3,974 authors and 108,676 documents spanning from 1991 to 2004. The number of documents acquired w.r.t years is illustrated in Fig. 4.5(a). We observe that the number of documents written by individual authors follows a *power law* distribution (Lotka's law) [44].

Topic #	manual namings	Topic #	manual namings
0	real-time system, performance	25	network traffic congestion control, protocols
1	rule mining, database	26	document retrieval, search engine
2	database query processing	27	language, automation machine
3	communication, channel capacity	28	mathematical derivation, proof
4	information theory	29	image segmentation, computer
5	programming language, compiler	30	multimedia, video streaming
6	scheduling, queueing	31	statistical learning theory
7	software engineering, system development	32	knowledge representation, learning
8	svm, learning, classification	33	protein sequence, dna structure
9	signal processing	34	robotics
10	ai, planning	35	system kernel, unix
11	matrix analysis, factorization	36	security, cryptography
12	dynamic flow control	37	mobile network, wireless protocols
13	dimension reduction, manifold	38	natural language, linguistic
14	decision tree, learning	39	np problem, complexity
15	numerical optimization	40	network package routing
16	mobile network, energy	41	user agents, interface
17	affiliation and venues	42	geometry, spatial objects
18	object oriented design	43	parallel processing
19	digital library services, web	44	distributed computing, network infrastructure
20	os cache strategy design	45	system architecture
21	circuit design	46	neural network, learning
22	concurrent control, distributed system	47	graph algorithms, coloring
23	game and marketing	48	linear programming
24	algorithm complexity	49	bayesian method, learning

Table 4.1. Topics discovered with manual labels.

4.4.1 Discovered topics

We train a Latent Dirichlet Allocation (LDA) model over our entire sample collection of CiteSeer by setting the topic number as $T = 50$, resulting in 50 discovered topics illustrated in Table 4.4. The setting of desired topic number is small because we only work on a small subset of authors in CiteSeer (3,974 authors out of 418,809). Due to the limited space, we cannot present all the automatically extracted top words for all topics. Instead, we manually tag all the topics with labels using ranked keywords in the word list for each topic.

For a more detailed description of some topics, in Table 4.4, we give a sample of six topics from Table 4.4 and their top words. Here the last row is manually labeled to summarize the topics. We are able to observe that LDA easily discovers the topics from a variety of areas [2].

[2]Note that Topic 17 denotes the affiliation and venues in which the keywords are *university,*

After the models are trained, we re-estimate each document with the LDA model to obtain the mixture of topics for each document. We further normalize the weights of the mixture components. It should be noted that this permits us to track the topic over time using some recently proposed online methods(e.g. [51]).

4.4.2 Topic trends

We visualize the four topic dynamics w.r.t. time in Fig. 4.6. Given a year, the strength of a topic is calculated as the normalized sum of all the probabilities of this topic inferred for all documents in this year. The topics trend is an indicator of the trend of interests in *social documents* and in our setting, the research interest trends.

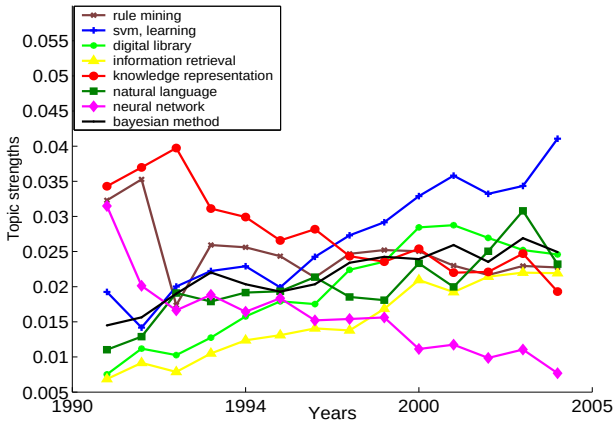
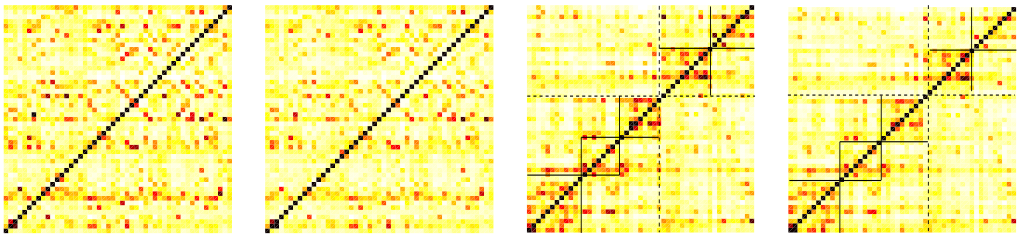


Figure 4.6. Topic probability of eight topics over 14 years in CiteSeer.

The eight topics we choose to plot are (1) query processing (Topic 02); (2) svm learning (Topic 08); (3) digital library (Topic 019); (4) information retrieval (Topic 026); (5) knowledge representation (Topic 032); (6) natural language processing (Topic 038); (7) neural network learning (Topic 046), and (8) Bayesian learning

department, email, conference, proceedings, etc which are also considered as topics since there was no deliberate removal of such information from the title/abstracts in CiteSeer.

(Topic 050) (similar results are in [62]). This raises the question of *where do the researchers in a declining trend go (ex. neural networks)? Do they switch to new topics, and which topics?* Our next goal is to automatically extract the dependencies among these discovered topics.



(a) Transition under 1st-order interaction (b) Transition under 2nd-order interaction (c) Transition under 1st-order interaction after block diagonalization (d) Transition under 2nd-order interaction after block diagonalization

Figure 4.7. Markov transition matrices before and after block diagonalization

4.4.3 Markov topic transition

In order to explore the temporal dependencies among a group of discovered topics, we identify the Markov topic transition matrix via maximum likelihood estimation of the 1st- and 2nd-order constraints brought about by the hidden social interactions of authors (interactions of single social actors, or collaboration between social actor pairs).

The Markov transition matrices Γ are shown in Fig. 4.7(a) and Fig. 4.7(b) to highlight the extraction of metastable topic states. The values of matrix entities are scaled with the color intensity with the darker color denoting large value. Fig. 4.7(a) and Fig. 4.7(b) visualize the Γ with 1st-order and 2nd-order social relationship, before block diagonalization. From Fig. 4.7(a) and Fig. 4.7(b), we observe that Γ is a sparse matrix, with large values in diagonal elements. The sparseness shows that these topics are separate though some transitions among

them exist. The large diagonal values indicates that the discovered topics in our case are relatively stable with mostly transitions to themselves. Authors in our CiteSeer sample prefer to remain in their own topics rather than switching between topics.

While the separateness among topics is for future investigation, we now take a closer look at the diagonal elements. Diagonal elements in Γ indicate the probability that an author (and author pair collaboration as well) will continue to work on the same topics over time. This *self-transition probability* shown in Fig. 4.8 allows us to rank the topics according to the authors reluctance to change topics.

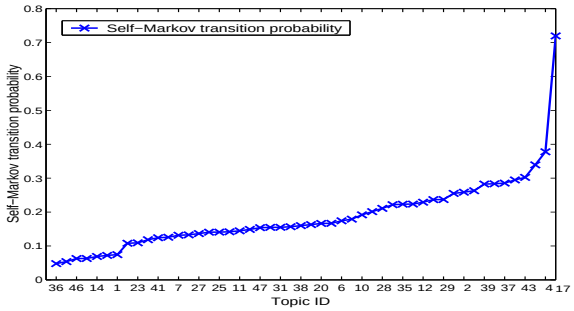


Figure 4.8. The self-transition probability ranking of topics. Topics with high probability are more stable.

Note that Topic 17 (affiliation and venue info.) is that with largest self-transition probability. The rational is obvious since most authors tend to continue including their affiliation/venue information which was part of the meta data used. In addition, we can see that generally the topics with heavy methodology requirements (e.g. np problem, linear system) and/or popular topics (e.g. mobile computing, network) are more likely to remain stable. By contrast, topics closely related to applications are more likely to have higher transition probabilities than other topics (e.g. data mining in database, security) all things being equal.

Second, in order to investigate the sparseness in matrix Γ, we perform metastable

#	Topic IDs										
mT_1	1	2	8	10	18	19	23	26	32	38	41
mT_2	0	5	7	20	21	22	27	28	35	43	45
mT_3	6	25	30	36	37	40	44				
mT_4	13	14	15	17	24	31	33	39	42	47	48
mT_5	3	4	9	11	12	16	29	34	46	49	
mT_1	data management, data mining										
mT_2	system, programming language, and architecture										
mT_3	network and communication										
mT_4	numerical analysis, machine learning										
mT_5	statistical methods and applications										

Table 4.2. Discovery of mTopics via block diagonal Markov transition matrix.

state recognition (introduced in $ 4.3), viewing Γ as the adjacency matrix of the Markov transition graph. In particular, we permute Γ in such a way that Γ is approximated by a block diagonal matrix. The resultant $\hat{\Gamma}$ is illustrated in Fig 4.7(c) and Fig. 4.7(d), on 1st-order and 2nd-order consideration of social relationship respectively.

The metastable states have in effect reduced the original Markov transition process to a new Markov process with fewer states and each diagonal block can be seen as a metastable state [11] which is a cluster of topics with tight intra-transition edges.

From Fig 4.7(c) and Fig. 4.7(d), we are able to initially break the two $\hat{\Gamma}$ into two major blocks, as noted by the dashed lines. Recursively, we can arrive at five smaller blocks, illustrated by solid lines, with each block as a metastable topic (or mTopic). Even though there exists a transition between topics, the transitions are more likely to occur within a metastable topic rather than between them. Table 4.2 gives the list of mTopics and the corresponding topics.

Comparing Table 4.2 with Table 4.4, we observe that the topic descendants discovered readily capture natural intuitions of the relationships among topics. Specifically, mTopics mT_1 includes topics on data management and data mining; mT_2 includes programming language, system and architecture; mT_3 covers network

and communication; mT_4 covers machine learning and numerical analysis; mT_5 are mainly statistical methods.

4.4.4 Transition within metastable topics

With metastable topics (mTopic) discovered according to the approach introduced in § 4.3, we are able to compute the accumulated transition probability among mTopics. Fig. 4.9 illustrates the uncoupled Markov transition graph among five mTopics we have discovered from the original stochastic matrix. Transitions with probability lower than 0.16 are hidden from the graph to clarify the major transition among the five mTopics. Such transition probabilities among metastable topics are very useful information for understanding the major trends of topics and their dependencies in social document collections.

Comparing Fig. 4.9 with the descriptions of each mTopic in Table 4.2, we can outline the major dependencies between mTopics. Out data indicates that mT_4 (numerical analysis) has been essential in these mTopics. And there is a transition to mT_5 (statistical methods) and which is tightly coupled with research in mT_1 (data management and data mining). Results also imply that researchers in mT_3 (networks) will be concerned with mT_2 (systems) and that data management research is coupled with systems issues due the high mutual transition probability between mT_1 and mT_2.

Next we look at the transitions within these metastable topics. Now that we know the topics within a metastable topic (mTopic) are very less likely to jump across mTopics, questions may be asked about how tightly the topics in the same metastable state are aggregated. We present the stochastic matrices of mT_1 and mT_4 in Fig. 4.10(a) and Fig. 4.10(b).

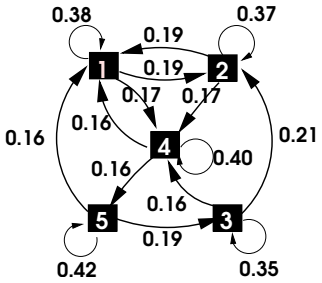
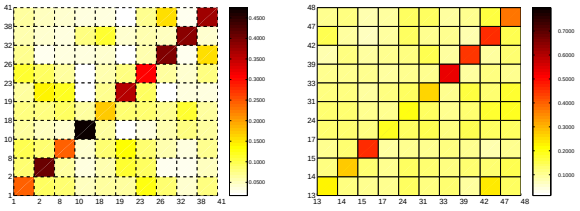


Figure 4.9. The Markov transition graph among mTopics. Transitions with probability lower than 0.16 are not shown.



(a) mTopic: mT_1 (b) mTopic: mT_4

Figure 4.10. The Markov transition structure in metastable topics

We observe that diagonal elements show the existence of high self-transition probabilities and that both matrices are almost symmetric, meaning the pairwise transition between topics in the same mTopic are largely balanced.

4.4.5 Who powers the topic transition

If one accepts the above interpretation of Markov transitions among the topics discovered in social document collections, a natural question to ask is what author or authors cause such a transition between topics, evaluating their roles as prominent social actors.

In particular, in the CiteSeer data setting, we seek to provide rank of authors based on their impact on the transition from one topic to another. We give a new metric $\delta(a_u)$ for the author impact ratio of a_u as measuring the difference between the obtained $P(t_i|t_j)$'s, with and without a_u.

Table 4.3. Top ranked authors according to their impact on three topic transitions.

T2→T1	**T49→T26**	**T1→T33**
Jiawei Han	W. Bruce Croft	Mark Gerstein
Jennifer Widom	David Madigan	Heikki Mannila
Timos Sellis	Norbert Fuhr	Mohammed Zaki
Dimitris Papadias	Andrew Mccallum	Limsoon Wong
Hans-peter Kriegel	James Allan	George Karypis
H. V. Jagadish	Thomas Hofmann	Jiawei Han
Jeffrey Naughton	John Lafferty	Susan Davidson
Divesh Srivastava	Hermann Ney	Dennis Shasha
Amr El Abbadi	Michael I. Jordan	Serge Abiteboul
Philip S. Yu	Ronald Rosenfeld	Jignesh M. Patel

Formally, consider how the transition probabilities change if an author a_u does not exist. Denote the estimation of $P(t_i|t_j)$ without a_u as $P(t_i|t_j)_{-a_u}$. One can then measure the importance of a_u w.r.t. topic $t_j \rightarrow t_i$ as $\delta(a_u, t_j \rightarrow t_i)$:

$$\delta(a_u, t_j \rightarrow t_i) = P(t_i|t_j) - P(t_i|t_j)_{-a_u}. \tag{4.14}$$

The new author ranking differs from previous ranking by citation counting, currently done in CiteSeer, Google Scholar, and ISI, by now incorporating social interactions while ranking social actors. In addition, the new ranking is dependent on the specified topic pairs thus quantifying the impact of social actors w.r.t. certain field(s).

Next, we choose all pairs of topics from the 50 discovered topics in our data and test our hypothesis. This ranking of social actors captures common knowledge of the importance of these social actors w.r.t. to different fields. Due space limitations, we select three topic transition instances ($T2 \rightarrow T1$, $T49 \rightarrow T26$, and $T1 \rightarrow T33$) and present the corresponding top ten ranked CiteSeer authors in Table 4.3.

Topic 00	Topic 01	Topic 02	Topic 03	Topic 05	Topic 07
real	data	queries	channel	program	software
time	database	data	coding	code	systems
system	mining	join	rate	analysis	development
simulation	spatial	patten	performance	java	tools
fault	relational	matching	bit	compiler	process
tolerance	query	clusters	capacity	data	engineering
embedded	temporal	analysis	transmission	language	components
events	large	algorithms	fading	programming	application
timing	rules	hierarchical	receiver	source	design
synchronization	association	large	interference	execution	component
execution	information	incremental	decoding	fortran	framework
scheduling	management	space	frequency	run	modeling
dynamic	discovery	aggregation	low	machine	specification
performance	support	evaluation	cdma	automatic	case
response	sql	views	distortion	compilation	study
distributed	frequent	cost	signal	optimization	reuse
task	patterns	efficient	systems	runtime	management
events	dbms	compression	block	dynamic	evaluation
clock	integration	approximate	modulation	static	object
period	schema	text	time	loops	oriented
real-time system performance	**association rule mining**	**query processing**	**communication capacity**	**program lang. compiler**	**software engr. system**

Table 4.4. Six topics discovered by LDA on CiteSeer subset. Each topic is described using 20 words with highest probabilities.

We observe that many commonly believed influential researchers in data management to data mining ($T2 \rightarrow T1$), Bayesian learning to search engine ($T49 \rightarrow T26$), and rule mining to bioinformatics ($T1 \rightarrow T33$) are well ranked [3].

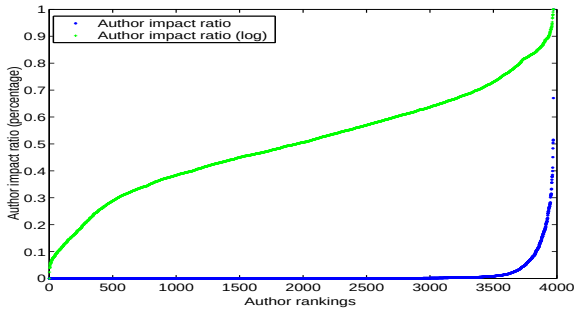


Figure 4.11. Ranking of authors w.r.t. their impact on the transition from Topic 02 to Topic 01.

Finally we give the distribution of impact over all authors in Fig. 4.11 for the transition of topic 02 to 01. The impact distribution is a power law, indicating that only a few social actors have large effects over a certain topic transitions.

[3] some researchers are not on the list because of no (indirect) collaboration with our seed authors and/or having the number of papers in CiteSeer necessary for our cut.

4.5 Summary

In this chapter, we model the topic dynamics in a social document corpus as a Markov chain and discover the probabilistic dependency between topics from the latent social interactions. We propose a novel method to estimate the Markov transition matrix of topics using social interactions of different orders. With the properties of Markov process with finite states, we apply the application of Markov metastable states as an approach for discovering the hierarchical clustering of topics and new topics. In addition, we give an experimental illustration of our methods using Markov transitions of topics to rank social actors by their impact on the CiteSeer database. An initial evaluation of our methodology on authors as social actors presents other methods for author impact besides citation counting. Future work could refine our estimation of topic dependency, use larger data sets, derive social ranking of actors independent of topics, explore better estimation methods and generate new communities of actors.

Chapter 5

Learning Social Actions to Rank Actors

5.1 Ranking Social Actors

An important topic in social network analysis (SNA) is the evaluation of roles of social actors [76], especially for ranking them by social impact. Applications include customer evaluation [15], viral marketing [60], and reviewer recommendations [78]. Online user communities generate a rich set of social relationships among users. For example, the authors in CiteSeer [20] are related to each other by their collaborations, citations, and document content. Traditional approaches for ranking networked entities seek to measure the centrality of vertices based on the network topology [5, 38]. However, in these cases, the network is constructed based on a single semantic type (e.g., hyper-links in PageRank), and the proposed methods do not address the important issue of dealing with the heterogeneous relationships among the actors, such as citations or collaborations. Recent work addresses networks with edges of multiple semantic types [54].

Recently, learning-based methods have been proposed for the construction of networks [1, 71]. The learning-based framework models the network flow in the network as a Markov transition matrix and optimizes an objective function (or loss function) defined on this matrix, usually subject to several constraints. The computed weighted network is then used by topology-based ranking methods (such as PageRank) to rank the vertices. The key to a successful learning-based approach is the choice of the objective functions and the extraction of the right constraints from the implicit user preferences that are specifically tailored to the domain in question.

This chapter introduces a new numerical optimization framework for learning the network flow of a social network [88]. The proposed approach defines two new objective functions, namely, the *constrained variation* and *Gini coefficient* while previous work was based on entropy methods [71, 1] and Laplacian graphs [86]. The *constrained variation* method builds on the standard PageRank transition matrix; The *Gini coefficient* method is similar to entropy maximization but has an efficient solution as a quadratic programming problem. Newly introduced *edge-wise* and *aggregate* constraints are derived from the implicit preferences inferred from multiple sources, including the content of the shared documents by the authors and the various social actions on them including collaborations, citations, and action temporality such as time delays in citations. The proposed approach makes it easy to extract preference constraints from social networks and is thus more practical for ranking actors in social networks. We also show the formulated optimization problems can be solved using standard quadratic programming (QP) methods. Our experimental evaluation on real-world large scale dataset (the CiteSeer collection) yields a new approach for topic-dependent ranking of authors with significant improvements.

5.2 Network Flow Modeling Framework

Recall from the related work that in the PageRank [5] the ranking scores are uniquely determined by the transition matrix P on the network. Given this, recent work has focused on the design of the Markov transition matrix [1, 71, 78].

Transition matrix learning [71], or network flow modeling, formulates the problem as a constrained entropy maximization of P [1]. The entropy is maximized for generalization and the matrix P is required to satisfy several constraints for being a Markov process as well as network flow, including $\forall v \in V, \quad \sum_j p_{v,j} = 1$ and $\sum_i p_{i,v} = \sum_j p_{v,j}$. Following the same entropy framework, constraints [1] were introduced to capture the *vertex-wise* preferences among vertices. Denote the *vertex-wise preference* for v over u as $u \prec_a v$. The *vertex-wise preference* constraints are:

$$\forall u \prec_a v, \quad \sum_i p_{i,u} \leq \sum_i p_{i,v}, \tag{5.1}$$

where the sum of flow into vertex u is smaller than that to v. However, a practical problem with Eq. 5.1 is the availability of such *vertex-wise preferences*, which is an absolute preference of the ranking order between any two vertices.

The new approach we propose models the flow in a network, which we denote by a square matrix $P \in \mathbb{R}^{|V|\times|V|}$, where each element is the weight on an edge E of graph $G = (V, E)$. To this end, we introduce a new numerical optimization

[1] In [71], the entropy of a Markov process, denoted by a stochastic matrix, is defined as the sum of entropies on each row of the matrix that is treated as a probability distribution. Alternatively, one can use conditional entropy. In particular, let X_t denote the state in a Markov process and thus the conditional entropy $H(X_t|X_{t-1}) = \sum_{i=1}^{N} P(X = i)H(X_t|X_{t-1} = i)$ where N is the number of states in the process, $P(X = i)$ is the stationary probability of state i, and $H(X_t|X_{t-1} = i)$ is the entropy associated with each row.

framework that searches for the matrix P [2]. In the following, we introduce the objective functions, the constraints for Markov property and preferences, and the framework for learning network flow.

5.2.1 Objective Functions

We propose two objective functions to be optimized, namely, the *constrained variation* and the *Gini coefficient*. The *constrained variation* objective function is the square distance between the learned matrix and the original PageRank edge weight. Denote the *constrained variation* as $\mathcal{R} : \mathbb{R}^{|V|\times|V|} \mapsto \mathbb{R}^+$, then:

$$\mathcal{R}(P) = \sum_{u,v\in V} (p_{u,v} - g_{u,v})^2, \tag{5.2}$$

where $g_{u,v}$ is the weight on edge (u, v) as in PageRank.

In contrast, the *Gini coefficient* approach uses the Gini coefficient as its objective function. The *Gini coefficient* is a measure of inequality of a distribution [21], and is usually used as an income inequality metric. Denote the *Gini coefficient* by $\mathcal{G} : \mathbb{R}^{|V|\times|V|} \mapsto \mathbb{R}^+$. Per definition,

$$\mathcal{G}(P) = \sum_{u,v\in V} p_{u,v}(1 - p_{u,v}). \tag{5.3}$$

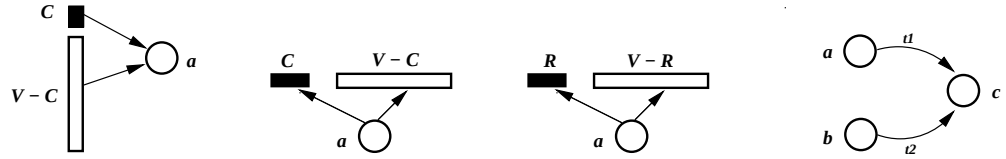
Comments: Note the two objective functions achieve extreme values in different scenarios. For *constrained variation*, the optimal value is at the closest point to the PageRank matrix on the space constructed by the constraints. This can be interpreted as a conservative approach that adjusts PageRank nominal values. For *Gini coefficient*, the optimal value is determined when the matrix is balanced.

[2]Similar to related work on PageRank, P will be further augmented by a dummy vertex for G to become a connected graph.

This is similar in philosophy to entropy based approaches [71, 1] where a more general and random condition of the matrix is favored.

5.2.2 Optimization Constraints

The search for optimal objective functions is bounded by several constraints: (1) the minimization of P should be a Markov matrix; (2) the constraints between pairs of elements in matrix P is determined by the preferences endorsed on graph edges; and (3) the constraints are inferred from aggregate preferences. Here the constraints (2) and (3) are new types of constraints we propose in our new framework for network flow modeling. The vertex-wise preference [1] is a special instance of our aggregate preference.



(a) Collaboration[1] (b) Collaboration[2] (c) Citation (d) Temporality

Figure 5.1. Preferences inferred from collaboration, citation and action temporality such as time delays in citations in academic community. Here a, b, c denote individual authors in the author set V; C is the set of collaborator of a; R is the set of authors whom a cites. t_1 and t_2 are the average delay in time from the date of cited documents to the date of citing documents.

5.2.2.1 Markov property

In order to guarantee that the learned P is representative of a Markov process, we require that the rows of P sum up to one, i.e.

$$\forall v \in V, \quad \sum_i p_{v,i} = 1. \tag{5.4}$$

This is different from related work [71] where the network flow is required to be balanced at each vertex (at each node, the sum of incoming flow is equal to the sum of outgoing flow). We relax the flow balance constraint and adopt the constraint of the more conventional Markov property where only the sum of probabilities that a Markov process flows out from each state is equal to one.

5.2.2.2 Edge-wise preferences: $\prec_\alpha$

The edge-wise preferences are inferred on pairs of edges, which prefer the network flow of one edge over that on another. Let $\prec_\alpha$ be a binary operator that denotes the preference between two edges in a network. For example, $e_1 \prec_\alpha e_2$ indicates a preference of the edge e_1 over e_2. Suppose such preferences compose a set $\lhd_\alpha = \{\langle e_1, e_2\rangle |\ e_1 \prec_\alpha e_2\}$. The corresponding edge-wise constraints require that the preferred edges hold a larger flow weight than their counterparts, i.e.:

$$\forall \langle e_{u,v}, e_{i,j}\rangle \in \lhd_r, \quad p_{u,v} \leq p_{i,j}. \tag{5.5}$$

Similar to vertex-wise preferences as required in related work [1], the edge-wise preferences is another type of implicit judgment that can be used to infer vertex-wise ranking.

5.2.2.3 Aggregate preferences: $\prec_\beta$

Parallel to the element-wise preference previously introduced, the aggregate preferences consider the agglomerate weights of network flow on multiple edges. Such preferences are quite useful when other direct preferences of individual edges are not available. Practical social networks are typically of this kind where social actors often form groups in their actions. Accordingly, the preferences are easier

to derive on groups of actors than on individuals or their singular relationships.

Formally, define a binary operator $\prec_\beta$ between two subsets of edges, say S_i and S_j. $S_i \prec_\beta S_j$ indicates that the summation of edges in S_i should be smaller than that in S_j: $|S_i| \leq |S_j|$, where $|S_i| = \sum_{p_{u,v} \in S_i} p_{u,v}$. Suppose the previously derived aggregate preference set is $\lhd_\beta = \{\langle S_i, S_j \rangle | S_i \prec_\beta S_j\}$. The aggregate constraints are then:

$$\forall \langle S_i, S_j \rangle \in \lhd_\beta, \quad |S_i| \leq |S_j|. \tag{5.6}$$

The *vertex-wise preference* used in related work [1] is an instance of the aggregate preference. Note the aggregate preference allows partial preferences and is therefore easier to obtain.

5.2.3 Optimization framework

With the objective functions and the constraints defined, we propose an optimization framework [3]:

$$\min_{0 \leq p_{u,v} \leq 1} \mathcal{R}(P) \quad or \min_{0 \leq p_{u,v} \leq 1} \mathcal{G}(P) \tag{5.7}$$

subject to

$$\text{E}q.\ 5.4, \text{E}q.\ 5.5, \text{ and Eq. } 5.6. \tag{5.8}$$

Therefore, the key to effective optimization is to determine the preference sets.

[3]Note here we only present the optimization framework without slack variables, which is a natural extension of the current formulation. In addition, the framework can be further improved by introducing a margin into the preference following the SVM paradigm in the case where there is not much preference data available.

The next section describes the inferences of preferences from author actions of social networks in the academic literature, using important indicators including collaborations, citations and temporal relationships among them.

5.3 Extraction of Preferences

This section gives a case study of the modeling network flow in CiteSeer, a repository of computer science publications. Here the social actors are authors and the heterogeneous implicit preferences of actors are inferred from the collaborations, citations, and the temporal characteristics of these actions.

The guidelines for inferring the preferences come from the meaning of edges in the learned Markov matrix. Per definition, an edge in a Markov graph (or an element in the Markov matrix) measures the probability that the process will move from the source of an edge to the sink. In the CiteSeer setting, the Markov process represents the diffusion of innovation through the networked authors. Here a larger weight on the edge $e_{u,v}$ denotes a higher probability that author u acknowledges v's impact and that readers will be more likely to reach v's work after knowing u.

5.3.1 Collaboration preferences

The first indicator of preference we explore is collaboration. The intuition is that the collaborators are more likely to acknowledge each other's work than randomly doing so. Accordingly, the information and innovations that diffuse through the author social network should have a larger probability to move among collaborators than among non-collaborators.

Given the observations of collaborations among authors, we are able to come up with a variety of possible implicit preferences regarding network flow, two of which

are illustrated in Fig. 5.1(a) and Fig. 5.1(b). In the figures, given an actor a from the complete authors set V and her collaborator set C, the implicit preferences we derive are: (1) information is more likely to flow from collaborators C to a than from the rest of the authors, $V - C$; (2) innovations that tend to flow from a to her collaborators C should be stronger than those from a to the rest of the authors, $V - C$. The edge set corresponding to (1) is $E^1_{C(a)} = \{(u, a)|u \in C(a)\}$ and the edge set corresponding to (2) is $E^2_{C(a)} = \{(a, v)|v \in C(a)\}$, and are respectively in a column and a row of matrix P with a as index, and $C(a)$ is the collaborator set of a. The preferences are then derived for the edge sets $E^1_{C(a)}$ and $E^2_{C(a)}$ over their complements, $\bar{E}^1_{C(a)}$ and $\bar{E}^2_{C(a)}$. Following the same notation, for each author $a \in V$, the aggregate preference is denoted by the operator as $\bar{E}^i_{C(a)} \prec_\beta E^i_{C(a)}, i = 1, 2$. Therefore, we obtain the first aggregate preference set due to collaboration which we denote as $\lhd^c$:

$$\lhd^c = \{\langle \bar{E}^i_{C(a)}, E^i_{C(a)} \rangle | a \in V, i = 1, 2\}. \tag{5.9}$$

Next, we discuss the preferences inferred from citations.

5.3.2 Citation preferences

The second type of preferences we propose is based on the citation relationship among authors. Similar to the collaboration preferences, the citation preferences assume a higher probability of citation from the citing authors to the cited authors. Fig. 5.1(c) illustrates the case where author a prefers her citation author set R over that of the rest of the authors, $V - R$. As a result, the sum of the edge weights from author a to her cited authors is preferred over that to the authors she has

not cited.

Let $E_{R(a)}$ be the set of edges pointing from a to the cited author set $R(a)$, and let $\bar{E}_{R(a)}$ be the set of edges pointing from a to the rest of the authors. It immediately follows that the derived aggregate preference set is

$$\lhd^r = \{\langle \bar{E}_{R(a)}, E_{R(a)} \rangle | a \in V\}. \tag{5.10}$$

Note that $E_{R(a)}$ and $\bar{E}_{R(a)}$ are all from the row of matrix P indexed by a and their union constitutes the full row.

Combining both aggregate preference sets, the complete aggregate preference set $\lhd_\beta$ is the union of the preferences derived from collaborations and citations: $\lhd_\beta = \lhd^c \cup \lhd^r$.

5.3.3 Temporal Preferences

The previous sections introduce the aggregate preference set $\lhd_\beta$. This section focuses on the edge-wise preference set $\lhd_\alpha$. Our assumption for inferring temporal preferences is that the edges with a strong network flow should have low latency in information delivery, thus leading to short delays in responsive actions, such as creating citations.

As illustrated in Fig. 5.1(d), we derive preferences from the shorter average delays from the date of cited documents to the date of citing documents, for every author. For example, in Fig. 5.1(d), the fact $t_1 \gg t_2$ leads to the preference that $(a, c) \geq (b, c)$. In particular, we pair-wisely compute the average delay of citations (if any), denoted by $t(i, j)$. The edge-wise preference set is generated as:

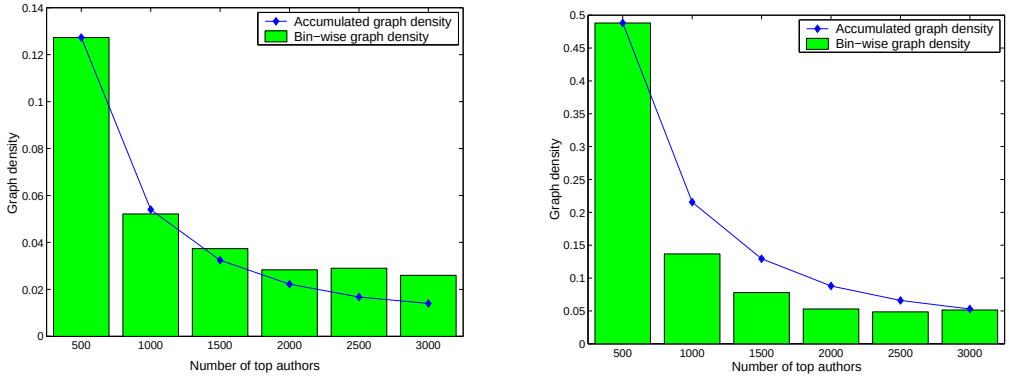
$$\lhd_\alpha = \{\langle (u,v),(i,j)\rangle | t(u,v) \gg t(i,j) > 0\} \tag{5.11}$$

where $\gg$ can be tuned by users according to the confidence of deriving preferences from different time granularity.

Given the preference set $\lhd_\alpha$ and $\lhd_\beta$, we have shown that the solution to the optimization in Eq. 5.7 - Eq. 5.8 can be obtained via quadratic programming (QP) [88].

5.4 Experiments on CiteSeer

For experiments, we use the data introduced in Chapter 4. For efficiency, we adopt a two-step approach. In the first step, the authors are sorted by their accumulated weight on each topic. In the second step, only a subset of the top authors are examined using the proposed numerical approach.



(a) Collaboration graph density (b) Citation graph density

Figure 5.2. Density of author collaboration/citation networks v.s. the number of top authors, on the topic 48: *database and data mining*. The bin-wise graph density only measures the subgraph of the vertices in the i-th bin of ranked authors. The accumulated graph density is computed on the subset of authors drawn from the first bin to the i-th bin. Note in general the citation network density is often significantly higher than that of the collaboration network.

A concern with such author subset generation is whether a significant amount of information will be compromised in the reduced problem and as a result, there are very few linkages in the graph to work on. To address this issue, we perform a simple statistical analysis on the graph densities of author subsets. Our preliminary investigation shows that the subsets of top authors actually endorse more inter-author information on average than the complete author set. This observation also implies that a leading author is more likely to collaborate with other leading authors. Given this observation, it is safe to work on a small subset of top authors without losing much information. Fig. 5.2(a) and Fig. 5.2(b) present the graph density of collaboration and citation networks, on topic 48, i.e. the topic of *database and data mining*. We can see that the density fall sharply as the size of author graph increases. In the following experiments, we work on the top 200 authors with the highest accumulated weights on a specific topic.

5.4.1 PageRank Matrix Formation

We compare our learning of network flow with PageRank [5], in which the network flow is designed based on certain metrics. This section describes the computation of the weighted adjacency matrix in PageRank. Two useful metrics for forming the PageRank matrix are the citations and collaborations relationship among authors. Thus, we construct two graphs, *citation graph* and *collaboration graph*.

For citation graph, the weight on edge from u to v is defined as the sum of the topic weights in the citing documents of u divided by the sum of topic weights in all citations u have formed. Denoting the edge weight as $s_1(v|u, T_i)$, we have:

$$s_1(v|u, T_i) = \frac{\sum_{d \in C(u \to v)} d_i}{\sum_{d \in C(u)} d_i} \tag{5.12}$$

where d_i is the weight of topic T_i for document d, $C(u)$ is the set of documents u have put citations on, $C(u \to v)$ is the set of documents composed by u that cites one of the documents authored by v.

For collaboration graph, for topic T_i and two authors u, v, the edge weight from u to v, $s_2(v|u, T_i)$, is defined as

$$s_2(v|u, T_i) = \frac{\sum_{d \in A(u,v)} d_i}{\sum_{d \in A(u)} d_i} \tag{5.13}$$

where d_i is the weight of topic T_i for document d, $A(u)$ (or $A(u, v)$) are the set of documents authored by u (or both u and v).

Then, using linear combination, we put the two graph together, yielding a new graph that considers both citations and collaborations. So the final weight from u to v with regard to topic T_i looks like:

$$s(v|u, T_i) = \lambda s_1(v|u, T_i) + (1 - \lambda) s_2(v|u, T_i); \tag{5.14}$$

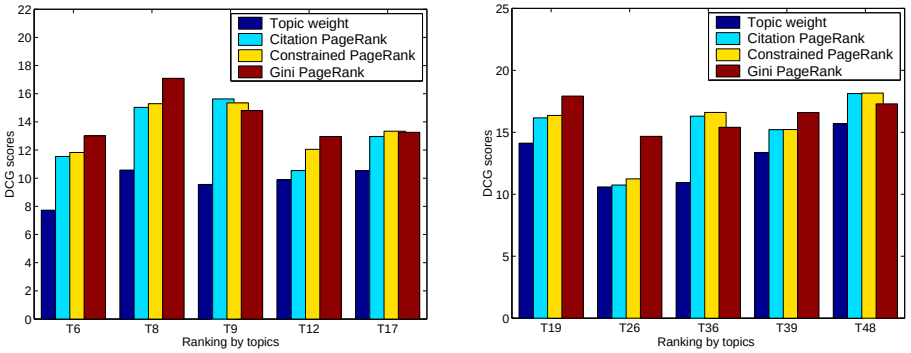
where $\lambda \in [0, 1]$ is the combination parameter. We prefer a larger λ because the denser graph raises less concern about sparsity issue of the matrix (The citation graph has more edges as discussed in the previous section). Otherwise, an ill-conditioned matrix will have problems in the eigen-decomposition required for obtaining the PageRank scores. The PageRank matrix obtained above is reasonable if the traditional weighted graph design in PageRank is used for network flow.

5.4.2 Ranking Quality: Quantitative Evaluation

Hereafter, a quantitative evaluation of the proposed two approaches, namely, Gini coefficient and constrained variation, is carried out against the other ranking methods, including metric-based and PageRank based methods. The methods we compare are itemized below:

- **Topic weight ranking (TW)**, given a topic, the ranking score for each author is the sum of topic weights of all documents published by this author;
- **Citation PageRank (CitePR)**, the ranking scores are given in the principal eigenvector of the weighted citation graph, as constructed according to Eq. 5.12;
- **Collaboration PageRank (CoPR)**, the ranking scores are given in the principal eigenvector of the weighted collaboration graph, as constructed in Eq. 5.13;
- **Constrained PageRank (CPR)**, the ranking is given by the principal eigenvector of the matrix learned using constrained variation as the objective function;
- **Gini PageRank (GiniPR)**, the ranking is given by the principal eigenvector of the matrix learned using Gini coefficient as the objective function;

We use the Discounted Cumulated Gain (DCG) metric [34] to evaluate the ranking quality of various approaches. In particular, four human judges are invited to provide feedback on the composite set of authors who occur in any of the top 20 domain specific ranking list, yielding the DCG_{20} scores. As suggested, assessments are carried out based on readily accepted professional achievement



(a) DCG_{20} scores for T6, T8, T9, T12, T17

(b) DCG_{20} scores for T19, T26, T36, T39, T48

Figure 5.3. DCG_{20} scores for T6, T8, T9, T12, T17, T19, T26, T36, T39, T48 in Table 4.4. The ranking using the learned matrices, i.e. the Gini PageRank and the constrained PageRank, normally outperform the ranking using the designed PageRank matrix. All PageRank-based rankings are significantly better than the counting-based ranking.

of these authors, such as winning of prestigious international awards (e.g. Turing Award), membership of national academy, and fellowship of ACM/IEEE, etc. Numerical assessment scores of $0, 1, 2$, and 3 are collected to reflect the judges' opinion with regard to whether an author is ranked top 20 in a certain field, which respectively imply the sentiment of *stronglydisagree*, *disagree*, *agree*, and *stronglyagree*. The average judge scores are used for computing the DCG. We repeat this ranking and assessment process for 10 of the 50 automatically generated topics, which our judge are most familiar with. In general, the judges reach a high agreement on the ranking quality, with the Kappa Coefficient [9] between 73%-92%.

The DCG scores attained for four methods on the 10 selected topics are presented in the Fig. 5.3(b) and Fig. 5.3(b). Each figure shows five categories of bars, each denoting the evaluations on a specific topic. The PageRank-based approaches clearly outperform the ranking using topic-weight counts. On average, the proposed constrained variation learning yields an improvement of 28.7%, and

the Gini coefficient of 35.4%, both over the count-based approach. In addition, we observe significant higher DCG scores of the two proposed approaches over the traditional PageRank on the citation-based graph (we omit collaboration-based PageRank here due to their very similar performances). On average, the new constrained variation-based learning generates an improvement of 2.3% while the new Gini coefficient-based learning brings an advance of 7.6%, both over the traditional PageRank, in ranking quality.

5.4.3 Ranking Quality: Case Study

To provide a closer look at the reported improvements, we follow by several case studies in two specific topics. The topics we choose are T19 and T48, i.e. topic *database and data mining* and topic *learning and classification*.

First, we compare the PageRank on the proposed Gini-based learning with the other metric-based ranking, as presented in Table 5.1. We can see the ranking based on topic-weight count generally complies with the number of publications as collected in CiteSeer. As a result, the commonly valued citation factor is not weighted enough to reflect the acknowledgement from the community. On the other hand, the ranking based on citation count fails to differentiate the research domains and misses the topology of the hidden collaboration network. Similar problems exist with CitePR, CoPR which only look at one aspect of the picture. By contrast, the proposed GiniPR well combines the factors of topic weights, citations, and collaborations. For example, *Rajeev Motwani* does not appear in the top 30 of the TW Rank because of having insufficient documents in CiteSeer. However, the Gini index ranks him in top 5 in database research due to the large number of citations to him. Another example is *Hector Garcia-Molina* who ranks 6th in

TopicWeight Rank	Pub #	Cite #	Col. #	CitePR	CoPR	GiniPR	Gini Rank
Jiawei Han	142	1312	13	$\Downarrow_{15}$	$\Downarrow_{1}$	$\Downarrow_{7}$	Jennifer Widom
Christian S. Jensen	104	390	11	$\Downarrow_{50}$	$\Uparrow_{1}$	$\Downarrow_{24}$	Hector Garcia-Molina
Jennifer Widom	113	3331	18	$\Uparrow_{1}$	$\Downarrow_{1}$	$\Uparrow_{2}$	Rakesh Agrawal
Rakesh Agrawal	129	4068	11	$\Uparrow_{3}$	$\Downarrow_{4}$	$\Uparrow_{1}$	**Rajeev Motwani**
Serge Abiteboul	115	2497	18	$\Uparrow_{1}$	$\Uparrow_{2}$	$\Downarrow_{1}$	**Jeffrey Scott Vitter**
Hector Garcia-Molina	169	2504	20	$\Uparrow_{3}$	$\Uparrow_{1}$	$\Uparrow_{4}$	Serge Abiteboul
Joseph Hellerstein	75	41	2	$\Downarrow_{131}$	$\Downarrow_{4}$	$\Downarrow_{59}$	Divesh Srivastava
Elke A. Rundensteiner	124	422	4	$\Downarrow_{80}$	$\Downarrow_{9}$	$\Downarrow_{86}$	Jiawei Han
Michael Stonebraker	144	857	6	$\Downarrow_{1}$	$\Downarrow_{9}$	$\Downarrow_{7}$	**Jeffrey D. Ullman**
Richard T. Snodgrass	68	471	9	$\Downarrow_{30}$	$\Uparrow_{1}$	$\Downarrow_{25}$	**Ramakrishnan Srikant**
Dan Suciu	98	1589	10	$\Uparrow_{5}$	$\Downarrow_{2}$	$\Downarrow_{1}$	**Gio Wiederhold**
Leonid Libkin	124	791	10	$\Downarrow_{11}$	$\Downarrow_{10}$	$\Downarrow_{13}$	Dan Suciu
Carlo Zaniolo	84	579	4	$\Downarrow_{25}$	$\Downarrow_{58}$	$\Downarrow_{20}$	**Christos Faloutsos**
Heikki Mannila	86	920	5	$\Downarrow_{4}$	$\Downarrow_{13}$	$\Downarrow_{4}$	Surajit Chaudhuri
Giuseppe De Giacomo	140	784	4	$\Downarrow_{62}$	$\Uparrow_{8}$	$\Downarrow_{81}$	**Luis Gravano**
Diego Calvanese	87	537	4	$\Downarrow_{53}$	$\Uparrow_{10}$	$\Downarrow_{54}$	Michael Stonebraker
Surajit Chaudhuri	51	548	6	$\Downarrow_{3}$	$\Downarrow_{11}$	$\Uparrow_{3}$	Alon Levy
Divyakant Agrawal	78	286	1	$\Downarrow_{102}$	$\Downarrow_{30}$	$\Downarrow_{72}$	Heikki Mannila
Amr El Abbadi	74	208	1	$\Downarrow_{120}$	$\Downarrow_{30}$	$\Downarrow_{90}$	**Raghu Ramakrishnan**
Jan Chomicki	77	560	6	$\Downarrow_{30}$	$\Downarrow_{1}$	$\Downarrow_{7}$	Richard T. Snodgrass
Maurizio Lenzerini	86	1087	4	$\Downarrow_{46}$	$\Uparrow_{11}$	$\Downarrow_{50}$	Rajeev Rastogi
Alon Levy	83	1470	6	$\Uparrow_{13}$	$\Downarrow_{3}$	$\Uparrow_{5}$	**Ben Shneiderman**
Divesh Srivastava	62	943	17	$\Uparrow_{6}$	$\Uparrow_{4}$	$\Uparrow_{16}$	**Peter Buneman**
Philip S. Yu	85	576	9	$\Downarrow_{18}$	$\Uparrow_{4}$	$\Downarrow_{4}$	**Timos Sellis**
Rajeev Rastogi	97	671	8	$\Downarrow_{5}$	$\Uparrow_{2}$	$\Uparrow_{4}$	Leonid Libkin
Joel Saltz	105	1081	7	$\Downarrow_{68}$	$\Uparrow_{12}$	$\Downarrow_{28}$	Christian S. Jensen
Hans-Peter Kriegel	63	458	2	$\Downarrow_{14}$	$\Downarrow_{121}$	$\Downarrow_{2}$	Jan Chomicki
Sudarshan Chawathe	61	376	5	$\Downarrow_{4}$	$\Downarrow_{1}$	$\Downarrow_{154}$	Philip S. Yu
Ling Liu	106	341	4	$\Downarrow_{37}$	$\Downarrow_{3}$	$\Downarrow_{30}$	Hans-Peter Kriegel
George Karypis	125	1211	4	$\Downarrow_{24}$	$\Downarrow_{7}$	$\Downarrow_{4}$	Ling Liu

Table 5.1. Top authors in T48, *database and data mining*, ranked by various methods. Rankings are provided by topic weight count (TW Rank), PageRank on citation graph (CitePR), PageRank on collaboration graph (CoPR), PageRank on learned graph using gini coefficient (GiniPR). Simple statistics included are the publication number, citation number, and the number of collaborators. Authors exclusively in Gini Rank are highlighted.

TW Rank is now 2nd in GiniRank since he has 20 collaborators within the same author bin (top 200 authors). Note that in the last column many of the highlighted authors missing from the first column have great impacts in the field.

Second, we compare the PageRank on the proposed *constrained variation* learning and the PageRanks on the networks by design. This is a case study on the topic 19: *learning and classification*. In the second column of Table 5.2, the top authors by their total citation number are shown. It is obvious that the citation counts fail to differentiate research domains by preferring many influential researchers from other domains. Both citation graph and collaboration graph-based PageRank seem to disagree strongly with the ranking by citation count. In particular, compared with citation count, the citation graph-based PageRank seems to give

Cite Rank	Pub #	Cite #	Col. #	CitePR	CoPR	GiniPR	CPR Rank
Robert E. Schapire	67	2030	13	-	$\Downarrow_6$	-	Robert E. Schapire
Sebastian Thrun	293	1832	9	$\Downarrow_{11}$	$\Uparrow_1$	$\Downarrow_{12}$	Yoav Freund
Michael I. Jordan	91	1540	4	$\Downarrow_1$	$\Downarrow_{167}$	-	Michael I. Jordan
Yoav Freund	68	1439	12	$\Uparrow_2$	$\Downarrow_1$	$\Uparrow_2$	David Haussler
Ron Kohavi	71	1395	7	$\Downarrow_5$	$\Downarrow_6$	$\Downarrow_3$	Avrim Blum
Leslie Pack Kaelbling	62	1383	6	$\Downarrow_9$	$\Downarrow_{125}$	$\Downarrow_{10}$	Michael Kearns
Jiawei Han	142	1312	1	$\Downarrow_{70}$	$\Downarrow_{79}$	$\Downarrow_{70}$	Trevor Hastie
Stephen Muggleton	45	1272	4	$\Downarrow_{21}$	$\Downarrow_{181}$	$\Downarrow_{23}$	Ron Kohavi
Daphne Koller	137	1250	7	$\Downarrow_3$	$\Downarrow_{37}$	-	Daphne Koller
Michael L. Littman	79	1229	7	$\Downarrow_1$	$\Downarrow_{156}$	$\Downarrow_{19}$	Yoram Singer
George Karypis	125	1211	1	$\Downarrow_{116}$	$\Downarrow_{61}$	$\Downarrow_{118}$	Manuela Veloso
Thomas G. Dietterich	53	1167	1	$\Downarrow_2$	$\Downarrow_{77}$	$\Downarrow_1$	Manfred K. Warmuth
William W. Cohen	68	1140	6	$\Downarrow_5$	$\Downarrow_{27}$	$\Downarrow_5$	Thomas G. Dietterich
Tomaso Poggio	88	1055	4	$\Downarrow_{29}$	$\Downarrow_{170}$	$\Downarrow_{28}$	Sebastian Thrun
Manuela Veloso	196	991	4	$\Downarrow_7$	$\Downarrow_{119}$	$\Uparrow_4$	Tom Mitchell
Marco Dorigo	80	976	1	$\Downarrow_{49}$	$\Downarrow_{57}$	$\Downarrow_{36}$	Leslie Pack Kaelbling
Trevor Hastie	88	941	1	$\Downarrow_{11}$	$\Downarrow_{49}$	$\Uparrow_{10}$	Peter Dayan
Michael Kearns	73	920	15	$\Uparrow_{12}$	$\Uparrow_{12}$	$\Uparrow_{12}$	William W. Cohen
David Haussler	65	864	9	$\Uparrow_{14}$	$\Downarrow_{119}$	$\Uparrow_{15}$	Sebastian B. Thrun
Thorsten Joachims	57	806	4	$\Downarrow_{20}$	$\Downarrow_{30}$	$\Downarrow_{19}$	Vladimir Vapnik
Zoubin Ghahramani	77	783	3	-	$\Downarrow_{158}$	$\Downarrow_1$	Yishay Mansour
David H. Wolpert	41	783	3	$\Downarrow_{16}$	$\Downarrow_{136}$	$\Downarrow_{15}$	Zoubin Ghahramani
Dayne Freitag	65	782	9	$\Downarrow_{13}$	$\Downarrow_{20}$	$\Downarrow_{11}$	Pat Langley
Eric Brill	49	779	1	$\Downarrow_{36}$	$\Downarrow_{75}$	$\Downarrow_{33}$	Nir Friedman
Yoram Singer	78	769	10	$\Uparrow_{18}$	$\Downarrow_4$	$\Uparrow_{15}$	Richard S. Sutton
Avrim Blum	295	768	10	$\Downarrow_6$	$\Uparrow_{24}$	$\Uparrow_{21}$	Leo Breiman
Nir Friedman	131	756	6	$\Uparrow_1$	$\Downarrow_{110}$	$\Uparrow_3$	Tommi Jaakkola
Yishay Mansour	115	754	11	$\Uparrow_8$	-	$\Uparrow_7$	Satinder Singh
Vladimir Vapnik	33	742	4	$\Uparrow_6$	$\Downarrow_{167}$	$\Uparrow_9$	Michael L. Littman
Richard S. Sutton	35	726	6	$\Uparrow_6$	$\Downarrow_{93}$	$\Uparrow_5$	Naftali Tishby

Table 5.2. Top authors in T19, *learning and classification*, ranked by various methods. Rankings are provided by citation count ranking (Cite #), PageRank on citation graph (CitePR), PageRank on collaboration graph (CoPR), PageRank on learned graph using Gini coefficient (GiniPR), and PageRank by constrained variation learning (CPR).

better ranking due to its topic consideration and the citation factor. Accordingly, we set the smoothing parameter $\lambda = 0.7$ to prioritize the citation graph in the objective function in learning. As a result, the CPR Rank generated on the matrix by constrained variation-based learning agrees better with the CoPR. In general, the CPR Rank is closer to traditional methods. Then when should one choose the CPR over GiniRank? We discuss this next.

The overall preliminary experiments show the change in ranking is smaller in constrained variation-based learning but is larger in Gini index-based learning. Thus we consider the constrained variation as a more conservative improvement over PageRank even though it usually brings less changes. On the other hand, the Gini index-based approach seems to differ much more from other alternatives and thus it is preferred when an reference approach is not available or poor. In

fact, as discussed before, the Gini index is very similar to the entropy measure, which reaches the minimum when the entropy is maximized. Previous studies has shown improved ranking quality when the entropy of the network flow matrix is maximized [71, 1]. Thus our result is also supported by the related work based on entropy.

5.5 Summary

We proposes a new method for ranking social network actors based on the social documents they share and act on in terms of citing others. The proposed method models the network flow by learning the implicit preferences from multiple preferences of actors in a social network. The problem can be shown to be described as a QP problem. Two variants of the new methods differ in their objective functions, with one minimizing the distance to the PageRank weighted graph (CPR) and the other maximizing the balance of the resulting network flow using the Gini coefficient (GiniPR). In particular, the CPR method behaves similar to the traditional PageRank approach and is thus more conservative. The GiniPR method parallels entropy maximization approaches and empirically performs better. Experimental evaluations are carried out on real-world dataset, the CiteSeer author records, showing significant improvements in the ranking quality.

Chapter 6

Co-Ranking in Heterogeneous Social Networks

6.1 The Co-Ranking Problem

Quantitative evaluation of researchers' contributions has become an increasingly important topic since the late 80's due to its practical importance for making decisions concerning matters of appointment, promotion and funding. As a result, bibliometrics indicators such as citation counts and different versions of the *Journal Impact Factor* [19, 42] are being widely used, although it is a subject of much controversy [77]. Accordingly, new metrics are constantly being proposed and questioned, leading to ever-increasing research efforts on bibliometrics [29, 42]. These simple counting metrics are attractive, because it is convenient to have a single number that is easy to interpret. However, it has become evident in recent research that the evaluation if the scientific output of individuals can be performed better by considering the network structures among the entities in question (e.g. [66, 43]).

Recently, a great amount of research has been concerned with ranking net-

worked entities, such as social actors or Web pages, to infer and quantify their relative importance, given the network structure. Several *centrality* measures have been proposed for that purpose [5, 38, 76]. For example, a journal can be considered influential if it is cited by many other journals, especially if those journals are influential, too. Ranking networked documents received a lot of attention, particularly because of its applications to search engines. (e.g. PageRank [5], HITS [38]). Ranking social network actors, on the other hand, is employed for exploring scientific collaboration networks [78], understanding terrorist networks [45, 78], ranking scientific conferences [66] and mining customer networks for efficient viral marketing [15]. While centrality measures are finding their way into traditional bibliometrics, let us point out that the evaluations of the relative importance of networked documents have been carried *independently*, in the similar studies, from social network actors, where the natural connection between researchers and their publications *authorship* and the social network among researchers are not fully leveraged.

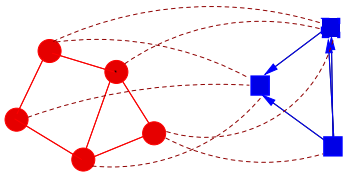


Figure 6.1. Three networks we use for co-ranking: a social network connecting authors, the citation network connecting documents, and the co-authorship network that ties the two together. Circles represent authors, rectangles represent documents.

This chapter introduce a new framework for co-ranking entities of different kinds in a heterogeneous network connecting the researchers (*authors*) and publications they produce (*documents*) [91]. The heterogeneous network is comprised of G_A, a social network based on social actions connecting authors, G_D, the citation network connecting documents, and G_{AD}, the bipartite authorship network that

ties the previous two together. Further details will be given in § 6.2. A simple example of a such a heterogeneous network is shown in Fig. 6.1.

We propose a co-ranking method in a heterogeneous network by coupling two random walks on G_A and G_D using the authorship information in G_{AD}. We assume that there is a mutually reinforcing relationship between authors and documents that could be reflected in the rankings. In particular, the more influential an author is, the more likely his documents will be well-received. Meanwhile, well-known documents bring more acknowledgments to their authors than those that are less cited. While it is possible to come up with a ranking of authors based solely on a social network and obtain interesting and meaningful results [43], these results are inherently limited, because they include no direct consideration neither of the number of publications of a given author (encoded in the authorship network) nor of their impact (reflected in the citation network).

6.2 Co-Ranking Framework

Denote the heterogeneous graph of authors and documents as $G = (V, E) = (V_A \cup V_D, E_A \cup E_D \cup E_{AD})$. There are three graphs (networks) in question. $G_A = (V_A, E_A)$ is the unweighted undirected graph (social network) of authors. V_A is the set of authors, while E_A is the set of bidirectional edges, representing social ties. The number of authors $n_A = |V_A|$ and authors are denoted as $a_i, a_j, \cdots \in V_A$. $G_D = (V_D, E_D)$ is the unweighted directed graph (citation network) of documents, where V_D is the document set, E_D is the set of links, representing citations between documents. The number of documents $n_D = |V_D|$. Individual documents are denoted as $d_i, d_j, \cdots \in V_D$. $G_{AD} = (V_{AD}, E_{AD})$ is the unweighted bipartite graph representing authorship. $V_{AD} = V_A \cup V_D$. Edges in E_{AD} connect each document

with all of its authors.

The framework includes three *random walks*, one on G_A, one on G_D and one on G_{AD}. A random walk on a graph is a Markov chain, its states being the vertices of the graph. It can be described by a square $n \times n$ matrix M, where n is the number of vertices in the graph. M prescribes the transition probabilities. That is, $0 \leq p(i,j) = M_{i,j} \leq 1$ is the conditional probability that the next state will be vertex j, given that the current state is vertex i. If there is no edge from vertex i to vertex j then $M_{i,j} = 0$, with the exception when there are no outgoing edges from vertex i at all. In that case we assume that $M_{i,j} = \frac{1}{n}$ for all vertices j. By definition, M is a *stochastic matrix*, i.e. its entries are nonnegative and every row adds up to one. A *simple random walk* on a graph goes equi-probably to any of the current vertex' neighbors.

In this chapter, "Markov chain" and "random walk" are used interchangeably to mean "time-homogeneous finite state-space Markov chain". After one step of a random walk, described by a stochastic matrix M, the probability distribution will be $M^T\mathbf{v}$, where M^T is the transpose of M. A *stationary probability distribution* $\mathbf{v_{st}} = \lim_{n\to\infty}(M^T)^n\mathbf{v}$ contains the limiting probabilities after a large number of steps of the random walk. It is a common convention that the PageRank ranking vector $\mathbf{r}$ satisfies $\|\mathbf{r}\|_1 = 1$, naturally, since $\mathbf{r}$ is a probability distribution. The co-ranking framework will produce two ranking vectors, $\mathbf{a}$ for authors and $\mathbf{d}$ for documents, also satisfying

$$\forall 1 \leq i \leq n_A, 1 \leq j \leq n_D, \quad a_i, d_j \geq 0; \tag{6.1}$$

$$\|\mathbf{a}\|_1 = 1, \|\mathbf{d}\|_1 = 1 \tag{6.2}$$

As mentioned above, we will have three random walks. The random walk on G_A (respectively, G_D) will be described by a stochastic matrix $\widetilde{A}$ (respectively, $\widetilde{D}$). We shall start from two random walks, described by stochastic matrices A and D, and then slightly alter them in § 6.2.1 to actually obtain $\widetilde{A}$ and $\widetilde{D}$. All of them are called *Intra-class random walks*, because they walk either within the authors' or the documents' network. The third random walk on G_{AD} is called the *Inter-class random walk*. It will suffice to describe it by an $n_A \times n_D$ matrix AD and an $n_D \times n_A$ matrix DA, since G_{AD} is bipartite. The design of A, D, AD and DA is postponed until § 6.4.

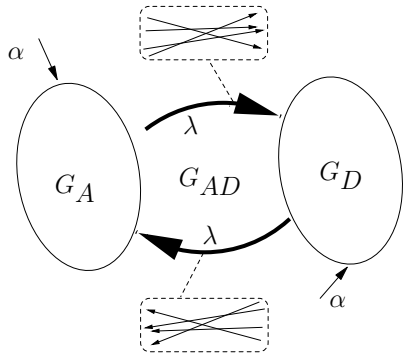


Figure 6.2. The framework for co-ranking authors and documents. G_A is the social network of authors. G_D is the citation network of documents. G_{AD} is the authorship network. α is the jump probability for the Intra-class random walks. λ is a parameter for coupling the random walks, quantifying the importance of G_{AD} versus that of G_A and G_D.

Before making everything precise, let us briefly sketch the co-ranking framework. The conceptual scheme is illustrated in Fig. 6.2. Two Intra-class random walks incorporate the *jump probability* α, which has the similar meaning to the damping factor as used in PageRank. They are coupled using the Inter-class random walk on the bipartite authorship graph G_{AD}. The coupling is regulated by λ. In the extreme case $\lambda = 0$ there is no coupling; this amounts to separately ranking authors and documents by PageRank. In general, λ represents the extent to which

we want the rankings of documents and their authors depend on each other[1].

6.2.1 PageRank: two random walks

First of all, we are going to rank the networks of authors and documents independently, according to the PageRank paradigm [5]. Consider a random walk on the author network G_A and let A be the transition matrix (A will be defined in § 6.4). Fix some α and say that at each time step with probability α we do not make a usual random walk step, but instead jump to any vertex, chosen uniformly at random. This is another random walk with the transition matrix

$$\widetilde{A} = (1-\alpha)A + \frac{\alpha}{n_A}\mathbf{1}\mathbf{1}^{\mathbf{T}} \tag{6.3}$$

Here $\mathbf{1}$ is the vector of n_A entries, each being equal to one. Let $\mathbf{a} \in \mathbf{R}^{n_A}$, $\|\mathbf{a}\|_1 = 1$ be the only solution of the equation

$$\mathbf{a} = \widetilde{A}^T\mathbf{a} \tag{6.4}$$

.

Vector $\mathbf{a}$ contains the ranking scores for the vertices in G_A. It is a standard fact that the existence and uniqueness of the solution of (6.4) follows from the random walk $\widetilde{A}$ being ergodic, and this is why we are using $\widetilde{A}$ instead of A. ($\alpha > 0$ guarantees irreducibility, because we can jump to any vertex in the graph.)

Documents can be ranked in the citation network G_D in a similar way. In

[1]This is a symmetric setting of parameters. An asymmetric setting of parameters can introduce $\alpha_A \neq \alpha_D$ and $\lambda_{AD} \neq \lambda_{DA}$. We do not expect that different α can make any difference. We do expect that different λ can make a difference, but we did not investigate that. Note, however, that in the latter case one would need a different normalization instead of (6.2), satisfying $\|\mathbf{a}\|_1\lambda_{AD} = \|\mathbf{d}\|_1\lambda_{DA}$.

particular,

$$\widetilde{D} = (1-\alpha)D + \frac{\alpha}{n_D}\mathbf{1}\mathbf{1}^{\mathbf{T}}, \tag{6.5}$$

6.2.2 (m, n, k, λ)–coupling of two Intra-class random walks

To couple these two random walks we construct a combined random walk on the heterogeneous graph $G = G_A \cup G_D \cup G_{AD}$. A probability distribution on such a graph will have the form $(\mathbf{a}, \mathbf{d})$, satisfying $\|\mathbf{a}\|_1 + \|\mathbf{d}\|_1 = 1$. We will use the stationary probabilities of the vertices in V_A to rank authors and the stationary probabilities of the vertices in V_D to rank documents. In fact, we will multiply all of them by 2 to ensure that $\|\mathbf{a}\|_1 = \|\mathbf{d}\|_1 = 1$. Of course, the greater the stationary probability (ranking score), the higher the rank of an author or a document.

The coupling is parameterized by four parameters, m, n, k and λ. Ordinary PageRank score is sometimes viewed as the probability that a *random surfer* will be on this web page at some moment in the distant future. Similarly, we present the combined random walk in terms of a random surfer (RS) who is capable of browsing over documents and their authors as well.

If at any given moment RS finds himself on the author side, the current vertex $v \in V_A$, then he can either make an *Intra-class step* (one step of the random walk parameterized by $\widetilde{A}$) or an *Inter-class step* — one step of the Inter-class random walk. Similarly, if RS finds himself on the document side, the current vertex $v \in V_D$, then one option is to make an *Intra-class step* (one step of the random walk parameterized by $\widetilde{D}$) while another option is to make one step of the Inter-class random walk. In general, one Intra-class step changes the probability distribution from $(\mathbf{a}, \mathbf{0})$ to $(\widetilde{A}\mathbf{a}, \mathbf{0})$ or from $(\mathbf{0}, \mathbf{d})$ to $(\mathbf{0}, \widetilde{D}\mathbf{d})$, while one Inter-class step changes the probability distribution from $(\mathbf{a}, \mathbf{d})$ to $(DA^T\mathbf{d}, AD^T\mathbf{a})$.

Now, the combined random walk is defined as follows:

1. If the current state of RS is some author, $v \in V_A$, then with probability λ take $2k+1$ Inter-class steps, while with probability $1-\lambda$ take m Intra-class steps on G_A.

2. If the current state of RS is some document, $v \in V_D$, then with probability λ take $2k+1$ Inter-class steps, while with probability $1-\lambda$ take n Intra-class steps on G_D.

It is convenient to write a subroutine *BiWalk* (Algo. 2) that takes $\mathbf{x}$, the probability distribution on one side of a bipartite graph and returns the distribution on the other side after taking $2k+1$ Inter-class steps. U is the transition matrix from the current side to the other and V is the transition matrix from the other side back to the current side.

Algorithm 2 Random walk on a Bipartite Graph

procedure $BiWalk(U, V, \mathbf{x}, k)$

$\mathbf{c} \leftarrow \mathbf{x}$
for $i = 1$ to k **do**
  $\mathbf{b} \leftarrow U^T\mathbf{c}$
  $\mathbf{c} \leftarrow V^T\mathbf{b}$
end for
$\mathbf{b} \leftarrow U^T\mathbf{c}$
return $\mathbf{b}$

Now, everything is ready to realize co-ranking in the following procedure, *CoupleWalk* (Algo. 3). It should be noted that the very recent work [32] of learning on subgraphs can be considered an implicit special version of our algorithm with infinite k and $m = n = 1$.

Algorithm 3 Coupling random walks for co-ranking

procedure $CoupleWalk(\widetilde{A}, \widetilde{D}, AD, DA, m, n, k, \lambda, \epsilon)$

1: $\mathbf{a} \leftarrow \frac{1}{n_A}\mathbf{1}$
2: $\mathbf{d} \leftarrow \frac{1}{n_D}\mathbf{1}$
3: **repeat**
4: $\quad \mathbf{a}' \leftarrow \mathbf{a}$
5: $\quad \mathbf{d}' \leftarrow \mathbf{d}$
6: $\quad \mathbf{a} \leftarrow (1-\lambda)(\widetilde{A}^T)^m\mathbf{a}' + \lambda BiWalk(DA, AD, \mathbf{d}', k)$
7: $\quad \mathbf{d} \leftarrow (1-\lambda)(\widetilde{D}^T)^n\mathbf{d}' + \lambda BiWalk(AD, DA, \mathbf{a}', k)$
8: **until** $|\mathbf{a} - \mathbf{a}'| \leq \epsilon$
9: return $\mathbf{a}$, $\mathbf{d}$

6.3 Convergence Analysis

We need to ensure that Algo. 3 converges. Note that $BiWalk(U, V, \mathbf{x}, k) = U^T(V^TU^T)^k\mathbf{x}$. Therefore, lines 6 and 7 in Algo. 3 can be rewritten as:

$$\mathbf{a}^{t+1} = (1-\lambda)(\widetilde{A}^T)^m\mathbf{a}^t + \lambda DA^T(AD^TDA^T)^k\mathbf{d}^t \tag{6.6}$$

$$\mathbf{d}^{t+1} = (1-\lambda)(\widetilde{D}^T)^n\mathbf{d}^t + \lambda AD^T(DA^TAD^T)^k\mathbf{a}^t \tag{6.7}$$

where $\mathbf{a}^t$ and $\mathbf{d}^t$ are the ranking vectors for authors and documents from the previous iteration; m, n are prescribed parameters. Now we concatenate $\mathbf{a}$ and $\mathbf{d}$ into a vector $\mathbf{v}$ such that $\mathbf{v} = [\mathbf{a}^T, \mathbf{d}^T]^T$. In particular, $\mathbf{v}^t = [(\mathbf{a}^t)^T, (\mathbf{d}^t)^T]^T$, is composed of $\mathbf{a}$ and $\mathbf{d}$ as in Algo. 3 after t iterations. Construct a matrix M, where

$$M = \begin{bmatrix} (1-\lambda)(\widetilde{A}^T)^m & \lambda DA^T(AD^TDA^T)^k \\ \lambda AD^T(DA^TAD^T)^k & (1-\lambda)(\widetilde{D}^T)^n \end{bmatrix}. \tag{6.8}$$

Clearly, $\mathbf{v}^{t+1} = M\mathbf{v}^t$, and M is a stochastic matrix that parameterizes the combined random walk. It is also easy to see that for $0 < \alpha, \lambda < 1$, this Markov

Chain is ergodic. Thus, the stationary probabilities can be found as $\lim_{n\to+\infty} M^n\mathbf{v}$, for any initial vector $\mathbf{v}$. In particular, $\mathbf{a}$ and $\mathbf{d}$ in Algo. 3 will converge to the ranking scores as we defined them. In practice, the convergence can be established numerically.

6.4 Random Walks in a Scientific Repository

This section sets up the co-ranking framework to be applied to co-ranking scientists and their publications. It includes defining three networks and the three corresponding random walks, parameterized by four stochastic matrices: A (giving rise to $\widetilde{A}$), D (giving rise to $\widetilde{D}$), AD and DA.

6.4.1 G_D: citation network, and D the Intra-class random walk

The citation document network G_D is defined as follows: there is a directed edge from d_i to d_j, if document d_i cites document d_j at least once. The graph is not weighted; we ignore repeated citations from the same document to the same document. Self-citations are technically allowed, but, presumably, there are none.

The design of D is straightforward. Namely, the Intra-class random walk on G_D is just a simple random walk on it. The transition probability

$$P(j|i) = D_{i,j} = \frac{n^D_{i,j}}{n^D_i}, \tag{6.9}$$

where $n^D_{i,j}$ is the indicator of whether document i cites j; n^D_i is the total number of citations document i makes. If a document does not cite anything (which effectively means that the citations of this documents are not in the corpus), let the transition

probabilities from this document be $\frac{1}{n_D}$.

6.4.2 G_A: social network, and A the Intra-class random walk

To define A, we come up with a more general definition to employs the notion of a *social event*. A social event could be any kind of activity, involving a group of authors. A co-occurrence of two authors in a social event is supposed to create or strengthen their social ties. In particular, we view collaborating on a paper or co-participating in a conference as such "co-occurrences". Let the set of social events be $\mathcal{E} = \{e_i\}$, where an event e_i is identified with the set of participating authors. We construct G_A as an unweighted graph, where two authors are connected by an edge if they co-occur in some social event $e \in \mathcal{E}$.

Intuitively, a paper of fewer authors infers stronger social ties among them on average (cf. [43]). To take this into account, we first make the graph G_A weighted. Define the social tie function $\tau(i, j, e_k) : \mathcal{A}\times\mathcal{A}\times\mathcal{E} \to [0, 1]$ representing the strength of a social tie between actor a_i and actor a_j resulting from their co-occurrence in the event e_k. The strength of the social tie depends on the size of the corresponding social event. If there are only two people taking part in an event (say, collaborating on a paper), we say that it infers a *unit social tie*. Otherwise, the tie is somehow normalized by the size of the event:

$$\tau(i, j, e_k) = \frac{\mathbb{I}(i, j \in e_k)}{|e_k|(|e_k| + 1)/2} \tag{6.10}$$

where $\mathbb{I}(i, j \in e_k)$ is the indicator function of whether authors i and j co-occur in the event e_k (that is, if $a_i \in e_k$ and $a_j \in e_k$; it can be that $a_i = a_j$). $|e_k| \geq 2$ is the number of authors involved in event e_k. For $|e_k| = 1$, only a self social tie of

that author is inferred. Adding up social ties inferred from all events, we obtain a cumulative matrix $T = (T_{i,j}) \in \mathbb{R}^{n_A \times n_A}$, by definition:

$$T_{i,j} = \sum_{e_k \in \mathcal{E}} \tau(i, j, e_k) \tag{6.11}$$

where $\mathcal{E}$ is the set of social events. Now G_A can be viewed as a weighted graph, with the weight on the edge connecting a_i and a_j being $T_{i,j}$.

In this chapter, we consider two kinds of social events. The first kind is a collaboration on a paper (even if the paper has a single author), in this case the 'event' includes exactly all the authors of this paper. The second kind is the appearance of names in conference proceeding lists. Each conference instance (i.e. *ACM SIGMOD '01*) is a separate event, consisting of the authors who took part in it. We treat the two kinds equally, and we find it appropriate because of the normalization (6.10).

We proceed to define the Intra-class random walk on G_A in a natural way, namely, the next step is chosen according to the weights on the edges. Technically, it amounts to normalizing T by rows. The transition probabilities from author a_i to author a_j (i.e. of the author a_j given a_i) can then be found as:

$$P(j|i) = A_{i,j} = \frac{T_{i,j}}{\sum_j T_{i,j}}. \tag{6.12}$$

Here T is symmetric due to the design of τ. A is not necessarily symmetric because row sums can be different. $\widetilde{A}$ is defined accordingly.

6.4.3 G_{AD}: the bipartite authorship network, and AD, DA: the Inter-class random walk on G_{AD}

The bipartite authorship graph G_{AD} is defined in the natural way. Namely, the entries in its adjacency matrix E_{AD} are the values of the indicator function of a document being written by an author, i.e.

$$E_{AD}(i,j) = \mathbb{I}(d_j \text{ is authored by } a_i). \tag{6.13}$$

Using the adjacency matrix E_{AD}, we define a weight matrix $W_{AD} = (w(i,j))$ as follows:

$$w(i,j) = \frac{E_{AD}(i,j)}{n_j^A}, \tag{6.14}$$

where n_j^A is the number of authors of the document d_j.

Then we proceed to define AD and DA, containing the conditional transition probabilities of a random surfer moving from author i to document j and vice versa, respectively, given that the next step is taken in the bipartite graph G_{AD}. That is, let

$$P(d_j|a_i) = AD_{i,j} = \frac{w(i,j)}{\sum_k w(i,k)}, \tag{6.15}$$

$$P(a_i|d_j) = DA_{j,i} = \frac{w(i,j)}{\sum_k w(k,i)}. \tag{6.16}$$

This completes the descriptions of networks and random walks. Note that (6.14) implies $\sum_k w(k,j) = 1$. The design of the matrices AD and DA is asymmetric to reflect the asymmetric relationship between authors and documents.

Indeed, it is better for an author to create many good documents; for a document it is better to have better authors, but not necessarily *more* authors.

6.5 Experiments on CiteSeer

For experiments, we use the CiteSeer datasets as introduced in Chapter 4. While performing the ranking on the full data collection is technically feasible, the bias in collection sizes towards certain domains can undermine the fairness of ranking scientists from different areas. Therefore, we start from categorizing the documents into domains. We selected five topics that are well-represented in the database: T6: stochastic and Markov processes, T8: WWW and information retrieval, T19: learning and classification, T36: statistical learning, and T48: data management. All experiments were carried out for each of these five topics.

6.5.1 Author Rankings

To evaluate the co-ranking approach, we perform a ranking of authors in each topic t by the methods listed below:

- **Publication count**, the number of papers (on the topic t) an author has in the document subset;
- **Topic weight**, the sum of topic weights of all documents, produced or co-authored by an author;
- **Number of citations**, the total number of citations to the documents of an author from the other documents on the same topic;

- **PageRank in the social network**, ranking by PageRank on the graph G_A, constructed as outlined in § 6.4;

- **Co-Ranking**, co-ranking authors and documents by the new method.

The parameter values we used in the Co-Ranking framework are $m = 2$, $n = 2$, $k = 1$, $\lambda = 0.2$, $\alpha = 0.1$. For different settings of m, n, k the top 20 authors and papers varied slightly, even less for different α.

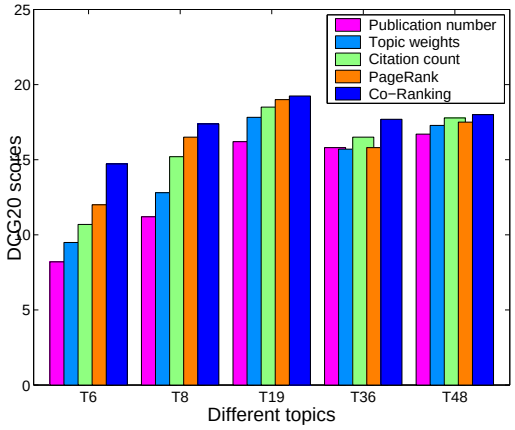


Figure 6.3. DCG_{20} scores for author rankings: number of papers, topic weights, number of citations, PageRank, and Co-Ranking.

We used a well-known metric, the Discounted Cumulated Gain (DCG) [34], in order to compare the five different rankings of authors. Top 20 authors according to each ranking (publication count, etc.) are merged in a single list, shuffled and submitted for judgment. Two human judges, one an author of this paper and the other one from outside, provide feedback. Numerical assessment scores of $0, 1, 2$, and 3 are collected to reflect the judges' opinion with regard to whether an author is ranked top 20 in a certain field, which respectively means *strongly disagree*, *disagree*, *agree*, and *strongly agree*, with the fact that these authors are ranked top 20 in the corresponding field. As suggested, assessments were carried out based on professional achievement of the authors such as winning of prestigious awards,

r	author names	con#	r	p#	r	cite#	r
1	Rakesh Agrawal	171	44	129	32	1915	1
2	Serge Abiteboul	209	12	115	42	1300	3
3	Jennifer Widom	234	5	113	44	1617	2
4	Jiawei Han	271	2	142	22	720	10
5	Hector Garcia-Molina	232	7	169	16	1247	4
6	Ian Foster	142	79	215	12	513	19
7	Azer Bestavro	97	198	174	14	354	42
8	Deborah Estrin	134	100	186	13	471	23
9	Subbarao Kambhampati	118	130	275	8	173	132
10	Michael Stonebraker	59	322	144	21	299	66
11	Christos Faloutsos	218	11	98	58	770	9
12	Moshe Y. Vardi	184	29	148	20	415	30
13	Rajeev Motwani	145	75	127	33	579	15
14	Richard T. Snodgrass	125	115	68	131	330	50
15	Joseph Hellerstein	63	305	75	103	132	208

Table 6.1. Top authors in the topic *data management* when $m = 2$, $n = 2$, $k = 1$. con# is the number of neighbors in the social network; p# is the number of papers; cite# is the number of citations; r denotes the ranks by the corresponding methods.

being a fellowship of ACM/IEEE, etc. The judges' assessment scores are averaged. We observe a high agreement between the two judges.

The DCG_{20} scores obtained are presented in Fig. 6.3. The figure shows five groups of bars corresponding to five topics. This evaluation shows that the new co-ranking method outperforms the other four ranking methods, achieving an average improvement of 27.8%, 19.1%, 10.6%, and 7.7% over rankings by the number of papers, the topic weights, the number of citations, and the PageRank.

We list the top 15 authors ordered by the Co-Ranking scores on the topics *data management* and *learning and classifications* in Table 6.1 and Table 6.2. Along with both tables, the ranks based on simple metrics are also presented. Note that in the top author lists, we observe a mix of famous scientists from different fields. This is due to the imperfect automatic categorization performed by LDA; manual categorization labels can be used instead.

r	author names	con#	r	p#	r	cite#	r
1	Sebastian Thrun	178	6	293	8	782	4
2	Bernd Girod	72	180	217	10	313	33
3	Jurgen Schmidhuber	152	21	160	14	446	18
4	Stephen Muggleton	99	88	45	200	492	11
5	Robert E. Schapire	133	35	67	105	1093	1
6	Avrim Blum	102	82	295	7	239	58
7	Trevor Hastie	68	199	88	52	263	53
8	Rakesh Agrawal	68	197	129	22	843	2
9	Manuela Veloso	155	18	196	11	491	12
10	Thomas G. Dietterich	74	173	53	159	514	8
11	Alex Pentland	126	47	110	36	369	21
12	Michael I. Jordan	172	9	91	50	566	7
13	David J.C. MacKay	22	379	73	91	349	25
14	David Haussler	113	61	65	112	351	24
15	David Heckerman	77	163	56	150	491	14

Table 6.2. Top authors in the topic *learning and classifications* when $m = 2$, $n = 2$, $k = 1$. con# is the number of neighbors in the social network; p# is the number of papers; cite# is the number of citations; r denotes the ranks by the corresponding methods.

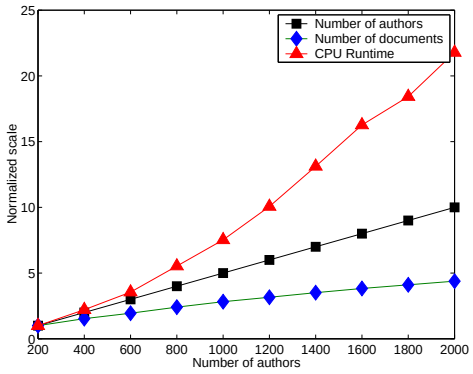


Figure 6.4. Average CPU runtime and number of documents w.r.t. the number of authors for five topics, where $m = 2$, $n = 2$, $k = 1$. Appropriate units have been chosen, so that a single normalized scale can be used. Everything is averaged over five topics.

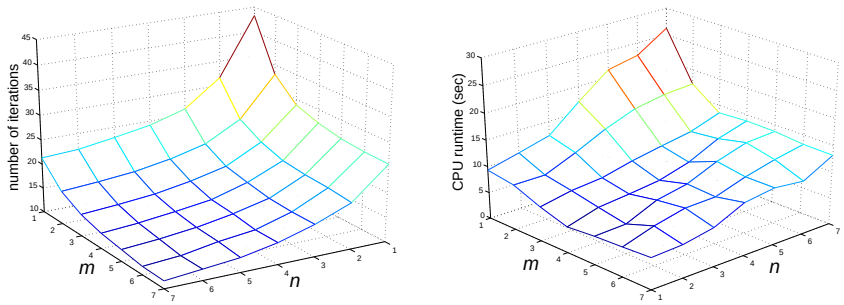
6.5.2 Parameter Effect

We ran Co-Ranking on 50 synthetic datasets with various settings of m, n, k, λ, and α and arrived at the following conclusions: (1) Large λ introduces more mutual dependence of the rankings between authors and documents. In particular, as λ increases, the ranking of authors becomes closer to the ranking by the number of publications; (2) In case of large α such as 0.5, the ranking of authors becomes more uniform, so that the documents of productive authors are neglected, and

also generally benefiting the documents with many authors. Since both effects are undesirable, keep α small; (3) For small m, especially $m = 1$, the weight of edges in G_A is not fully taken into account, but only the local differences in weights matter; (4) Prevent large k. It completely eliminates the effect of authors on documents and vice versa, except for the authorship information: the bipartite random walk forgets everything, as expected from a Markov chain after many steps; (5) For small n, the structure of the citation network is less important, making the Co-Ranking more like a citation counting.

6.5.3 Convergence and Runtime

Finally, we present some observations about the computational complexity: We observed that the algorithm converges faster for larger α. This is expected because a Markov chain takes a shorter time to reach the stationary status if the transition matrix is closer to uniform.



(a) Number of iterations until convergence (b) CPU runtime until convergence

Figure 6.5. Effect of m-n on convergence.

We fix $k = 1$, $\lambda = 0.2$, $\alpha = 0.1$ and vary m and n. Fig. 6.5(a) and Fig. 6.5(b) show the effect of m and n on the number of iterations before convergence and the runtime of the program. It can be seen that for large and increasing m and n the number of iterations decreases slowly. This is because the Intra-class random

walks have enough steps to become nearly stationary before the next Inter-class step.

The computational complexity of Algo. 2 is $O(k \times n_A \times n_D)$. The complexity of Algo. 3 is $O(t \times n_A \times n_D \times (n + m + 2k + 1))$, where n, m, k are parameters and t is the number of steps before convergence. Fig. 6.4 shows the average CPU runtime w.r.t. to the number of authors. The Co-Ranking was implemented in Python and tested on Intel CoreDuo 1.66 GHz, 1G RAM, Windows O.S.

6.6 Summary

This chapter proposes a new link analysis ranking approach for co-ranking authors and documents respectively in their social and citation networks. Starting from the PageRank paradigm as applied to both networks, the new method is based on coupling two random walks into a combined one, presumably exploiting the mutually reinforcing relationship between documents and their authors: good documents are written by reputable authors and vice versa. Experiments on a real world data set suggest that Co-Ranking is more satisfactory than counting the number publications or the total number of citations a given scientist has received. Also, it appears competitive with the PageRank algorithm as applied to the social network only. We did not evaluate the ranking of documents due to the lack of any objective criteria.

Chapter 7

Communities in Heterogeneous Social Networks

7.1 Discovering Communities in Heterogeneous Social Networks

Well known graph-theoretic methods for discovering communities from networks include spectral graph partitioning [58, 13], hierarchical community discovery [79], and clustering based on random walks [28]. Despite the wide range of choices for partitioning homogeneous networks, research on discovering communities from heterogeneous social networks is rather limited [1]. Treating heterogeneous graphs the same as homogeneous ones leads to difficulty in normalization since different edge types may be incomparable [18]. However, observations of real-world networks often indicate diverse network structures, many of which can be modeled as heterogeneous networks of social actors and the other node types such as

[1] Here we define a heterogeneous graph as a graph where there are multiple types of vertices and edges.

documents (e.g. emails, blogs, collaborative publications) or social events. In this chapter, we are particularly interested in communication documents as these data sources represent the most widely available sources of information regarding social networks.

In this chapter [85], we addresses the community discovery problem in a temporal heterogeneous social network consisting of authors, document content, and the venues in which the documents are published, all of which are constructed by different types of social actions over time. We propose a new framework that addresses the two main challenges in this new problem: (a) handling of the heterogenous network and (b) incorporation of the temporal aspect of the data. For (a), we formulate community discovery in a heterogeneous social network (the social network is a network of authors, words, and publication venues) as a tripartite graph partitioning problem. A normalized cut (NCut) cost function is defined over the partitions. We show that partitioning a tripartite graph is a quadratically constrained quadratic programming (QCQP) problem. For (b), we introduce a new method for incorporating prior knowledge, such as prior community membership, into the current discovery process. The discovery of temporal communities is then performed by threading communities discovered at consecutive time periods using the output from the previous period as prior knowledge. At each time period, the constrained graph partitioning method is able to capture both the current graph topology and historical information regarding the vertex membership. This problem is efficiently solved using a proposed fractional orthogonal iteration algorithm (instead of pursuing the semidefinite program (SDP) as in [18], which is computationally intractable). We evaluate the proposed approach on synthetic datasets with various settings in order to explore the properties of the new algorithm. A great improvement in clustering precision is observed. In addition, we show the

results of applying this method to a sample dataset obtained from CiteSeer.

Let us now consider consider social networks of researchers in the context of their collaborations on published work. The data in focus includes the co-occurrences of authors with documents, documents with words, and documents with venues. All data are associated with time stamps, which are the years of publication. The data is collapsed on documents yielding the (1) author-word co-occurrences and (2) word-venue co-occurrences, over a certain amount of time. Thus, within each time period there are two correlated bipartite graphs, $G(V_X, V_Y, W_{XY})$ and $G(V_Y, V_Z, W_{YZ})$, where V_X is the author set, V_Y is the word set, V_Z is the venue set, W_{XY} is the bipartite edge weights between V_X and V_Y, and W_{YZ} is the edge weights for V_Y and V_Z. Here $G(V_X, V_Y, W_{XY})$ and $G(V_Y, V_Z, W_{YZ})$ share the vertex set V_Y. We refer to $G(V_X, V_Y, W_{XY})$ and $G(V_Y, V_Z, W_{YZ})$ as a *bipartite graph couple*, which can be seen as a generalized social network of authors, words, and documents. Two static communities in such a social network are illustrated in Fig. 7.1, where a *static community*, at a specific time, is defined on the snapshot below:

Definition 6. *A static community in a static social network is a composite of closely associated authors, words, and venues. Entities within the same community are closely related while entities in different communities are loosely associated if at all.*

Over the entire time period, the underlying social network structure is dynamic. Accordingly, instead of observing a single static social network over the entire data set, a sequence of static social networks of various structures is generated, with consecutive snapshots showing significant overlap of entities. The definition of a temporal community thus embodies the temporal aspects of the network:

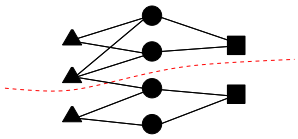


Figure 7.1. A static social network. triangles denote the authors, circles denote the words, and rectangles denote the venues. The graph between authors and words is inferred from the document authorship and the graph between words and venues is based on the publication records of documents. Two static communities are separated by the dashed line.

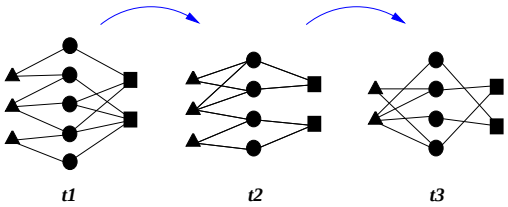


Figure 7.2. A dynamic social network. Three snapshots are included in the network with various numbers of authors (denoted by triangles), venues (denoted by rectangles), and words (denoted by circles).

Definition 7. *A temporal community in a dynamic social network is a threaded sequence of static communities at each time period. In a temporal community, the structure of a static community at a specific time depends on the previous* N *temporal networks, where* N *is a parameter that can be defined as the* order *of the temporal community.*

A dynamic social network is illustrated in Fig. 7.2. Three snapshots are included, each having different network structures. It can be seen that each static social network is a bipartite graph couple.

The goal is to cluster authors, words and venues given their changing relationships over time. The desired number of communities k is assumed and given as a parameter.

7.2 Graph Partitioning

We start from the discovery of static communities from a static social network. Suppose there are two bipartite graphs, $G_{XY} = G(V_X, V_Y, W_{XY})$ and $G_{YZ} = G(V_Y, V_Z, W_{YZ})$, where V_X is the author set, V_Y is the word set, and V_Z is the venue set; $W_{XY} \in \mathbb{R}^{+n_X \times n_Y}$ is a matrix where the elements represent the number of co-occurrences of an author and a word; and $W_{YZ} \in \mathbb{R}^{+n_Y \times n_Z}$ is a matrix whose elements are the number of co-occurrences of a word and a venue (n_X, n_Y, n_Z are the size of V_X, V_Y, V_Z). Note G_{XY} and G_{YZ} share V_Y.

Consider a community with two types of vertices from V_X and V_Y, say which are represented by two subsets S_i^X and S_j^Y. The weight of the community is:

$$W(S_i^X, S_j^Y) = \sum_{u \in S_i^X, v \in S_j^Y} w_{u,v}. \tag{7.1}$$

Likewise, the weight between a subset of vertices and the vertex set that they are from are from is denoted as $W(S_i^X, Y)$ or $W(X, S_i^Y)$. Given k as the desired number of communities, the cost function of *Normalized Cut (NC)* is defined as [82]:

$$J_2 = \sum_{i=1}^{k} \frac{W(S_i^X, \overline{S_i^Y}) + W(\overline{S_i^X}, S_i^Y)}{W(S_i^X, Y) + W(X, S_i^Y)} \tag{7.2}$$

where S_i^X, S_i^Y are the subsets of V_X and V_Y in community i; $\overline{S_i^X}$, $\overline{S_i^Y}$ are the subsets of V_X and V_Y not in community i. The sets $\{S_i^X\}_{i=1}^k$, $\{S_i^Y\}_{i=1}^k$ that minimize the cost J_2 belong to the discovered k communities.

Now define several indicator matrices. Let $X = [X_1, ..., X_k]$, where X_i is an indicator vector of whether the corresponding element belongs to community i, with 1 indicating so or 0 otherwise. Similarly, we have $Y = [Y_1, ..., Y_k]$ and $Z = [Z_1, ..., Z_k]$.

Define D_{XY} and D_{YZ} as diagonal matrices where the elements are the sums of rows in W_{XY} and W_{YZ}. Define D_{YX} and D_{ZY} as diagonal matrices where elements are the sums of columns in W_{XY} and W_{YZ}. After some manipulations, we can rewrite Eq. 7.2 as:

$$J_2 = \sum_{i=1}^{k} \frac{X_i^T D_{XY} X_i + Y_i^T D_{YX} Y_i - 2X_i^T W_{XY} Y_i}{X_i^T D_{XY} X_i + Y_i^T D_{YX} Y_i} \tag{7.3}$$

$$= k - \sum_{i=1}^{k} \frac{2X_i^T W_{XY} Y_i}{X_i^T D_{XY} X_i + Y_i^T D_{YX} Y_i}. \tag{7.4}$$

The problem of searching for best solutions to the above minimization problem has been shown to be NP-hard. In order to obtain a solution efficiently, prior work relaxes the elements in X_i and Y_i to real values instead of the discrete set $\{0, 1\}$ [82]. Extending this work, we further scale X_i and Y_i to the denominator. In particular, assuming $X_i = D_{XY}^{-\frac{1}{2}} \hat{X}_i$ and $Y_i = D_{YX}^{-\frac{1}{2}} \hat{Y}_i$, we let $\hat{X}_i^T \hat{X}_i = \hat{Y}_i^T \hat{Y}_i = 1$. Thus, J_2 becomes:

$$J_2 = k - \sum_{i=1}^{k} \hat{X}_i^T D_{XY}^{-\frac{1}{2}} W_{XY} D_{YX}^{-\frac{1}{2}} \hat{Y}_i. \tag{7.5}$$

Here $D_{XY}^{-\frac{1}{2}} W_{XY} D_{YX}^{-\frac{1}{2}}$ is in fact the normalized edge weight matrix. The minimization of cost function J_2 is carried out over $\hat{X}_i$ and $\hat{Y}_i$ for $i = 1, ..., k$. Traditionally, the different minimizers are assumed to be orthogonal to each other [81], i.e. $\hat{X}^T \hat{X} = \mathbf{I}$ and $\hat{Y}^T \hat{Y} = \mathbf{I}$. We impose the same constraint on our solution.

Now let us generalize the cost function for a bipartite graph couple, where we have an additional set of vertices Z and the edge weights with Y in W_{YZ}. Similarly, define $\hat{X} = [\hat{X}_1, ..., \hat{X}_k]$, $\hat{Y} = [\hat{Y}_1, ..., \hat{Y}_k]$ and $\hat{Z} = [\hat{Z}_1, ..., \hat{Z}_k]$, where $\hat{X}^T \hat{X} = \hat{Y}^T \hat{Y} = \hat{Z}^T \hat{Z} = \mathbf{I}$. Let J_{XY} be the cost function of partitioning graph

G_{XY} and J_{YZ} be the cost function for G_{YZ}. We introduce a parameter λ to balance the costs on both graphs. Based on Eq. 7.5, we define the new cost function J_3 on the bipartite graph couple as:

$$\begin{aligned} J_3 = & \quad \lambda J_{XY} + (1-\lambda) J_{YZ} \\ = & \quad k - \lambda \textstyle\sum_{i=1}^{k} \hat{X}_i^T D_{XY}^{-\frac{1}{2}} W_{XY} D_{YX}^{-\frac{1}{2}} \hat{Y}_i \\ & -(1-\lambda) \textstyle\sum_{i=1}^{k} \hat{Y}_i^T D_{YZ}^{-\frac{1}{2}} W_{YZ} D_{ZY}^{-\frac{1}{2}} \hat{Z}_i \end{aligned} \tag{7.6}$$

where the second and third terms represent the cost functions on G_{XY} and G_{YZ}.

Thus, the minimization of cost function J_3 over $\hat{X}$, $\hat{Y}$, and $\hat{Z}$ becomes a maximization of the negative term in J_3:

$$\begin{aligned} & \min_{\hat{X},\hat{Y},\hat{Z}} J_3 \\ \equiv \quad & \max_{\hat{X},\hat{Y},\hat{Z}} \lambda \textstyle\sum_{i=1}^{k} \hat{X}_i^T D_{XY}^{-\frac{1}{2}} W_{XY} D_{YX}^{-\frac{1}{2}} \hat{Y}_i \\ & +(1-\lambda) \textstyle\sum_{i=1}^{k} \hat{Y}_i^T D_{YZ}^{-\frac{1}{2}} W_{YZ} D_{ZY}^{-\frac{1}{2}} \hat{Z}_i \end{aligned} \tag{7.7}$$

subject to

$$\hat{X} = [\hat{X}_1, ...\hat{X}_k], \quad \hat{X}^T \hat{X} = \mathbf{I}; \tag{7.8}$$

$$\hat{Y} = [\hat{Y}_1, ..., \hat{Y}_k], \quad \hat{Y}^T \hat{Y} = \mathbf{I}; \tag{7.9}$$

$$\hat{Z} = [\hat{Z}_1, ..., \hat{Z}_k], \quad \hat{Z}^T \hat{Z} = \mathbf{I}; \tag{7.10}$$

where $\mathbf{I}$ is an identity matrix.

Now let us rewrite the problem in matrix form. Define $\widehat{W}_{XY} = D_{XY}^{-\frac{1}{2}} W_{XY} D_{YX}^{-\frac{1}{2}}$

and $\widehat{W}_{YZ} = D_{YZ}^{-\frac{1}{2}} W_{YZ} D_{ZY}^{-\frac{1}{2}}$. Define $U = [U_1, ..., U_k]$, where $U_i = [\hat{X}_i^T, \hat{Y}_i^T, \hat{Z}_i^T]^T$; Let there be a matrix M such that:

$$M = \begin{bmatrix} 0 & \lambda\widehat{W}_{XY} & 0 \\ \lambda\widehat{W}_{XY}^T & 0 & (1-\lambda)\widehat{W}_{YZ} \\ 0 & (1-\lambda)\widehat{W}_{YZ}^T & 0 \end{bmatrix}. \tag{7.11}$$

It is easy to verify that the cost function in Eq. 7.7 is $\frac{1}{2}\sum_{i=1}^{k} U_i^T M U_i$. The problem thus becomes to minimize the trace of the matrix (The trace of a square matrix is defined as the sum of the diagonal elements):

$$\max_{U} \operatorname{tr}(U^T M U) \tag{7.12}$$

subject to

$$U = [\hat{X}^T, \hat{Y}^T, \hat{Z}^T]^T \tag{7.13}$$

$$\hat{X}, \hat{Y}, \hat{Z} \text{ satisfy Eq. 7.8 - Eq. 7.10} \tag{7.14}$$

Here the optimization problem is a quadratically constrained quadratic programming problem [4]. Note that Eq. 7.8 - Eq. 7.10 is not equivalent to $U^T U = \mathbf{I}$. Constraints on U apply to its segments (i.e. $\hat{X}, \hat{Y}, \hat{Z}$) respectively.

7.3 Partitioning Temporal Graphs

The problem of community discovery has been formulated as a graph partitioning issue. Next we present a constrained graph partitioning method that threads community discovery across consecutive time periods.

7.3.1 Graphs with consistent vertices

We first focus on the case where graphs have consistent vertices. For each time period, we have M^t and U^t as described in Eq. 7.11 and Eq. 7.8 - Eq. 7.10, where $t = 1, ..., T$ are the time stamps and U^t contains the community membership of authors, words, and venues. Assume that the graphs have consistent vertices; thus, all U^t have the same dimensions. Now, let us define a cost function on the difference between U^{t_i} and U^{t_j} for an arbitrary time stamp pair t_i, t_j, denoted $c(U^{t_i}, U^{t_j})$. The discovery of community structure at time t seeks to minimize the weighted sum of the distances between the current and previous community membership back to $t - \delta$:

$$\min_{U^t} \sum_{\pi=t-\delta}^{t-1} \alpha_\pi c(U^\pi, U^t) \tag{7.15}$$

where α_π is the weight on the distance to the community membership at π time periods ago. The weights on different historic periods are prescribed parameters. Hereafter, for simplicity, we concern ourselves only with the first-order dependency case where $\delta = 1$ and $\alpha_\pi = 1$.

A key issue is the design of the cost function $c(\dot{U}, U)$. Here we let the cost function be the negative cosine distance between two subspaces. Suppose $\dot{X}$, $\dot{Y}$, and $\dot{Z}$ are the reference subspaces of X, Y, Z. We know that $\|\dot{X}\|^2 = \|\dot{Y}\|^2 = \|\dot{Z}\|^2 = 1$. Thus, the square of cosine distances between the desired subspace and the reference subspace are respectively $\|\dot{X}^T\hat{X}\|^2$, $\|\dot{Y}^T\hat{Y}\|^2$, and $\|\dot{Z}^T\hat{Z}\|^2$. In addition, we know that the cosine distances are within $[0, 1]$. We thus seek to maximize the cosine distances to minimize the cost imposed by the distance from the reference subspaces. In particular, define the cost function $c(\dot{U}, U)$:

$$-c(\dot{U}, U) = \alpha\|\dot{X}^T\hat{X}\|^2 + \beta\|\dot{Y}^T\hat{Y}\|^2 + \gamma\|\dot{Z}^T\hat{Z}\|^2 \tag{7.16}$$

$$= \alpha.\mathrm{tr}(\hat{X}^T\dot{X}\dot{X}^T\hat{X}) + \beta.\mathrm{tr}(\hat{Y}^T\dot{Y}\dot{Y}^T\hat{Y}) + \gamma.\mathrm{tr}(\hat{Z}^T\dot{Z}\dot{Z}^T\hat{Z}) \tag{7.17}$$

$$= \mathrm{tr}(U^T\dot{U}\dot{U}^TU), \tag{7.18}$$

where $\dot{U} = [\sqrt{\alpha}\dot{X}^T, \sqrt{\beta}\dot{Y}^T, \sqrt{\gamma}\dot{Z}^T]^T$, α, β and γ are the weight parameters of the membership differences in authors, words, and venues. Here, notice that $\dot{U}\dot{U}^T$ is essentially the covariance matrix between the vertices in the reference time period. Since we have assumed consistent vertices in the graphs across different time periods, we essentially minimize the conflicts between the discovered U and the referenced covariance.

7.3.2 Graphs with evolving vertices

Now we generalize the previous section to graphs with evolving vertices. In practice, some vertices may disappear and other new ones may show up, thus the $\dot{U}$ obtained from previous period can disagree with the dimensionality of the U in the current time period. We introduce an additional step to adapt $\dot{U}$ to address this issue.

First, some vertices from previous time period may disappear. Since each vertex corresponds to a row in $\dot{U}$, we can delete these rows from $\dot{U}$, forming a matrix with the same number of columns but a smaller number of rows, $\dot{U}'$. We call the first step shrink(). Thus we have:

$$\dot{U}' = \mathrm{shrink}(\dot{U}) = [\dot{X}'^T, \dot{Y}'^T, \dot{Z}'^T]^T \tag{7.19}$$

where $\dot{U}'$ is the adapted subspace with disappeared vertices removed. $\dot{X}'$, $\dot{Y}'$, and $\dot{Z}'$ still correspond to the remaining $\dot{X}$, $\dot{Y}$, and $\dot{Z}$. Second, some new vertices may appear in the current time period. In this case, we have no prior knowledge about their membership. Therefore, we require zero co-variances of them with others, corresponding to zeros in the corresponding rows. Name this second step expand():

$$\dot{U}'' = \text{expand}(\dot{U}') = [\dot{X}'^T, 0, \dot{Y}'^T, 0, \dot{Z}'^T, 0]^T, \tag{7.20}$$

where $[\dot{X}'^T, 0]^T$, $[\dot{Y}'^T, 0]^T$, and $[\dot{Z}'^T, 0]^T$ respectively correspond to the newly observed X^t, Y^t, and Y^t; all $0's$ has the appropriate number of rows and k columns. We then arrive at the new reference covariance matrix $c(\dot{U}, U)$ as:

$$\dot{C} = \dot{U}''\dot{U}''^T, \tag{7.21}$$

which leads to the new cost function $c(\dot{U}, U)$ on U and reference $\dot{U}$ defined as: $-c(\dot{U}, U) = \text{tr}(U^T\dot{C}U)$, where $\dot{C}$ is given in Eq. 7.19 - Eq. 7.21.

Note the handling of new vertices here. Since the reference $\dot{U}''$ still has values in the rows corresponding to the old vertices, these previously observed vertices will be made consistent with the previous period. On the other hand, the new vertices will not be affected by such prior knowledge of the previous time period because of the zeros in the rest of $\dot{U}''$. To see this, note that the $\text{tr}(U^T\dot{C}U)$ has zero diagonals in the indices of those newly observed vertices regardless of the values of U in the corresponding rows.

Given the above, the combined community discovery problem at each time period is written as:

$$\begin{aligned}\min_U \widetilde{J} &= \min_U J_3 + c(\dot{U}, U) \\ &\equiv \max_U \mathrm{tr}(U^T M U) + \mathrm{tr}(U^T \dot{C} U) \\ &= \max_U \mathrm{tr}(U^T (M + \dot{C}) U) \end{aligned} \tag{7.22}$$

subject to

$$U = [\hat{X}^T, \hat{Y}^T, \hat{Z}^T]^T \tag{7.23}$$

$$\hat{X}, \hat{Y}, \hat{Z} \text{ satisfy Eq. 7.8 - Eq. 7.10} \tag{7.24}$$

$$M \text{ is given in Eq. 7.11} \tag{7.25}$$

$$\dot{U} = [\sqrt{\alpha}\dot{X}^T, \sqrt{\beta}\dot{Y}^T, \sqrt{\gamma}\dot{Z}^T]^T \tag{7.26}$$

$$\dot{C} \text{ is given by Eq. 7.19 - Eq. 7.21.} \tag{7.27}$$

where α, β and γ are the weight parameters for the membership differences in authors, words, and venues; $\dot{U}$ is the reference membership matrix. We arrive at a quadratically constrained quadratic programming problem.

7.4 Efficient Approximate Solutions

This section gives an efficient algorithm to solve the problem formulated in Eq. 7.22 - Eq. 7.27. It can be seen that Eq. 7.22 has a quadratic cost function of the matrix U. Here Eq. 7.22 can be rewritten as:

$$\max_U \sum_i U_i^T (M + \dot{C}) U_i) \tag{7.28}$$

where the U_i's are column vectors in U. We can see that this is a sum of a sequence of quadratic functions each corresponding to a subset of constraints in Eq. 7.23 - Eq. 7.27. Thus we have a sequence of quadratically constrained quadratic programming (QCQP) sub-problems. Note these QCQP problems are not isolated because their solution vectors U_i are required to be orthogonal.

For each QCQP sub-problem alone, there exists a standard solution using semidefinite programming (SDP) [4]. For example, a related work [18] studied the binary clustering case and proposed an approximate solution using an interior-point method. However, we note that our optimizer here is a matrix ($U = [X^T, Y^T, Z^T]^T$) instead of a single vector. Thus, to apply SDP on each column vector and combine them together is overly complex. Nevertheless, one might construct a very high-dimensional vector by columns of U and still translate the problem into SDP, but difficulty still arises from the exploding dimensionality of the problem. Recall that $U \in \mathbb{R}^{(n_X+n_Y+n_Z)\times k}$, where n_X, n_Y, and n_Z are the numbers of authors, words, and venues. The translated SDP problem will have a $k(n_X+n_Y+n_Z)$-dimensional vector as the minimizer (with a $k(n_X+n_Y+n_Z) \times k(n_X+n_Y+n_Z)$ semidefinite matrix of constraints), which can easily surpass the capacity of most SDP solvers.

Instead, we propose an efficient algorithm that searches for approximate solutions. The new algorithm is based on algorithms for eigenvectors. First we are aware that the Eq. 7.22, without constraints, reaches the maximum when U contains the first k eigenvectors of the symmetric matrix $A = M + \dot{U}\dot{U}^T$. This is a standard result from matrix theory [23]. In addition, we have $\forall U \in \{U|U^TU = \mathbf{I}\}$, $\quad U^TAU \leq \lambda_1 + ... + \lambda_k$, where $\lambda_1, ...\lambda_k$ are the first k largest eigenvalues of A. Second, we seek to preserve the constraints as much as possible while maximizing $\widetilde{J}$. We modify the *orthogonal iteration* method which is used to calculate the eigenvector space without constraints. The idea is to incorporate the constraints

into the classical method. The new algorithm, *fractional orthogonal iteration*, is presented below:

Algorithm 4 fractional orthogonal iteration

1: $\dot{U} = [\sqrt{\alpha}\dot{X}^T, \sqrt{\beta}\dot{Y}^T, \sqrt{\gamma}\dot{Z}^T]^T$;
2: $\dot{U}' \leftarrow \text{shrink}(\dot{U})$ as in Eq. 7.19
3: $\dot{U}'' \leftarrow \text{expend}(\dot{U}')$ as in Eq. 7.20
4: $\dot{C} \leftarrow \dot{U''}\dot{U''}^T$
5: $A = M + \dot{C}$
6: $[U, D] \leftarrow \text{eig}(A, k)$
7: **for** $i = 1, 2, 3, ...$ **do**
8: $\begin{bmatrix} \hat{X} \\ \hat{Y} \\ \hat{Z} \end{bmatrix} \leftarrow A \times U$
9: $Q_X R_X \leftarrow \hat{X}$ // QR factorization
10: $Q_Y R_Y \leftarrow \hat{Y}$ // QR factorization
11: $Q_Z R_Z \leftarrow \hat{Z}$ // QR factorization
12: $U \leftarrow \begin{bmatrix} Q_X \\ Q_Y \\ Q_Z \end{bmatrix}$
13: **end for**
14: $U \leftarrow M \times U$
15: run k-means on U to obtain the desired partitioning, where each row in U denotes the original data object of the same index.

Here $\text{eig}(A, k)$ calculates the k-dimensional eigenvector space of A without constraints. This is the initial value for the subsequent orthogonal iteration. In the algorithm, step 9 - step 11 produce the normalized $\hat{X}$, $\hat{Y}$ and $\hat{Z}$ as specified in the constraints. Step 8 performs the power iteration as in the original *orthogonal iteration* method for calculating eigenvectors. Up to step 15, the algorithm has projected the original bipartite graph couple into an approximate k-dimensional eigenspace. The distribution of the points in the new space preserves the distribution of objects at the current time period, in addition to imposing the community membership from the last period. Then we run k-means to cluster the heterogeneous objects as current communities.

7.5 Experiments on CiteSeer

A synthetic data generator was created to test the proposed method in various conditions, including different edge density-to-noise ratio, various proportions of $X/Y/Z$, different settings of λ, and different numbers of clusters (k). Two connected graphs G_{XY} and G_{YZ} are generated for the prescribed K and sizes of X, Y, and Z. All clusters contain the same number of entities with specified proportions of $X, Y, andZ$. The densities of all the clusters are the same, but the edge weights vary randomly. Random noise is added to the graph and density is determined by the given noise-signal ratio parameter (nsr). Setting $nsr = 1$ yields a random graph without cluster structures. Presumably, the community structures in the graph XY diminish as the noise-signal ratio (nsr) grows. Low nsr indicates that graph partitioning will be easier. The table below includes a complete list of parameters and their meanings.

abbr.	**usages**
fsi	fractional subspace iteration
par	partitioning static graphs using fsi
t-par	partitioning temporal graphs using fsi
k	number of clusters
$density$	the edge density of the graph clusters
nsr	noise-signal ratio, noise density / cluster density
x/z	the size of X / the size of Z
λ	the weight parameter in Eq. 7.11

7.5.1 Precision w.r.t. graph conditions

First, we focus on the clustering precision w.r.t. different densities and nsr for $k = 2$. As illustrated in Fig. 7.4(a) we present four values of nsr, indicating increasing difficulty for partitioning. In general, we observe that the precision decreases as nsr grows. In each subfigure, we can see that the clustering precision

grows quickly as the graph clusters become denser. On graphs with less noise, the precision grows faster than on the highly noisy graphs. Comparatively, the proposed fsi algorithm outperforms the traditional *subspace iteration* algorithm (without consideration of constraints) for different nsr. We are able to see that the special scaling introduced in fsi improves the *subspace iteration*. The fsi usually outperforms *subspace iteration* by a greater amount in the more difficult situations (large nsr). All precisions are measured using k-means with random initial medians. For each case, the k-means is repeated for 10 times and the averages are presented.

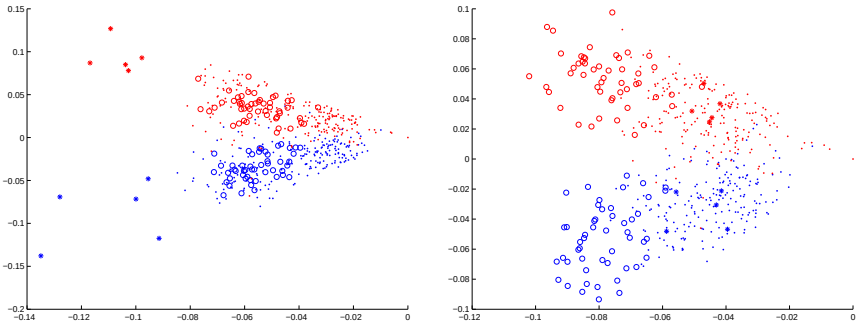
Second, we perform fsi on different settings of x/z ratios for a fixed setting of λ. In real world datasets, the sizes X and Z are usually not balanced. We compare fsi with *subspace iteration* for imbalanced data against fsi by varying the x/z ratio. Fig. 7.4(b) shows different settings of x/z for different densities. Recall that a large x/z indicates that the size of X is much greater than that of Z. Without loss of generality, we assume $x/z \geq 1$. We can see that for sparse graphs (small density) the fsi outperforms *subspace iteration* greatly (illustrated in the subfigure on the bottom). In simple cases (large density), the fsi generally outperforms *subspace iteration* for small x/z; however, fsi underperform *subspace iteration* slightly for small x/z on dense graphs. Note that real-world graphs are usually very sparse; thus, fsi could be favored on many real-world datasets.

7.5.2 Precision w.r.t. parameter settings

Here we test different settings of parameters and their impact on community discovery precision. A set of experiments were run with different settings of λ in different x/z ratios. The results illustrated in Fig. 7.4(c) show that the favorable

λ are different when x/z varies. When the X outnumbers Z by a large margin, a greater value in λ is favored; similarly, small λ performs better when there are few X entities compared with Z. This suggests that graphs with more edges deserve a larger weight in the cost evaluation.

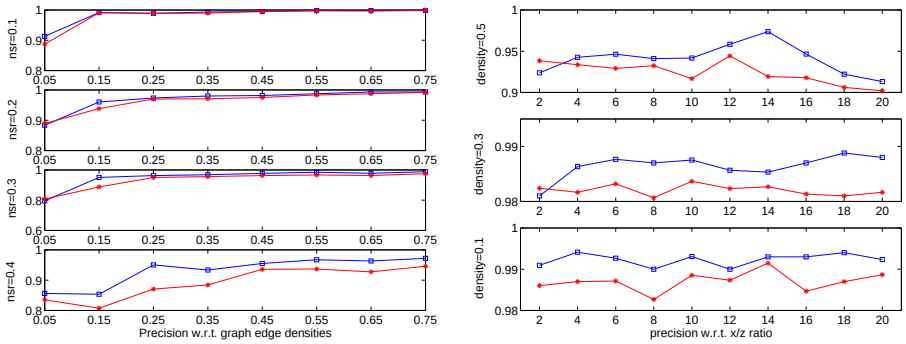
In order to better visualize the effect of λ with different x/z, we present the subspace scatter plots for different λ. Note that here $|X|/|Y|/|Z| = 50 : 200 : 5$. The X outnumber Z, indicated by a great x/z ratio. In Fig. 7.3, we show precisions for $\lambda = 0.5, 0.8$. Here $k = 2$ so we have 2-D subspaces. In this case, a large λ better scales the edges in YZ and thus better embeds Z into the subspace.



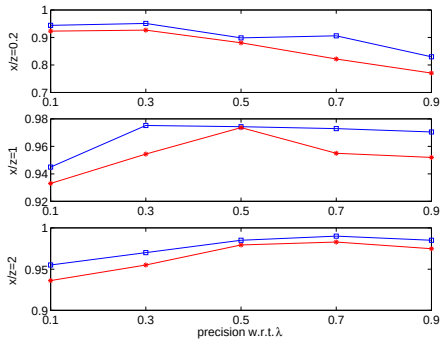
(a) $\lambda = 0.5$ (b) $\lambda = 0.8$

Figure 7.3. Subspace plots for different λ when $|X|/|Y|/|Z| = 50{:}200{:}5$. Different clusters are colored differently. Entities of different types have different markers (circles, dots, stars for X, Y, Z). Here $k = 2$.

Finally, we compare fsi with *subspace iteration* on different numbers of clusters, at different *subspace iteration*. We can see that, for large density, fsi still outperforms *subspace iteration* for large numbers of clusters. However, the *subspace iteration* seems to work better than fsi for the case of many clusters on sparse graphs. In practice, we can substitute fsi by recursively performing k-means using $k = 2$ for bi-partitioning the graph, similar to [82].



(a) Edge density and nsr ratio (b) x/z ratio and edge density



(c) λ and x/z ratio

Figure 7.4. Clustering precision w.r.t. different graph conditions

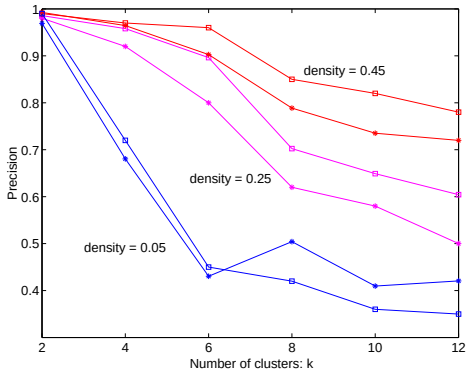
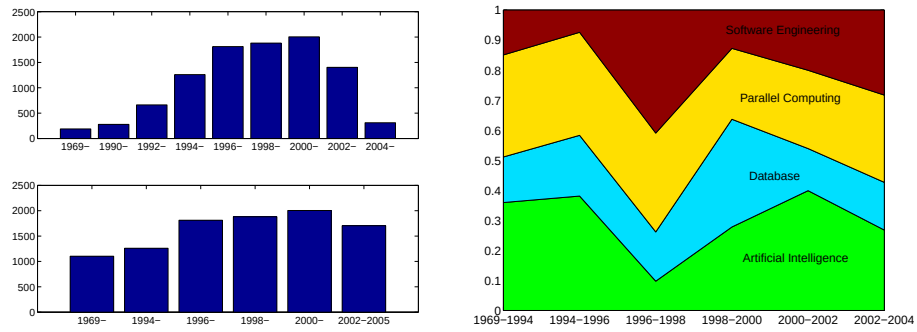


Figure 7.5. The precision w.r.t. k, at different densities: $density = 0.05$, $density = 0.25$, $density = 0.45$.

7.5.3 Real-world dataset and experiments

A real-world data set for further experimentation was generated by sampling documents from CiteSeer using combined document meta-data from CiteSeer, the



(a) Publication number (b) Community size

Figure 7.6. Amount of publications and community size over time. Two different grouping methods are shown, one by uniform grouping of years and the other by proportional grouping.

ACM Guide, and the DBLP for enhanced data accuracy and coverage. A set of venues was chosen from five fields in computer science (software engineering, data mining, artificial intelligence, databases, and distributed computing), such that data from each field included at least 2000 distinct author names and at least ten years of significant coverage. All documents contained in CiteSeer from each venue were obtained and the top 100 key phrases were extracted from each document using the KEA key phrase extraction algorithm [17]. The final dataset contained 12,677 authors and 45,295 key phrases from 30 distinct venues ranging over the years 1969 to 2004. The total number of documents used was 13,310.

Experiments on this data set began by empirically determining the appropriate number of clusters. While it is an open problem to determine the dimension of a subspace for embedding a graph, we used simple heuristics. We ran the proposed community discovery algorithm (fsi) with different k and chose the k corresponding to the smallest $\widetilde{J}$ (or the greatest $\gamma = \mathrm{tr}(U^T(M + \dot{C})U)$) as in Eq. 7.22. We observed that the γ initially grows dramatically as k increases, but grows at a much lower rate as k becomes large. Thus we chose the smallest k that gave the near maximum γ. This gave us $k = 4$.

Then we ran the temporal community discovery (t-par) algorithm with $k = 4$ with various settings of λ. For screening the results, we judge the quality of discovery by examining the grouping of venues since their number is small. We observed that the quality is better for greater λ, supporting the results from synthetic datasets that suggest λ should be set proportionally to $|X|/|Z|$. Here we set $\lambda = 0.6$.

We observe that the resulting communities of authors, venues, and words are well grouped. Four communities are discovered for *artificial intelligence and machine learning*, *database and data mining*, *parallel and distributed computing*, and *software engineering*. We present two discovered communities and their authors in Table 7.1 and Table 7.2. In our experiments, we used the discovered venue set to manually produce community labels. The key phrases (ranked by frequency) were considered as the summarization of a community.

Table 7.1 includes a subset of authors discovered in the *artificial intelligence and machine learning* community over six time periods. For presentation, we rank the authors by their number of chapters within the corresponding periods. We can observe that the community memberships of authors are relatively stable but change over time. In the experiments, we observed that the top authors remained as the "core" members of the corresponding community and there were many more authors who had joined and left from the communities during these six time periods. Similarly, authors from the *database and data mining* community are presented in Table 7.2. The top venues, which due to space limit we cannot show in the table, are for JMLR, PAMI, ICML, AAAI/IAAI, UAI, IJCAI, JAIR, and *PODS, SIGMOD, VLDB, SIGMOD Record, ICDE* for Table 7.2.

We used the discovered clusters of words as the description for the corresponding communities. Summarizations of two communities are presented in Table 7.3

1969-94	1994-96	1996-98	1998-2000	2000-02	2002-04
M I Jordan	M I Jordan	W L Johnson	S Thrun	D Koller	A Blum
L P Kaelbling	L P Kaelbling	N Friedman	C Boutilier	A W Moore	S Thrun
J Y Halpern	Z Ghahramani	D Koller	T Sandholm	M I Jordan	S Zilberstein
S P Singh	S P Singh	R E Schapire	D Koller	M L Littman	P Stone
Z Ghahramani	M K Warmuth	Y Singer	N Friedman	S Thrun	J Langford
M K Warmuth	T G Dietterich	R Dechter	Y Singer	D Schuurmans	T Eiter
T G Dietterich	T Dean	T J Sejnowski	A Mccallum	J Shawe-taylor	P Domingos
T Dean	Y Bengio	H S Seung	L P Kaelbling	S P Singh	A K Jain
Y Bengio	P Smets	D Poole	S P Singh	N Friedman	S Baker
P Smets	W Maass	M I Jordan	P R Cohen	N Cristianini	S Chawla
W Maass	V Tresp	N Tishby	R Khardon	A Mccallum	R Dechter
V Tresp	D Weinshall	R Greiner	M J Kearns	P Domingos	C Guestrin
D Weinshall	D Geiger	Y Mansour	K Nigam	Y Bengio	C Boutilier
D Geiger	S Kambhampati	M K Warmuth	N Cristianini	D Freitag	M J Kearns
D Poole	A Saffiotti	Y Freund	J Shawe-taylor	A Y Ng	T Lukasiewicz
R E Schapire	R E Schapire	D P Helmbold	C Baral	M K Warmuth	A Demiriz
S Kambhampati	D S Nau	C Boutilier	A W Moore	G E Hinton	S P Singh
C Baumlckstroumlm	H A Simon	M L Littman	D Fox	N Tishby	D Koller
F Bacchus	F Bacchus	P Dayan	D Roth	A J Smola	D Schuurmans
A Saffiotti	D Poole	A J Grove	M P Wellman	G Raumltsch	S Prabhakar

Table 7.1. Machine learning community during 1969-2004 in a CiteSeer sample.

1969-94	1994-96	1996-98	1998-2000	2000-02	2002-04
M Yannakakis	M Yannakakis	R Hull	A Mendelson	G Gottlob	S Abiteboul
V Vianu	V Vianu	A Mendelzon	J Paredaens	V Vianu	L Popa
A Gupta	J Y Halpern	Z M Zsoyoglu	C Papadimitriou	H Garcia-molina	T Milo
Garciacute	Garciacute	H Garcia-molina	H Garcia-molina	J Widom	P G Kolaitis
J Widom	J Widom	D Suciu	S Abiteboul	A Y Halevy	P S Yu
J F Naughton	H Garcia-molina	A Silberschatz	D Florescu	C Faloutsos	F Neven
H Garcia-molina	J F Naughton	A Y Levy	A Y Levy	D Suciu	C Beeri
C Faloutsos	C Faloutsos	L Libkin	R Motwani	D Gunopulos	R Rastogi
A Kemper	J Hammer	G Moerkotte	L Lakshmanan	S Lee	J Han
K Ramamritham	A Biliris	S Seshadri	T Milo	J Han	D Srivastava
G Moerkotte	K Ramamritham	S Abiteboul	S Cluet	W Fan	M N Garofalakis
I S Mumick	A Kemper	J Widom	J Han	R Rastogi	J Widom
A Biliris	C Baumlckstroumlm	R Agrawal	D Suciu	C S Jensen	A Y Halevy
J Hammer	G Moerkotte	R Ramakrishnan	J S Vitter	H V Jagadish	C Li
M Chen	I S Mumick	S Sudarshan	R Rastogi	D Kossmann	J Madhavan
P S Yu	K Lin	K Ramamritham	G D Giacomo	D Srivastava	W Fan
T Milo	S Berson	A Kemper	C S Jensen	K Chakrabarti	B Babcock
D Suciu	D Suciu	D Florescu	D Srivastava	S Muthukrishnan	C Y Chan
J Han	D Kossmann	P Atzeni	O Shehory	D S Weld	C Koch
K Lin	C A Knoblock	M Benedikt	M Lenzerini	G D Giacomo	J Gehrke

Table 7.2. Database community during 1969-2004 in a CiteSeer sample.

and Table 7.4. Words are ranked by their frequency of occurrence within the data. Those words that did not occur in the previous period are highlighted. Over the six time periods, we can see the emergence of new words, which presumably indicate the evolution of interests of the community.

Finally, we show the changes in communities' sizes over time in Fig. 7.6(b). The size of a community is measured by the number of distinct authors discovered within a particular time period. The sizes of the four communities are scaled to sum up to one.

years	words
1994-96	learning model training probability value image set action input points output variables goal point values search policy agent function selection examples error units distance knowledge classification representation recognition region test
1996-98	learning **state** model image value training probability **network** set values variables **class** error points input point action **vector** representation **sequence** agent search **distribution** recognition units **random** output classification case robot
1998-00	learning model state value training set image probability values action points policy error search point sequence **actions** noise function knowledge distribution classification robot parameters estimate **text** optimal estimation accuracy representation
2000-02	learning model training set error image probability **matrix** point sequence distribution **kernel** classification **random** features state estimation function representation **input** accuracy **strategy vector** text **prediction** parameters bound **approach selection**
2002-04	learning model set probability **policy points** training sequence image **variables** optimal **algorithm** function matrix **search** point error **distance erent** random bound classification max **robot** estimate representation case **expected** distribution vector

Table 7.3. Frequent words in the machine learning community during 1994-2004 in a CiteSeer sample.

years	words
1994-96	query data database queries object path event cost type user execution objects table class transaction local rules server client join name formula update rule attribute attributes view pages plan read
1996-98	query data queries database object cost **tree information** view user attributes pages objects rules join plan table update transaction type attribute **constraints** page access server disk **requests real-time label** client
1998-00	query data queries user information database pages rules constraints plan **path** attributes attribute view join formula table sources update objects **request** strategy **documents level instance items** rule **web** spatial **application**
2000-02	data query queries **points** information path **cost** xml database attributes **values** pages **tree** constraints table join plan **type** objects page **distance management example document** attribute update labeled items documents web
2002-04	data query **node** queries xml path values tree database attributes table document join name plan **service cache** objects **return selection** constraints type **patterns** label mapping attribute tuples index items **root**

Table 7.4. Most frequent words in the database community during 1994-2004 in a CiteSeer sample.

7.6 Summary

This chapter addresses an emerging problem of temporal community discovery from communication documents, by which one can observe the temporal trends in community membership over time. The problem is formulated as a tripartite graph partitioning problem with prior knowledge available of entity covariances. Temporal communities are discovered by threading the partitioning of graphs in different time periods, using a new constrained partitioning algorithm. Evaluation of the new algorithm is carried out on several synthetic datasets and a real-world dataset prepared from CiteSeer. Experiments on synthetic data reveal the properties of the new algorithm in various graph conditions. Experiments on CiteSeer data show the effectiveness of the proposed approach in author community discovery and community summarization.

Chapter 8

Recommendations using Heterogeneous Social Networks

8.1 Recommender Systems for Networked Data

Recommender systems continue to play important and new roles in business on the World Wide Web [73, 31, 74]. The most popular method adopted by contemporary recommender systems is *Collaborative Filtering* (CF), where the core assumption is that *similar* users on *similar* items express *similar* interests. The heart of memory-based CF methods is the measurement of similarity: either the similarity of users (a.k.a user-based CF) or the similarity of items (a.k.a items-based CF) or a hybrid of both. The user-based CF computes the similarity among users, usually based on user profiles or past behavior [31], and seeks consistency in the predictions among similar users. But it is known that user-based CF often suffers from the *data sparsity problem* because most of the user-item ratings are missing in practice. The item-based CF, on the other hand, allows input of additional item-wise information and is also capable of capturing the interactions

among them [73]. This is a major advantage of item-based CF when it comes to dealing with items that are networked, which are usually encountered on the Web. For example, consider the problem of document recommendation in a digital library such as the CiteSeer (http://citeseer.ist.psu.edu). As illustrated in Fig. 8.1, let documents be denoted as vertices on a directed graph where the edges indicate their citations. The similarity among documents can be measured by their cocitations (cociting the same documents or being cocited by others) [1]. In this case, document B and C are similar because they are cocited by E.

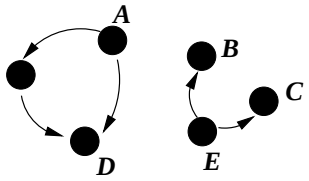
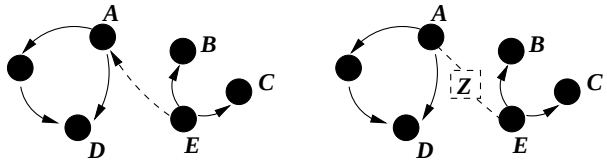


Figure 8.1. An example of citation graph.

Working with networked items for CF is of recent interest. A recent work approaches this problem by leveraging the item similarities measured on an item graph [73]. They model item similarities by an undirected graph and, given several vertices labeled interesting, perform label propagation to rank the remaining vertices. The key issue in label propagation on graphs is the measurement of vertex similarity, where related work simply borrows the recent results of the Laplacian on directed graphs [7] and semi-supervised learning of graphs [86]. Nevertheless, using a single graph Laplacian to measure the item similarity can overfit in practice, especially for data on the Web, where the graphs tend to be noisy and sparse in nature. For example, if we revisit Fig. 8.1 and consider two quite common scenarios, as illustrated in Fig. 8.2, it is easy to see why measuring item similarities based on a single graph can sometimes cause problems. The first case is called

[1] In fact, the term *cocitation* in this chapter refers to two concepts in information sciences: *bibliographic coupling* and *cocitation*.

missing citations, where for some reason a citation is missing (or equivalently is added) from the citation graph. Then the similarity between A and B (or C) will not be encoded in the graph Laplacian. The second case, called *same authors*, shows that if A and E are authored by the same researcher Z, using the citation graph only will not capture the similarity between D and B, which presumably should be similar because they are both cited by the author Z.



(a) Missing citations (b) Same authors

Figure 8.2. Two common scenarios: *missing citations* and *same authors*, which give rise to the problems for measuring item similarities based on a single citation graph.

Needless to say, the cases presented above are just two of the many problems caused by the *noise* and *sparsity* of the citation graph. Noise in a citation graph is a result of a missing citation link or an incorrect one. Fortunately, real world data can usually be described by different semantics or can be associated with other data. In the focus of this chapter, where only relational data is concerned, we work with several graphs regarding the same set of items. For example, in the case of document recommendation, and in addition to the document citation graph, we also have a document-author bipartite graph that encodes the authorship, and a document-venue bipartite graph that indicates where the documents were published. Such relationship between documents and other objects can be used to improve the measurement of document similarity. The idea of this work is to combine multiple graphs to calculate the similarities among items. The items can be the full vertex set of a graph (as in the citation graph) or can be a subset

of a graph (as in document-author bipartite graph) [2]. By doing so, we let data from different semantics regarding the same item set complement each other[3].

In this chapter [94], we implement a model of learning from multiple graphs by seeking a single low-dimensional embedding of items that captures the relative similarities among them. Based on the obtained item embedding, we perform label propagation, giving rise to a new recommendation framework using semi-supervised learning on graphs. In addition, as introduced in the reference but not here [94], we address the scalability issue and propose an incremental version of our new method, where an approximate embedding is calculated only for the new items. The new methods are evaluated on two real world datasets prepared from CiteSeer. We compare the new batch method with a baseline modified from a recent semi-supervised learning algorithm on a directed graph and a basic user-based CF method using Singular Value Decomposition (SVD). Also, we compare the new incremental method with the new batch method in terms of recommendation quality and efficiency. We observe significant quality improvement in our batch method and significant efficiency improvement with tolerable quality loss for our incremental method.

8.2 Recommendation by Label Propagation

Label propagation is one typical kind of *transductive learning* in the semi-supervised learning category where the goal is to estimate the labels of unlabeled data using other partially labeled data and their similarities. Label propagation on a network has many different applications. For example, recent work shows

[2]Note the difference between this work and the related work [84] where multiple graphs with the *same set of vertices* are combined.

[3]Note the difference with another related work [74] is that we are not working with the user-rating matrix but rather starting from a directed graph of items.

that trust between individuals can be propagated on social networks [26] and user interests can be propagated on item graphs for recommendations [73].

In this work, we focus on using label propagation for document recommendation in digital libraries. Let the document set be $\mathcal{D}$, where $|\mathcal{D}|$ is the number of documents. Suppose we are given the document citation graph $G_D = (V_D, E_D)$, which is an unweighted directed graph. Suppose the pair-wise similarities among the documents are described by the matrix $S \in \mathbb{R}^{|\mathcal{D}|\times|\mathcal{D}|}$ measured based on G_D. A few documents have been labeled "interesting" while the remaining are not, denoted by positive and zero values in the label vector y. The goal is to find the score vector $f \in \mathbb{R}^{|\mathcal{D}|}$ where each element corresponds to the propagated interests. Then document recommendation can be performed by ranking the documents by their interest scores. A recent approach addressed the graph label propagation problem by minimizing the regularization loss below [86]:

$$\Omega(f) \equiv f^T(I - S)f + \mu\|f - y\|^2, \tag{8.1}$$

where $\mu > 0$ is the regularization parameter. The first term is the cost function for the *smoothness constraint*, which prefers small differences in labels between nearby points; the second term is the *fitting constraint* that measures the difference of f from given data label y. Setting the $\partial\Omega(y)/\partial f = 0$, we can see that the solution f^* is essentially the solution to the linear equation:

$$(I - \alpha S)f^* = (1 - \alpha)y, \tag{8.2}$$

where $\alpha = 1/(1 + \mu)$. One solution to the above is given in a related work using a

power method [86]:

$$f^{t+1} \leftarrow \alpha S f^t + (1-\alpha) y \tag{8.3}$$

where f^0 is the random guess and $f^* = f^\infty$ is the solution. Here, notice that $\mathcal{L} = (I - \alpha S)$ is essentially a variant Laplacian on this graph using S as the adjacency matrix; and $\mathcal{K} = (I - \alpha S)^{-1} = \mathcal{L}^{-1}$ is the graph diffusion kernel. Thus, one essentially applies $f^* = (1-\alpha)\mathcal{L}^{-1} y$ (or $f^* = (1-\alpha)\mathcal{K} y$)to rank documents for recommendation.

Now the interesting question is how to calculate S (or equivalently the kernel $\mathcal{K}$) among the set $\mathcal{D}$. However, there has been limited amount of work on obtaining S. For graph data, recent work borrows the results from spectral graph theory [6, 7], where the similarity measures on both undirected and directed graphs have been given. For undirected graph, S_u is simply the normalized adjacency matrix:

$$S_u = \Pi^{-1/2} W \Pi^{-1/2} \tag{8.4}$$

where Π is a diagonal matrix such that $We = \Pi e$ and e is an all-one column vector. For directed graph, where the adjacency matrix is first normalized as a random walk transition matrix P $(= \Pi^{-1} W)$, the similarity measure S_d is calculated as:

$$S_d = \frac{\Phi^{1/2} P \Phi^{-1/2} + \Phi^{-1/2} P^T \Phi^{1/2}}{2} \tag{8.5}$$

where Φ is a diagonal matrix where each diagonal contains the stationary probability on the corresponding vertex [4].

Note that the similarity measures given above are derived from a single graph on

[4] In practice when some nodes have no outgoing or incoming edges, the probability distribution over nodes can incorporate certain randomness so that P denotes an ergodic Markov chain.

$\mathcal{D}$. However, many real world data can be described by multiple graphs, including those within $\mathcal{D}$ and between $\mathcal{D}$ and another set. Such information is of more importance to combine especially when a single view of the data is sparse or even incomplete. In the following, we introduce a new way to integrate three general types of graphs. Instead of estimating S directly, we seek to learn a low-dimensional latent linear space.

8.3 Learning Multiple Graphs

The immediate goal of this section is to determine the relative positions of all documents in a k-dimensional latent semantic space, say $X \in \mathbb{R}^{|\mathcal{D}|\times k}$, which will combine the social inferences in document citations, authorship and venues. In the sequel, we assume k is a prescribed parameter which we do not seek to determine automatically. Note a contribution of this work is the different strategies used for different graphs based on their characteristics, which are described in the following subsections.

We begin by a formulation of our problem. Let $\mathcal{D}$, $\mathcal{A}$, $\mathcal{V}$ be the sets of documents, authors and venues and $|\mathcal{D}|$, $|\mathcal{A}|$, $|\mathcal{V}|$ be their sizes. We have three graphs, one directed graph G_D on $\mathcal{D}$; one bipartite graph G_{DA} between $\mathcal{D}$ and $\mathcal{A}$; and one bipartite graph G_{DV} between $\mathcal{D}$ and $\mathcal{V}$, which describe the relationship among documents, between documents and authors, and between documents and venues. Let the adjacency matrices of G_D, G_{DA}, G_{DV} be D, A and V. We assume all relationships in question are described by nonnegative values. For example, G_D can be considered as to describe the citation relationship among $\mathcal{D}$ and $D_{i,j} = 1$ if document d_i cites d_j ($D_{i,j} = 0$ if otherwise); G_A can be considered as the authorship relationship (an author composes a document) or the citation relationship (an

author cites a document) between $\mathcal{D}$ and $\mathcal{A}$.

8.3.1 Learning from Citation Matrix: D

In this section, we relate the document embedding X to the citation matrix D, which is the adjacency matrix of the directed graph G_D.

The citation matrix D include two kinds of document co-occurrences: cociting and being cocited. A cociting relationship among a set of documents means that they all cite a same document; A cocited relation refer to that several documents are cited together by an another document. In many related work (e.g. [86]) on directed graphs, these two kinds of document co-occurrences are used to infer the similarity among documents. Probably the most well recognized way to represent the similarities among the nodes of a graph is associated with the graph Laplacian [7], say $\mathcal{L} \in \mathbb{R}^{|\mathcal{D}|\times|\mathcal{D}|}$, which is defined as:

$$\mathcal{L} = I - \alpha S_d, \tag{8.6}$$

where S_d is the similarity matrix on directed graphs as measured in Eq. 8.5; $\alpha \in (0, 1)$ is a parameter for the Laplacian to be invertible; I is an identity matrix. Note that S is symmetric and positive-semidefinite. In practice, different weights can be assigned to similarities measured from cociting and cocited relations in Eq. 8.5 which now assumes an equal importance of both parts.

Next we give the method to learn from G_D.

Objective function: Suppose we have a document embedding $X = [\mathbf{x}_1, ...\mathbf{x}_k]$ where $\mathbf{x}_i$ contains the distribution of values of all documents on the i-th dimension of a k-dimensional latent space. The overall "lack-of-smoothness" of the distribu-

tion of these vectors w.r.t. to the Laplacian $\mathcal{L}$ can be measured as

$$\Omega(X) = \sum_{1 \le i \le k} \mathbf{x}_i^T \mathcal{L} \mathbf{x}_i = \mathrm{Tr}(X^T \mathcal{L} X), \tag{8.7}$$

where $X = [\mathbf{x}_1, ...\mathbf{x}_k]$. Here we seek to minimize the overall "lack-of-smoothness" so that the relative positions of documents in X will reflect the similarity in S_d.

Constraint: In addition to the objective function of X, we enforce a constraint on X so as to avoid getting a trivial solution (Note that $X = 0$ minimizes Eq. 8.7 if there is no constraint on X). We choose to use the newly proposed log-determinant heuristic on $X^T X$, a.k.a the log-det heuristic, denoted by $\log|X^T X|$ [16]. It has been shown that the $\log|Y|$ is a smooth approximation for the rank of Y if Y is a positive semidefinite matrix. It is obvious the gram matrix $X^T X$ is positive semidefinite. Thus, when we maximize $\log|X^T X|$, we effectively maximize the rank of X, which is at most k. Another way to understand $\log|X^T X|$ is to note that $|X^T X| = \prod_i \lambda_i(X^T X) = \prod_i \sigma_i(X)^2$, where $\lambda_i(Y)$ is the i-th eigen-value of Y and $\sigma_i(X)$ is the i-th singular value of X. Therefore, a full-ranked X is preferred when $\log|X^T X|$ is maximized. For more reasons on using the log-det heuristic, refer to the *Comments* below and [16].

Using the log-det heuristic, we arrive at the combined optimization problem:

$$\min_X \left\{ \mathrm{Tr}(X^T \mathcal{L} X) - \log|X^T X| \right\} \tag{8.8}$$

where $\mathrm{Tr}(A)$ is the trace function defined as the sum of diagonal elements of A. It has been shown that $\max\{\log|X^T X|\}$ (or equivalently $\min\{-\log|X^T X|\}$) is a convex problem [16]. So Eq. 8.8 is still a convex problem.

Comments: First, it is interesting to notice that we did not use the tradi-

tional constraint on X (such as the orthonormal constraint of the subspace used in PCA [81]). The reason of choosing log-det heuristic in our case is because that (1) the orthonormal constraint is non-convex; (2) the orthonormal constraint cannot be solved by gradient-based methods and thus cannot be efficiently solved and cannot be easily combined with the other two factorizations in the following sections; (3) the log-det, $\log|X^TX|$, has a small problem scale ($k \times k$) and can be solved effectively by gradient-based methods. Second, note a key difference of this work from related work on link matrix factorization (e.g. [95]) is that we seek to determine X to comply with the graph Laplacian (not to factorize the link matrix) which gives us a convex problem that is global optimal.

8.3.2 Learning from Author Matrix: A

Here, we show how to learn from an author matrix, A, which is the adjacency matrix of the bipartite graph, G_{DA}, that captures the relationship between $\mathcal{D}$ and $\mathcal{A}$. We can use G_{DA} to encode two kinds of information between authors and documents, one being the authorship and the other being the author-citation-ship. To encode authorship, we let $A \in \mathbb{I}^{|\mathcal{D}|\times|\mathcal{A}|}$, where $A_{i,j}$ indicates whether the i-th paper is authored by the j-th author; To encode author-citation-ship, we assume $A \in \mathbb{R}^{|\mathcal{D}|\times|\mathcal{A}|}$, where $A_{i,j}$ can be the number of times that document i is cited by author j (or the logarithm of the citation count for rescaling).

We can consider both kinds of author-document relationship using matrix factorization, where authors in both cases are considered social features of documents, inferring similarities between documents. The basic intuition is that the document related to a same set of authors should be relatively close in the latent space X. The inference of this intuition to citation recommendation is that the other work

of an author will be recommended given a reader is interested in several work by similar authors.

Given the authorship matrix $A \in \mathbb{R}^{|\mathcal{D}|\times|\mathcal{A}|}$, we want to use X to approximate it. Let the authors be described by an author profile matrix $W \in \mathbb{R}^{|\mathcal{A}|\times k}$. We can approximate A by XW^T as:

$$\min_{X,W} \|A - XW^T\|_F^2 + \lambda_1 \|W\|_F^2, \tag{8.9}$$

where X and W are the minimizers. To prevent overfitting, the second term is used, where λ_1 is the parameter. Note that later we will combine Eq. 8.8 and Eq. 8.9; So we do not show the constraint on $\|X\|_F^2$ here. It is worth mentioning that the idea of using two latent semantic spaces to approximate a co-occurrence matrix is similar to that used in document content analysis (e.g. the LSA [10]).

8.3.3 Learning from Venue Matrix: V

In the above, we have given the method for learning a representation of $\mathcal{D}$ from a directed citation graph G_D and an undirected bipartite graph G_{DA}. In this section, we are given an additional piece of categorical information, which can be described by the bipartite venue graph G_{DV}, where one set of nodes are the documents from $\mathcal{D}$ and the other set are the venues from $\mathcal{V}$.

Similar to A, we have the venue matrix $V \in \mathbb{I}^{|\mathcal{D}|\times|\mathcal{V}|}$, where $V_{i,j}$ denotes whether document i is in venue j. However, a key difference here is that each row in V has at most one nonzero element because one document can proceed in at most one venue. Although we could as well employ XW^T to approximate V (as in Sec. 8.3.2), we will show that the special property of V can help us cancel the variable matrix W, and thus reducing the optimization problem size for better

efficiency. Accordingly, we follow a similar but different approach. In particular, let us consider to use V to predict the X via linear combinations. Suppose we have W_2 as the coefficient, we seek to minimize the following:

$$\min_{X,W_2} \|VW_2^T - X\|_F^2. \tag{8.10}$$

One can understand Eq. 8.10 in this way: Here each column of W_2 can be considered as a cluster center of the corresponding class (i.e., the venues). Then solving Eq. 8.10 in fact simultaneously (1) pushes the representation of documents close to their respective class centers; and (2) optimizes the centers to be close to their members.

Next, we cancel W_2 using the unique property of our venue matrix V. Setting the derivative to be zero, we have $0 = \partial\|VW_2^T - X\|_F^2/\partial W_2 = 2(V^TVW_2 - V^TX)$, suggesting that $W_2 = (V^TV)^{-1}V^TX$. Note that V^TV is diagonal matrix and is thus invertible. Plug in W_2 back to Eq. 8.10. We arrive at the optimization where W_2 is canceled:

$$\min_{X} \|V(V^TV)^{-1}V^TX - X\|_F^2, \tag{8.11}$$

where $(V^TV)^{-1}V^T$ is the pseudo inverse of V. Here since V^TV is $|\mathcal{V}| \times |\mathcal{V}|$ diagonal matrix, its inverse can be computed in $|\mathcal{V}|$ flops. Meanwhile, $V(V^TV)^{-1}V^T$ is block diagonal where each block denotes a complete graph among all documents within the same venue. Note that Eq. 8.9 cannot be handled in the same way because $(A^TA)^{-1}$ is a dense matrix, resulting in a $|\mathcal{D}| \times |\mathcal{D}|$ dense matrix of $A(A^TA)^{-1}A^T$, which in practice raises scalability issues.

8.3.4 Learning Document Embedding

We have arrived at a combined optimization formulation given the above sub-problems. We will combine Eq. 8.8, Eq. 8.9 and Eq. 8.10 in a unified optimization framework. Define the new objective $J(X, W)$ as a function of X, W. We have an optimization below to learn the document embedding matrix X:

$$\begin{aligned} J(X,W) = \quad & (\mathrm{Tr}(X^T\mathcal{L}X) - \log|X^TX| \\ & +\alpha\|A - XW^T\|_F^2 + \lambda\|W\|_F^2 \\ & +\beta\|V(V^TV)^{-1}V^TX - X\|_F^2) \end{aligned} \tag{8.12}$$

where λ is the weight of regularization on W; α is the weight for learning from A; β is the weight for learning from V.

The optimization illustrated above can be solved using standard Conjugate Gradient (CG) method, where the key step is the evaluation of objective function and the gradient. Below, we show the gradients for the combined optimization in Eq. 8.12:

$$\begin{aligned} \frac{\partial J}{\partial X} = \quad & 2\mathcal{L}X - 2X(X^TX)^{-1} \\ & +2\alpha(XW^TW - AW) + \\ & +2\beta(VV^\dagger - I)^T(VV^\dagger - I)X \end{aligned} \tag{8.13}$$

$$\frac{\partial J}{\partial W} = \quad 2\alpha(WX^TX - A^TX) + 2\lambda W \tag{8.14}$$

where $V^\dagger = (V^TV)^{-1}V^T$ is the pseudo inverse of V. When searching for the solutions, we vectorize the gradients of X, W into a long vector. In implementa-

tion, different calculation order of matrix product leads to very different efficiency. For example, it is much more efficient to calculate $(VV^\dagger - I)^T(VV^\dagger - I)X$ as $(V^\dagger)^T V^T V V^\dagger X - 2VV^\dagger X + X$ because V and $V^\dagger$ are very sparse.

After X is calculated, we can use linear model in the recommendation, i.e. $f^* = X(X^T X)^{-1} X^T y$, which has been shown to arrive at the same solution of the power method in Eq. 8.3 [86]. By doing so, we can obtain efficiency advantage over the power method as in Eq. 8.3.

8.4 Experiments on CiteSeer

We continue to use the CiteSeer datasets as introduced in previous chapters (Ch. 4). In particular, two datasets were prepared with different sizes. The first dataset, referred to as DS_1, has 400 authors, 9,197 documents, 50 venues, and 19,844 citations; The second dataset, referred to as DS_2, which is larger in size, has 800 authors, 15,073 documents, 100 venues, and 38,614 citations.

8.4.1 Evaluation Metrics

The performance of recommendation can be measured by a wide range of metrics, including user experience studies and click-through monitoring. For experimental purpose, we will evaluate the proposed method against citation records by cross-validation. In particular, we randomly remove t documents, use the remaining documents as the seeds, perform recommendations, and judge the recommendation quality by examining how well these removed documents can be retrieved. As suggested by real user usage patterns, we are only interested in the top recommended documents. Quantitatively, we define the recommendation *precision* (p) as the percentage of the top recommended documents that are in fact from the true

citation set. The *recall* (r) is defined as the percentage of true citations that are really recommended in the top m documents. The *F-score*, which combines *precision* and *recall* is defined as $f = (1+\delta^2)rp/(r+\delta^2 p)$, where $\delta \in [0, \infty)$ determines how relatively important we want the recall to be (Here we use $\delta = 1$, i.e. F-1 score, as in many related work.). We have introduced a parameter in evaluation, m, which is the number of top documents we evaluate the f-score at.

8.4.2 Recommendation Quality

This section introduces the experiments on recommendation quality. We compare the recommendation by our algorithm with two other baselines: one based on Laplacian on directed graphs [7] and label propagation using graph Laplacian [86] (named as *Lap*) and the other based on Singular Vector Decomposition of the author matrix (named as *SVD*). We chose to compare with the *Lap* method to see whether the fusion of different graphs can effectively produce additional information than the original graph citation graph; We chose the *SVD* on author matrix as another baseline because we would like compare our method against the traditional CF method on the additional graph information (as one can argue that the significant improvement of the new method is purely due to the use of the additional information).

	f \ m	**m=t**	**m=5**	**m=10**
$DS1$	f(lap)	0.013	0.048	0.192
	f(svd)	0.035	0.086	0.138
	f(new)	**0.108**	**0.242**	**0.325**
$DS2$	f(lap)	0.011	0.046	0.156
	f(svd)	0.027	0.072	0.109
	f(new)	**0.083**	**0.158**	**0.229**

Table 8.1. The f-score calculated on different numbers of top documents, m.

Table 8.1 and Table 8.2 list the f-scores of three different methods (our new

	f \ t	**t=1**	**t=2**	**t=3**	**t=4**
$DS1$	f(lap)	0.041	0.048	0.075	0.086
	f(svd)	0.062	0.088	0.099	0.103
	f(new)	**0.197**	**0.242**	**0.248**	**0.252**
$DS2$	f(lap)	0.037	0.047	0.068	0.077
	f(svd)	0.049	0.072	0.082	0.086
	f(new)	**0.121**	**0.158**	**0.181**	**0.182**

Table 8.2. The f-score w.r.t. different numbers of left-out documents, t.

method with *Lap* and *SVD*) on two datasets (DS_1 and DS_2). Table 8.1 for different number of top documents evaluated on (denoted by m). We are able to see that the new method outperforms both *Lap* and SVD significantly on both datasets in different settings of parameters. In general, the new method are $3-5$ times better in f-score than *Lap* and 2.5 times better than SVD. The *Lap* method under-performs SVD on the very top documents but beats it if evaluated on more top documents. In addition, we notice that the f-scores get better in general as we look at more top documents. Also, the f-scores on the smaller dataset DS_1 are generally higher than those on the larger dataset DS_2. Here, we can see that the recommendation quality can be significantly improved by using the author matrix as the additional information. Note that the different information, when used individually, such as the *Lap* on the citation graph or the SVD on the author graph, can be not as good. However, if the multiple information are combined, the performance is greatly improved[5].

[5]In our experiments, additionally, we work with different methods of formulating the author matrix, A, for example, using the number of citations from authors to documents in A. The experiments show that using the citation-ship in A can be even better. Due to space limit, here we present the experiments with authorship in A only.

8.4.3 Parameter Effect

The effect of parameters for the new method is experimented in this section. We experiment with different settings of dimensionality, or k, and weights on authors and venues, or α and β. In Table 8.3, we show the f-scores for different k's. It occurs that the f-scores become higher for greater k. We believe this is because the higher dimensional space can better captures the similarities in the original citation graphs. However, on the other hand, we observe that it takes longer training time for greater k. Seeking k thus become a trade-off between quality and efficiency. In our experiments, we chose $k = 100$ as greater k do not seem to give much better results. The CPU time for training at different k's are illustrated in Table 8.4.

f \ k	**k=50**	**k=100**	**k=150**	**k=200**
$DS1$	0.203	0.242	0.249	0.262
$DS2$	0.095	0.158	0.181	0.197

Table 8.3. The f-score w.r.t. different setting of dimensionality, k.

	t(lap)	**t(new)**			
time \ k		**k=50**	**k=100**	**k=150**	**k=200**
$DS1$	694s	**440s**	**502s**	**558s**	**621s**
$DS2$	940s	**638s**	**743s**	**820s**	**910s**

Table 8.4. The CPU time for recommendations w.r.t. different dimensionalities.

Fig. 8.3 illustrates the f-scores for different settings of α and β, which are respectively the weights on authors and venues. Here α and β are obtained by testing on a held-out set. We determine which of the two components obtains greater improvement if incorporated, search for the best parameter for this component, fix it, and then search for the best parameter for the other component. In our experiments, we observe that adding the author component tends to improve the recommendation quality better so we first tune α, which yields different f-scores, as shown by the blue curve in Fig. 8.3. Then we fix the $\alpha = 0.1$ and tune β, arriving

at the best f-score at $\beta = 0.05$.

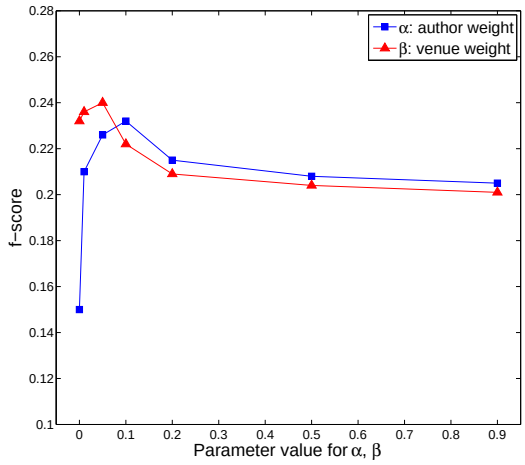


Figure 8.3. f-scores for different settings of weights on the authors, α, and on the venues, β. The α is tuned first for $\beta = 0$; Then β is tuned for the fixed best $\alpha = 0.1$.

8.5 Summary

We address the item-based collaborative filtering problem for items that are networked. We propose a new method for combining multiple graphs in order to measure item similarities. In particular, the new method seeks a single low-dimensional embedding of items that captures the relative similarities among them in the latent space. We formulate this as an optimization problem, where the learning of three general types of graphs are formulated as three sub-problems, each using a factorization strategy tailored to the unique characteristics of the graph type. Based on the obtained item embedding, a new recommendation framework is developed using semi-supervised learning on graphs. In addition, we address the scalability and propose an incremental version of the new method. The new methods are evaluated on two real world datasets prepared from CiteSeer. Experiments have

demonstrated significant quality improvement for our batch method and significant efficiency improvement with tolerable quality loss for our incremental method.

Chapter 9

Conclusions

This research consists of a series of new methods for data mining of social documents and social networks. In general, the steps are proposing new content models for user generated social documents, presenting the connection between social content and social actions, and proposing new techniques for data mining heterogeneous social networks constructed by various social actions.

In particular, contributions of this dissertation include: (1) New content models for emails and social annotations, estimated using Gibbs sampling and improved by entropy filtering [90, 93]. The models have been applied for semantic community discovery and language modeling-based information retrieval. (2) Exploration of the connection between content evolution and social actions for hierarchical clustering of topics in document analysis [87]. The topic dynamics in social document corpora are modeled as a Markov chain and the dependency among these topics are estimated using social interactions of different orders. The topic clustering is done by a Markov metastable state detection. (3) New methods for ranking and co-ranking in heterogeneous social networks constructed by multiple kinds of social actions [91, 88]. The ranking of social actors occurs through modeling the network

flow by learning the implicit preferences of social actors' actions. The co-ranking of authors and documents is achieved by coupling two random walks into a combined one, presumably exploiting the mutually reinforcing relationship between documents and their authors. (4) New methods for discovering temporal communities from communication documents, by which one can observe the temporal trends in community membership over time [85]. The problem is formulated as a tripartite graph partitioning problem with entity covariance, prior knowledge available. Temporal communities are discovered by threading the partitioning of graphs in different time periods, using a new, constrained partitioning algorithm. (5) A new framework for combining multiple graphs to measure document similarities, applied for digital libraries' document recommendations [94]. This framework seeks documents' single, low-dimensional embedding that captures the relative similarities in the latent space. The formulation of this is as an optimization problem, where the learning of three general types of graphs constitute three sub-problems, each using a factorization strategy tailored to the unique characteristics of the graph type. Based on the obtained item embedding, a new recommendation framework is developed using semi-supervised learning with graphs.

In addition, due to physical constraints other research has mention but details are absent. These considered omissions include new methods for learning user click-throughs in Web searches [83], clustering results comparisons [89], and discovering organizational structures from corporate email corpora [92]. Also, the incremental method for learning from multiple graphs is an intentional omission in Chapter 8 [94].

Considerations for future research include: (1) Exploration of the connection between social actions and the topology and evolution of social networks (the connection between social actions and social content is addressed in this dissertation);

(2) Consideration and measurement of negative social actions and social ties (this study and traditional literature, positively weighted social ties leaving open the question of usefulness of introducing negative ties to explain some observations). (3) Information flow in a social network. This study attempts to measure the flow of information in social networks by learning from heterogeneous social actions. An interesting aspect would be to explore whether or not information can flow over social networks and how that can be captured in the social content. More importantly, would be to investigate how such an information flow correlates with social actions. (4) Prediction of social actions. This dissertation presents results for predicting document citations for recommendations. A useful endeavor would be explore the predictability of other kinds of social actions, such as collaborations or acknowledgments. (5) Focus on scalability. Many approaches proposed in this study deal with sparse matrices. More efficient solutions and experiments on very large datasets are recommended to deal with the ever-growing Web data.

Bibliography

[1] A. Agarwal, S. Chakrabarti, and S. Aggarwal. Learning to rank networked entities. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 14–23. ACM Press, 2006.

[2] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, 284(5):34–43, 2001.

[3] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[4] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[5] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *WWW7: Proceedings of the seventh international conference on World Wide Web 7*, pages 107–117. Elsevier Science Publishers B. V., 1998.

[6] F. Chung. *Spectral Graph Theory*. American Mathematical Society, 1997.

[7] F. Chung. Laplacians and the cheeger inequality for directed graphs. *Annals of Combinatorics*, 9, 2005.

[8] A. Clauset, M. Newman, and C. Moore. Finding community structure in very large networks. In *Physics Review*, 2004.

[9] J. Cohen. A coefficient for agreement for nominal scales. *Education and Psychological Measurement*, pages 37–46, 1960.

[10] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

[11] P. Deuflhard, W. Huisinga, A. Fischer, and C. Schutte. Identification of almost invariant aggregates in reversible nearly uncoupled Markov chains. *Linear Algebra and its Applications*, 315(1–3):39–59, 2000.

[12] I. S. Dhillon, S. Mallela, and D. S. Modha. Information-theoretic co-clustering. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 89–98, New York, NY, USA, 2003. ACM Press.

[13] C. Ding. A tutorial on spectral clustering. In *Proc. of the 25th International Conference on Machine Learning*, July 2004.

[14] C. Ding, X. He, H. Zha, M. Gu, and H. D. Simon. A min-max cut algorithm for graph partitioning and data clustering. In *ICDM '01: Proceedings of International Conference on Data Mining*, pages 107–114, 2001.

[15] P. Domingos and M. Richardson. Mining the network value of customers. In *KDD '01: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 57–66. ACM Press, 2001.

[16] M. Fazel, H. Hindi, and S. P. Boyd. Log-det heuristic for matrix rank minimization with applications to hankel and euclidean distance matrices. In *Proceedings of American Control Conference*, 2003.

[17] E. Frank, G. W. Paynter, I. H. Witten, C. Gutwin, and C. G. Nevill-Manning. Domain-specific keyphrase extraction. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, pages 668–673, 1999.

[18] B. Gao, T.-Y. Liu, X. Zheng, Q.-S. Cheng, and W.-Y. Ma. Consistent bipartite graph co-partitioning for star-structured high-order heterogeneous data co-clustering. In *KDD '05: Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 41–50, New York, NY, USA, 2005. ACM Press.

[19] E. Garfield. Citation analysis as a tool in journal evaluation. *Science*, 178(60):471–479, November 1972.

[20] C. L. Giles, K. D. Bollacker, and S. Lawrence. Citeseer: an automatic citation indexing system. In *DL '98: Proceedings of the third ACM conference on Digital libraries*, pages 89–98, 1998.

[21] C. Gini. Measurement of inequality of incomes. *The Economic Journal*, pages 124–126, 1921.

[22] S. Golder and B. A. Huberman. Usage patterns of collaborative tagging systems. *Journal of Information Science*, pages 198–208, 2006.

[23] G. H. Golub and C. F. V. Loan. *Matrix Computations*. The Johns Hopkins University Press, 1996.

[24] T. Griffiths and M. Steyvers. Finding scientific topics. In *National Academy of Sciences*, 2004.

[25] D. Gruhl, R. Guha, D. Liben-Nowell, and A. Tomkins. Information diffusion through blogspace. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 491–501, 2004.

[26] 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 Press.

[27] H. Han, L. Giles, H. Zha, C. Li, and K. Tsioutsiouliklis. Two supervised learning approaches for name disambiguation in author citations. In *Proceedings of the 4th ACM/IEEE joint conference on Digital libraries*, 2004.

[28] D. Harel and Y. Koren. Clustering spatial data using random walks. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 281–286, 2001.

[29] J. E. Hirsch. An index to quantify an individual's scientific research output. *Proceedings of the National Academy of Sciences*, 102:16569, 2005.

[30] T. Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*, 42(1-2):177–196, 2001.

[31] T. Hofmann. Latent semantic models for collaborative filtering. *ACM Trans. Inf. Syst.*, 22(1):89–115, 2004.

[32] J. Huang, T. Zhu, R. Greiner, D. Zhou, and D. Schuurmans. Information marginalization on subgraphs. In *PKDD*, pages 199–210, 2006.

[33] P. Jackson. *Introduction to expert systems*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1986.

[34] K. Jarvelin and J. Kekalainen. IR evaluation methods for retrieving highly relevant documents. In *Proceedings of the 23rd annual international ACM SIGIR conference on research and development in information retrieval*, pages 41–48, 2000.

[35] F. Jelinek and R. Mercer. Interpolated estimation of markov source parameters from sparse data. In *Pattern recognition in Practice*, 1980.

[36] X. Ji and W. Xu. Document clustering with prior knowledge. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 405–412, New York, NY, USA, 2006. ACM Press.

[37] H. Kautz, B. Selman, and M. Shah. Referral web: Combining social networks and collaborative filtering. *Communications of the ACM*, March 1997.

[38] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, 1999.

[39] R. Kumar, J. Novak, P. Raghavan, and A. Tomkins. On the bursty evolution of blogspace. In *Proceedings of the 12th international conference on World Wide Web*, pages 568–576, 2003.

[40] O. Kurland, L. Lee, and C. Domshlak. Better than the real thing?: iterative pseudo-query processing using cluster-based language models. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 19–26, New York, NY, USA, 2005. ACM Press.

[41] J. Lafferty and C. Zhai. Document language models, query models, and risk minimization for information retrieval. In *SIGIR '01: Proceedings of the 24th annual international conference on Research and development in information retrieval*, pages 111–119, 2001.

[42] S. Lehmann, A. D. Jackson, and B. E. Lautrup. Measures and mismeasures of scientific quality, 2005.

[43] X. Liu, J. Bollen, M. L. Nelson, and H. Van de Sompel. Co-authorship networks in the digital library research community. *arXiv.org:cs/0502056*, 2005.

[44] A. J. Lotka. The frequency distribution of scientific productivity. *Journal of the Washington Academy of Sciences*, 16(12):317–323, 1926.

[45] S. A. Macskassy and F. J. Provost. Suspicion scoring based on guilt-by-association, collective inference, and focused data access. In *NAACSOS conference proceedings*, June 2005.

[46] N. Matsumura, D. E. Goldberg, and X. Llora. Mining directed social network from message board. In *WWW '05: Special interest tracks and posters of the 14th international conference on World Wide Web*, pages 1092–1093. ACM Press, 2005.

[47] A. McCallum, A. Corrada-Emmanuel, and X. Wang. The author-recipient-topic model for topic and role discovery in social networks: Experiments with enron and academic email. In *University of Massachusetts Amherst, Technical Report*, 2004.

[48] A. K. McCallum. Multi-label text classification with a mixture model trained by em. In *AAAI'09 Workshop on Text Learning*, 1999.

[49] Q. Mei and C. Zhai. Discovering evolutionary theme patterns from text: an exploration of temporal text mining. In *KDD '05: Proceeding of the eleventh intl. conf. on Knowledge discovery in data mining*, 2005.

[50] B. Miller and J. Riedl. A hands-on introduction to collaborative filtering. In *Proc. of the ACM conf. on computer supported cooperative work*, 1996.

[51] S. Morinaga and K. Yamanishi. Tracking topic dynamic trends using a finite mixture model. In *KDD '04: Proceedings of the tenth intl. conf. on Knowledge discovery and data mining*, pages 811–816, 2004.

[52] M. Newman. Fast algorithm for detecting community structure in networks. In *Physics Review*, 2004.

[53] M. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69:026113, 2004.

[54] Z. Nie, Y. Zhang, J.-R. Wen, and W.-Y. Ma. Object-level ranking: bringing order to web objects. In *WWW '05: Proceedings of the 14th international conference on World Wide Web*, pages 567–574, New York, NY, USA, 2005. ACM Press.

[55] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using em. *Mach. Learn.*, 39(2-3):103–134, 2000.

[56] G. Pinski and F. Narin. Citation influence for journal aggregates of scientific publications: Theory, with application to the literature of physics. *Inf. Process. Manage.*, 12(5):297–312, 1976.

[57] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *SIGIR '98: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 275–281, New York, NY, USA, 1998. ACM Press.

[58] A. Pothen, H. D. Simon, and K.-P. Liou. Partitioning sparse matrices with eigenvectors of graphs. *SIAM Journal on Matrix Analysis and Applications*, 11(3):430–452, 1990.

[59] W. M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, pages 622–626, 1971.

[60] M. Richardson and P. Domingos. Mining knowledge-sharing sites for viral marketing. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 61–70. ACM Press, 2002.

[61] C. P. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer Publisher, 2nd Edition, 2005.

[62] M. Rosen-Zvi, T. Griffiths, M. Steyvers, and P. Smyth. The author-topic model for authors and documents. In *UAI '04: Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 487–494. UAI Press, 2004.

[63] E. Seneta. *Non-negative Matrices and Markov Chains*. Springer-Verlag, 1981.

[64] J. Shetty and J. Adibi. The enron email dataset database schema and brief statistical report. In *University of Southern California, Technical Report*, 2004.

[65] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):888–905, 2000.

[66] A. Sidiropoulos and Y. Manolopoulos. A new perspective to automatically rank scientific conferences using digital libraries. *Inf. Process. Manage.*, 41(2):289–312, 2005.

[67] e. Stephen Dill. Semtag and seeker: bootstrapping the semantic web via automated semantic annotation. In *Proceedings of the 12th international conference on World Wide Web*, pages 178–186, 2003.

[68] M. Steyvers, P. Smyth, M. Rosen-Zvi, and T. Griffiths. Probabilistic author-topic models for information discovery. In *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 306–315. ACM Press, 2004.

[69] T. Tao, X. Wang, Q. Mei, and C. Zhai. Language model information retrieval with document expansion. In *HLT-NAACL*, 2006.

[70] R. Todorov and W. Gilazel. Journal citation measures: a concise review. *J. Inf. Sci.*, 14(1):47–56, 1988.

[71] J. A. Tomlin. A new paradigm for ranking pages on the world wide web. In *Proceedings of the 12th international conference on World Wide Web*, pages 350–355. ACM Press, 2003.

[72] J. R. Tyler, D. M. Wilkinson, and B. A. Huberman. Email as spectroscopy: automated discovery of community structure within organizations. *Communities and technologies*, pages 81–96, 2003.

[73] F. Wang, S. Ma, L. Yang, and T. Li. Recommendation on item graphs. In *ICDM '06: Proceedings of the Sixth International Conference on Data Mining*, pages 1119–1123, Washington, DC, USA, 2006. IEEE Computer Society.

[74] J. Wang, A. P. de Vries, and M. J. T. Reinders. Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 501–508, New York, NY, USA, 2006. ACM Press.

[75] X. Wang and A. McCallum. Topics over time: a non-markov continuous-time model of topical trends. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 424–433, New York, NY, USA, 2006. ACM.

[76] S. Wasserman and K. Faust. *Social Network Analysis: Methods and Applications*. Cambridge University Press, 1994.

[77] P. Weingart. Impact of bibliometrics upon the science system: Inadvertent consequences? *Scientometrics*, 62(1):117–131, 2005.

[78] S. White and P. Smyth. Algorithms for estimating relative importance in networks. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 266–275. ACM Press, 2003.

[79] A. Y. Wu, M. Garland, and J. Han. Mining scale-free networks using geodesic clustering. In *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 719–724. ACM Press, 2004.

[80] X. Wu, L. Zhang, and Y. Yu. Exploring social annotations for the semantic web. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, pages 417–426, New York, NY, USA, 2006. ACM Press.

[81] H. Zha, C. Ding, M. Gu, X. He, and H. Simon. Spectral relaxation for k-means clustering. In *Neural Information Processing Systems*, volume 14, 2001.

[82] H. Zha, X. He, C. Ding, H. Simon, and M. Gu. Bipartite graph partitioning and data clustering. In *CIKM '01: Proceedings of the tenth international conference on Information and knowledge management*, pages 25–32, New York, NY, USA, 2001. ACM Press.

[83] D. Zhou, L. Bolelli, J. Li, C. L. Giles, and H. Zha. Learning user clicks in web search. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*. ACM Press, 2007.

[84] D. Zhou and C. J. C. Burges. Spectral clustering and transductive learning with multiple views. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 1159–1166, 2007.

[85] D. Zhou, I. Councill, H. Zha, and C. L. Giles. Discovering temporal communities from social network documents. In *ICDM'07: Proceedings of the 7th IEEE International Conference on Data Mining*, 2007.

[86] D. Zhou, J. Huang, and B. Scholkopf. Learning from labeled and unlabeled data on a directed graph. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*, pages 1036–1043, 2005.

[87] D. Zhou, X. Ji, C. L. Giles, and H. Zha. Topic evolution and social interactions : How authors effect research. In *CIKM '06: Proceedings of the 15th ACM International Conference on Information and Knowledge Management*, pages 247–248. ACM Press, 2006.

[88] D. Zhou, H. Li, J. Li, W. chien Lee, H. Zha, and C. L. Giles. Learning to rank social network actors. In *ICDM'07 Workshops: Proceedings of a Workshop at the 7th IEEE International Conference on Data Mining*, 2007.

[89] D. Zhou, J. Li, and H. Zha. A new mallows distance based metric for comparing clusterings. In *ICML '05: Proceedings of the 22nd International Conference on Machine Learning*, pages 1028–1035. ACM Press, 2005.

[90] D. Zhou, E. Manavoglu, J. Li, C. L. Giles, and H. Zha. Probabilistic models for discovering e-communities. In *WWW '06: Proceedings of the 15th international conference on World Wide Web*, pages 173–182. ACM Press, 2006.

[91] D. Zhou, S. Orshanskiy, H. Zha, and C. L. Giles. Co-ranking authors and documents in a heterogeneous network. In *ICDM'07: Proceedings of the 7th IEEE International Conference on Data Mining*, 2007.

[92] D. Zhou, Y. Song, H. Zha, and Y. Zhang. Towards discovering organizational structure from email corpus. In *ICMLA '05: Proceedings of the Fourth International Conference on Machine Learning and Applications (ICMLA'05)*, pages 279–284. IEEE Computer Society, 2005.

[93] D. Zhou, S. Zheng, J. Bian, H. Zha, and C. L. Giles. Exploring social annotations for information retrieval. In *WWW '08: Proceedings of the 17th international conference on World Wide Web*, 2008.

[94] D. Zhou, S. Zhu, K. Yu, X. Song, B. Tseng, H. Zha, and C. L. Giles. Learning multiple graphs for document recommendations. In *WWW '08: Proceedings of the 17th international conference on World Wide Web*, 2008.

[95] S. Zhu, K. Yu, Y. Chi, and Y. Gong. Combining content and link for classification using matrix factorization. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, 2007.

Vita

Ding Zhou

Ding Zhou was born in Changsha, Hunan, China. He received his bachelor's degree in Computer Science from Fudan University in Shanghai in July 2004. In August 2004, he enrolled in the Ph.D. program in Computer Science and Engineering at The Pennsylvania State University where he developed research in computational social network analysis and engaged in a wide range of projects involving data mining and machine learning from user generated data, such as heterogeneous social networks, social actions, and social texts. His social network research began with mining email texts. Concentration in mining social networks in scientific documents arose from membership in the CiteSeer^X project, a well known search engine for computer science scientific literature. During his Ph.D. study, he has published 21 papers in many venues including conferences for machine learning and information retrieval.

His research has received attention and collaboration with industry. He interned at Yahoo!, Google, and NEC Labs America in 2005, 2006, and 2007 and collaborated with researchers from Lawrence Berkeley National Laboratory. Currently he is a research scientist with *Facebook*.