Discovery and analysis of social networks based on online user activities

Lauw, Hady Wirawan

2008

Lauw, H. W. (2008). Discovery and analysis of social networks based on online user activities. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/14249>

<https://doi.org/10.32657/10356/14249>

DISCOVERY AND ANALYSIS OF SOCIAL NETWORKS BASED ON ONLINE USER ACTIVITIES

HADY WIRAWAN LAUW

School of Computer Engineering

A thesis submitted to the Nanyang Technological University
in fulfillment of the requirement for the degree of
Doctor of Philosophy

2008

Acknowledgments

I would like to express my gratitude to the following people and organizations for their support and contributions, without which this dissertation would not have been possible.

Dr Lim Ee Peng, my supervisor, who provides much-needed guidance, kind support, and constructive evaluation of my work, and whose streams of suggestions and ideas make this research as much his as mine. He continues to be a great mentor and example.

Dr Pang Hwee Hwa, co-supervisor, for his involvement, guidance, and advice throughout my Ph.D. research. I appreciate that despite his busy schedule at SMU, he makes the time and routinely travels to NTU for meetings.

Dr Wang Ke, collaborator, for pursuing joint research, co-authoring papers, and giving advice on research during (and beyond) his one year of Visiting Professorship at NTU.

Tan Teck Tim and **Center for IT Services** for providing the wireless network usage data used in this research, and collaborating on the collection of data and the interpretation of results.

Dr Sourav Bhowmick and **Dr Ng Wee Keong**, current and former Directors of Centre for Advanced Information Systems (CAIS), for generally ensuring a conducive research environment in the centre.

Professors, Researchers, and Fellow Students at CAIS for various discussions, seminars, exchanges of ideas, and their kind friendship.

Lai Chee Keong, Chua Chiew Song, and Tan Lay Choo, current and former support staff at CAIS, for the technical and administrative support in the course of my research at the lab.

Koh Soh Hong, Elaine, Secretary to Head of Division (Information Systems) for her generous help with any administrative need I encountered over the course of my research.

Last but not least, the **Agency for Science, Technology and Research (A*STAR)**, for the generous financial support in the form of an A*STAR Graduate Scholarship throughout my pursuance of this PhD program .

Contents

Acknowledgments	1
List of Figures	5
List of Tables	7
Abstract	9
1 Introduction	10
1.1 Research Scope	12
1.2 Contributions	16
1.3 Applications	19
1.4 Organization of the Dissertation	20
2 Social Network Overview	21
2.1 Fundamental Concepts	21
2.2 Social Network Discovery	23
2.3 Social Network Analysis	27
2.4 Social Network Application	37
3 Social Network Discovery by Mining Spatio-Temporal Events	41
3.1 Overview	41
3.2 STEvent: Spatio-Temporal Event Model	42
3.3 Computational Algorithms	49
3.4 Experiments on Cyber Location Data	54
3.5 Experiments on Physical Location Data	62
3.6 Summary	67

4	Bias and Controversy in Collaborative Rating Networks	69
4.1	Overview	69
4.2	Bias and Controversy Framework	72
4.3	Deviation Measures	77
4.4	Convergence of the Proposed Models	79
4.5	Experiments on Real-Life Data	82
4.6	Experiments on Synthetic Data	88
4.7	Discussion	96
5	Quality and Leniency in Collaborative Rating Networks	98
5.1	Overview	98
5.2	Leniency-aware Quality (LQ) Model	101
5.3	Solution Types	105
5.4	Experiments on Real-Life Data	108
5.5	Experiments on Synthetic Data	115
5.6	Discussion	123
6	Rating Dependencies in Collaborative Rating Networks	125
6.1	Overview	125
6.2	Rating Dependencies (RD) Model	127
6.3	Correlation Measures	129
6.4	Solution Types	132
6.5	Experiments	135
6.6	Discussion	143
7	Conclusion	144
7.1	Future Work	147
A	List of Publications	150
	References	153

List of Figures

1.1	Social Network Example	11
1.2	Research Directions in Social Networks	12
1.3	Collaborative Rating Network	14
2.1	Taxonomy of Social Network Discovery	24
2.2	Taxonomy of Social Network Analysis	28
3.1	Algorithm: Construction of Events	51
3.2	Algorithm: Construction of Links	53
3.3	Algorithmic Behavior (<i>cyberdata</i> Aug-Sep04)	61
3.4	Time Taken vs. Data Size (<i>cyberdata</i> Aug-Sep04)	62
3.5	Algorithmic Behavior (<i>physicaldata</i> Aug-Sep04)	66
4.1	Example Rating Network	70
4.2	Framework for Analyzing Bias and Controversy in Rating Data	73
4.3	Histogram of Deviation Values	78
4.4	Percentrank Scatterplots: <i>Naive</i> vs. <i>IR</i>	84
4.5	Vary k : Recall (<i>IR</i> vs. <i>Naive</i>)	91
4.6	Vary k : Distribution of Bias/Controversy	92
4.7	Vary k : Recall (Upper vs. Lower Half Evidence)	93
4.8	Vary n : Recall (<i>IR</i> vs. <i>Naive</i>)	94
4.9	Vary n : Variance in Controversy Proportion	95
4.10	Vary n : Recall (Upper vs. Lower Half Evidence)	95
5.1	Rating Data Examples	99
5.2	Quality Rank Scatterplots: <i>Naive</i> vs. <i>Ranked</i>	111

5.3	Quality Rank Scatterplot: <i>Naive</i> vs. <i>Riggs</i>	112
5.4	Quality Rank Scatterplots: <i>Exact</i> vs. <i>Ranked</i>	112
5.5	Vary n : Quality Recall and Leniency Recall	119
5.6	Vary n : Leniency Class Distribution Error	120
5.7	Vary k : Quality Recall	121
5.8	Vary m : Quality Recall	122
5.9	Vary α : Quality Recall	123
6.1	<i>Naive</i> vs. <i>Exact</i> : Reviewer Dependency	137
6.2	<i>Naive</i> vs. <i>Exact</i> : Object Dependency	138
6.3	<i>Ranked</i> vs. <i>Exact</i> : Reviewer Dependency	139

List of Tables

3.1	Table of Notations	44
3.2	Parameter Values (<i>cyberdata</i> Aug-Sep04)	55
3.3	Demographic Similarity (<i>cyberdata</i> Aug-Sep04)	56
3.4	Highly Similar Event-based Pairs (<i>cyberdata</i> Aug-Sep04)	57
3.5	Demographic Similarity across Levels (<i>cyberdata</i> Aug-Sep04)	59
3.6	Demographic Similarity across Periods (<i>cyberdata</i> at level 1)	59
3.7	Parameter Values (<i>physicaldata</i> Aug-Sep04)	63
3.8	Demographic Similarity across Levels (<i>physicaldata</i> Aug-Sep04)	64
3.9	Demographic Similarity across Periods (<i>physicaldata</i> at level 3)	64
4.1	Table of Notations	70
4.2	Data Size	83
4.3	Network Connectivity	83
4.4	Rank Similarity at Top k%: <i>Naive</i> vs. <i>IR</i>	86
4.5	Case Examples: Bias	87
4.6	Case Examples: Controversy	88
4.7	Data Generation Parameters	89
5.1	Table of Notations	99
5.2	Quality and Leniency for Examples in Figure 5.1	103
5.3	LQ Solutions	108
5.4	Data Size	110
5.5	Profile of Object <i>mu1006829</i>	113
5.6	Profile of Object <i>mu1016864</i>	114
5.7	Profile of Reviewer <i>edmaidel</i>	115

5.8	Profile of Reviewer <i>heymrdj2k</i>	115
5.9	Data Generation Parameters	116
5.10	Generated Rating Scores (e_{ij} values)	116
5.11	Vary k : Percentages of Objects with High $\mathbf{d}_j.d_-$ or $\mathbf{d}_j.d_+$	121
5.12	Vary m : Percentages of Objects with High $\mathbf{d}_j.d_-$ or $\mathbf{d}_j.d_+$	122
6.1	Table of Notations	127
6.2	Comparative Scenarios: LC vs. MI	130
6.3	Data Size	135
6.4	Profile of Object <i>boulevard_dry_stout</i>	140
6.5	Objects reviewed by <i>wingdman</i>	140
6.6	Profile of Reviewer <i>adamldemarco</i>	141
6.7	Reviewers rating <i>hoegaarden_original_ white_beer</i>	141
6.8	Profile of Reviewer <i>loraxc</i>	142

Abstract

This dissertation documents our research on the discovery and analysis of social networks based on user activities online, focussing on two types of data that have not previously attracted much work, namely: spatio-temporal data and collaborative rating data.

Spatio-temporal data has spatial and temporal information about users. Our research aim for spatio-temporal data is to discover the latent social network that relates users together. We adopt spatio-temporal co-occurrence as a basis to predict social associations, reasoning that frequent co-occurrences by several users in space and time suggest some form of interaction or association. Our proposed model *STEvent* works by mining events, which are distinct co-occurrences among two or more actors, and inferring links between pairs of actors who participate in many common events. The effectiveness of the *STEvent* model is verified through experiments on two real-life datasets.

Collaborative rating data involves a set of users (acting as reviewers) who collaboratively evaluate a set of objects, by contributing rating scores to those objects. Rating data has a bipartite social network representation, with reviewers and objects as nodes, and ratings as links. Our research agenda on rating data is to analyze a set of rating-related behaviors of reviewers and objects, focussing on three research issues involving rating-related behaviors, namely: (a) analyzing score deviations to determine the *bias* of reviewers and the *controversy* of objects, (b) analyzing score inflations/deflations to determine the *quality* of objects and the *leniency* of reviewers, and (c) analyzing score correlations to determine the *rating dependencies* of reviewers and objects. A common thread across these three is the notion of mutual dependency between reviewers' and objects' behaviors. For each, we propose a model that takes into account this mutual dependency, and develop the necessary computational framework to simultaneously solve the behaviors of all reviewers and objects in the network. These models are shown to be effective through various experiments on real-life and synthetic datasets.

Chapter 1

Introduction

Online Activities

The prevalence of Internet and mobile computing technologies means that individuals now conduct more activities online (over both wired and wireless networks). The Web becomes a platform for many different types of activities, such as information publishing (on Web sites and homepages), information sharing (on newsgroups and forums of various topics of interest), communication (over emails and instant messaging networks), gaming (over massively multi-player online games, e.g., World of Warcraft¹), and shopping (at online retailers, e.g., Amazon², or auction sites, e.g., eBay³). All these activities bring people together regardless of geographical distance.

With the recent advent of Web 2.0, those online activities become even more social and collaborative in nature. Information publishing tools such as blogs invite conversations in the form of comments. Wikis allow multiple contributors to collaborate in writing articles. Information sharing applications (e.g., image sharing on Flickr⁴ and video sharing on YouTube⁵) now support social features, such as tagging and rating by users to collaboratively organize and evaluate content. Through the use of reviews and

¹www.worldofwarcraft.com

²www.amazon.com

³www.ebay.com

⁴www.flickr.com

⁵www.youtube.com

Figure 1.1: Social Network Example

ratings, users can cooperate to identify the good products to purchase, or the reputable users to transact with. It is therefore not a surprise that a large number of online communities form on the Web 2.0 and users of these communities are connected with one another in some social networks.

Social Networks

A social network models the structure of associations among social entities as a graph. For example, the social network in Figure 1.1 reveals the pattern of connectivity among eight nodes. Nodes may represent various types of social entities, such as people, organizations, countries, objects, etc. Links represent the relationships among these entities, such as friendship, liking/disliking, transactions, etc. Various network analysis techniques can be employed to derive useful information on nodes, links, paths, as well as clusters and subgraphs. Social network theory has been applied widely in various application domains, including law enforcement, business, sociology, and economics [WF94].

Since much of the online activities described above are social in nature, social network is a suitable framework to study online activities. When carrying out online activities, users maintain some form of social identities, e.g., homepages on the Web, nicknames on various sites, and avatars on online games. Activities of and interactions among these social identities can usually be tracked and captured (e.g., which homepages link to which

Figure 1.2: Research Directions in Social Networks

other homepages, who comments on whose blog, who attacks whom in an online game). While some types of data may be proprietary or private (e.g., email messages), other types of data are publicly available (e.g., blog posts and Wikipedia articles). Mining such data promises to uncover useful insights on human activities and interactions. As online activities increasingly mirror offline ones, the insights would likely have applicability to offline situations too.

1.1 Research Scope

Our objective is to apply the social network framework to the study of online activities, so as to derive useful insights on human activities and interactions. Below, we first review the main branches in social network research, before identifying the specific research issues that will be covered in this dissertation.

We propose an organization of prior research work on social networks into a taxonomy as shown in Figure 1.2, with three main divisions namely: discovery, analysis, and application. We will further expand on each division in Chapter 2. *Discovery* refers to the process of discovering or constructing a social network from some data (e.g., co-

CHAPTER 1. INTRODUCTION

authorship of academicians on publications). Traditionally, social networks involve only users as actors. The recent work on social network discovery however has also studied social networks that involve other types of actors, e.g., papers, blogs. *Analysis* of a social network is carried out to identify useful insights about nodes, links, or subgraphs within a network (e.g., finding out the influential academicians). *Application* concerns how the discovered social network and the analytical outcomes can be used in different types of applications.

Note that the social network framework lends itself to broad interpretation and application, because different meanings may be attached to the sets of nodes and links of different social networks. Hence, to bring out the more specific and useful insights, we need to consider the data modeled by a social network. Various types of data have been studied previously, including kinship, club membership, co-authorship, etc. [WF94]. However, online activities give rise to new types of user-generated data that have not attracted much work. Two such data types that are the focus of this dissertation are *spatio-temporal data* and *collaborative rating data*.

Spatio-Temporal Data

User-generated spatio-temporal data has at least three attributes: user, location, and time. Such data is usually generated by spatially and temporally sensitive activities, such as an individual's movements across various physical locations or cyber locations (Web pages) over time. As this data results from online activities, some of which are social interactions, it contains latent information about the social associations among individuals.

Our main research agenda for spatio-temporal data is to discover the latent social network that relates users together based on the recorded activities or interactions taking the form of spatio-temporal tuples as described above. Once discovered, the social network can feed into various analysis techniques or applications. To address this research

Figure 1.3: Collaborative Rating Network

objective, we propose spatio-temporal co-occurrence as a basis for inferring links between individuals. The intuition is that a pair of individuals who frequently co-occur in space and time may be engaged in rendezvous events. The more frequent the co-occurrences, the likelier is a link between the pair.

Collaborative Rating Data

Collaborative rating refers to the activities in which users assume the role of reviewer, and assign rating scores to objects, in order to collaboratively evaluate the objects. Rating is a common activity in various contexts, including peer review (academicians rating papers), product review (users rating products), and information sharing (users rating photos or videos shared by others). A rating network has a bipartite social network representation as shown in Figure 1.3. The two distinct types of nodes are reviewers and objects. A link exists from reviewer r_i to object o_j if r_i has rated o_j . The link weight e_{ij} is the rating score that r_i assigns to o_j .

Since collaborative rating data are readily available in the network form, the discovery process is trivial. The main research challenge is the analysis of collaborative rating data. Existing work on social network analysis mostly focuses on structural analysis that is blind to the semantics of a rating link. Hence, this dissertation proposes to mine rating-related behaviors of nodes (reviewers and objects) in a rating network. Specifically, we

CHAPTER 1. INTRODUCTION

identify three sets of rating-related behaviors: (a) *bias and controversy*, (b) *quality and leniency*, and (c) *rating dependencies*.

Bias and Controversy. In a rating network, a reviewer often deviates from co-reviewers when assigning a score to an object. We seek to mine score deviations to reveal the *bias* of reviewers and the *controversy* of objects. On one hand, a reviewer who has high *bias* may tend to assign a score that deviates from co-reviewers. On the other hand, an object that has high *controversy* may draw deviating scores from reviewers.

Quality and Leniency. An important purpose of rating systems involves aggregating the rating scores assigned to an object to derive an overall score that reflects the *quality* of the object. One reviewer behavior that may affect the rating scores is *leniency*, which refers to the tendency of a reviewer to assign higher scores. We therefore seek to mine the *leniency* information from rating scores and use it to derive the *quality* of objects more equitably.

Rating Dependencies. In this research, we examine rating systems that allow reviewers to assign different types of scores based on specific rating criteria. We seek to mine, for each reviewer and for each object, the *dependency* between scores on any two given criteria. A reviewer has high *dependency* between a pair of criteria when his/her rating scores on objects based on the two criteria exhibit strong correlation. On the other hand, an object is said to have high *dependency* between a pair of criteria when the ratings it receives on the two criteria exhibit strong correlation.

The knowledge of the above rating-related behaviors is useful in various application contexts. It allows us to know whether the evaluation process has been conducted fairly and objectively. It also helps to meet the discriminatory objective of rating, which is to more accurately identify the best objects or weed out the worst ones. In some cases, it reveals some information about the reviewers and objects, such as the suitability of a reviewer based on his/her past record of bias, or the preferences of a reviewer who is heavily dependent on a pair of rating criteria.

In summary, as indicated in Figure 1.2, the scope of this dissertation includes: (a) *social network discovery from spatio-temporal data*, as well as social network analysis of collaborative rating data to derive (b) *bias and controversy*, (c) *quality and leniency*, and (d) *rating dependencies*.

1.2 Contributions

The main contributions of this dissertation can be summarized as follows. The first three concern our work on spatio-temporal data, and the rest on collaborative rating data.

Novel Definition of Spatio-Temporal Event

Our definition of spatio-temporal event based on co-occurrences in space and time between at least two actors meeting specific conditions (see Definition 3.1) is novel. We further introduce four event weight measures that measure the extent to which an event is indicative of a social association. These measures are: *spatial and temporal precisions* (measuring how closely in space and time actors are within the event) and *spatial and temporal uniqueness* (measuring the uniqueness of the event’s location and time among other events). A link between two actors has a weight which is aggregated from the weights of events participated by both actors.

Efficient Algorithms for Event-based Social Network Discovery

We propose efficient algorithms that construct a social network from spatio-temporal data in two phases: Phase 1 constructs events from spatio-temporal data, and Phase 2 constructs links from events. By minimizing the number of times a tuple is processed, our Phase 1 algorithm requires $O(|\mathcal{D}| \times l_{max})$ complexity, where $|\mathcal{D}|$ is the number of tuples and l_{max} is the number of location granularity levels. A brute force approach that examines every sequence of tuples of size 2, size 3, and so on up to size $|\mathcal{D}|$, to see

CHAPTER 1. INTRODUCTION

which ones meet the event definition will be very time consuming, with up to $O(|\mathcal{D}|^2)$ complexity.

By indexing events by location and time, and selectively investigating pairs of actors, we also reduce our Phase 2's complexity to $O(|\mathcal{E}| \times \log |\mathcal{E}| + |G_{E_{cand}}|)$, where $|\mathcal{E}|$ is the number of events and $|G_{E_{cand}}|$ is the number of candidate links to be considered. A brute force approach that iterates over all possible links to find out the set of supporting events and iterates over all events to determine event weights will be very time consuming ($O(|\mathcal{E}|^2)$).

Verification of Discovered Social Networks

We verify the effectiveness of the discovered social network using a similarity-based measure, relying on a well-known result in sociology that associated pairs are likely to be similar [Fel81, Car91]. For this purpose, we conduct experiments on two sets of real-life data of different location semantics, *physical location* and *cyber location* (Web pages) respectively, which allow us to compare the different social networks discovered using the different location semantics. We also repeat the experiments for data collected at different time periods and for various levels of location granularity to see if the good performance achieved on both data sets can be repeated under the different conditions.

Novel Definitions of Specific Rating-related Behaviors

We propose the problem of mining rating-related behaviors from collaborative rating data, identifying three sets of rating-related behaviors: (a) bias of reviewers and controversy of objects, (b) leniency of reviewers and quality of objects, and (c) reviewer dependency and object dependency. While some of these terms have been mentioned in various domains, our specific definitions and formal modeling them are *novel*. They are ‘*data-centric*’; they can be detected from rating scores alone, without additional information such as demographic attributes of reviewers, etc. They are also *specific* to each

reviewer or to each object, unlike prior work that mainly focuses on systematic behaviors affecting reviewers or objects in general.

Mutual Dependency between Reviewers' and Objects' Behaviors

We propose the principle of mutual dependency between reviewers' and objects' behaviors within a rating network. This mutual dependency is expressed in two ways. First, the behaviors of reviewers and the behaviors of objects are mutually dependent as determining one requires knowing the other. For example, as score deviation may come about due to a reviewer's bias or an object's controversy, determining bias requires knowing the controversy of rated objects (vice versa). Second, a reviewer's behavior is dependent on those of other reviewers, because a reviewer's behavior is inferred not only from his/her rating scores on rated objects but also from how other reviewers rate the same objects. Similarly, an object's behavior is dependent on those of other objects. This mutual dependency principle is incorporated into the formal models of the rating-related behaviors.

Efficient Computational Framework for Rating-related Behaviors

Due to the mutual dependency principle, the whole rating network must be solved as an integrated system of equations involving all reviewers and objects. We determine how this system of equations, which can be compactly represented in terms of matrices, can be solved using iterative methods [AR87, GVL96]. Starting from random values, each iteration moves the system closer to the final solution, which is achieved when the iterations converge. We discuss the convergence properties of such iterations. Iterative methods are time-efficient, with time complexity proportional to the number of iterations to convergence, which generally is small. Moreover, the iterations can be prematurely terminated for approximate results. Iterative methods are also memory-efficient. They can be implemented with an adjacency list representation of the rating network, which reduces the

memory requirements tremendously for a sparse rating network, as compared to matrix manipulation techniques that usually require an adjacency matrix representation.

Verification of Rating-related Behaviors

We verify the effectiveness of the proposed behavioral models against baseline models, which ignore the mutual dependency principle. We conduct experiments on real-life data, which allow us to see whether the proposed models produce a significant difference from baseline results, and whether such differences are in favor of our models. For some models, we also conduct experiments on synthetic data, which allow us to study the models' performance with respect to pre-determined ground truth. For this purpose, we propose an appropriate data generation mechanism, and suggest the use of recall measures for performance evaluation. By varying the settings of a small set of important parameters, we investigate whether the proposed models can consistently outperform their baselines and how the models behave over different parameter settings.

1.3 Applications

We believe that the contributions made in this thesis will find applications in various fields. The spatio-temporal event framework provides one more avenue to construct social networks. The network constructed can feed directly into various social network analytical techniques (see Section 2.3). It may assist such specialized fields as criminal network analysis, provided some movement data of suspects are available. Another application, social search, involves querying one's social network to look up interesting actors or paths, which will require a network to have been constructed in the first place.

On the other hand, as rating features are prevalent on various online social media (e.g., blogs, wikis, and content sharing sites), e-commerce platforms, and recommender systems, the work on user behaviors in this thesis can be applied to rating data extracted

from these sites. For example, a recommender system that is aware of rating-related behaviors can make a more informed recommendation. The identification of high-quality recommendable objects will benefit from taking into account the bias and leniency of reviewers or the controversy of objects. The peer review exercise whereby academicians evaluate each other's work will also benefit from this work. Knowing the various behaviors of reviewers and objects will enrich the evaluation process, and perhaps help in the future assignment of reviewers to objects.

1.4 Organization of the Dissertation

The rest of this dissertation is organized as follows.

- In Chapter 2, we review prior work in social network research, covering the three main research directions: social network discovery, analysis, and application.
- In Chapter 3, we discuss our proposed *STEvent* (*Spatio-Temporal Event*) model to discover social network from spatio-temporal co-occurrences.
- In Chapter 4, we address the problem of measuring the *bias* of reviewers and the *controversy* of objects by mining score deviations in a collaborative rating network.
- In Chapter 5, we address the problem of summarizing rating scores assigned by reviewers to derive the *quality* of objects, taking into account the *leniency* behavior of reviewers.
- In Chapter 6, we address the problem of measuring *rating dependencies*, or the tendencies of reviewers (or objects) to assign (or draw) correlated scores across pairs of scoring criteria.
- In Chapter 7, we conclude this dissertation with a summary of research results and an overview of future research directions.

Chapter 2

Social Network Overview

In this chapter, we give an overview of prior work in social network research. After reviewing a set of fundamental social network concepts, we organize the prior work into three sections, covering social network discovery, analysis, and application respectively. Where appropriate, we draw relevant comparisons between our work and the prior work.

As the study of social networks goes beyond computer science, the following discussion and references (especially in Section 2.3) necessarily draw from various domains of scientific enquiry, including psychological sciences, social sciences, and commerce. Cutting across these disparate domains is a graph-theoretical foundation of network analysis [BH83, Dav68, WF94], whereby a social network is represented as a graph (for instance with adjacency matrix data structure), with actors as nodes and links as edges. As the focus of this thesis is on social networks based on online activities, the discussion of the work will be in this context.

2.1 Fundamental Concepts

We review the basic terminology of social network that will be used throughout the dissertation.

CHAPTER 2. SOCIAL NETWORK OVERVIEW

Actor

An actor is an entity whose relationships to other actors are mapped onto a social network. Examples of actors include people, objects, organizations, countries, etc.

Link

A link directly relates a pair of actors. There could be diverse meanings attached to a link, including: *evaluation* (e.g., liking/disliking, respect, friendship), *affiliation* (e.g., person belonging to a club), *interaction* (e.g., communicating, collaborating), etc.

A link is either *directed* from one actor to another, or *undirected* if it is symmetrically shared between the two actors. A *dichotomous* link is either present or absent, while a *valued* link is weighted with a range of values, with higher values usually indicating stronger relationships. A valued link may also be *unsigned*, with positive link weights, or *signed*, where link weight may be positive or negative (e.g., liking or disliking).

Path

A path connects a pair of actors through an unbroken chain of links. The length of a path is the number of links that make up the chain.

Subgroup

A subgroup comprises a subset of actors in a social network, as well as all the links between them. The actors to be included in a subgroup are selected based on specific criteria, which will be discussed later.

Relation

A social network may have several types of links. A relation is the set of all links of a specific type. For example, if we define two relations $\mathcal{R}_{friend}$ and $\mathcal{R}_{work}$, then all links based on friendship make up $\mathcal{R}_{friend}$ and all links based on working relationship make up $\mathcal{R}_{work}$.

Mode

A social network may have several types of actors. Mode refers to the number of distinct types of actors. If all actors are of the same type (e.g., people), the network is a one-mode network. If there are two types of actors (e.g., people and organizations), it is a two-mode network.

2.2 Social Network Discovery

The problem of social network discovery can be expressed as follows: *given a finite set of actors, find out which pairs of actors have a link between them and, if applicable, what the weight of each link is.* The solution to this problem requires some criterion to decide whether there is sufficient evidence to infer a link between two nodes and to quantify the strength of that link. Below, we list four such criteria that have been used in prior work, namely: *self-reported*, *communication*, *similarity*, and *co-occurrence*. As shown by the taxonomy in Figure 2.1, the former two usually give rise to directed links; the latter two, to undirected links. Note that for each criterion, links can be inferred from either offline or online activities.

2.2.1 Self-Reported

Self-reported links refer to links discovered from the involved actors themselves. A directed link from actor a_i to another actor a_j exists if a_i has reported it. Such links are directed since a_j may not necessarily report a link to a_i . Even if a pair of actors mutually report links to each other, they may not attach equal weights to the link.

Classical social network research discovers self-reported links through carefully constructed procedures such as questionnaires, interviews, direct observations of interactions, manual sifting through archival record, or various experiments [WF94]. The discovery

CHAPTER 2. SOCIAL NETWORK OVERVIEW

Figure 2.1: Taxonomy of Social Network Discovery

effort is time- and resource-intensive, covers a small number of actors, and is usually restricted to specific settings (e.g., people in a company/school).

Online settings lower the barrier and create incentives for a user to report links to others. Someone maintaining a homepage or a blog often lists hyperlinks to Web sites or blogs of friends (e.g., LiveJournal [KNRT04]), to increase her connectivity within the community, which helps to increase traffic to their homepage or blog. Similarly, profile pages of community-centric sites such as Facebook or Friendster [Boy04] commonly display a self-professed list of friends within the community. Consequently, there are voluminous and diverse self-reported links that can be harvested from these sources.

2.2.2 Communication

Communication, defined generally as transfer of information or resources, is commonly exhibited by socially related people. Communication-based links are usually directed from the originator to the recipient. If desired, an undirected link may be inferred from bi-directional links. Links are usually weighted by the frequency and intensity (e.g, conversation length) of the communication.

Evidence of communication can be drawn from direct observation of interactions or interviews, e.g., asking a group of people to give accounts of work communication [GHW97]. Much of modern communication is computer-mediated, over the Internet, which often leaves a trail in the form of usage logs that can be mined for evidence of sustained communication. Sources of online communication include records of email [CKY05, DC05], Instant Messaging (IM) [AT06, RDHT04, RT04], newsgroups [ARSX03, BCMS04], phone logs [KNV06], etc.

2.2.3 Similarity

Similarity has its foundation on the well-received sociological idea that friends tend to be alike [Fel81, Car91]. This leads to the premise that the more people have in common, the likelier it is that they are related. Similarity-based links are naturally undirected, since the notion of similarity is symmetric.

Prior work on similarity-based links involves identifying the relevant attributes of users that may indicate relationship, and a suitable similarity measure. Homepages with similar content and linkages may represent a group of related individuals [AA03]. Two people whose sets of communication partners overlap may be affiliated to a common group [SW93]. Other forms of similarity include sharing the same opinions or areas of interest [RD02], or even sharing similar vocabulary choices in email messages [KS05].

2.2.4 Co-occurrence

Co-occurrence assumes that if several actors occur together more frequently than random chance alone would allow, they are likely associated in some way. Like similarity, it is also undirected by nature. Prior work on co-occurrence-based links can be organized into two streams: *transactional*, where there is a clear boundary within which two actors are said to co-occur, and *spatio-temporal*, where the boundary of co-occurrence is defined by space and/or time.

Transactional Co-occurrence

The term *transaction* is borrowed from work on frequent pattern mining [AIS93, AS94]. It refers to a discrete instance within which a few items may co-occur, e.g., a supermarket transaction involving a number of product items. A frequent pattern involves a set of items that co-occur together in many transactions, and thus are likely to be associated with one another. Applied to social network discovery, a transaction in an offline setting may refer to a party attended by a pair of actors [WF94], a movie that a pair of actors act in [KNV06], or a publication which a pair of researchers co-author [LC03, LC04]. In an online setting, a transaction may refer to a Web page where the names of a pair actors co-occur [FMT04].

Spatio-Temporal Co-occurrence

The boundary of a transaction is not always clear-cut, especially when it involves continuous dimensions such as space and time. Suppose that we have a set of tuples $\{\langle a, s, t \rangle\}$, where each tuple records an actor a appearing at location s at time t , and we wish to infer links between pairs of actors based on co-occurrences. A transaction must then be defined in terms of space and/or time. For example, a spatial transaction can be derived by discretizing the space dimension using a sliding window [SH01]. A temporal transaction can be a time interval within which two IM users must be online together (and thus are more likely to engage in a conversation) [RDHT04, RT04].

In turn, a spatio-temporal co-occurrence is defined over both space and time. Our work (*STEvent* in Chapter 3) concerns social network discovery from spatio-temporal co-occurrences. That spatio-temporal movement data are a possible indicator of social association has been suggested in prior work [CP03, EP06, TMRL02]. However, our work is distinguished from the prior work in the following ways:

- It focuses on the analysis of movement data and algorithm development to infer associations, while the others focus on the development of movement-tracking devices.
- It generalizes the spatio-temporal co-occurrence beyond movement over physical locations to include other location types such as cyber locations.
- It is the first to attempt at verification of the inferred associations through analysis of demographic data.

2.3 Social Network Analysis

Social network analysis attempts to find useful structures, patterns, or insights that exist within a social network. As shown in the taxonomy in Figure 2.2, such studies may look for “important” actors in the network (*actor analysis*), “important” paths connecting a subset of actors (*path analysis*), and subgroups that exist within a network (*subgroup analysis*).

Note that we do not distinguish between social networks derived from offline or online activities. Most analytical methods simply assume a readily available social network. Neither do we distinguish between directed links or undirected links. Most analytical methods can be adapted to both types of links. The common workaround is to define analysis for directed links and treat undirected links as bi-directional links, or to define analysis for undirected links and ignore the direction of directed links.

2.3.1 Actor Analysis

The problem of actor analysis can generally be expressed as follows: *given a social network, measure or rank the “importance” of every actor in the network*. There are various definitions of importance, which usually represents a certain property or behavior of an

CHAPTER 2. SOCIAL NETWORK OVERVIEW

Figure 2.2: Taxonomy of Social Network Analysis

actor. As shown in the taxonomy in Figure 2.2, prior work in actor analysis has largely focused on the following definitions of importance: *centrality*, *influence*, *reputation*, and *anomaly*. Our work on collaborative rating data (Chapters 4, 5, and 6) concerns a fifth definition of importance: *rating-related behaviors*. Below, we discuss various work in each category, and relate our work to them.

Centrality

Centrality equates importance of actors to occupying strategic or central locations in a network [WF94]. Such actors are more visible and are involved in more relationships with other actors. Social network researchers have developed the following measures of centrality, that are mostly based on the structural properties of a graph.

Degree. The degree centrality of an actor is her number of links. The intuition is that central actors should be the most active, and should have the most connections to others in its vicinity. This measure has been applied to law enforcement, where it is used to

CHAPTER 2. SOCIAL NETWORK OVERVIEW

identify the key players in a price fixing conspiracy [XC03], and the supposed ringleader of 911 terrorist network (Mohammed Atta) [Kre02].

Closeness. The closeness centrality of an actor is the inverse of the average path length from the actor to all other actors in the network. The reasoning is that an important actor should have easy access to others members of the network.

Betweenness. The betweenness centrality of an actor is the number of distinct shortest paths (connecting any pair of actors) that pass through it. Actors with high betweenness values are in a position to control communication channels, either by impeding or accelerating or just by getting informed of such communication.

Eigenvector Centrality. The eigenvector centrality of an actor is the sum of the eigenvector centralities of other actors with links to the actor [Bon72, Bon91]. This measure takes into account not just the number of links that an actor has, but also the quality of those links. Intuitively, a central actors is one whom many other central actors link to. The most well-known and successful application of eigenvector centrality is for ranking Web pages based on hyperlinks for Web search, e.g, PageRank [PBMW98], HITS [Kle99], and various other link analysis algorithms [BRRT05, Hav03].

Our work on rating-related behaviors is different from centrality in the following ways. Firstly, our work takes into account the context in which links are assigned (ratings) and factor that into the behavioral analysis. Centrality relies solely on structural properties of a graph, and ignores the meaning of links. Secondly, centrality is primarily based on the notion of popularity, with the number of links having a high impact on any centrality measure. This is incongruent with rating-related behaviors, which mostly are not affected by having many links (ratings).

Influence

Influence equates importance of actors to ability to propagate the adoption of an idea or a product to other actors in the network. The mode of propagation could be through

CHAPTER 2. SOCIAL NETWORK OVERVIEW

various channels such as word-of-mouth or persuasion. This measure finds application in viral marketing, which depends on identifying high-influence individual to promote products and services to their acquaintances [DR01, KKT03, LAH07, ONL02, RD02].

The propagation framework is as follows [KKT03]. Each actor is in one of two states: active or inactive. Initially, only one or a few *seed* actors are active, while the rest are inactive. The propagation of active state proceeds in discrete iterations. In each iteration, an inactive actor may get activated by its active neighbors. Actors that are active in the previous iterations remain active. The iterations terminate after a preset number of iterations, or when no further activation is possible. The influence of an actor (or a small subset of actors) is measured by using the actor(s) as seed actor(s) and counting the final number of active actors at the end of the iterations. The mechanism by which an actor is activated generally falls into either the *threshold model* or the *cascade model*.

Threshold Model. In the threshold model [Gra87], each actor a_j has a threshold activation value of θ_j , and the link weight w_{ij} from a_i to a_j reflects a_i 's degree of influence on a_j . Actor a_j is activated in the iteration when $(\sum_{\text{active } a_i \in \text{neighbors}(a_j)} w_{ij}) \geq \theta_j$.

Cascade Model. In the cascade model [GLM01], the link weight w_{ij} from a_i to a_j reflects the probability that a_i can successfully activate a_j . In each iteration of the propagation process, each active actor a_i is given a chance to activate an inactive neighbor a_j with a probability of success equal to w_{ij} .

Our work on rating-related behaviors is fundamentally different from influence. While we consider two-mode networks (involving reviewers and rated objects), influence considers one-mode networks of people. Influence models a context in which actors could be of two states (active and inactive) and studies how the propagation of active state may be facilitated by starting the propagation process with different seed actors. Our work models a context in which reviewers assign ratings on objects, and studies what behaviors (of reviewers and of objects) may affect the ratings assigned.

Reputation

Reputation is often equated with trustworthiness. In online settings, interaction between strangers is common. Thus, platforms that support such interactions (e.g., online auction sites) often institute a reputation system that allows users to evaluate how trusted an actor is by others in the network. All things being equal, one would rather transact with actors of higher reputation. There are two main criteria for inferring the reputation of an actor: *past behaviors* and *trust evaluation by others*.

Past Behaviors. One way to determine how trustworthy an actor will be in the future is to see how trustworthy the actor has been in the past. The auction site eBay¹ maintains a *feedback score* for each registered user. On completing a transaction, a buyer and a seller may give a feedback point to each other, which can be 1 (positive rating), 0 (neutral rating), or -1 (negative rating). The feedback score (reputation) of an actor is his/her running total of feedback points [RKZF00]. In product review site Epinions², a user may write product reviews and get paid based on the number of people who read the reviews. Each review may also be rated by other users. The reputation of a user is a function of the rating scores received by the user's reviews [CS01].

Trust Evaluation by Others. Some systems such as FOAF [GH06] and Epinions [GKRT04] maintain a social network, where each link denotes a trust relationship. Thus, another way to determine how trustworthy an actor is to see how many other actors in the network trust her [KSGM03, XL04]. For example, the work on EigenTrust [KSGM03] measures the reputation of an actor as the sum of the reputations of other actors with trust links to the actor (akin to eigenvector centrality applied on a network of trust relationship).

Reputation is related to our work on rating-related behaviors (Chapters 4, 5, and 6) in that many reputation systems make use of rating systems to allow users to rate past

¹www.ebay.com

²www.epinions.com

CHAPTER 2. SOCIAL NETWORK OVERVIEW

transactions or to indicate their trust levels on others. However, our work focuses on behaviors of reviewers and objects during the assignment of ratings itself, which apply to rating systems in general (not just those for quantifying reputation).

Anomaly

In contrast to centrality, anomaly equates importance to being different from or having few connections to other actors. For instance, key players (bosses) in a criminal network may intentionally keep a distance from others for fear of detection by the police and let their underlings carry out their wishes [XC03]. Finding anomalous actors is akin to outlier detection [AAR96, KN97], which is concerned with identifying data points that are situated at a distance from the majority of data points. In prior work, anomalous actors have been defined as those with low closeness centrality values [XC03], or those least visited by random walks starting from other actors in the network [SQCF05].

Our work on bias and controversy in rating networks (Chapter 4) identifies reviewers who tend assign scores very different from co-reviewers. Such reviewers may appear to be outliers. However, as rating networks are bipartite (with reviewers and objects), how outlying a reviewer is depends on how controversial the rated objects are. This notion of mutual dependency between actors is not usually factored into anomaly detection.

2.3.2 Path Analysis

The problem of path analysis can generally be expressed as follows: *given a social network and ≥ 2 seed actors, identify the set of “important” paths connecting the seed actors*. The important paths are those that are most likely undertaken from one seed actor to another. Prior work is organized based on how each defines what make up the important paths. As shown in the taxonomy in Figure 2.2, the four main criteria are: *graph-theoretic distance*, *electrical conductance*, *random walk*, and *novelty*.

Graph-theoretic Distance

Several distance measures in graph theory that could serve to measure the importance of a path include *shortest path* and *maximum flow*.

Shortest Path. The shortest path is the path with minimum number of links (for dichotomous links), or the path with maximum total weight (for valued links). This measure has been used to identify strongest association paths between entities in a criminal network [XC04]. For instance, if two criminals are known to be cooperating, they are likely to use the shortest path between them. Individuals along this association path are themselves potential suspects in the criminal activity.

Maximum Flow. In the maximum flow approach, the social network is modeled as a flow graph. One seed actor is designated the source node, and the other the sink node. Each link in the network is a channel for the flow of material, which is limited by the capacity (link weight). The maximum flow path allows the greatest flow of materials from the source to the sink.

Electrical Conductance

A social network could also be modeled as an electrical circuit. Each seed actor is assigned a potential (source node 1V and sink node 0V). Each link is like a resistor with a certain conductance value (link weight). The best path is the one that delivers the highest electrical current from the source node to the target node. The electrical conductance model for mining interesting connections between individuals in a social network was first proposed by [FMT04], and further improved upon by [KNV06].

Electrical conductance is superior to graph-theoretic distance measures in two ways. Unlike the shortest path approach, this model takes into account the popularity of intermediate nodes in a path. Popular nodes allow greater leakage of electricity, corresponding to weaker and incidental connections to a popular person that a normal person would

CHAPTER 2. SOCIAL NETWORK OVERVIEW

have. Unlike the maximum flow approach, this model takes into account the length of a path in determining interestingness. Longer paths accumulate resistance which impedes the flow of electricity, similar to weaker social relationship to be expected from a longer social path.

Random Walk

Another way to measure path importance is using the random walk mechanism. Random walk is a traversal of a social network graph, which starts from a seed actor and picks the next neighboring actor to visit randomly (either with uniform probability or with probability proportional to link weight). If we start independent random walks from each seed actor, intuitively the paths that are most commonly traversed by these random walks in aggregate are the most important paths connecting the seed actors. The work on center-piece subgraph [TF06] applies the random walk model to find interesting co-authorship connections. Unlike the electrical conductance model, the center-piece subgraph may also include good paths that connect only a subset of (not all) seed actors.

Novelty

Path importance may also be defined in terms of novelty or uniqueness. A given social network may consist of a few relations (e.g., friendship relation, work relation). Thus, a path may be constructed by links of a few different relations. The novelty of a path is how rarely the combination of relation types in its links can be found in other paths. A novel path captures a unique and exclusive relationship between the seed actors. For example, [LC03] discovered paths denoting student-teacher relationships based on their exclusive co-authorship with each other. [LC04] found that the only two mafia groups to be involved in a gang war in a simulated criminal database were connected by paths made up of novel combinations of evidence links (e.g., money transactions, meetings, killings).

2.3.3 Subgroup Analysis

In a social network, for every actor, there is a relatively small subset of other actors that the actor knows well; that small subset constitutes a subgroup. In general, members of a subgroup interact more frequently and intensively with other members than with non-members. A network consists of one or more subgroups, which may or may not overlap with each other. The subgroup analysis problem can be concisely stated as follows: *given a social network, identify the subgroups in the network*. In prior work, there are various definitions of what constitutes a subgroup. As shown in the taxonomy in Figure 2.2, these definitions fall into one of three categories: *connectivity*, *graph partitioning*, and *subgraph isomorphism*.

Connectivity-based Subgroups

Connectivity-based subgroups are defined in terms of how connected members in a subgroup are [Fre92, Fre96, WF94]. Here we look at three such criteria: mutuality, reachability, and nodal degree.

Mutuality. Mutuality-based subgroups, called *cliques*, are maximal complete subgraphs of at least three actors. This definition captures the idea of cohesiveness, where everyone knows everyone else. However, due to its strictness, cliques are relatively rare in real-life data.

Reachability. Reachability only requires that any pairwise members of a subgroup is reachable from each other through a path of a length not more than n links. If the path may involve an actor outside the subgroup, the subgroup is called *n-clique*. A more restrictive version, *n-clan*, can be derived by rejecting those *n-cliques* that require a path involving a non-member.

Nodal Degree. Another way to relax the mutuality requirement is to allow each actor to have a lower degree than mutuality would have required. Given k and n , a subgroup

of n members is termed a k -plex if at most k links can be missing from each actor to its neighbors, or a k -core, if at least k links must be present from each actor to its neighbors.

Graph Partitioning

Graph partitioning assumes that a social network consists of a set of disjoint subgroups. Finding those subgroups involves removing a set of links such that the social network graph is partitioned into disjoint subgraphs. This method has been used to find subgroups in networks with unsigned links as well as those with signed links.

Unsigned Links. In a network of unsigned links, the objective is to partition the graph into components, such that each component is relatively dense, but the cut (the set of links to be removed) between any two components is relatively sparse. As there could be many possible cuts, the best cut is the one that maximizes the value of some goodness function. This method has been used to partition a collection of newsgroups [BCMS04] and Web pages [FLG00, FLGC02] into subgroups consisting of newsgroups or Web pages of similar topics.

Signed Links. In a network of signed links, the objective is to partition the graph into components, by removing negative links, such that each component consists of as many positive links as possible. For example, [ARSX03] split contributors of newsgroups on controversial issues (e.g., politics, abortion) into two camps: those who are for or against a particular issue. [YCL07] split a network of political parties and a network of tribes into subgroups of similarly aligned parties/tribes.

Subgraph Isomorphism

Subgraph isomorphism assumes that a subgroup has a non-random pattern of linking among its members (subgraph pattern), which is shared by a number of other subgroups. Hence, finding subgroups within a network is equivalent to finding subgraph patterns

CHAPTER 2. SOCIAL NETWORK OVERVIEW

that have many isomorphic instances in the network. Below, we review two approaches to derive such subgroups: *Apriori-like algorithms* and *compression-based approach*.

Apriori-like Algorithms. A subgraph pattern is frequent if the number of isomorphic instances meets the specified threshold value. To reduce the space of subgraph patterns whose frequencies have to be determined, most of the proposed algorithms [CF06, IWM00, KK01, KK07, YH02, YH03] follow the general principle of the Apriori algorithm that was first proposed by [AS94] for mining association rules from transaction databases. Adapted to graph data, the principle states that a subgraph pattern has a higher frequency than any of its supergraphs (other patterns that subsume the subgraph). If a subgraph pattern is not frequent, none of its supergraphs need to be considered.

Compression-based Approach. Unlike the apriori-like algorithms that find all subgraph patterns whose frequencies meet the threshold, the compression-based approach employs a greedy algorithm to find a subset of subgraph patterns that together result in a good compression of the original graph [CH00]. Using the Minimum Description Length (MDL) principle, compression is achieved by replacing all isomorphic instances of a subgraph pattern with a more concise representation called “concept”. [MH04] used this approach to identify substructures in a terrorist network, revealing the chain-like communication channels used by terrorist cells.

2.4 Social Network Application

Below, we list a number of applications (mostly online applications) with social networking aspects. While the list is by no means exhaustive, it sufficiently paints a picture of how the techniques reviewed earlier in this chapter may be used in real-life applications.

Criminal Network Analysis

Organized crimes involve a group of criminals cooperating with each other to accomplish an illegal purpose, such as terrorism [Kre02, Rot02], drug trafficking, money laundering,

CHAPTER 2. SOCIAL NETWORK OVERVIEW

and other gang-related offenses [XC05]. The organized nature of these crimes means that different members of the group take on specialized roles and communicate selectively with only a subset of other members. Since the relationship and communication patterns among the criminals can naturally be mapped onto a social network, various social network analytical tools can be employed to learn of the key actors [XC03], the various paths connecting key actors to identify other collaborators who mediate communication between the key actors [XC04], the various subgroups which might be responsible for different tasks [XC05], or the evolution of the criminal network [XMKC04]. Examples of such criminal analysis applications that are documented in the literature include *CrimeNet Explorer* [XC05] and *Terrorism Knowledge Discovery Project* [RQC⁺04].

Online Social Media

Online social media refers to online applications for disseminating and sharing information that also support socially-oriented features. Examples of such applications include: blogs (e.g., LiveJournal³), wikis (e.g., Wikipedia⁴), content sharing (e.g., Flickr⁵ for photos, YouTube⁶ for videos), online communities (Facebook [LES06], Friendster [Boy04], MySpace [PL05]), and social bookmarking (e.g., del.icio.us). Such applications often allow users to assign *tags* (textual annotations) to objects in order to collaboratively organize content, to assign *ratings* to collaboratively evaluate content, and to maintain one's *social network* in order to track the latest goings-on, activities, and interests of friends. The social aspects of these activities lend themselves to social network analysis. For example, by analyzing the pattern of hyperlinking among blog posts, we can identify the opinion leaders among bloggers [SHT07]. By analyzing the edit history of Wikipedia articles, we can identify the most authoritative authors [HLS⁺07].

³www.livejournal.com

⁴en.wikipedia.org

⁵www.flickr.com

⁶www.youtube.com

Social Search

Social search refers to querying one's social network to look up interesting actors or paths. For instance, one may look for actors whose profile fit the description given in a query, e.g., someone looking for potential dates [LES06]. Alternatively, one may look for actors holding a specific piece of information [YS03, ZvA04]. This is especially useful for information that is not widely available and may not be indexed in public databases. For example, the answer to the question “Which camera shop in my local neighborhood would offer a good deal to students of my university?” is probably known by a university friend who is an avid photographer. One may also search for interesting association paths. *ReferralWeb* [KSS97] allows a user to explore the chains of referrals leading to a target actor. Users of such a system may be a businessman who wishes to get an introduction to a potential business partner or a graduating student who needs a reference letter from a well-known academician.

Recommender Systems

Recommender systems are online applications that generate personalized recommendations (e.g., which book to buy) based on information provided by the users [AT05, HKTR04, RV97]. Some recommender systems require the user to manually enter a personal profile of interests, preferences, or expertise. Others may infer this information implicitly from the user's past activities, e.g., user's purchasing history at Amazon⁷ or user's ratings on movies at GroupLens⁸. A similarity-based social network can then be constructed based on this information. The system could then generate recommendations to an actor based on what other similar or related actors have purchased or rated highly.

⁷www.amazon.com

⁸www.grouplens.org

Academic Peer Review

Peer review refers to the collaborative exercise in which academicians evaluate each other's work, in order to determine which papers should be accepted for publications in conference proceedings and journals, or which research proposals should be granted funding. Questions that often come up during the peer review process include how to identify the best papers or proposals taking into account the varying rating scores assigned by different reviewers [RW01, WMM02], and how to best assign reviewers to objects (papers or proposals) taking into account such factors as the match in topics between reviewers and objects and the workload of reviewers [DN92, GS97, HP06].

Social network techniques would likely be useful in deriving the answers to these questions as many academic activities can be mapped onto social network representation. For example, there is a wealth of research on social networks based on co-authorship [BHKL06, KNV06, LC03, TF06], co-citation (being cited together in publications) [CC99, Cul86, SWB97, Sma73, Sma77, WM98], bibliographic coupling (citing common publications) [Kes63], etc. Social network analysis can be employed to generate insights that would help to improve and inform the peer review process, e.g., identifying the authorities in specific fields [WM98], or tracking which communities are growing or shrinking [BHKL06].

Chapter 3

Social Network Discovery by Mining Spatio-Temporal Events

3.1 Overview

With greater use of mobile computing and Web technologies, comes a greater amount of data about users on the move or on the Internet. Knowingly or unknowingly, users are tracked when they carry wireless devices or when they visit Web pages at different sites. Such user data have spatio-temporal properties. We aim to mine from such data movement behaviors that may suggest social associations among users. We focus on one movement behavior known as co-occurrence, whereby two or more users collocating around the same time implies that there is some interaction among them.

We consider a specific type of spatio-temporal data. We use $\mathcal{D}$ to denote the collection of tuples. Each tuple $d = \langle a, t, s \rangle$ in $\mathcal{D}$ codes for a time $d.t$ and a location $d.s$ at which an actor $d.a$ is observed. Each time value is expressed at a particular atomic unit (e.g., seconds). It is not necessary that the actors' locations are tracked at regular intervals. We use semantic locations, whereby each location has a coarse granularity and has some semantic meaning. Examples include physical locations (e.g., rooms) and cyber locations (e.g., URL addresses). Such locations can be tracked more easily due to their coarse granularity, and other location models such as xyz coordinates could be transformed into semantic locations.

Our objective is to discover a social network graph $G(G_V, G_E)$ from $\mathcal{D}$, in which the nodes in G_V represent actors and the edges in G_E represent weighted associations between pairs of actors. These associations will be inferred from spatio-temporal co-occurrences among interacting actors without having the direct knowledge about the events behind the co-occurrences.

An association between a pair of actors can be inferred from their participation in common events [WF94]. An event refers to some form of social collectivity or activity (e.g., clubs, meetings). For instance, conferences gather academicians to exchange knowledge and contacts. This leads to the notion of event-supported links between two actors, where the link weight is related to the number of events common to the two actors.

We propose a novel model called *STEvent* (for *Spatio-Temporal Event*) that discovers spatio-temporal events from the data and uses the events to build a network of associations among actors. Spatio-temporal events are distinct co-occurrences of several actors in space and time, which may reveal rendezvous acts by these actors. Common participation by a pair of actors in many events suggests the existence of an association.

The rest of this chapter is organized as follows. Section 3.2 describes and formalizes our spatio-temporal event model. Section 3.3 discusses a two-phase algorithm that implements the proposed model efficiently. We report our experimental results on *Cyber Location Data* (capturing users' movement behavior over cyber locations) in Section 3.4 and on *Physical Location Data* (capturing users' movement behavior over physical locations) in Section 3.5. Section 3.6 summarizes our research findings.

3.2 STEvent: Spatio-Temporal Event Model

3.2.1 Event Definition

The formal definition of a spatio-temporal event is given below, assuming certain user-specified semantic location granularity and time duration δ_{max} .

Definition 3.1 *A spatio-temporal event is a subset of tuples, $e \subseteq \mathcal{D}$, meeting all of the conditions below:*

- $\forall d_i, d_j \in e, d_i.s = d_j.s,$
*i.e., tuples are of the **same location***
- $\forall d_i, d_j \in e, |d_i.t - d_j.t| \leq \delta_{max},$
*i.e., tuples are **separated in time by at most** δ_{max}*
- $|\{d.a \mid d \in e\}| \geq 2,$
*i.e., tuples involve **two or more actors***
- *for any event $e' \subseteq \mathcal{D}, (e' \subseteq e) \vee (e \subseteq e') \Rightarrow (e' = e),$*
*i.e., each event is **maximal***

The first two conditions specify the constraints of a co-occurrence. The semantic location granularity and time interval δ_{max} limit the furthest that actors could be separated in space and time respectively to be considered as co-occurring with one another. The third condition requires that an event must involve more than one actor. Finally, requiring each event to be maximal places a constraint on the number of times that a tuple may be included in events. Its purpose is to ensure, as much as possible, that each event stands for a single underlying interaction.

We enumerate some notations related to events in Table 3.1. The set of all events defined over database $\mathcal{D}$ is denoted as $\mathcal{E}$. An event $e \in \mathcal{E}$ has several properties. The set of distinct actors represented by tuples in an event is its *actor set*, $e.\mathcal{A} = \{d.a \mid d \in e\}$. An event's *start time*, $e.t^- = \min_{d \in e}\{d.t\}$, and *end time*, $e.t^+ = \max_{d \in e}\{d.t\}$, are the times of its earliest and latest tuples respectively. Correspondingly, its *duration* is defined by $e.\delta = |e.t^- - e.t^+|$. The *area* $e.\Delta$ of an event measures the scope of its semantic location $e.s$. We do not specify the exact form of this property, other than that for two locations,

Table 3.1: Table of Notations

<i>Notation</i>	<i>Description</i>
a	an actor
$\mathcal{D}$	the set of all tuples
d	a tuple
$\mathcal{E}$	the set of all events
e	an event
$e.\mathcal{A}$	event e 's actors
$e.s$	event e 's location
$e.t^-$	event e 's start time
$e.t^+$	event e 's end time
$e.\delta$	event e 's duration
$e.\triangle$	event e 's area
$e.w$	event e 's weight
$e.w_{p-s}$	event e 's spatial precision
$e.w_{p-t}$	event e 's temporal precision
$e.w_{u-s}$	event e 's spatial uniqueness
$e.w_{u-t}$	event e 's temporal uniqueness
$e.l$	event e 's location level
$\langle a_x, a_y \rangle$	the link between actors a_x and a_y
$\langle a_x, a_y \rangle.w$	the weight of link $\langle a_x, a_y \rangle$

where one contains the other, the area value should be monotonic with respect to the granularity of the semantic location, i.e., the containing location should have no smaller area than the contained location. Lastly, its *weight* $e.w$ is a goodness measure related to the quality of relationship among actors of that event. The remaining notations will be explained when introduced in subsequent sections.

3.2.2 Event Weight

In assigning weight to events, we use an event's spatial and temporal properties to gauge its adeptness in representing an actual interaction. The four weight measures that we adopt are spatial and temporal precision, as well as spatial and temporal uniqueness.

- *Spatial precision* ($e.w_{p-s}$) measures how closely in space actors are from each other when participating in an event. A finer location granularity should have a higher spatial precision value. For instance, we would be more confident that two people are friends if they stand very closely together than if they stand widely apart. We

define the spatial precision $e.w_{p-s} \in (0, 1]$ of an event e as the inverse of the event's area $e.\Delta$, normalized with respect to the maximum such value as shown in Eq. 3.1.

$$e.w_{p-s} = \frac{\frac{1}{e.\Delta}}{\max_{e' \in \mathcal{E}} \left\{ \frac{1}{e'.\Delta} \right\}} \quad (\text{Eq. 3.1})$$

- *Temporal precision* ($e.w_{p-t}$) measures the closeness in time between the occurrences of the earliest and the latest actors. When several actors are spotted in quick succession to each other, they are more likely to have been related. We define the temporal precision $e.w_{p-t} \in (0, 1]$ of an event e in terms of the event's duration as shown Eq. 3.2. Addition of a unit of time δ_{unit} to the denominator is meant to ensure a non-zero minimum value for the case of $e.\delta = \delta_{max}$.

$$e.w_{p-t} = 1 - \frac{e.\delta}{(\delta_{max} + \delta_{unit})} \quad (\text{Eq. 3.2})$$

- *Spatial uniqueness* ($e.w_{u-s}$) measures the uniqueness of an event's location among other events. Intuitively, co-occurrences at unique locations are more predictive of meaningful interaction. The spatial uniqueness function $e.w_{u-s} \in (0, 1]$ is given in Eq. 3.3. Counting only events other than itself ensures a non-zero minimum value.

$$e.w_{u-s} = 1 - \frac{|\{e' \in \mathcal{E} \mid (e' \neq e) \wedge (e'.s = e.s)\}|}{|\mathcal{E}|} \quad (\text{Eq. 3.3})$$

- *Temporal uniqueness* ($e.w_{u-t}$) measures the uniqueness of an event's time. An event that takes place when few other events are taking place are less likely to have been due to chance. Two events overlap each other temporally if they share at least a non-zero period of time. This is reflected in the function for temporal uniqueness $e.w_{u-t} \in (0, 1]$ given in Eq. 3.4.

$$e.w_{u-t} = 1 - \frac{|\{e' \in \mathcal{E} \mid (e' \neq e) \wedge (e'.[t^-, t^+] \cap e.[t^-, t^+] \neq \emptyset)\}|}{|\mathcal{E}|} \quad (\text{Eq. 3.4})$$

Finally, **event weight** $e.w \in (0, 1]$ is the product of the four preceding measures as shown in Eq. 3.5. Having non-zero value for each measure prevents any one measure from nullifying the contribution of the other measures. An event's weight can be interpreted as the probability that the event predicts an actual association between participating actors or the strength of such an association.

$$e.w = e.w_{p-s} \times e.w_{p-t} \times e.w_{u-s} \times e.w_{u-t} \quad (\text{Eq. 3.5})$$

3.2.3 Supporting Locations with Multi-Level Granularity

Earlier, we define an event in terms of locations at a single, user-specified level of granularity. We now extend that definition to include locations with multiple levels of granularity.

We associate a location with a position (e.g., URL address, physical address), a region with a quantifiable area, and a granularity level (how specific the indicated position is). For example, a location can be a cyber location, positioned at URL address <http://www.ntu.edu.sg/sce/research-areas>. Its level 1 is www.ntu.edu.sg, level 2 is [sce](http://www.ntu.edu.sg/sce), and level 3 is [research-areas.asp](http://www.ntu.edu.sg/sce/research-areas.asp). A location may also be a physical location, with the following granularity levels: building (level 1), floor (level2), and room (level 3). Each higher level is a more specific location than the previous one, with higher granularity level and smaller area.

We code granularity levels as $l \in \{1, 2, \dots, l_{max}\}$, with 1 and l_{max} representing the coarsest and finest levels of granularity respectively. For a tuple d with location $d.s$, we refer to its level of granularity as $d.s.l$. We represent a containing location at a level l' coarser than $d.s.l$ as $d.s(l')$. A tuple that supports an event at a particular location level also supports all coarser levels.

CHAPTER 3. SOCIAL NETWORK DISCOVERY BY MINING SPATIO-TEMPORAL EVENTS

We can transform the database $\mathcal{D}$ into $\mathcal{D}' = \{d_j = \langle a, t, d_i.s(l') \rangle \mid d_i \in \mathcal{D}, 1 \leq l' \leq d_i.s.l\}$ such that each tuple $d_i \in \mathcal{D}$ is replaced by l_{max} number of tuples $d_j \in \mathcal{D}'$, for the same actor and the same time value but with locations expressed at various levels of granularity. Thus, the only necessary change to Definition 3.1 involves replacing $\mathcal{D}$ by $\mathcal{D}'$.

If no natural area quantity is known (e.g., for URL addresses), we propose quantifying a location's area as a function of its granularity level. The containing location should have an area at least as large as the contained locations of finer granularity. We propose that an event's area is the inverse of the granularity level of its location. If for an event e , its location level is $e.l$, then we may express its area as $e.\Delta = \frac{1}{e.l}$. Using this area function, the spatial precision can be rewritten as in Eq. 3.6.

$$e.w_{p-s} = \frac{e.l}{\max_{e' \in \mathcal{E}} \{e'.l\}} \quad (\text{Eq. 3.6})$$

If several events involve the same set of actors at around the same time, then we should take into account only the one with the finest location granularity. Knowing that two actors are in the same city is redundant if we know they are at the same home unit.

Definition 3.2 *An event e_{sub} is a subevent of another event e_{sup} , or alternatively e_{sup} is a superevent of e_{sub} , if the following conditions are met:*

- $(e_{sup}.\Delta > e_{sub}.\Delta) \wedge (e_{sup}.s \text{ contains } e_{sub}.s)$,
i.e., the superevent's location has a coarser granularity and contains the subevent's
- $(e_{sup}.t^- \leq e_{sub}.t^-) \wedge (e_{sub}.t^+ \leq e_{sup}.t^+)$,
i.e., the subevent's time period sits within the superevent's
- $e_{sub}.\mathcal{A} \subseteq e_{sup}.\mathcal{A}$,
i.e., actors participating in the subevent participate in the superevent as well

The first condition captures the essence of the subevent-superevent relationship as having arisen from locations of different granularity levels. If a subevent and its superevent have arisen from the same tuples in the original database $\mathcal{D}$, then the latter two conditions are natural consequents of the first condition. The definition of subevent-superevent relation is used to determine events associated with a pair of actors as described in the following section.

3.2.4 Event-based Links

In deriving links from events, we may want to impose a certain threshold (*min_event_weight*) on the minimum weight that an event should have to support links between actors.

Definition 3.3 *An event e supports a link $\langle a_x, a_y \rangle$ between two actors, a_x and a_y , if $(\{a_x, a_y\} \subseteq e.\mathcal{A}) \wedge (e.w \geq \text{min_event_weight})$, for a given threshold *min_event_weight*.*

For a pair of actors, we group together all such events as the *event set* of the pair. Furthermore, owing to the multi-level granularity of semantic locations, we should take care to only include the most restrictive subevents supporting a linkage between the pair.

Definition 3.4 *For a given link $\langle a_x, a_y \rangle$, its event set is $\mathcal{E}_{\langle a_x, a_y \rangle} \subseteq \mathcal{E}$, such that:*

- $\mathcal{E}_{\langle a_x, a_y \rangle} = \{e \in \mathcal{E} \mid (\{a_x, a_y\} \subseteq e.\mathcal{A}) \wedge (e.w \geq \text{min_event_weight})\}$
- $\forall e \in \mathcal{E}_{\langle a_x, a_y \rangle} \nexists e' \in \mathcal{E}_{\langle a_x, a_y \rangle}, e' \text{ is a subevent of } e$

The size of the event set hints at the strength of relationship between the pair of actors. Intuitively, the greater the cardinality of an event set, more events profess to establish the linkage between the concerned pair, and correspondingly not only the linkage between the pair is more likely, it is also likely to be stronger. To factor this in quantifying the relationship strength of a pair, we define the **link weight** for a pair of actors $\langle a_x, a_y \rangle$ by the summation of the weight of the events in its event set, as given in Eq. 3.7.

$$\langle a_x, a_y \rangle.w = \sum_{e \in \mathcal{E}_{\langle a_x, a_y \rangle}} e.w \quad (\text{Eq. 3.7})$$

To control the number of links to be included in the output social network, we may impose a threshold *min_link_weight*.

Definition 3.5 *A link $\langle a_x, a_y \rangle$ exists if $\langle a_x, a_y \rangle.w \geq \text{min_link_weight}$, for a given threshold *min_link_weight*.*

We can now restate the spatio-temporal event-based social network discovery problem more concretely. Given database $\mathcal{D}$, maximum duration δ_{max} , thresholds *min_event_weight* and *min_link_weight*, find social network graph $G(G_V, G_E)$, where:

- $G_V = \{a \mid \exists \langle a_x, a_y \rangle \in G_E, a \in \{a_x, a_y\}\}$
- $G_E = \{\langle a_x, a_y \rangle \mid \langle a_x, a_y \rangle.w \geq \text{min_link_weight}\}$

3.3 Computational Algorithms

We propose algorithms to solve the above-mentioned problem in two phases: (a) *Phase 1 - Construction of Events* and (b) *Phase 2 - Construction of Links*.

As events are subsets of tuples (subject to conditions in Definition 3.1), and each tuple may support multiple events of different location granularity levels, the complexity of event construction (Phase 1) is driven mainly by the number of tuples $|\mathcal{D}|$ and the number of location granularity levels l_{max} . Particularly, $|\mathcal{D}|$ is likely a large number, as data points may be continuously generated as time passes. A brute force approach that examines every sequence of tuples of size 2, size 3, and so on up to size $|\mathcal{D}|$, to see which ones meet the event definition will be very time consuming, with up to $O(|\mathcal{D}|^2)$ complexity. Our proposed algorithm for Phase 1, with $O(|\mathcal{D}| \times l_{max})$ complexity, achieves better efficiency, requiring only a single pass through all the tuples.

Constructing links from events requires finding out for each link the set of supporting events and the weights of those events. Hence, the complexity of link construction (Phase 2) is driven mainly by the number of actors $|\mathcal{A}|$ and the number of events $|\mathcal{E}|$. Both are likely large numbers. Particularly, the number of events keeps growing with more tuples being generated over time. A brute force approach that iterates over all possible links to find out the set of supporting events ($O(|\mathcal{A}|^2 \times |\mathcal{E}|)$) and iterates over all events to determine event weights ($O(|\mathcal{E}|^2)$) will be very time consuming. Our proposed algorithm for Phase 2 employs strategies such as indexing events by space and time, which speeds up event weight determination. It also maintains a temporary set $G_{E_{cand}}$ of candidate links (links with at least one supporting event), which prunes the number of links in consideration, as $|G_{E_{cand}}|$ will be much smaller than the total number of possible links ($|\mathcal{A}| \times (|\mathcal{A}| - 1) \div 2$). These strategies result in much better efficiency for Phase 2, with a final complexity of $O(|\mathcal{E}| \times \log |\mathcal{E}| + |G_{E_{cand}}|)$.

3.3.1 Phase 1 - Construction of Events

This phase deals with parsing the database, creating events, and assigning tuples to these events. Algorithm 3.1 lists the required steps. It takes as input the database $\mathcal{D}$ and the maximum duration δ_{max} . It returns as output the set of all events $\mathcal{E}$ constructed from $\mathcal{D}$.

First, two sets of events, $\mathcal{E}_{cand}$ and $\mathcal{E}$, are initialized as empty sets. $\mathcal{E}_{cand}$ is a temporary store of recently created events that may still be affected by incoming tuples. $\mathcal{E}$ is the output set of events. We also initialize and maintain two index structures: $I_{s \rightarrow e}$ that maps a location to events of that location, and $I_{t \rightarrow e}$ that maps a time interval to events with intersecting time periods. These indices will be used in Phase 2 to determine event weights efficiently.

Tuples are traversed in chronological order (line 3). Each tuple d is assigned to a candidate event (an event in $\mathcal{E}_{cand}$) that has the same location as d and that is not

CHAPTER 3. SOCIAL NETWORK DISCOVERY BY MINING SPATIO-TEMPORAL EVENTS

Input: database $\mathcal{D}$, maximum duration δ_{max} , min_link_weight

Output: events $\mathcal{E}$, index of events by location $I_{s \rightarrow e}$, index of events by time $I_{t \rightarrow e}$

Algorithm:

```

1:  $\mathcal{E} = \emptyset$ ,  $\mathcal{E}_{cand} = \emptyset$ 
2: initialize indices  $I_{s \rightarrow e}$  and  $I_{t \rightarrow e}$ 
3: for each tuple  $d \in \mathcal{D}$  in the order of  $d.t$  do
4:   for each event  $e \in \mathcal{E}_{cand}$ ,  $(d.t - e.t^- > \delta_{max})$  do
5:     if  $(|e.\mathcal{A}| > 1) \wedge (\nexists e' \in \mathcal{E}, (e \subseteq e'))$  then
6:        $\mathcal{E} = \mathcal{E} \cup \{e\}$ 
7:        $updateIndex(I_{s \rightarrow e}, e.s, e)$ 
8:        $updateIndex(I_{t \rightarrow e}, e.[t^-, t^+], e)$ 
9:     end if
10:     $\mathcal{E}_{cand} = \mathcal{E}_{cand} - \{e\}$ 
11:     $e_{child} = \{d_i \in e \mid d.t - d_i.t \leq \delta_{max}\}$ 
12:     $\mathcal{E}_{cand} = \mathcal{E}_{cand} \cup \{e_{child}\}$ 
13:  end for
14:  for each location granularity level  $l = 1$  to  $l_{max}$  do
15:    if  $\exists e \in \mathcal{E}_{cand}, (e.s = d.s(l))$  then
16:       $e = e \cup \{d\}$ 
17:    else
18:      create new event  $e = \{d\}$  with  $e.s = d.s(l)$ 
19:       $\mathcal{E}_{cand} = \mathcal{E}_{cand} \cup \{e\}$ 
20:    end if
21:  end for
22: end for
23: return  $\mathcal{E}$ ,  $I_{s \rightarrow e}$ ,  $I_{t \rightarrow e}$ 

```

Figure 3.1: Algorithm: Construction of Events

expired (whose duration would not breach the limit of δ_{max} with the addition of d). If there is no such candidate event, a new one will be created. A tuple may update several candidate events of different location granularity levels (lines 14–21).

$\mathcal{E}_{cand}$ is continually cleared of expired events whose temporal properties do not allow them to accept more tuples. If such events are well-constructed, namely: they consist of at least two actors, and are not just a subset of another event, they are transferred to the output set $\mathcal{E}$. The indices $I_{s \rightarrow e}$ and $I_{t \rightarrow e}$ are also updated with these events (lines 4–9).

When an event e expires, the event e is removed from $\mathcal{E}_{cand}$. A new *child event*

is created to replace e in $\mathcal{E}_{cand}$. The child event e_{child} is “descended” from e , being a subset of e containing tuples less than δ_{max} apart from the incoming tuple d , such that $|d.t - e_{child}.t^-| \leq \delta_{max}$. Therefore, at any point of time, there will be only one event of a particular location in $\mathcal{E}_{cand}$. (lines 10–12).

After all the tuples have been traversed, the final set of events $\mathcal{E}$ is returned as output of this phase (line 23).

To gauge the complexity of the algorithm, we look at the most deeply-nested iteration, which is the updating of events with the current tuple (lines 15–16). This step is done once for every level of location granularity (up to l_{max} iterations), and for every tuple of the database ($|\mathcal{D}|$ iterations). In the worst case, the complexity of this phase is $O(|\mathcal{D}| \times l_{max})$.

3.3.2 Phase 2 - Construction of Links

In this phase, the events $\mathcal{E}$ generated in the previous phase are evaluated, and links are generated from them. As output, this phase returns the nodes G_V and the links G_E of the desired social network graph $G(G_V, G_E)$. Algorithm 3.2 describes the required steps.

First, we initialize G_V , G_E , and G_{Ecand} as empty sets (line 1). G_{Ecand} is a temporary store of links. In the first outermost loop iterating over each event (lines 2–18), the algorithm computes event weights and determines the set of supporting events for each link. Computing spatial and temporal precisions ($e.w_{p-s}$ and $e.w_{p-t}$) are trivial if the maximum duration and area are known beforehand (line 3). However, computing spatial and temporal uniqueness ($e.w_{u-s}$ and $e.w_{u-t}$) requires knowing the count of events sharing the same spatial or temporal properties, which can be derived by querying the indices $I_{s \rightarrow e}$ and $I_{t \rightarrow e}$ (lines 4–7). Event weight is the product of the four measures (line 15).

If an event’s weight is above the threshold *min_event_weight*, this event can support links between pairs of actors (lines 9–17). An event of n actors supports $n(n-1)/2$ links. These links are candidate links, entered into the temporary store G_{Ecand} , as the ultimate

CHAPTER 3. SOCIAL NETWORK DISCOVERY BY MINING SPATIO-TEMPORAL EVENTS

Input: $\mathcal{E}$, min_event_weight , min_link_weight , $I_{s \rightarrow e}$, $I_{t \rightarrow e}$

Output: nodes G_V , links G_E

Algorithm:

```

1:  $G_V = \emptyset$ ,  $G_E = \emptyset$ ,  $G_{E_{cand}} = \emptyset$ 
2: for each event  $e \in \mathcal{E}$  do
3:   compute  $e.w_{p-s}$  and  $e.w_{p-t}$ 
4:    $countEventsSharingLocation = queryIndex(I_{s \rightarrow e}, e.s)$ 
5:    $countEventsSharingTime = queryIndex(I_{t \rightarrow e}, e.[t^-, t^+])$ 
6:    $e.w_{u-s} = 1 - countEventsSharingLocation/|\mathcal{E}|$ 
7:    $e.w_{u-t} = 1 - countEventsSharingTime/|\mathcal{E}|$ 
8:    $e.w = e.w_{p-s} \times e.w_{p-t} \times e.w_{u-s} \times e.w_{u-t}$ 
9:   if  $e.w \geq min\_event\_weight$  then
10:    for each pair  $a_x, a_y \in e.\mathcal{A}$  do
11:       $G_{E_{cand}} = G_{E_{cand}} \cup \{\langle a_x, a_y \rangle\}$ 
12:      if  $\nexists e' \in \mathcal{E}_{\langle a_x, a_y \rangle}$ , ( $e'$  subevent of  $e$ ) then
13:        remove superevents of  $e$  from  $\mathcal{E}_{\langle a_x, a_y \rangle}$ 
14:         $\mathcal{E}_{\langle a_x, a_y \rangle} = \mathcal{E}_{\langle a_x, a_y \rangle} \cup \{e\}$ 
15:      end if
16:    end for
17:  end if
18: end for
19: for each link  $\langle a_x, a_y \rangle \in G_{E_{cand}}$  do
20:    $\langle a_x, a_y \rangle.w = \sum_{e \in \mathcal{E}_{\langle a_x, a_y \rangle}} (e.w)$ 
21:   if  $\langle a_x, a_y \rangle.w \geq min\_link\_weight$  then
22:      $G_E = G_E \cup \{\langle a_x, a_y \rangle\}$ 
23:      $G_V = G_V \cup \{a_x, a_y\}$ 
24:   end if
25: end for
26: return  $G_V$ ,  $G_E$ 

```

Figure 3.2: Algorithm: Construction of Links

weight of these links are not yet known. For each candidate link $\langle a_x, a_y \rangle$, its event set $\mathcal{E}_{\langle a_x, a_y \rangle}$ is updated while keeping watch of subevent-superevent relationships in its event set. The outcome is the set of candidate links $G_{E_{cand}}$.

In the second outermost loop (lines 19–25), the weight of each candidate link is evaluated by summing up the weights of events due to the pair of actors. Links whose weights are beyond the required threshold min_link_weight are entered into the output

set of links G_E . The corresponding actors are also included in the set of nodes G_V . The two sets, which is a graph representation of the desired social network, are then returned as output of this phase (line 26).

Complexity-wise, the first outermost loop iterates through $|\mathcal{E}|$ events. Determining event weights requires querying the indices, which involves $\log |\mathcal{E}|$ complexity. The generation of candidate links is more difficult to estimate. For an event e , the number of pairs generated would be $|e.\mathcal{A}|(|e.\mathcal{A}| - 1)/2$, but the average value of $|e.\mathcal{A}|$ is not known beforehand. A simplifying assumption is that each event introduces an equal number of pairs into the candidate set, in which case the number of candidate links per event is $|G_{Ecand}|/|\mathcal{E}|$. The estimated complexity of the first outermost loop is then $O(|\mathcal{E}| \times (\log |\mathcal{E}| + |G_{Ecand}|/|\mathcal{E}|)) = O(|\mathcal{E}| \times \log |\mathcal{E}| + |G_{Ecand}|)$. Given that the second outermost loop has a complexity of $O(|G_{Ecand}|)$, the overall complexity of this phase is $O(|\mathcal{E}| \times \log |\mathcal{E}| + |G_{Ecand}|)$.

3.4 Experiments on Cyber Location Data

The main objective of experiments is to verify the validity of the links discovered by *STEvent*, by examining the demographic similarity of pairs of actors. In addition, we study the behavior of our proposed algorithms with different parameter settings. We briefly explore the computational efficiency of *STEvent*.

3.4.1 Data

The data were collected as a log of Web pages (given by URL addresses) accessed by users of the wireless network in our campus. The users include undergraduate/graduate students and staff members. We call this *Cyber Location Data* (or *cyberdata*). Each tuple $\langle a, t, s \rangle$ consists of a user login name a , a time stamp t , and a URL address s . To protect privacy, all user login names were anonymized. While not a location in the geographical

Table 3.2: Parameter Values (*cyberdata* Aug-Sep04)

Parameter	Default Value	Range
$ \mathcal{D} $	2.5 million	0 – 2.5 million
δ_{max}	2 hours	10 minutes – 16 hours
<i>min_link_weight</i>	0	0 – 100
<i>min_event_weight</i>	0	0

sense of the word, a URL address possesses some semantic meaning as coded by the words forming the address as well as by the content of the Web page that it points to.

We pre-processed this data in the following way. Although the data spanned the period from August 2004 to March 2005, we did not use data for November and December 2004 as the usage level was very low during this period, which was the university’s holiday period. We retained only data concerning users who appeared at least once in each of the six remaining months (August to October 2004 and January to March 2005). There were a total of 533 such users. We also opted to use a single level of location granularity, choosing the URL hostname for most experiments. For example, for the URL `http://www.ntu.edu.sg/sce/research-areas.asp`, the hostname is `www.ntu.edu.sg`. There were about 131 thousand unique URL hostnames. The data size after pre-processing was about 9.5 million tuples or 550MB.

3.4.2 Demographic Similarity

Perhaps the ideal way to verify the relationships extracted by *STEvent* is to seek direct confirmation from the concerned actors. However, the data provider ruled out approaching the concerned actors for privacy reasons. A result from sociology is that people who are more closely related tend to have greater similarity to each other [Car91]. We propose an alternative verification scheme, which looks at whether strongly-related pairs of actors are more likely to be similar than any pair of two randomly-picked actors.

We run the algorithms on the portion of *cyberdata* for the two-month period *Aug-Sep04*, with a single location granularity level (URL hostname) and the parameter values

Table 3.3: Demographic Similarity (*cyberdata* Aug-Sep04)

<i>Common Features</i>	<i>Random (%)</i>	<i>STEvent (%)</i>
at least 1	55.5	86.0
at least 2	18.5	22.0
at least 3	3.3	5.3

given in Table 3.2 ($\delta_{max} = 2h$, $min_event_weight = 0$). We extract the top 100 links in terms of weight and call this our *STEvent* result set. For comparison, we draw links at random from the same population of actors and call it the *Random* result set.

Demographic Similarity for *cyberdata* Aug-Sep04

For each actor in either result set, we obtain information on up to three attributes namely: *department* (e.g., business, biology, civil engineering), *status* (e.g., undergraduate, post-graduate, staff), and *year of admission* (e.g., 2004). We count the number of common attribute values between each pair of actors. Higher values indicate higher similarity.

STEvent Result Set. Table 3.3 shows the distribution of the number of common attribute values among the 100 pairs of actors in *STEvent* result set. Since not all pairs can be compared on all three attributes, we present the number of pairs with at least n common attributes as a fraction of all pairs that can be compared on n attributes. For instance, in Table 3.3, for *STEvent*, 5.3% of all pairs who have all three attributes present have all three attribute values in common.

Random Result Set. Table 3.3 also shows the corresponding distribution for the *Random* result set, obtained by progressive sampling on the same population of actors. Evidently, *STEvent*'s distribution shows greater similarity between pairs than the *Random*'s. Statistical χ^2 goodness-of-fit test [WMMY02] at 5% level of significance also suggests that the *STEvent*'s distribution is sufficiently dissimilar from *Random*'s to imply that the improvement by *STEvent* over *Random* is significant.

We further examine the event locations for a select set of top-ranked pairs to learn of the nature of these relationships. In Table 3.4, we list twelve pairs who are among the

CHAPTER 3. SOCIAL NETWORK DISCOVERY BY MINING SPATIO-TEMPORAL EVENTS

Table 3.4: Highly Similar Event-based Pairs (*cyberdata* Aug-Sep04)

<i>Pairs</i>		<i>Common Attributes</i>	<i>Sample URL Locations</i>
1	a_1	Postgraduate	Center for Aerospace Structures, Univ. of Colorado
	a_2	Civil Engineering 2003	South East University (China) ScienceDirect Digital Library
2	a_2	Postgraduate	Singapore Millennium Foundation Scholarship
	a_3	Civil Engineering	South East University (China) ScienceDirect Digital Library
3	a_1	Postgraduate	BJPTA.gov.cn (China)
	a_3	Civil Engineering	South East University (China) ScienceDirect Digital Library
4	a_1	Postgraduate	Xi'an Jiaotong University (China)
	a_4	Civil Engineering	US Naval Facilities Engineering Command US Federal Real Property Management
5	a_3	Postgraduate	Sohu Sports (China)
	a_4	Civil Engineering	ScienceDirect Digital Library
6	a_2	Postgraduate	Chinese Software Developer Network
	a_4	Civil Engineering	ScienceDirect Digital Library
7	a_5	Postgraduate	Sina Entertainment, Finance (China)
	a_6	Electrical Engineering	BlogCN
8	a_6	Postgraduate	Sina Entertainment, Sports (China)
	a_7	Electrical Engineering	IEEE Xplore National Kidney Foundation (Singapore)
9	a_5	Postgraduate	Sina Entertainment, Finance (China)
	a_7	Electrical Engineering	IEEE Xplore HardwareZone (Singapore)
10	a_8	Postgraduate	Nucleic Acids Research Journal (NAR)
	a_9	Biology 2003	National Center for Biotechnology Information (NCBI) ScienceDirect Digital Library
11	a_9	Postgraduate	National Center for Biotechnology Information (NCBI)
	a_{10}	Biology	Blizzard Entertainment
12	a_{11}	Postgraduate	AsiaOne (Singapore)
	a_{12}	Mechanical Engineering	Zaobao (Singapore) ScienceDirect Digital Library

top 50 links in terms of weight. For each pair, we provide their common demographic attribute values as well as a number of cyber locations that they have in common.

Pairs 1 to 6. The first six pairs involve four Civil Engineering graduate students (a_1 , a_2 , a_3 , and a_4). Their interest in Chinese universities (South East China University and Xi'an Jiaotong University) indicates probable prior affiliation to these institutions. In addition, other China-based URL's such as BJPTA.gov.cn, Chinese Software Developer Network, and Sohu Sports suggest their common country of origin (China). Their access

of ScienceDirect reveals their research occupation.

Pairs 7 to 9. The next three pairs involve three Electrical Engineering graduate students (a_5 , a_6 , and a_7). Their access of Entertainment, Finance, and Sports sections of the China-based Sina portal indicates their common interests and probable country of origin. Their research occupation is evidenced by their access of IEEE Xplore, an established digital library for Electrical Engineering.

Pairs 10 and 11. The next two pairs involve three Biology graduate students (a_8 , a_9 , and a_{10}). Nucleic Acid Research journal and National Centre for Biotechnology Information database indicate their common research interests. a_9 and a_{10} are likely to have a common interest in gaming as well, as evidenced by their access of the Web site of Blizzard Entertainment (an American computer game developer).

Pair 12. The last pair involves two Mechanical Engineering graduate students (a_{11} , and a_{12}). Their common URL's include newspaper portals (AsiaOne and Zaobao) as well as the ScienceDirect digital library.

For each of the above pairs, there is a strong impression of the probable existence of some sort of relationship supported by common occupation, interest, or country of origin.

Demographic Similarity at Various Location Granularity Levels

Previously, we have used only a single level of location granularity at the level of URL hostname (level 1). Here, we investigate the demographic similarity across two other levels (levels 2 and 3) of directory structure of a given URL address. As an example of levels in URL address, for the URL `http://www.ntu.edu.sg/sce/research-areas.asp`, its level 1 is `www.ntu.edu.sg`, level 2 is `sce`, and level 3 is `research-areas.asp`. Each higher level is a more specific location than the previous one (see Section 3.2.3).

Intuitively, we suspected that stronger relationship could be detected at more specific locations. However, this expectation is not supported by the demographic similarity

Table 3.5: Demographic Similarity across Levels (*cyberdata* Aug-Sep04)

<i>Common Features</i>	<i>Random (%)</i>	<i>STEvent (%)</i>		
		<i>Level 1</i>	<i>Level 2</i>	<i>Level 3</i>
at least 1	55.5	86.0	84.8	84.8
at least 2	18.5	22.0	23.2	25.3
at least 3	3.3	5.3	5.9	9.4

Table 3.6: Demographic Similarity across Periods (*cyberdata* at level 1)

<i>Common Features</i>	<i>Random (%)</i>	<i>STEvent (%)</i>				
		<i>Aug-Sep04</i>	<i>Sep-Oct04</i>	<i>Oct04-Jan05</i>	<i>Jan-Feb05</i>	<i>Feb-Mar05</i>
at least 1	55.5	86.0	84.0	78.0	70.0	65.0
at least 2	18.5	22.0	23.5	19.4	18.0	18.0
at least 3	3.3	5.3	6.5	11.5	10.5	6.1

distributions in Table 3.5, which are relatively uniform across the three levels. χ^2 homogeneity test at 5% level [WMMY02] confirms that there is insufficient evidence to conclude otherwise, at least for this particular data (*cyberdata*). This implies the adequacy of URL hostname (level 1) in representing the deeper levels of location granularity.

Demographic Similarity at Various Time Periods

Previously, we have used data from the two-month period (*Aug-Sep04*). These experiments are repeated for four more overlapping two-month periods: *Sep-Oct04*, *Oct04-Jan05*, *Jan-Feb05*, and *Feb-Mar05*. Note that *Oct04-Jan05* does not include the holiday months of November and December 2004.

The demographic similarity distributions shown in Table 3.6 vary slightly among these periods. Nevertheless, χ^2 goodness-of-fit test at 5% level [WMMY02] confirms that *STEvent* distribution for each period is sufficiently dissimilar from that of *Random* (thus marking each period's improvement over *Random* significant). On average, every two consecutive periods share about 60% of the top 100 links in common. This indicates a remarkable consistency in the identification of the strongest links. In the short term, the top links in one period are likely to feature as top links in the next period.

3.4.3 Algorithmic Behavior

Here, we study the effects of variation of parameters on the behavior of the algorithms. At any one time, we vary one parameter and keep the rest fixed. When fixed the parameters would have the default values specified in Table 3.2.

Vary $|\mathcal{D}|$

We vary the data size from 0 to 2.5 million tuples by starting with an empty set and then incrementally adding one day's worth of data. Figure 3.3.a suggests that the growth of events is approximately linear to the data size. This makes sense as the rate at which events take place in real life should be more or less constant. As more events are discovered, the number of candidate links also increases as shown by Figure 3.3.a.

Figure 3.3.b shows the growth in the number of actors with at least one event. Not every actor is active every day. As we incrementally increase the data size by one day's worth of data, the number of participating actors initially increases. After a month, each actor has participated in at least one event. Thereafter the number of actors is stable. We also track the density of the graph formed by the candidate links, defined as the ratio of the number of candidate links to the maximum possible such number or $\frac{|G_{E_{cand}}|}{n(n-1)/2}$ for n actors. Initially, the density is rather flat, as the number of links increases with the number of actors. By the end of August, the number of actors is stable but as more events occur, more pairs of actors can be connected by at least one event. This is evident in the later increase of density. We expect that the density would flatten again once all the possible links have been discovered.

Vary δ_{max}

In Figure 3.3.c, as δ_{max} is varied from 10 minutes to 16 hours, initially we see a minor surge in the number of events, which reaches the peak around 1 and 2 hours. Apparently,

CHAPTER 3. SOCIAL NETWORK DISCOVERY BY MINING SPATIO-TEMPORAL EVENTS

3.3.a: $|\mathcal{E}|$, $|G_{E_{cand}}|$ vs. $|\mathcal{D}|$ 3.3.b: Density vs. $|\mathcal{D}|$ 3.3.c: $|\mathcal{E}|$, $|G_{E_{cand}}|$ vs. δ_{max} 3.3.d: $|G_{E_{cand}}|$, $|G_E|$ vs. min_link_weight Figure 3.3: Algorithmic Behavior (*cyberdata* Aug-Sep04)

longer δ_{max} makes it easier for several tuples to belong to an event together. However it then declines and eventually levels off. An exceedingly long δ_{max} would “combine” several shorter events at the same location into a long-running event. On the other hand, the number of candidate links keeps increasing, though at increasingly slower pace, as it gets less and less restrictive for two tuples to join together in an event.

Vary min_link_weight

Previously, with min_link_weight set at 0, the candidate links are equivalent to the discovered links. Figure 3.3.d shows that as we increase this value to 100, the number of candidate links $|G_{E_{cand}}|$ remains constant, but the number of discovered links $|G_E|$ drops

Figure 3.4: Time Taken vs. Data Size (*cyberdata* Aug-Sep04)

precipitously, from 77677 at $\text{min_link_weight} = 0$, to 573 at $\text{min_link_weight} = 20$, to 28 at $\text{min_link_weight} = 100$.

3.4.4 Time Complexity

We vary the data size and chart the time taken for each phase of the algorithm. To get these timings, the algorithms were implemented in C++ and run on an Intel Pentium 4 1.7GHz machine with 512MB RAM. Figure 3.4 shows that Phase 1 (P1) time grows linearly with the data size, confirming the theoretical complexity of $O(|\mathcal{D}|)$. Phase 2 (P2)'s is slightly above linear, feasibly approaching the theoretical complexity of $O(|\mathcal{E}| \times \log |\mathcal{E}| + |G_{E_{cand}}|)$. Overall, P2 is running at a multiple of up to 7 times of P1. The longest total time taken for the two phases approaches 10 minutes; this is when min_link_weight is 0. It is expected that for real usage, higher values of min_link_weight will be set, leading to shorter completion times.

3.5 Experiments on Physical Location Data

The objectives of experiments on this data are to cross-validate the experiments on *cyberdata* to see if the results are consistent, as well as to find new insights peculiar to

Table 3.7: Parameter Values (*physicaldata* Aug-Sep04)

<i>Parameter</i>	Default Value	Range
$ \mathcal{D} $	14 thousand	0 – 14 thousand
δ_{max}	2 hours	10 minutes – 16 hours
<i>min_link_weight</i>	0	0 – 25
<i>min_event_weight</i>	0	0

Physical Location Data (or *physicaldata*).

3.5.1 Data

The data were collected as a log of base stations (situated at known physical locations) that each user connects to in order to gain access to the wireless network. Each tuple $\langle a, t, s \rangle$ of this data consists of a user a , a time stamp t , as well as a location s . A location corresponds to the closest base station to which this user’s wireless device is connected to. Each base station serves an area within 25–100m radius, with more base stations placed in crowded areas. As our users mostly frequented relatively crowded shared spaces, it was likely that the closest base station would be nearby. A location consists of three levels, from the most general to the most specific: building (level 1), floor (level 2), and room (level 3). Unless otherwise specified, we use all three levels of location granularity.

Each tuple originally referred to a wireless device, identified by a device id. With the availability of other types of data from DHCP and firewall servers, a device id could be mapped back to a real user, with some data loss. DHCP data allowed mapping a device id to an IP address allocated to that device at a particular point of time. Firewall data allowed mapping an IP address to a user name, which identified a real user. The mapping was time-sensitive, i.e., a device id could be successfully mapped to a user name only if there were matching DHCP and Firewall records within a small time window (5 minutes). Unsuccessful mapping resulted in data loss. Increasing the mapping time window would reduce the data loss but would also reduce the mapping confidence.

Table 3.8: Demographic Similarity across Levels (*physicaldata* Aug-Sep04)

<i>Common Features</i>	<i>Random (%)</i>	<i>STEvent (%)</i>		
		<i>Level 1</i>	<i>Level 2</i>	<i>Level 3</i>
at least 1	53.5	91.0	97.0	100.0
at least 2	19.3	66.0	74.0	87.0
at least 3	13.2	64.1	66.7	71.1

Table 3.9: Demographic Similarity across Periods (*physicaldata* at level 3)

<i>Common Features</i>	<i>Random (%)</i>	<i>STEvent (%)</i>				
		<i>Aug-Sep04</i>	<i>Sep-Oct04</i>	<i>Oct04-Jan05</i>	<i>Jan-Feb05</i>	<i>Feb-Mar05</i>
at least 1	53.5	100.0	100.0	100.0	98.0	100.0
at least 2	19.3	87.0	90.0	84.0	78.0	83.8
at least 3	13.2	71.1	74.4	69.1	61.8	59.3

We retained only data concerning users who appeared at least once in each of the six months (August to October 2004 and January to March 2005). There were 75 such users, moving over 63 unique level-3 locations. The data size after pre-processing was 34 thousand tuples or 1.41MB. Because of the data loss due to mapping and the much smaller size of *physicaldata*, we decided to use *cyberdata* as the primary data for experiments and *physicaldata* as a secondary data for verification.

3.5.2 Demographic Similarity

Using *physicaldata* in place of *cyberdata*, we repeat the experiments in Section 3.4.2 (demographic similarity) with the parameter settings given in Table 3.7. We derive a *STEvent* result set (top 100 pairs) for each level and each period. Table 3.8 outlines the varying distributions across levels for *Aug-Sep04* when the multi-level location granularity (see Section 3.2.3) is enabled for three different levels: building (level 1), floor (level2), and room (level 3). Table 3.9 in turn outlines those across different periods when the multi-level location is enabled at level 3.

We compare *STEvent* to the *Random*. This *Random* distribution is slightly different from that for *cyberdata*, especially for pairs sharing 3 common attribute values. This

difference is not surprising, as these pairs are drawn from a smaller set of actors (75 vs. 533 actors), with a different underlying distribution in terms of proportions of attribute values. On the other hand, for *physicaldata*, the *STEvent* similarity values are much higher as compared to *Random*. Applying a similar χ^2 goodness-of-fit test at 5% level [WMMY02] reveals that all *STEvent* distributions, for each level or each period, are sufficiently dissimilar from *Random* distribution, confirming the significant improvement that visual inspection alone has suggested.

While the χ^2 homogeneity test at 5% level [WMMY02] for values in Table 3.9 suggests that the *STEvent* distribution is more or less uniform for all periods, that for values in Table 3.8 suggests that the distributions of the different levels are not uniform; in fact they seem to be improving at deeper levels (more specific locations). The latter is a result that *cyberdata* is not able to produce.

3.5.3 Algorithmic Behavior

Using *physicaldata* in place of *cyberdata*, we repeat the experiments in Section 3.4.3 (algorithmic behavior) with the parameter settings given in Table 3.7. The results are illustrated in Figure 3.5. A quick visual inspection comparing Figure 3.3 and Figure 3.5 reveals a remarkably similar set of trends in all four sub-figures, despite the significant difference in data sizes between the two data sets. Other than noting this consistency between the two data sets, we would not go into any further detail as we believe the previous discussion on *cyberdata* still applies here.

A comparable figure to Figure 3.4 on time complexity is not shown here as given the relatively much smaller data size, the time taken for experiments on *physicaldata* is almost negligible and does not show any clear or notable trend.

CHAPTER 3. SOCIAL NETWORK DISCOVERY BY MINING SPATIO-TEMPORAL EVENTS

3.5.a: $|\mathcal{E}|$, $|G_{E_{cand}}|$ vs. $|\mathcal{D}|$ 3.5.b: Density vs. $|\mathcal{D}|$ 3.5.c: $|\mathcal{E}|$, $|G_{E_{cand}}|$ vs. δ_{max} 3.5.d: $|G_{E_{cand}}|$, $|G_E|$ vs. min_link_weight Figure 3.5: Algorithmic Behavior (*physicaldata* Aug-Sep04)

3.5.4 Comparison: Cyber Location vs. Physical Location

To summarize the relationship between the two sets of data, we highlight their differences and similarities as follows.

The two data sets have data sizes (tuples, actors, locations) of different orders of magnitude, with *cyberdata* being the much larger one. Seeking overlap between the top 100 pairs of the two data sets in each period does not yield many common pairs (10%, 11%, 11%, 9%, 7%). Moreover the demographic similarity distributions of *physicaldata* is generally much higher than those of *cyberdata*. These hint at the different semantics of

relationship that can be mined from those data sets. *physicaldata* co-occurrences seem to correlate more with the notion of similarity than *cyberdata* co-occurrences.

On the other hand, it has also been shown affirmatively how the algorithmic behaviors of the two data sets are consistent in their trends, how the demographic similarity for each respective data set is similar across the five periods of interest, and how their respective demographic similarity distributions (across location levels and periods) consistently produce statistically significant improvements over *Random* distributions.

3.6 Summary

In this chapter, we address the problem of social network discovery from spatio-temporal data. Our proposed model *STEvent* works by mining events from the spatio-temporal data, which are distinct co-occurrences among two or more actors, and inferring links between pairs of actors who participate in many common events. To implement the *STEvent* model, we describe a two-phase algorithm, which first discovers events from the data, and then uses the events to derive links. The *STEvent* model is tested on two sets of real-life data: *Cyber Location Data* (a log of Web pages accessed by users at different times) and *Physical Location Data* (a log of physical locations from where users connect to the campus wireless network over time).

The following are the main findings derived from this research:

- The top links discovered by *STEvent* demonstrate a relatively high degree of similarity, which is another well-recognized predictor of social association [Fel81, Car91]. The demographic similarity among pairs of actors is much higher for the top *STEvent* links than for *Random* links. This result stands across various experiments, involving different data, time periods, and levels of location granularity.

CHAPTER 3. SOCIAL NETWORK DISCOVERY BY MINING SPATIO-TEMPORAL EVENTS

- The results for *Physical Location Data* are significantly better than for *Cyber Location Data*. This hints that different types of data may yield links of different relationship semantics or qualities. Apparently, co-occurrences in physical locations are better predictors of links among users with similar demographic attributes.
- The proposed algorithms are efficient. Its time complexity is approximately linear to the data size (see Figure 3.4). Most of our experiments with *Cyber Location Data*, involving 9.5 million tuples or 550MB worth of data, complete in a matter of minutes. Experiments with the much smaller *Physical Location Data* (34 thousand tuples or 1.41MB) complete in a matter of seconds.

Chapter 4

Bias and Controversy in Collaborative Rating Networks

4.1 Overview

In any rating system, some of the most important questions are whether the rating has been conducted fairly, and whether any reviewer is biased. Various studies on peer review [McC06, WMM02] and student evaluation [Pre97] have shown that reviewers often assign varying scores when rating the same object. One reason behind this variability could be “bias” on the part of some reviewers, which causes them to deviate from other reviewers. Another reason could be “controversy” on the part of objects being rated, thus attracting varied opinions from reviewers. Knowing bias and controversy would contribute towards more just and equitable treatment of reviewers and objects in a rating system. For instance, the biased reviewers and the controversial objects may be highlighted to the rating system administrator(s) for his/her attention.

To investigate the notions of “bias” of reviewers and “controversy” of objects within a rating system, we model a rating system as a bipartite network with weighted edges. Consider the example rating network shown in Figure 4.1. Reviewers and objects are the two distinct types of nodes. If a reviewer r_i rates an object o_j , an edge from r_i to o_j is created and is weighted with the assigned rating score $e_{ij} \in [0, 1]$. For example, in

CHAPTER 4. BIAS AND CONTROVERSY IN COLLABORATIVE RATING NETWORKS

Figure 4.1: Example Rating Network

Table 4.1: Table of Notations

Notation	Description
r_i	a reviewer
o_j	an object
e_{ij}	rating score by r_i on o_j
d_{ij}	deviation by r_i on o_j
b_i	r_i 's bias
c_j	o_j 's controversy
x_i	r_i 's reviewer evidence
y_j	o_j 's object evidence

Figure 4.1, r_1 assigns $e_{11} = 0.6$ to o_1 . Given such a rating network, we seek to measure the bias $b_i \in [0, 1]$ of each reviewer r_i and the controversy $c_j \in [0, 1]$ of each object o_j . For ease of reference, the above notations are listed in Table 4.1. The remaining notations in the table will be explained when introduced in subsequent sections.

A straightforward approach to measuring bias and controversy is to use standard statistical measures. Reviewers who deviate much from their co-reviewers are considered biased, and objects which attract much deviation are considered controversial. Suppose that deviation $d_{ij} \in [0, 1]$ measures the degree of deviation by r_i from his/her co-reviewers on o_j , and that d_{ij} can be derived from the scores by o_j 's reviewers, we can express the above intuition with Eq. 4.1 to measure b_i and Eq. 4.2 to measure c_j . b_i is the average of deviations d_{ij} 's by r_i on objects o_j 's rated by r_i ; c_j is the average of deviations d_{ij} 's on o_j by its reviewers r_i 's. We call this pair of equations the *Naive* approach.

$$b_i = \mathcal{Avg}_j d_{ij} \quad (\text{Eq. 4.1})$$

$$c_j = \mathcal{Avg}_i d_{ij} \quad (\text{Eq. 4.2})$$

Example 4.1 *Let us compare the bias of r_1 and r_5 in Figure 4.1. Suppose that d_{ij} is derived as the absolute distance between r_i 's rating score and the mean rating score by the reviewers r_k 's of object o_j as shown in Eq. 4.3. We have $d_{11} = 0.3$ and $d_{53} = 0.375$. Naive concludes that r_1 is less biased than r_5 , since $(b_1 = d_{11} = 0.3) < (b_5 = d_{53} = 0.375)$.*

$$d_{ij} = \left| e_{ij} - \mathcal{Avg}_k e_{kj} \right| \quad (\text{Eq. 4.3})$$

However, the more intuitively correct outcome is that r_1 is more biased than r_5 . Visual inspection reveals that the co-reviewers of o_1 and o_2 respectively are much more in consensus than co-reviewers of o_3 (with 4 divergent scores). This suggests that o_3 is likely a controversial object, and r_5 's deviation is due to the controversy of o_3 . In contrast, since r_1 is the only deviating reviewer of o_1 , it is likely that r_1 's deviation is due to bias. Thus, Naive leads to the wrong conclusion that r_1 is less biased than r_5 .

Naive's main weaknesses are its equal treatment of score deviations by all reviewers and objects, and its isolated consideration of each reviewer's bias or object's controversy. In determining an object's controversy, *Naive* treats score deviations by all reviewers equally, although some reviewers may be more biased than others. *Naive* also considers each object in isolation, looking only at how each object's reviewers may deviate on the object, but ignoring how the same reviewers behave on other objects.

We therefore develop a framework that comprehensively examines the bias of every reviewer and the controversy of every object within a rating network. Several observations underlie our approach in determining bias and controversy. First, evaluation is

“subjective”, as reviewers and objects have varying bias and controversy respectively. Second, bias and controversy are mutually dependent, as determining the bias of a reviewer requires knowing the controversy of his/her rated objects (vice versa). In addition to bias and controversy, our approach also quantifies *evidence*, which represents the degree of confidence with which the bias of a reviewer or the controversy of an object has been derived.

The rest of this chapter is organized as follows. Section 4.2 describes our proposed bias and controversy framework. Section 4.3 describes two ways to derive deviation measures, and identifies which is more suitable. Section 4.4 discusses the convergence of an iterative method implementation of the proposed models. Sections 4.5 and 4.6 verify the effectiveness of our approach through experiments on real-life and synthetic data respectively. Section 4.7 concludes this chapter.

4.2 Bias and Controversy Framework

We adopt a framework for analyzing rating data as given in Figure 4.2. Given rating data, the *Bias & Controversy Model* determines the bias of each reviewer and the controversy of each object. The information on bias/controversy helps the *Evidence Model* to provide information on the degree of support for the computed bias and controversy. This information on bias, controversy and evidence would complement the rating data in the *Analytical and Decision Process* by analysts. For instance, this process may involve moderating or excluding scores by biased reviewers and giving special attention or treatment to controversial objects. The bias/controversy information may also be retained for future use, e.g., for better assignment of reviewers to objects in the future. Below, we discuss our proposed models for bias/controversy and evidence respectively.

Figure 4.2: Framework for Analyzing Bias and Controversy in Rating Data

4.2.1 Bias and Controversy

Considering the *subjectivity* of evaluation, we define *bias* for a reviewer and *controversy* for an object based on the following principle.

- *A reviewer is more biased if s/he deviates more on less controversial objects.* A reviewer's deviations on controversial objects might well be due to the controversy of these objects. In contrast, if the objects are not controversial, any deviation would suggest bias on the reviewer's part.
- *An object is more controversial if there is greater deviation by less biased reviewers.* Similarly, we should attribute to an object's controversy mainly the deviations by its less biased reviewers.

Thus, we propose the *Inverse Reinforcement* (or simply, *IR*) bias and controversy model that attributes to a reviewer only deviation due to his/her bias, and not to the controversy of rated objects, by reducing the amount of deviation attributed to bias by the amount of controversy that could have contributed to that deviation.

Inverse Reinforcement (IR) Model

The *IR* model consists of a pair of equations: Eq. 4.4 for measuring bias and Eq. 4.5 for measuring controversy. In Eq. 4.4, we compute the bias b_i of a reviewer r_i by aggregating

CHAPTER 4. BIAS AND CONTROVERSY IN COLLABORATIVE RATING NETWORKS

(using the $\mathcal{Avg}$ function) the score deviation d_{ij} for each rated object o_j , and reducing d_{ij} by its controversy c_j . Here, $\bar{c}_j$ is the complement of controversy c_j , i.e., $\bar{c}_j$ is higher when c_j is lower. The score deviation d_{ij} can be derived from the scores by o_j 's reviewers in several ways, as will be shown in Section 4.3. Eq. 4.4 captures the idea that b_i is higher when d_{ij} is higher, since a highly biased reviewer would likely deviate on objects. On the other hand, b_i is lower when c_j is higher, since a highly controversial object is expected to have high deviations by its reviewers.

$$b_i = \mathcal{Avg}_j d_{ij} \cdot \bar{c}_j \quad (\text{Eq. 4.4})$$

$$c_j = \mathcal{Avg}_i d_{ij} \cdot \bar{b}_i \quad (\text{Eq. 4.5})$$

In determining controversy with Eq. 4.5, we consider an object o_j to have higher controversy c_j if on aggregate it tends to have high deviation d_{ij} by reviewer r_i with low bias b_i . Here, $\bar{b}_i$ denotes the complement of b_i . High d_{ij} by a reviewer with high b_i would contribute less to c_j , since a biased reviewer is expected to deviate on objects. Thus, the relationship between b_i and c_j is inversely reinforcing, i.e., b_i is higher when c_j is lower (and vice versa). For this reason, we call our proposed model *Inverse Reinforcement* or *IR* model. In Eq. 4.4 and Eq. 4.5, we require $d_{ij} \in [0, 1]$, $b_i, \bar{b}_i \in [0, 1]$, and $c_j, \bar{c}_j \in [0, 1]$.

To get an overall picture of the behavior of a reviewer, we aggregate the deviations weighted by controversy over all rated objects. In the above equations, we use *average* as the aggregation function, which is suitable as it takes into account repeated deviation by a reviewer. *Summation* is unsuitable, as it unfairly attributes higher bias to a reviewer who simply has rated many objects. *Maximum* may unfairly over-penalize a reviewer who deviates only on a small minority (even one) of rated objects, while *minimum* may under-penalize a reviewer who deviates on the majority (but not all) of rated objects.

A key point in this proposed model is the mutual dependency between b_i and c_j in Eq. 4.4 and Eq. 4.5. This mutual dependency means that b_i and c_j have to be determined simultaneously, as neither is known beforehand. Moreover, this dependency also extends to all reviewers and objects, which are inter-connected within the rating network. The bias of a reviewer r_i depends on the controversy of r_i 's objects, which in turn depends on the bias of r_i 's co-reviewers, and so on. Therefore, the bias of all reviewers and the controversy of all objects have to be determined simultaneously as well. We will describe how to derive all b_i 's and c_j 's simultaneously later in Section 4.4.

Naive Model as a Special Case

Earlier, we have presented the *Naive* model, comprising Eq. 4.1 and Eq. 4.2, which defines b_i 's and c_j 's directly and entirely from d_{ij} 's. The main difference between *Naive* and *IR* is that *Naive* exhibits no similar dependency between b_i and c_j . Thus, *Naive* is a suitable baseline alternative to *IR*. Comparing the *Naive* and *IR* equations, we further observe that *IR* is the more general model that would degenerate into *Naive* under particular conditions. When all objects have equal controversy ($\bar{c}_j$ is constant), *IR*'s equation for bias Eq. 4.4 practically degenerates into *Naive*'s Eq. 4.1. Similarly, when all reviewers have equal bias ($\bar{b}_i$ is constant), *IR*'s Eq. 4.5 degenerates into *Naive*'s Eq. 4.2.

IR work by measuring a reviewer's bias by his/her deviations on non-controversial objects. Such deviations provide the supporting evidence which can be modeled separately by an evidence model as shown in Section 4.2.2. When a reviewer rates mostly controversial objects, there may be little evidence to say that s/he is biased.

4.2.2 Evidence Model

Considering the possible existence of biased reviewers and controversial objects, we define *evidence* for a reviewer and for an object based on the following principle.

CHAPTER 4. BIAS AND CONTROVERSY IN COLLABORATIVE RATING NETWORKS

- *A reviewer has high evidence if s/he has rated mainly objects with low controversy and high evidence.*
- *An object has high evidence if its reviewers mainly have low bias and high evidence.*

We propose a model to measure evidence, called the *Evidence* model, which consists of Eq. 4.6 to determine x_i (the evidence of r_i) and Eq. 4.7 to determine y_j (the evidence of o_j). Like b_i and c_j , x_i and y_j are also mutually dependent. However, the relationship between x_i and y_j are mutually reinforcing. Note that evidence are measured after bias and controversy have been determined. Since $\bar{b}_i, \bar{c}_j \in [0, 1]$, we would have $x_i, y_j \in [0, 1]$.

$$x_i = \mathcal{Avg}_j \bar{c}_j \cdot y_j \quad (\text{Eq. 4.6})$$

$$y_j = \mathcal{Avg}_i \bar{b}_i \cdot x_i \quad (\text{Eq. 4.7})$$

Measuring the evidence of a reviewer or an object helps to reveal whether the reviewer's bias or the object's controversy is well-supported by evidence. In the analysis of rating outcome, it informs an analyst inspecting the overall rankings by bias and controversy of the 'confidence level' with which each reviewer's bias or each object's controversy has been derived. Evidence can also be used to filter the rating outcome, for instance by focusing only on the subset of reviewers or objects meeting a certain level of evidence.

In practice, there are several ways to boost the evidence of reviewers. One way is to have a reviewer evaluate more objects so as to increase the reviewer's probability of evaluating objects with low controversy. Another way is to ensure each reviewer is allocated at least a few objects with low controversy would directly increase evidence. The latter is possible only if we have some prior knowledge on the controversy of objects. In general, a higher level of evidence overall should improve *IR* model's ability to measure bias and controversy. This will be verified with the help of experiments in Section 4.6.

4.3 Deviation Measures

Deviation measure is an important component in the bias and controversy model. We consider two deviation measures: *deviation from mean* and *deviation from co-reviewers*, and justify why the latter is more suitable for our purpose.

4.3.1 Deviation from Mean

The first measure is given by Eq. 4.3 and has been defined in Section 4.1. We call it *deviation from mean*. It is straightforward, but does not fully capture the degree of disagreement among reviewers. For one reason, it assumes that the mean is the “true” score. Any reviewer who gives a score close to the mean has minimal deviation. For example, suppose r_1 , r_2 , and r_3 assign to o_j the following scores: $e_{1j} = 0.0$, $e_{2j} = 0.5$, and $e_{3j} = 1.0$, then we have $d_{1j} = 0.5$, $d_{2j} = 0.0$, and $d_{3j} = 0.5$. It unfairly claims that r_2 has not deviated at all, when clearly all the reviewers cannot agree on o_j ’s score.

For another reason, deviation as measured from the mean tends to be a small value, even approaching zero. This is disadvantageous since the outcome of computation is determined by the relative ratios among various d_{ij} values, and very small d_{ij} values may make the system too sensitive to small changes. For example, in Figure 4.3.a, we plot a histogram of d_{ij} values obtained by applying this measure on a real-life data (“1 Million MovieLens Dataset”). This data will also be used for experiments in Section 4.5. Figure 4.3.a shows that the resulting d_{ij} values are quite small; close to 40% are below 0.1; close to 70% are below 0.2.

4.3.2 Deviation from Co-Reviewers

We propose an alternative measure: *deviation from co-reviewers*. Eq. 4.8 shows that this measure takes deviation d_{ij} as the average distance between r_i ’s score and the score by each co-reviewer r_k . m_j is the number of reviewers of o_j (including r_i).

CHAPTER 4. BIAS AND CONTROVERSY IN COLLABORATIVE RATING NETWORKS

Figure 4.3: Histogram of Deviation Values

$$d_{ij} = \begin{cases} \frac{1}{m_j-1} \sum_{k \neq i} |e_{kj} - e_{ij}| & \text{if } m_j > 1 \\ 0 & \text{otherwise} \end{cases} \quad (\text{Eq. 4.8})$$

This measure overcomes the disadvantages of *deviation from mean*. It better captures the notion of disagreement from the majority. A reviewer with e_{ij} closer to the majority (as opposed to the mean) would have a lower d_{ij} . If no majority of reviewers can agree on the score of o_j , all reviewers will generally have high deviation. For the same example of r_1 , r_2 , and r_3 assigning $e_{1j} = 0.0$, $e_{2j} = 0.5$, and $e_{3j} = 1.0$ to o_j , we have $d_{1j} = 0.75$, $d_{2j} = 0.50$, and $d_{3j} = 0.75$. We find that the difference between d_{1j} and d_{2j} as determined by *deviation from co-reviewers* ($d_{1j} = 0.75$ vs. $d_{2j} = 0.50$) is more reasonable than that as determined by *deviation from mean* ($d_{1j} = 0.50$ vs. $d_{2j} = 0$).

It is also less likely to have deviation values close to zero, creating a more stable system that is not too sensitive to small changes. Figure 4.3.b plots a histogram of d_{ij} values using *deviation from co-reviewers* on the same data (“1 Million MovieLens Dataset”). Only a very small proportion ($< 1\%$) are in the bin closest to zero ($d_{ij} \leq 0.1$).

For these reasons, *deviation from co-reviewers* is a more suitable deviation measure

than *deviation from mean*. We use *deviation from co-reviewers* in the subsequent implementation of the *Naive* and *IR* models for experiments.

4.4 Convergence of the Proposed Models

We show how the proposed *IR* (for bias and controversy) and *Evidence* models can be solved by iterative methods [AR87, GVL96] that provide a converged solution. We reformulate the problem as finding an eigenvector of a square matrix, and discuss the convergence of such formulation for bias and controversy, as well as for evidence.

Bias and Controversy

Assuming that the linear relationships $b_i + \bar{b}_i = 1$ and $c_j + \bar{c}_j = 1$ hold, Eq. 4.4 and Eq. 4.5 can be re-written as Eq. 4.9 and Eq. 4.10. n is the total number of objects, m is the total number of reviewers, n_i is the number of objects of r_i , and m_j is the number of reviewers of o_j . Without affecting the correctness of these equations, we may simply assume $d_{ij} = 0$ if r_i has not rated o_j .

$$b_i = \frac{\sum_{j=1}^n d_{ij} \cdot (1 - c_j)}{n_i} \quad (\text{Eq. 4.9})$$

$$c_j = \frac{\sum_{i=1}^m d_{ij} \cdot (1 - b_i)}{m_j} \quad (\text{Eq. 4.10})$$

The system of equations comprising Eq. 4.9 for every r_i and Eq. 4.10 for every o_j can be more compactly represented in matrix form as Eq. 4.11 and Eq. 4.12. B is the $m \times 1$ vector of b_i 's. C is the $n \times 1$ vector of c_j 's. $\mathbf{1}$ is a column vector of appropriate length whose elements are all 1's. I is an $m \times n$ matrix, whose each element $I_{ij} = d_{ij} \div n_i$. J is an $m \times n$ matrix, whose each element $J_{ij} = d_{ij} \div m_j$.

$$B = I (\mathbf{1} - C) \quad (\text{Eq. 4.11})$$

$$C = J^T(\mathbf{1} - B) \quad (\text{Eq. 4.12})$$

Substituting Eq. 4.12 into Eq. 4.11, we get the recursive equation Eq. 4.13 in terms of B . Hereonafter, we focus on deriving the solution of B . The solution of C can similarly be derived from a recursive equation obtained by substituting Eq. 4.11 into Eq. 4.12.

$$B = (I\mathbf{1} - IJ^T\mathbf{1}) + IJ^TB \quad (\text{Eq. 4.13})$$

We transform Eq. 4.13 into the eigenvector equation Eq. 4.14 in terms of B as follows. For any $w \times 1$ column vector W , we use the notation W^m to denote $w \times m$ matrix formed by replicating W across m columns. If B is normalized such that $\sum_{i=1}^m |b_i| = \beta$, then $\beta^{-1}W^mB = W$ holds. We use this notation to factorize B out from the right-hand side of Eq. 4.13. This results in Eq. 4.14, which has the recursive form of $B = ZB$, where B is the variable and $Z = [\beta^{-1}(I\mathbf{1} - IJ^T\mathbf{1})^m + IJ^T]$ is a constant matrix. Specifically, we are interested in B as the principal eigenvector of Z . A similar formulation has previously been attempted in the work on PageRank [PBMW98] and HITS [Kle99].

$$B = [\beta^{-1}(I\mathbf{1} - IJ^T\mathbf{1})^m + IJ^T] B \quad (\text{Eq. 4.14})$$

Iterative methods [AR87, GVL96] can be used to derive B . The iterative form is given by $B_{k+1} = ZB_k$. Starting from the initial B_0 , the output of the k -th iteration is used as input for the $(k+1)$ -th iteration. Subject to the assumption that the square asymmetric matrix Z is *diagonalizable* (it has linearly-independent eigenvectors) and has a uniquely largest eigenvalue [GVL96], then as k increases, B_k will converge to the principal eigenvector of Z almost independently of the initial B_0 . These conditions for convergence can be tested, e.g., by inspecting the eigenvalues or eigenvectors of the square matrix [AR87]. The converged B contains the output bias b_i of each reviewer r_i .

The above iterations require periodic normalization of B , which returns B to an invariant state such that convergence can be observed as (or almost) no change between two consecutive iterations. In this case, normalization maintains that the maximum element of B is always 1, i.e., after each iteration all elements of B are divided by the maximum element. We find that other normalization methods used in previous work [Kle99, PBMW98], such as L_1 or L_2 , are not suitable for our application. L_p -normalization divides all b_i 's by a constant to meet the following condition: $\sum_{i=1}^m |b_i|^p = 1$. For small p and large m , L_p -normalization leads to very small b_i 's that may even approach zero, in which case, *IR*'s Eq. 4.4 would degenerate into *Naive*'s Eq. 4.1.

For a given number of reviewers and objects, the computational complexity of the above iterative computation is $O(K)$, where K is the number of iterations to convergence. We observe empirically that K is usually small. For the experiments in Sections 4.5 and 4.6 with thousands of reviewers and objects, the number of iterations ranges from 12 to 176, with a median of 17. The running time ranges from less than 1 second to 60 seconds, with a median of less than 1 second. Moreover, as each iteration moves B_k closer to the converged B_K , this implementation allows the option of prematurely terminating the iterations for an approximate result if time proves to be a constraint.

Since a rating network could be sparse, an adjacency matrix representation of the network may incur extra memory requirement, especially for a large number of reviewers and objects. A more memory-efficient computation can be achieved by representing the network as an adjacency list (which saves memory), and to break down a matrix equation (which is a composite of linear equations) into the individual linear equations (which can be computed using the adjacency list).

Evidence

The solution of evidence largely follows that of bias and controversy. The system of equations that consists of Eq. 4.6 for every r_i and Eq. 4.7 for every o_j can be expressed

in terms of matrices, as given in Eq. 4.15 and Eq. 4.16. X is the $m \times 1$ vector of x_i 's. Y is the $n \times 1$ vector of y_j 's. U is an $m \times n$ matrix, whose each element $U_{ij} = (1 - c_j) \div n_i$. V is an $m \times n$ matrix, whose each element $V_{ij} = (1 - b_i) \div m_j$. Substituting Eq. 4.15 and Eq. 4.16 into each other, we get the recursive forms of $X = (UV^T)X$ and $Y = (V^TU)Y$. We solve them for X and Y respectively using iterative methods as shown above.

$$X = U Y \quad (\text{Eq. 4.15})$$

$$Y = V^T X \quad (\text{Eq. 4.16})$$

4.5 Experiments on Real-Life Data

The objective of experiments on real-life data is to show that *IR*'s results are more intuitively correct than *Naive*'s. To highlight the difference between *IR* and *Naive*, we compare their rankings for bias and controversy. Subsequently, we showcase how the results by *IR* are more reasonable, with the aid of specific case examples.

4.5.1 Data

We chose the “1 Million MovieLens Dataset” from GroupLens (www.grouplens.org) for our experiments. The data contain ratings by users of the movie recommendation site MovieLens (www.movielens.org). The data are suitable due to its large data size and high level of participation. There were 6040 reviewers, 3706 objects (movies) and 1000209 scores as shown in Table 4.2. Each reviewer rated at least 20 objects.

The data were further pre-processed as follows. Each reviewer originally assigned scores from 1 (worst) to 5 (best). These scores were normalized to the range from 0.2 to 1 by a simple division by 5. We also ensured that each reviewer had at least 3 objects and each object had at least 3 reviewers, by iteratively removing reviewers with less than

Table 4.2: Data Size

	<i>Original</i>	<i>Filtered</i>
<i>reviewers</i>	6040	6040
<i>objects</i>	3706	3503
<i>scores</i>	1000209	999917

Table 4.3: Network Connectivity

	<i>minimum</i>	<i>median</i>	<i>max</i>
<i>objects per reviewer</i>	19	96	2290
<i>reviewers per object</i>	3	140	3428

3 objects and objects with less than 3 reviewers until there was no more such reviewer or object. This removed the occasional reviewers/objects and gave greater support when inferring the “behavior” of reviewers/objects. As shown in Table 4.2, the data size after filtering were still large, with 6040 reviewers and 3503 objects. The reviewers and objects actively participated in the rating as shown by the high number of objects per reviewer (with median of 96) and reviewers per object (with median of 140) given in Table 4.3.

4.5.2 Comparison of Ranked Lists

This part of the experiment compares the ranked list produced by the *IR* model with that by *Naive* model. To do so, we derive a rank for each reviewer (object). Reviewers and objects are ranked in descending order of bias and controversy respectively. Same values share the same rank. For example, if the three highest values are the same, they share rank 1, and the next highest is of rank 4. Due to the large number of reviewers and objects, the ranks may range from 1 to the thousands. For simplicity and ease of comparison, we convert these ranks into “percentranks”, which range from 1 to 100. Percentrank of a reviewer r_i is derived from its rank as follows: $\text{percentrank}(r_i) = \lceil \text{rank}(r_i) \times 100 \div m \rceil$, where m is the total number of reviewers. A reviewer with percentrank 1 is in the top 1% most biased in the population. A reviewer with percentrank 2 is in the top 2% (but below the top 1%), and so on. Percentrank of an object in terms of controversy may be derived in a similar manner.

CHAPTER 4. BIAS AND CONTROVERSY IN COLLABORATIVE RATING NETWORKS

Figure 4.4: Percentrank Scatterplots: *Naive* vs. *IR***Percentrank Comparison.**

We compare the percentranks assigned to each reviewer or object by *IR* and *Naive*. Figure 4.4.a and Figure 4.4.b are scatterplots of bias and controversy percentranks respectively. A point in Figure 4.4.a represents a reviewer, while in Figure 4.4.b it represents an object. Values on the *x-axis* are percentranks assigned by *IR*, while those on the *y-axis* are percentranks assigned by *Naive*. If the percentranks assigned by *IR* and *Naive* are generally similar, these points would line up along the diagonal. This is not the case in Figure 4.4.a and Figure 4.4.b. Instead, there are large variances around the diagonal. The circled point on Figure 4.4.b shows that an object may be given as extremely different percentranks as 70 by *IR* and 10 by *Naive*. For many reviewers/objects, *IR* and *Naive* have assigned them different percentranks. The top percentranks see greater similarity than the middle percentranks. One possible reason could be that *IR* and *Naive* would tend to agree on very blatant bias or controversy.

Similarity at Top $k\%$

In practice, the top ranks are usually of utmost interest. We now go on to show that there are still significant differences even at the top ranks. For this comparison, we will use actual ranks, rather than percentranks. This is because percentrank is a less precise measure than ranks and ignores rank differences in the same percentrank group. We compare the top $k\%$ ranked list by *IR* with the *Naive*'s ranking of the same reviewers/objects. In other words, we have two ranked lists, X being the top $k\%$ of *IR*, and Y being the ranked list of *Naive* after deleting all objects not among the top $k\%$ of *IR*.

We use the following two well-known measures to quantify the difference of the rankings of X and Y . Both measures output a similarity value, which ranges from 0 (reversed ranking) to 1 (identical ranking).

- *Kendall* similarity [DKNS01, FKS03] counts the number of pairs for which X and Y agree on their relative ranks as shown in Eq. 4.17, where n is the size of X and Y . For an item t , its rank in X is $rank_X(t)$ and in Y is $rank_Y(t)$. *Kendall* penalizes each pair of items (t_1, t_2) where $rank_X(t_1) > rank_X(t_2)$ but $rank_Y(t_1) < rank_Y(t_2)$.

$$Kendall(X, Y) = \frac{|\{(t_1, t_2) | X \text{ and } Y \text{ agree on order of } (t_1, t_2)\}|}{\frac{1}{2}n(n-1)} \quad (\text{Eq. 4.17})$$

- *Spearman* similarity [DKNS01, FKS03] counts, for each item t , the absolute difference between $rank_X(t)$ and $rank_Y(t)$. The differences are aggregated across all items as shown in Eq. 4.18. N is the maximum possible aggregate difference.

$$Spearman(X, Y) = 1 - \frac{\sum_{t=1}^n |rank_X(t) - rank_Y(t)|}{N} \quad (\text{Eq. 4.18})$$

Table 4.4: Rank Similarity at Top $k\%$: *Naive* vs. *IR*

(a) Bias				(b) Controversy			
	<i>Top1%</i>	<i>Top5%</i>	<i>Top10%</i>		<i>Top1%</i>	<i>Top5%</i>	<i>Top10%</i>
<i>Kendall</i>	0.83	0.78	0.77	<i>Kendall</i>	0.77	0.82	0.83
<i>Spearman</i>	0.87	0.85	0.85	<i>Spearman</i>	0.83	0.88	0.89

Table 4.4(a) gives the similarity values between *IR* and *Naive*'s bias ranked lists at the top $k\%$, where $k \in \{1, 5, 10\}$. Table 4.4(b) gives the corresponding values for controversy. These similarity values are relatively high, as *IR* results in different rank positions only when reviewers (or objects) are especially affected by controversial objects (or biased reviewers). However, these values are still significantly below 1 (identical ranking), which means *IR* still ranks differently from *Naive*. In fact, the seemingly minor differences at the top ranks are deceptive, as we have seen at least one example of extreme difference (the circled object in Figure 4.4.b, with percentrank 70 by *IR* vs. percentrank 10 by *Naive*).

4.5.3 Case Examples

We now highlight specific examples to showcase that the bias/controversy percentranks produced by *IR* are more intuitively correct than *Naive*'s.

Reviewer Examples

First, we discuss the case of *user-2339*, whose profile is given in Table 4.5(a). This reviewer is assigned bias percentrank 4 by *Naive* and 13 by *IR* (see upper left-hand subsection of table). The lower bias percentrank by *IR* is because *user-2339* has deviated mostly on very controversial objects. *user-2339* rates 23 objects, and her median d_{ij} is 0.26 (see upper right-hand subsection of table). We observe that the twelve objects with higher (above-median) deviation are mainly controversial objects. Seven have controversy percentranks of 20 or below, three between 21–40, and two between 41–60 (see

Table 4.5: Case Examples: Bias

(a) Profile of user-2339 (with 23 objects)					(b) Profile of user-837 (with 69 objects)				
<i>Bias</i>		<i>d_{ij}</i>			<i>Bias</i>		<i>d_{ij}</i>		
<i>Naive</i>	<i>IR</i>	<i>min</i>	<i>med</i>	<i>max</i>	<i>Naive</i>	<i>IR</i>	<i>min</i>	<i>med</i>	<i>max</i>
4	13	0.10	0.26	0.53	16	6	0.10	0.22	0.66
<i>IR Controversy (d_{ij} ≥ med)</i>					<i>IR Controversy (d_{ij} ≥ med)</i>				
<i>1-20</i>	<i>21-40</i>	<i>41-60</i>	<i>61-80</i>	<i>81-100</i>	<i>1-20</i>	<i>21-40</i>	<i>41-60</i>	<i>61-80</i>	<i>81-100</i>
7	3	2	0	0	0	0	2	14	19

(c) Evidence			
<i>Reviewers</i>	<i>Bias</i>		<i>Evidence</i>
	<i>Naive</i>	<i>IR</i>	<i>IR</i>
user-2339	4	13	87
user-837	16	6	1

lower subsection of table). Thus, *IR* has not overly penalized *user-2339* for the high deviation that could very well be due to high controversy, and lowers her bias percentrank accordingly.

Our second case is *user-837*, whose profile is given in Table 4.5(b). *user-837* is assigned bias percentrank 16 by *Naive* and 6 by *IR*. The higher bias percentrank by *IR* is because *user-837* has deviated more on less controversial objects. The thirty-five objects on which *user-837* has higher deviations (with $d_{ij} \geq \text{user-837's median}$) have mainly low controversy (nineteen objects between percentranks 81–100, fourteen between 61–80, and two between 41–60). It is more likely that the high deviations are due to *user-837*'s own bias, rather than the objects' controversy. *IR* justifiably assigns a higher bias percentrank than *Naive*.

We can further show that the difference between *IR* and *Naive* in the above two cases can be attributed to evidence. When a reviewer has higher evidence, *IR* has a stronger case of the reviewer's bias, thus assigns a higher bias percentrank. Table 4.5(c) shows the bias percentranks (by *Naive* and *IR*) and evidence percentrank (by *IR*) of *user-2339* and *user-837* respectively. *user-2339* has low evidence percentrank of 87, and thus is

Table 4.6: Case Examples: Controversy

(a) Profile of object-749 (with 4 reviewers)					(b) Profile of object-3522 (with 3 reviewers)				
<i>Controversy</i>		<i>d_{ij}</i>			<i>Controversy</i>		<i>d_{ij}</i>		
<i>Naive</i>	<i>IR</i>	<i>min</i>	<i>med</i>	<i>max</i>	<i>Naive</i>	<i>IR</i>	<i>min</i>	<i>med</i>	<i>max</i>
4	63	0.20	0.27	0.47	10	6	0.20	0.30	0.30
<i>IR Bias (d_{ij} ≥ med)</i>					<i>IR Bias (d_{ij} ≥ med)</i>				
<i>1-20</i>	<i>21-40</i>	<i>41-60</i>	<i>61-80</i>	<i>81-100</i>	<i>1-20</i>	<i>21-40</i>	<i>41-60</i>	<i>61-80</i>	<i>81-100</i>
2	0	0	0	0	0	0	1	0	1

(c) Evidence			
<i>Objects</i>	<i>Controversy</i>		<i>Evidence</i>
	<i>Naive</i>	<i>IR</i>	<i>IR</i>
object-749	4	63	100
object-3522	10	6	1

assigned a lower bias percentrank by *IR*. *user-837* has high evidence percentrank of 1, and thus is assigned a higher bias percentrank by *IR*. Thus, *IR*'s taking evidence into account results in a different and more correct conclusion than *Naive*'s.

Object Examples

Similar observations can be made for case examples of objects. For instance, *object-749* (in Table 4.6(a)) is assigned a much lower controversy percentrank by *IR* (63 by *IR* vs. 4 by *Naive*) because the reviewers who deviate the most have very high bias percentranks. On the other hand, *object-3522* (in Table 4.6(b)) is assigned a higher controversy percentrank by *IR* (6 by *IR* vs. 10 by *Naive*) because the reviewers who deviate the most have very low percentranks. Table 4.6(c) reiterates that the controversy percentrank differences between *IR* and *Naive* is attributable to evidence.

4.6 Experiments on Synthetic Data

The main limitation of real-life data is a lack of “ground truth” on bias and controversy. To address this, we conduct experiments on synthetic data with known ground truth.

Table 4.7: Data Generation Parameters

<i>Parameter</i>	<i>Description</i>	<i>Default Value</i>
k	percentage of <i>biased</i> reviewers and <i>controversial</i> objects	30%
n	number of reviewers assigned to each object	10

Our objectives are to verify: (a) whether *IR* would be able to recall the ground truth successfully, (b) whether *IR* would have a better recall than *Naive*, and (c) whether a higher level of evidence would lead to a better recall.

4.6.1 Data Generation

Synthetic data generation is a complex task, because the propagation effects of bias and controversy in a rating network cannot be very precisely controlled. If the data generation scheme is too complex, we run the risk of not knowing the exact nature of the data. Thus, we chose to keep the data generation simple. However, by varying a number of well-chosen parameters, we could still draw meaningful insights.

The synthetic data simulate the scenario where there are two classes of reviewers: *biased* and *unbiased*. Similarly, there are two classes of objects: *controversial* and *non-controversial*. The different classes are associated with different behaviors. We synthetically generated rating scores based on these behaviors.

The synthetic data generation was a three-step process:

- (a) *Assign bias and controversy.* Given k , we labeled $k\%$ of reviewers *biased* and objects *controversial*, and the rest *unbiased* and *non-controversial*.
- (b) *Assign reviewers to objects.* Given n , we randomly assigned n reviewers to each object. The actual number of objects for each reviewer might vary, as the random assignment would not be perfectly even.
- (c) *Generate rating scores.* For each reviewer r_i assigned to an object o_j , we generated the rating score e_{ij} , such that $e_{ij} = 0.5$ if the reviewer was *unbiased* and the object was *non-controversial*, and a random number in $[0, 1]$ otherwise.

Metrics

Given synthetic data (without the pre-determined labels), *IR* and *Naive*'s task is to reclassify the reviewers into two classes (the top $k\%$ reviewers are considered *biased*, while the rest *unbiased*) and the objects into two classes (the top $k\%$ objects are considered *controversial*, while the rest *non-controversial*).

To measure the performance of each model, we adopt the following metrics:

- *Bias recall* is the percentage of the pre-determined $k\%$ *biased* reviewers that are also ranked among the top $k\%$ most biased by each model.
- *Controversy recall* is the percentage of the pre-determined $k\%$ *controversial* objects that are also ranked among the top $k\%$ most controversial by each model.

The recall is averaged over 20 sets of independently generated synthetic data. Since the sets of reviewers/objects are all of the same size ($k\%$), recall and precision are the same.

In the following experiments, we focus on the variation of recall due to two parameters: (a) the class proportion k , and (b) the number of reviewers per object n . The default values of k and n when fixed are given in Table 4.7. The number of reviewers and objects are always fixed at 1000 each. We have carried out other experiments showing that for a fixed k and n , increasing the number of reviewers and objects does not affect recall.

4.6.2 Varying Proportion of *biased* and *controversial* Classes k

We study how recall is affected by the change in the proportion of biased reviewers and controversial objects, by fixing $n = 10$, and varying k from 1% to 90%.

Recall

Figure 4.5.a and Figure 4.5.b plot the bias and controversy recalls respectively at various k . We observe that: (a) *IR* generally outperforms *Naive* in recall, and (b) for *IR* and *Naive*, recall generally decreases with increasing k , but increases slightly for $k > 70\%$.

CHAPTER 4. BIAS AND CONTROVERSY IN COLLABORATIVE RATING NETWORKS

Figure 4.5: Vary k : Recall (IR vs. Naive)

We can explain the above observations as follows.

- When there are few biased reviewers and controversial objects, the propagated effects of bias or controversy are limited and isolated. This makes it relatively easy to distinguish the few biased or controversial ones from the rest. Thus, recall is high when $k < 10\%$ for both *IR* and *Naive*.
- As there are increasingly more biased reviewers and controversial objects in the system, the effects of propagation increase as well. It becomes easier to confuse those who are truly biased and those who simply have rated controversial objects. Recall decreases for $10\% \leq k \leq 70\%$ proportion. However, *IR*'s recall decreases slower as *IR* better compensates for the effects of bias/controversy.
- Once the biased reviewers and controversial objects make up the vast majority ($k \geq 80\%$), it is no longer meaningful to identify them. At $k = 90\%$, the recall is also 90%, implying that a simple majority classifier could achieve the same recall.

Figure 4.6: Vary k : Distribution of Bias/Controversy

Distribution

The above explanations are further supported by Figure 4.6.a and Figure 4.6.b, which plot IR 's distribution of bias and controversy values across the percentranks, for three different k settings i.e., $k \in \{10\%, 30\%, 70\%\}$. For $k = 10\%$, there is a clear cut-off at 10 percentrank separating high and low bias (or controversy) values, which explains the high recall of IR . For $k = 30\%$, the cut-off at 30 percentrank is gentler but still evident. IR 's recall is lower than before, but is still high. For $k = 70\%$, there is no clear cut-off at 70 percentrank. IR 's recall is lowest at this point.

Evidence

Next, we study how evidence may affect recall. We compute the evidence of every reviewer and object (x_i and y_j values) using IR 's bias and controversy values (see Section 4.2.2). We then split the reviewers into two groups: *IR-Upper Half Evidence*, consisting of the top 50% of reviewers in terms of evidence, and *IR-Lower Half Evidence*, which consists of the remaining reviewers. For each group, we measure bias recall, which is the percentage of the l predetermined *biased* reviewers that are also ranked among the top l most biased

Figure 4.7: Vary k : Recall (Upper vs. Lower Half Evidence)

reviewers within the group. Recall and precision are practically the same as they both share the same denominator (l). A similar exercise is carried out for objects.

Figure 4.7.a and Figure 4.7.b respectively plot the bias recall and controversy recall at various k , comparing *IR-Upper Half Evidence*, *IR-Lower Half Evidence*, and *IR* (all reviewers/objects). The recall curves of *IR* are identical to those in Figures 4.5.a and 4.5.b. The recall curves of *IR-Upper Half Evidence* are higher than those of *IR*, which in turn are higher than those of *IR-Lower Half Evidence*. Higher evidence means each reviewer rates more non-controversial objects, and each object gets more unbiased reviewers. This makes it easier to identify bias or controversy, which leads to higher recall.

4.6.3 Varying Number of Reviewers per Object n

Recall

We study how n affects recall, by fixing $k = 30\%$ and varying n from 10 to 50. Figure 4.8.a and Figure 4.8.b respectively plot the varying bias and controversy recalls at various n . From these figures, we observe that: (a) *IR*'s recall consistently beats or equals *Naive*'s, and (b) for both *IR* and *Naive*, recall tends to increase with n .

CHAPTER 4. BIAS AND CONTROVERSY IN COLLABORATIVE RATING NETWORKS

Figure 4.8: Vary n : Recall (IR vs. Naive)

Probabilistically, $k = 30\%$ of a reviewer's objects are controversial. If all reviewers share the same proportion of controversial objects, a biased reviewer (who deviates on all objects) is easily distinguishable from an unbiased reviewer (who deviates only on $k = 30\%$ of objects). In practice, due to random assignment, a reviewer may have been assigned a varying proportion of controversial objects. An unbiased reviewer assigned mostly controversial objects may seem to deviate as much as a biased reviewer. This causes misclassification and hurts recall. However, as n increases, the proportion of controversial objects (controversy proportion) assigned to every reviewer converges towards the intended proportion ($k = 30\%$), and as a result recall increases as well.

In Figure 4.9, we plot the range of controversy proportions for various n . The points on each line denotes the minimum, average, and maximum values of controversy proportion among all reviewers. At $n = 10$, the maximum controversy proportion is 90%. A reviewer who deviates on 90% of objects would seem like a biased reviewer (who deviates on 100% of objects). With increasing n , the range tends to narrow. By $n = 50$, the maximum controversy proportion is only 50%. A reviewer who deviates on 50% of objects is unlikely to be confused with a biased reviewer.

CHAPTER 4. BIAS AND CONTROVERSY IN COLLABORATIVE RATING NETWORKS

Figure 4.9: Vary n : Variance in Controversy Proportion

4.10.a: Bias

4.10.b: Controversy

Figure 4.10: Vary n : Recall (Upper vs. Lower Half Evidence)

Evidence

Figure 4.10.a and Figure 4.10.b respectively plot the varying bias and controversy recalls at various n , comparing the different evidence groups (*IR-Upper Half Evidence*, *IR-Lower Half Evidence*, and *IR*). From these figures, we again observe that higher evidence generally leads to higher recall. The recall curves of *IR-Upper Half Evidence* are higher than those of *IR*, which in turn are higher than those of *IR-Lower Half Evidence*.

4.7 Discussion

Summary

In this chapter, we address the problem of measuring bias and controversy in collaborative rating networks. We develop the *Inverse Reinforcement* or *IR* model, which attributes to a reviewer only deviation due to his/her bias, and not to the controversy of rated objects. We further propose the *Evidence* model to determine the degree of support for each outcome of bias or controversy. Experiments on real-life data show that *IR* results in bias and controversy rankings which are significantly different from *Naive*'s. The differences between *IR* and *Naive* can be resolved in favor of *IR* and are often due to *IR*'s awareness (and *Naive*'s ignorance) of evidence. Experiments on synthetic data, with pre-determined ground truth, show that *IR* consistently achieves better bias recall and controversy recall than *Naive* over different parameter settings. We also show that higher evidence level generally leads to higher bias recall and controversy recall.

Related Work

The study of cognitive biases concerns people's predisposed opinions that may come from specific heuristics or mental shortcuts [Baz90, BL96, SHA99]. There are various types of cognitive biases, depending on their causes. For instance, base rate fallacy is caused by applying an inappropriate base rate (prior probability of an event); bandwagon effect by overreliance on other people's choices; congruence bias by overreliance on hypotheses that match one's prior belief; choice-supported bias by the tendency for positive attribution to previous choices; distinctive bias by the different frames of reference when evaluating objects jointly or separately.

Thus, hypothesizing on the possible causes is central to the study of cognitive biases. For example, [BH06] investigates whether reviewers' scores on peer review submissions are dependent on the reviewers' also being authors. Our approach is different as we focus on

CHAPTER 4. BIAS AND CONTROVERSY IN COLLABORATIVE RATING NETWORKS

detecting deviations attributable to reviewers or objects, without delving on their possible causes. Testing each possible hypothesis would require much additional information on reviewers or objects than is available. Such studies are also more appropriately done within the cognitive sciences.

Chapter 5

Quality and Leniency in Collaborative Rating Networks

5.1 Overview

In any rating system, one of the primary objectives is to identify the best objects based on the rating scores assigned by reviewers. In this chapter, we therefore address the *score summarization problem*: how to aggregate the rating scores given to an object to arrive at an overall score that reflects the *quality* of the object as well as possible. We use the following notations. A reviewer r_i may assign a rating score $e_{ij} \in [0, 1]$ to an object o_j . We seek to determine the quality q_j of every object o_j . Table 5.1 lists the new notations used in this chapter, in addition to those listed earlier in Table 4.1.

A straightforward approach of determining quality is to average the scores given to an object as shown in Eq. 5.1. This approach, which we term the *Naive* model, treats the scores by different reviewers equally.

$$q_j = \mathcal{A}vg_i e_{ij} \quad (\text{Eq. 5.1})$$

The *Naive* model would have been adequate if all reviewers were to rate all (or many) objects, as supported by the law of large numbers [GS82]. However, this is not a realistic and practical criterion supported by most social media applications, where users may

CHAPTER 5. QUALITY AND LENIENCY IN COLLABORATIVE RATING NETWORKS

Table 5.1: Table of Notations

Notation	Description
q_j	o_j 's quality
l_i	r_i 's leniency

	o_1	o_2	o_3	o_4	o_5		o_1	o_2	o_3	o_4	o_5
r_1	0.7	0.7	0.7	—	—	r_1	0.7	0.5	0.5	—	—
r_2	0.4	—	—	0.5	—	r_2	0.4	—	—	0.3	—
r_3	0.4	—	—	—	0.5	r_3	0.4	—	—	—	0.3
r_4	—	0.4	0.4	0.5	0.5	r_4	—	0.5	0.5	0.6	0.6
r_5	—	0.4	0.4	0.5	0.5	r_5	—	0.5	0.5	0.6	0.6
5.1.a: Rating Data 1						5.1.b: Rating Data 2					

Figure 5.1: Rating Data Examples

either voluntarily find or be assigned the objects to rate. Therefore, we consider rating scenarios where most objects are rated by few (less than 10) reviewers, which befit most social media applications. For instance, for the data that we collected from Epinions (see Section 5.4.1), 72% of objects receive 10 or fewer rating scores, and 60% of objects receive 5 or fewer rating scores. When an object receives a small number of rating scores, one or two reviewers could adversely skew its aggregate quality.

Example 5.1 *Figure 5.1.a and Figure 5.1.b show two sets of rating data under the same reviewer/object assignment. The matrix elements are the e_{ij} scores. A '—' denotes that the reviewer has not evaluated the object. In both data sets, o_1 receives the same set of scores (0.7 from r_1 , 0.4 from r_2 , and 0.4 from r_3). Using the averaging approach (Naive model), we arrive at the same overall score for o_1 ($q_1 = 0.5$) in both data sets. However, a more reasonable outcome is that o_1 should receive a lower overall score in the first than in the second data set.*

Consider Figure 5.1.a first. The varying scores received by o_1 suggest that either r_1 's score is too high or r_2 and r_3 's scores are too low. If we consider the scores of other objects, we observe that r_1 also assigns higher scores than her co-reviewers on o_2 and o_3 . In contrast, r_2 and r_3 tend to agree with their co-reviewers on o_4 and o_5 respectively. The

CHAPTER 5. QUALITY AND LENIENCY IN COLLABORATIVE RATING NETWORKS

record suggests that it is more likely that r_1 is lenient, and r_2 and r_3 are not. Thus, it makes sense to trust the scores by r_2 and r_3 more.

In Figure 5.1.b, r_1 tends to agree with her co-reviewers on o_2 and o_3 , while r_2 and r_3 show a record of assigning lower scores than the majority of their co-reviewers on o_4 and o_5 . In this case, it makes sense to trust r_1 's score more, even though r_1 is the minority.

Thus, *Naive* may arrive at the wrong quality of an object, as it considers only the face value of the rating scores, effectively treating all reviewers equally when in fact reviewers may not be on equal grounds when assigning the scores. Therefore, we propose to model the variance among reviewers in terms of *leniency*, or the tendency of a reviewer to assign a higher score to an object than the object deserves (as determined by the quality of the object). Once determined, the leniency information can be used to adjust the rating scores appropriately to arrive at q_j values that better reflect the quality of objects.

The behavioral concepts studied in this chapter (quality and leniency) are different from bias and controversy discussed in Chapter 4. The controversy of an object is an orthogonal concept to the quality of an object in that a controversial object can be of either high quality or low quality, and therefore cannot be used to measure the quality of an object. Bias is also different from leniency, in that a biased reviewer deviates on most objects, while a lenient reviewer generally assigns higher scores than other reviewers, but might well agree with the consensus once his/her scores have been corrected for leniency.

The rest of this chapter is organized as follows. Section 5.2 describes our proposed model. Section 5.3 discusses the two types of solutions derivable from the model: the *Exact* solution that represents leniency and quality as numeric measures, and the *Ranked* solution that represents leniency and quality as ranked measures. Sections 5.4 and 5.5 verify the effectiveness of our approach through experiments on real-life and synthetic data respectively. Section 5.6 concludes this chapter.

5.2 Leniency-aware Quality (LQ) Model

Given rating data, we seek to determine the quality q_j of each object o_j , by also determining the leniency l_i of each reviewer r_i and using it to derive q_j . We develop the *Leniency-aware Quality* or *LQ* model, that solves leniency and quality simultaneously, based on the two following insights about leniency.

- *Networked Approach to Leniency.* The leniency of a reviewer can only be determined relative to her co-reviewers. A reviewer who tends to give a higher rating score than a majority of co-reviewers on rated objects has a tendency of being lenient. Similarly, when considering the quality of an object, we need to consider how other objects have been rated by its reviewers.
- *Mutual Dependency between Leniency and Quality.* The two measures of interest: *leniency* and *quality* are mutually dependent. On one hand, to determine a reviewer's leniency, we need to know the quality of objects rated by the reviewer as a baseline to measure leniency. On the other hand, to determine the quality of an object, we need to know the leniency of its reviewers.

Our *LQ* model consists of a pair of equations (Eq. 5.2 and Eq. 5.3) that determine the leniency of reviewers and the quality of rated objects respectively. To measure how lenient a reviewer r_i is, we need to know how r_i 's rating scores compare to the quality of rated objects. Suppose that q_j is known, the extent to which the given score e_{ij} is inflated or deflated can be measured by $\frac{e_{ij}-q_j}{e_{ij}}$. Note that the inflation (or deflation) is measured relative to the base score e_{ij} ¹. If r_i regularly inflates her rating scores, we have even more evidence that r_i is lenient. Hence, to determine l_i , we aggregate $\frac{e_{ij}-q_j}{e_{ij}}$ over the set of objects that r_i has rated as shown in Eq. 5.2. Here, we use *average* as the

¹The case of $e_{ij} = 0$ should be avoided by replacing such e_{ij} with an appropriately small value.

CHAPTER 5. QUALITY AND LENIENCY IN COLLABORATIVE RATING NETWORKS

aggregation function. Consequently, $l_i > 0$ denotes a *lenient* reviewer, $l_i < 0$ denotes a *strict* reviewer, and $l_i = 0$ denotes a *neutral* reviewer.

$$l_i = \mathcal{A}vg_j \left(\frac{e_{ij} - q_j}{e_{ij}} \right) \quad (\text{Eq. 5.2})$$

Note that q_j in Eq. 5.2 is not known beforehand. It is to be determined as an aggregation (here, we assume *average*) of rating scores assigned to o_j . However, suppose that we know l_i of each r_i who has rated o_j , we can then compensate for each r_i 's tendency to inflate or deflate rating scores. This compensation approach to derive q_j is shown in Eq. 5.3. If $l_i < 0$, we revise the rating score e_{ij} upwards. If $l_i > 0$, we revise it downwards. The adjustment is proportional to the base score e_{ij} . $\alpha \in [0, 1]$ is a user-determined compensation factor, which controls the extent to which the scores may be adjusted to compensate for leniency. Larger α would lead to larger compensation.

$$q_j = \mathcal{A}vg_i [e_{ij} \cdot (1 - \alpha \cdot l_i)] \quad (\text{Eq. 5.3})$$

By compensating, Eq. 5.3 estimates the score that would have been assigned by a lenient reviewer had she been neutral ($l \approx 0$). That way, the quality of two objects rated by different sets of reviewers, who may use different ranges within the rating scale, would be more comparable. Eq. 5.3 can even return $q_j < \min_i(e_{ij})$, if all or most of o_j 's reviewers have $l_i > 0$, or $q_j > \max_i(e_{ij})$ if all or most reviewers have $l_i < 0$. This is possible as the *LQ* model does not look at each object in isolation, but instead considers the broader context (how its reviewers have rated other objects, how those other objects are rated by other reviewers, and so on). In contrast, the *Naive* model (Eq. 5.1) confines q_j to $[\min_i(e_{ij}), \max_i(e_{ij})]$. Thus, the *LQ* model is better positioned to “salvage” an object from a very skewed assignment of reviewers (such as an object whose reviewers are all strict).

Table 5.2: Quality and Leniency for Examples in Figure 5.1

(a) Rating Data 1			(b) Rating Data 2		
<i>Naive</i>	<i>LQ</i>		<i>Naive</i>	<i>LQ</i>	
$q_1 = 0.50$	$q_1 = 0.48$	$l_1 = 0.31$	$q_1 = 0.50$	$q_1 = 0.56$	$l_1 = 0.10$
$q_2 = 0.50$	$q_2 = 0.48$	$l_2 = -0.14$	$q_2 = 0.50$	$q_2 = 0.47$	$l_2 = -0.55$
$q_3 = 0.50$	$q_3 = 0.48$	$l_3 = -0.14$	$q_3 = 0.50$	$q_3 = 0.47$	$l_3 = -0.55$
$q_4 = 0.50$	$q_4 = 0.53$	$l_4 = -0.14$	$q_4 = 0.50$	$q_4 = 0.51$	$l_4 = 0.10$
$q_5 = 0.50$	$q_5 = 0.53$	$l_5 = -0.14$	$q_5 = 0.50$	$q_5 = 0.51$	$l_5 = 0.10$

The two variables l_i and q_j are mutually dependent and must be determined simultaneously. This dependency extends to all reviewers and objects connected to one another within the rating network. This is because to know a given l_i requires us to know the q_j of all objects rated by r_i . However, for each q_j , we need to know the leniency of r_i as well as those of r_i 's co-reviewers on o_j . This dependency could only be resolved by considering the leniency of every reviewer and the quality of every object simultaneously.

Note that the *Naive* approach is a special case of the above model. When $\alpha = 0$, no adjustment for leniency is done, and Eq. 5.3 is reduced into *Naive*'s Eq. 5.1.

Example 5.2 Table 5.2(a) and Table 5.2(b) display the quality and leniency computed using *Naive* and *LQ* models for Figure 5.1.a and Figure 5.1.b respectively. For this example, *LQ* uses the Exact solution (to be introduced in Section 5.3) at $\alpha = 0.5$. In both scenarios, *Naive* gives all objects the same quality of 0.50. We claim, however, that the different rankings of objects by *LQ* are more intuitive.

Consider Table 5.2(a) first. *LQ* considers objects rated by r_1 (o_1 , o_2 , and o_3) to be of lower quality than the rest (o_4 and o_5). Note that r_1 is considered lenient ($l_1 = 0.31$ by *LQ*) due to r_1 's tendency to give higher scores than her co-reviewers on o_1 , o_2 , and o_3 . Adjusting for r_1 's leniency, *LQ* arrives at the net lower quality of o_1 , o_2 , and o_3 (0.48 by *LQ*), as compared to that of o_4 and o_5 (0.53 by *LQ*).

In Table 5.2(b), *LQ* considers objects rated by r_2 or r_3 (o_1 , o_4 , and o_5) to be of higher quality than the rest (o_2 and o_3). Adjusting for the strict scoring by r_2 and r_3 ($l_2 = -0.55$,

$l_3 = -0.55$ by LQ), LQ results in the higher quality of objects rated by r_2 or r_3 ($q_1 = 0.56$, $q_4 = 0.51$, $q_5 = 0.51$ by LQ) than those of o_2 and o_3 ($q_2 = 0.47$, $q_3 = 0.47$ by LQ).

Relative and Absolute Compensation Modes

Eq. 5.2 (and correspondingly Eq. 5.3) models leniency in relative terms. A reviewer's leniency is assumed to induce her to inflate (or deflate) her scores by a certain fraction or percentage. For instance, a reviewer with $l_i = 0.1$ tends to inflate her scores by 10%. In Eq. 5.2, the difference between e_{ij} and q_j is taken relative to e_{ij} . In turn, in Eq. 5.3, the compensation component $(1 - \alpha \cdot l_i)$ adjusts the score e_{ij} in relative terms as well. We term this approach the *Relative* compensation mode.

Another approach is to model leniency in absolute terms, which we term the *Absolute* compensation mode. In this approach, l_i is an absolute value by which r_i inflates (or deflates) her scores. For instance, a reviewer with $l_i = 0.1$ tends to inflate her scores by 0.1. This compensation mode gives rise to a different pair of leniency and quality equations (Eq. 5.4 and Eq. 5.5). In determining leniency, the difference between e_{ij} and q_j is taken as an absolute value ($e_{ij} - q_j$) in Eq. 5.4. In determining quality, the adjustment to e_{ij} is absolute ($e_{ij} - \alpha \cdot l_i$) in Eq. 5.5.

$$l_i = \mathcal{A}vg_j (e_{ij} - q_j) \quad (\text{Eq. 5.4})$$

$$q_j = \mathcal{A}vg_i (e_{ij} - \alpha \cdot l_i) \quad (\text{Eq. 5.5})$$

The main difference between the two compensation modes is the underlying assumption on the mechanism by which a lenient reviewer would inflate (or deflate) her scores. However, since the equation for quality adjusts for leniency accordingly, the outcomes of the two modes may not be very different. Our experiments in Section 5.5 show that there are only minor differences between the two modes in terms of the quality of objects.

5.3 Solution Types

A solution to the LQ model tells us the relative comparison among objects in terms of quality, and among reviewers in terms of leniency. We identify two approaches to reach a solution. The first approach, which we call the *Exact* solution, treats the model as a linear system of equations to be solved for exact values of quality and leniency. The second approach, which we call the *Ranked* solution, treats the model as a ranking problem and solves it for a ranking of objects by quality and a ranking of reviewers by leniency. The two solutions may not be identical. *Ranked* solution is valuable as sometimes no *Exact* solution exists, but a unique ranking still exists and knowing the ranking suffices for the application. For instance, a conference program chair may only be interested in ranking all submitted papers by quality so as to accept the best papers. Below, we characterize these two solutions for the *Relative* mode. Similar discussions apply to *Absolute*.

The set of equations comprising Eq. 5.3 for every o_j and Eq. 5.2 for every r_i can be more compactly expressed as matrix equations Eq. 5.6 and Eq. 5.7. For m reviewers and n objects, Q is $n \times 1$ vector of q_j 's, L is $m \times 1$ vector of l_i 's, and $\mathbf{1}$ is a vector of appropriate length containing all 1's. U is $m \times n$ matrix whose element $u_{ij} = [(c_{ij} \cdot e_{ij}) / \sum_i c_{ij}]$, where $c_{ij} = 1$ when r_i has evaluated o_j , and 0 otherwise. V is $m \times n$ matrix whose element $v_{ij} = (c_{ij} / \sum_j c_{ij})$. W is $m \times n$ matrix whose element $w_{ij} = [(c_{ij} / e_{ij}) / \sum_j c_{ij}]$. Q and L are variables, and the rest are inputs.

$$Q = U^T \mathbf{1} - \alpha U^T L \quad (\text{Eq. 5.6})$$

$$L = V \mathbf{1} - W Q \quad (\text{Eq. 5.7})$$

Substituting Eq. 5.7 into Eq. 5.6, we get a recursive equation in terms of Q given in Eq. 5.8, where $X = (U^T \mathbf{1} - \alpha U^T V \mathbf{1})$ and $Y = (\alpha U^T W)$. Intuitively, any q_j (in left-hand

side Q) could be expressed in terms of other q_j 's (in right-hand side Q), as determined by X and Y that govern how these objects are connected in the network.

$$Q = X + YQ \quad (\text{Eq. 5.8})$$

Subsequently, we distinguish between the *Exact* solution, which solves Eq. 5.8 as a linear system of equations, and the *Ranked* solution, which derives a unique ranking from an eigenvector equation modified from Eq. 5.8.

5.3.1 Exact Solution

The *Exact* solution is the unique value of Q (and the corresponding L) satisfying Eq. 5.8. The matrix equation Eq. 5.8 stands for a system of n linear equations in terms of various q_j 's. From linear algebra [AR87], we know that such a system of linear equations may be in one of three situations:

Case 1: consistent and uniquely determined, there is one unique solution, which is the intersection point of the n linear equations

Case 2: consistent and underdetermined, there are infinitely many solutions, which lie on the line or plane where the linear equations meet

Case 3: inconsistent, there is no solution as the linear equations do not meet

Hence, *Exact* solution exists only under Case 1, which produces a unique Q . This solution is given in Eq. 5.9. For the solution to be unique, $(I - Y)$ must be invertible, which is true if and only if $\det(I - Y) \neq 0$. Once Q is determined, L can be derived using Eq. 5.7. Elements of Q and L are the exact values of quality and leniency that we are interested in. Failing the test $\det(I - Y) \neq 0$, Eq. 5.8 falls under Case 2 or Case 3, for which *Exact* solution does not exist.

$$Q = (I - Y)^{-1}X \quad (\text{Eq. 5.9})$$

5.3.2 Ranked Solution

For the *Ranked* solution, we are only interested in the ranking by quality (and by leniency). We could derive such a ranking from Eq. 5.10, which is modified from Eq. 5.8 by adding a non-zero, real-valued scalar variable λ . Intuitively, Eq. 5.10 says that any q_j (in left-hand side Q) could be expressed in terms of other q_j 's (in right-hand side Q), after rescaling by λ . In other words, the Q that satisfies Eq. 5.10 would preserve the relative ratio among q_j elements (and the ranking by quality).

$$\lambda Q = X + YQ \quad (\text{Eq. 5.10})$$

Due to the λ variable, *Ranked*'s Eq. 5.10 is fundamentally different from *Exact*'s Eq. 5.8. Thus, the two solutions may not produce identical rankings. For *Ranked*, we are only interested that such a λ exists, but the value or sign of λ is not important, as once λ is known, we could always rescale λQ back to Q (normalization).

Since we are only interested in the direction of vector Q (Q or any scaling of Q is acceptable), we can re-formulate Eq. 5.10 as an eigenvector equation (Eq. 5.11). The $n \times n$ matrix X^n is formed by replicating the $n \times 1$ vector X across n columns. β is the inverse of the sum of elements of Q , i.e., $\beta = (\sum_j q_j)^{-1}$. We see that Q is in fact an eigenvector of $(\beta X^n + Y)$. In fact, what we want is the dominant eigenvector.

$$\lambda Q = (\beta X^n + Y) Q \quad (\text{Eq. 5.11})$$

As β and Q are mutually dependent, we could break this dependency by fixing the value of β in order to derive a unique dominant eigenvector Q . An intuitive choice for β value is the inverse of the sum of quality by the *Naive* model. This has the advantage

Table 5.3: LQ Solutions

		<i>Solution Type</i>	
		<i>Exact</i>	<i>Ranked</i>
<i>Compensation Mode</i>	<i>Relative</i>	R-Exact	R-Ranked
	<i>Absolute</i>	A-Exact	A-Ranked

of preserving the sum of quality before and after compensation, which would prevent a *general* inflation or deflation of quality (for the quality of some objects to go up, those of others must come down). For a fixed value of β , the eigenvector equation can be more simply expressed as Eq. 5.12, where $Z = \beta X^n + Y$.

$$\lambda Q = Z Q \quad (\text{Eq. 5.12})$$

Iterative methods [AR87, GVL96] can be used to solve Eq. 5.12 to get the dominant eigenvector Q . The iterative form is $Q_{k+1} = Z Q_k$. The only variable is Q , as λ is removed by normalizing Q after each iteration. Normalization in this case returns Q to the state of $\sum_j q_j = \beta^{-1}$, where β^{-1} is the sum of quality by *Naive*. Subject to the assumption that Z is *diagonalizable* (it has linearly-independent eigenvectors) and has a uniquely largest eigenvalue [GVL96], then as k increases, Q_k will converge to the dominant eigenvector of Z almost independently of the initial Q_0 . Once converged, the elements of Q (and the corresponding L) are used to rank objects (and reviewers).

5.4 Experiments on Real-Life Data

The experimental objective is to verify the efficacy of the proposed *LQ* model. Table 5.3 shows *LQ*'s four possible solutions, owing to two compensation modes (*Relative* and *Absolute*) and two solution types (*Exact* and *Ranked*). In these experiments, we set $\alpha = 0.5$, for which *Exact* and *Ranked* solutions exist. Experiments on different α values will be covered by our experiments on synthetic data (see Section 5.5).

We compare the four LQ solutions against one another as well as against two comparative models. The first is the simple averaging *Naive* model (Eq. 5.1). The second is a weighted averaging model proposed by [RW01], which we call *Riggs* after its first author. *Riggs* determines the quality of an object by taking a weighted average of rating scores as shown in Eq. 5.13. The weight w_i of a reviewer r_i is measured by how closely the reviewer's scores (e_{ij}) are to the computed quality q_j on average as shown in Eq. 5.14.

$$q_j = \frac{\sum_i w_i \times e_{ij}}{\sum_i w_i} \quad (\text{Eq. 5.13})$$

$$w_i = 1 - \underset{j}{Avg} |e_{ij} - q_j| \quad (\text{Eq. 5.14})$$

5.4.1 Data

The data were collected from Epinions product review site. We crawled pages from the site for two days (Nov. 28–29, 2005), starting from a seed page². The collected pages represented only a subset of all reviewers, objects (products), and ratings in Epinions.

We rescaled the rating scores, originally on the scale of 1 to 5 stars, to a new range of 0.2 to 1.0 by a simple division by 5. We retained only ratings within the *Videos & DVDs* category, which was one of the categories with the most number of objects within the collected data. We ensured that each object had at least 3 reviewers, and each reviewer had at least 3 objects, by iteratively removing objects with less than 3 reviewers, and reviewers with less than 3 objects, until there was no more such object/reviewer. This removed the occasional reviewers/objects, and lent greater support when inferring the “behavior” of reviewers/objects.

Table 5.4 shows the final data size, with 172 reviewers, 165 objects, and 1157 ratings. The small size of the data allows for more detailed analysis of the experimental results.

²http://www.epinions.com/member/community_lists.html/show_~6/display_list_~true/vert_~3321654/year_~1900/sec_~community_member_list/pp_~1/pa_~1

Table 5.4: Data Size

	<i>count</i>
<i>reviewers</i>	172
<i>objects</i>	165
<i>ratings</i>	1157
<i>ratings per object</i>	3 – 14 (median: 7)
<i>ratings per reviewer</i>	3 – 39 (median: 5)

There are also few ratings per object (median of 7) and per reviewer (median of 5). This is suitable for testing the hypothesis, mentioned in Section 5.1, that *LQ* model would be especially useful when a typical object receives few ratings and is susceptible to distortion by a small subset of its reviewers.

5.4.2 Comparison of Ranked Lists

All the comparative solutions were run on the above dataset to compute quality (and leniency for *LQ* solutions). For each solution, objects (or reviewers) were ranked in descending order of quality (or leniency). The highest value was given rank 1. Same values shared the same rank. For example, if the next three highest values were the same, they would share rank 2. Here, we compare the quality ranked lists generated by different solutions, to see how different *LQ* solutions are from one another, as well as from *Naive* or *Riggs*.

Naive vs. Ranked

First, we compare the ranked lists produced by *R-Ranked* and *A-Ranked* to that by *Naive*. Figure 5.2.a shows a scatterplot of quality ranks for *R-Ranked* vs. *Naive*. Each point represents an object. Values on the *x*- and *y*-axes represent quality ranks computed by *R-Ranked* and *Naive* respectively. Figure 5.2.b is the corresponding scatterplot for *A-Ranked* vs. *Naive*. We draw the following observations from both figures:

- As shown in both Figures 5.2.a and 5.2.b, there are significant variances around the diagonal, revealing that both *R-Ranked* and *A-Ranked* rank objects differently

CHAPTER 5. QUALITY AND LENIENCY IN COLLABORATIVE RATING NETWORKS

Figure 5.2: Quality Rank Scatterplots: *Naive* vs. *Ranked*

from *Naive*. In particular, there are 29 objects sharing rank 1 by *Naive*. The same objects are given ranks ranging from 2 to 62 by *R-Ranked* (Figure 5.2.a) and from 1 to 49 by *A-Ranked* (Figure 5.2.b). Thus, the *LQ* solutions are more successful at differentiating even very competitive objects. This is useful in such situations as selecting the very best papers at conferences or proposals for funding.

- There are greater variances at the top ranks than at the bottom ranks. There is a lower density of objects at the lower end of the quality spectrum, which makes it harder for a low-quality object to displace another higher- or lower-ranked object.

Naive vs. Riggs

For comparison, we show the scatterplot of quality ranks for *Naive* vs. *Riggs* in Figure 5.3. Evidently, *Naive* is much more similar to *Riggs* than to any *LQ* solution. This suggests that the distortion in rating scores caused by leniency is much more significant than that by reputation (*Riggs*), and that *LQ* results in a more significant correction of *Naive*'s quality ranking than *Riggs* does.

CHAPTER 5. QUALITY AND LENIENCY IN COLLABORATIVE RATING NETWORKS

Figure 5.3: Quality Rank Scatterplot: *Naive* vs. *Riggs*

5.4.a: Relative Compensation

5.4.b: Absolute Compensation

Figure 5.4: Quality Rank Scatterplots: *Exact* vs. *Ranked***Exact vs. Ranked**

Figures 5.4.a and 5.4.b are the scatterplots for *R-Ranked* vs. *R-Exact* and *A-Exact* vs. *A-Ranked* respectively. The points line up along the diagonal, implying that for this experiment the *Exact* and *Ranked* solutions are practically identical. As leniency and quality are mutually dependent, the scatterplots for leniency are similar to Figures 5.4.a and 5.4.b. Due to the *Exact/Ranked* similarity, thereafter we use only *Ranked* to represent *LQ*.

Table 5.5: Profile of Object *mu1006829*

object		Quality (Rank)			
		<i>Naive</i>	<i>Riggs</i>	<i>R-Ranked</i>	<i>A-Ranked</i>
mu1006829		1.00 (1)	1.00 (1)	0.95 (55)	0.97 (43)
reviewers	e _{ij}	Leniency (Rank)			
		<i>R-Ranked</i>		<i>A-Ranked</i>	
head-driller	1.0	0.16 (8)		0.13 (9)	
jiggyjay	1.0	0.13 (10)		0.11 (11)	
melliangel	1.0	0.09 (20)		0.07 (27)	
mrsocko316	1.0	0.07 (31)		0.06 (37)	
sadgit	1.0	0.06 (40)		0.05 (64)	
drdevience	1.0	0.05 (50)		0.05 (54)	
mike_bracken	1.0	0.04 (69)		0.08 (22)	
mcmaster	1.0	0.02 (79)		0.03 (83)	
jeremy1456	1.0	-0.08 (123)		0.02 (84)	

5.4.3 Case Examples

Here, we showcase how *LQ* model is more intuitively correct, by providing two examples of objects upon which *LQ* solutions disagree with *Naive* or *Riggs* on their quality ranks, and showing how the disagreement can be explained in favor of *LQ*. These are followed by two examples of reviewers with very different leniency ranks, showing how the difference comes about due to their rating behaviors.

Object Examples

Table 5.5 describes the profile of object *mu1006829*, showing its quality values (and ranks) computed by different solutions, its rating scores, and the leniency values (and ranks) of its reviewers. For *mu1006829*, the quality ranks assigned by *R-Ranked* (rank 55) and *A-Ranked* (rank 43) are much lower than those assigned by *Naive* (rank 1) or *Riggs* (rank 1). This is because *mu1006829*'s reviewers are generally lenient (with $l_i > 0$). *R-Ranked* and *A-Ranked* recognize and compensate for their tendency to inflate the rating scores, resulting in a lower quality rank for *mu1006829*.

The second object example *mu1016864*, whose profile is shown in Table 5.6, receives much higher quality ranks from *R-Ranked* (rank 1) and *A-Ranked* (rank 14) than from

CHAPTER 5. QUALITY AND LENIENCY IN COLLABORATIVE RATING NETWORKS

Table 5.6: Profile of Object *mu1016864*

object		Quality (Rank)			
		<i>Naive</i>	<i>Riggs</i>	<i>R-Ranked</i>	<i>A-Ranked</i>
mu1016864		0.98 (35)	0.98 (36)	1.05 (1)	1.00 (14)
reviewers	e_{ij}	Leniency (Rank)			
		<i>R-Ranked</i>		<i>A-Ranked</i>	
<i>edmaidel</i>	1.0	-0.80 (170)		-0.21 (171)	
<i>george_chabot</i>	1.0	-0.14 (137)		-0.06 (142)	
<i>janesbit1</i>	1.0	-0.06 (116)		-0.05 (134)	
<i>matthewn</i>	1.0	-0.05 (110)		-0.04 (128)	
<i>cripper</i>	1.0	-0.01 (97)		-0.01 (105)	
<i>icariusrex</i>	1.0	0.00 (95)		0.02 (87)	
<i>munkus</i>	1.0	0.00 (94)		0.01 (96)	
<i>ninput</i>	0.8	-0.54 (167)		-0.12 (156)	

Naive (rank 35) and *Riggs* (rank 36). *mu1016864*'s reviewers are mostly strict (with $l_i < 0$). In particular, *edmaidel* is very strict ($l_i = -0.80$ by *R-Ranked*, $l_i = -0.21$ by *A-Ranked*). Note that the leniency values are in relative terms for *R-Ranked* and in absolute terms for *A-Ranked*. These reviewers' tendency to deflate their rating scores are taken into account by *R-Ranked* and *A-Ranked*, which then lift their quality ranks correspondingly. The quality of *mu1016864* determined by *R-Ranked* is 1.05, which is higher than the maximum e_{ij} (1.0). The higher value of 1.05 received by *mu1016864* is due to *mu1016864*'s receiving perfect e_{ij} scores even from reviewers who are very strict on other objects.

Reviewer Examples

A reviewer's leniency is determined by her rating behavior: whether she consistently rates higher or lower than the derived quality. Table 5.7 shows the profile of *edmaidel*, a strict reviewer ($l_i = -0.80$ by *R-Ranked*, $l_i = -0.21$ *A-Ranked*) and very low leniency ranks (rank 170 by *R-Ranked*, rank 171 by *A-Ranked*, out of 172 reviewers). Comparing *edmaidel*'s rating score e_{ij} and the quality q_j of each rated object, we observe that the rating scores are consistently higher across the six objects ($1.0 < 1.05$, $0.4 < 0.89$, $0.4 < 0.49$, $0.4 < 0.42$, $0.2 < 0.59$, $0.2 < 0.46$ for e_{ij} vs. q_j by *R-Ranked*).

Table 5.7: Profile of Reviewer *edmaidel*

reviewer		Leniency (Rank)	
		<i>R-Ranked</i>	<i>A-Ranked</i>
edmaidel		-0.80 (170)	-0.21 (171)
objects	e_{ij}	Quality (Rank)	
		<i>R-Ranked</i>	<i>A-Ranked</i>
<i>mu1016864</i>	1.0	1.05 (1)	1.00 (14)
<i>mu1015561</i>	0.4	0.89 (103)	0.89 (102)
<i>mu1134655</i>	0.4	0.49 (161)	0.49 (161)
<i>mu1145001</i>	0.4	0.42 (163)	0.40 (163)
<i>mu1130966</i>	0.2	0.59 (158)	0.59 (158)
<i>mu1148492</i>	0.2	0.46 (162)	0.47 (162)

Table 5.8: Profile of Reviewer *heymrdj2k*

reviewer		Leniency (Rank)	
		<i>R-Ranked</i>	<i>A-Ranked</i>
heymrdj2k		0.24 (2)	0.17 (3)
objects	e_{ij}	Quality (Rank)	
		<i>R-Ranked</i>	<i>A-Ranked</i>
<i>mu1003625</i>	1.0	0.85 (125)	0.85 (122)
<i>mu1083887</i>	0.8	0.67 (151)	0.69 (150)
<i>mu1152188</i>	0.8	0.64 (155)	0.65 (152)
<i>mu1098652</i>	0.6	0.34 (165)	0.33 (165)

Table 5.8 shows the profile of reviewer *heymrdj2k*, with positive leniency values and very high leniency ranks (0.24 and rank 2 by *R-Ranked*, 0.17 and rank 3 by *A-Ranked*). *heymrdj2k*'s rating scores on her four rated objects are consistently higher than the respective quality values ($1.0 > 0.85$, $0.8 > 0.67$, $0.8 > 0.64$, $0.6 > 0.34$ for e_{ij} vs. q_j by *R-Ranked*).

5.5 Experiments on Synthetic Data

Most real-life data do not have “ground truth” information on quality or leniency. Our experiments with synthetically generated data address the following objectives: (a) to verify *LQ*'s effectiveness against a known ground truth, and (b) to study the effects of various parameters on *LQ*'s recall of the ground truth.

Table 5.9: Data Generation Parameters

Parameter	Description	Default Value
k	percentage of non- <i>neutral</i> reviewers	60%
m	percentage of <i>lenient</i> among non- <i>neutral</i> reviewers	50%
n	number of reviewers assigned to each object	10

Table 5.10: Generated Rating Scores (e_{ij} values)

	<i>strict</i> ($l_i = -0.4$)	<i>neutral</i> ($l_i = 0$)	<i>lenient</i> ($l_i = 0.4$)
<i>low-quality</i> ($q_j = 0.4$)	0.29	0.40	0.67
<i>high-quality</i> ($q_j = 0.6$)	0.43	0.60	1.00

5.5.1 Data Generation

Synthetic data generation is a rather complex process, as the propagation effect within a network (such as a rating network) and the interaction between data generation parameters cannot be very precisely controlled. We choose to keep the data generation scheme simple, with few well-chosen parameters that still allow us to draw meaningful insights.

The synthetic data simulates the scenario where there are two classes of objects: *low-quality* and *high-quality*, and three classes of reviewers: *strict* ($l_i < 0$), *neutral* ($l_i = 0$), and *lenient* ($l_i > 0$) associated with different rating behaviors. There are 3 parameters: k , m , and n , as described in Table 5.9.

The data generation schemes involves the following steps:

- (a) *Assign quality values.* Label 50% of objects *low-quality*, and assign them $q_j = 0.4$. Label the rest *high-quality*, and assign them $q_j = 0.6$.
- (b) *Assign leniency values.* Select $k\%$ of reviewers, label $m\%$ of them *lenient* (and assign $l_i = 0.4$), and label the other $(100 - m)\%$ *strict* (and assign $l_i = -0.4$). Label the remaining $(100 - k)\%$ of reviewers *neutral* (and assign $l_i = 0$).
- (c) *Assign reviewers to objects.* Randomly assign n reviewers to every object. On average, there would be n objects per reviewer, but the actual number may vary among reviewers.

CHAPTER 5. QUALITY AND LENIENCY IN COLLABORATIVE RATING NETWORKS

- (d) *Generate rating scores.* For each r_1 assigned to o_j , compute rating score e_{ij} as in Eq. 5.15. This equation is simply a rearrangement of Eq. 5.3, moving e_{ij} to the left-hand side and assuming $\alpha = 1$ (to maximize the distortion of rating scores).

$$e_{ij} = q_j \div (1 - l_i) \quad (\text{Eq. 5.15})$$

As shown in Table 5.10, *low-quality* objects receive rating scores in $\{0.29, 0.40, 0.67\}$, depending on whether the reviewer is *strict*, *neutral*, or *lenient*. *high-quality* objects receive rating scores in $\{0.43, 0.60, 1.00\}$. As the score ranges of the two object classes overlap, there is a chance of confusion, such as when a *low-quality* object with mainly *lenient* reviewers mistaken for a *high-quality* object, or when a *high-quality* object with mainly *strict* reviewers mistaken for a *low-quality* object.

Metrics

Given a synthetic data (without the pre-determined labels), each solution computes the quality values and attempts to reclassify the objects into the two quality classes with minimal mistake. To measure the performance of a solution, we use the following metrics:

- *Quality recall* is defined as the percentage of ground-truth *high-quality* objects that are among the top 50% in terms of quality values as computed by the solution. Quality recall ranges from 0% to 100%. Due to the 50:50 split between the two quality classes, recall of *high-quality* is the same as recall of *low-quality*. Since the sets to be compared are of the same size, recall and precision are also the same.
- *Leniency recall* (for *LQ* solutions only) is defined as the *Pearson's correlation coefficient* [WMMY02] between the vector of ground-truth leniency values L^G and the vector of derived leniency values L^S , which is determined as follows. Using the computed leniency ranking of reviewers, we classify the top $k\% \times m\%$ of reviewers

lenient (and assign them $l_i = 0.4$), the next $(100 - k)\%$ *neutral* (and assign them $l_i = 0$), and the last $k\% \times (100 - m)\%$ *strict* (and assign them $l_i = -0.4$), mimicking the ground truth distribution. Leniency recall ranges from -100% (negatively correlated from ground truth) to 100% (perfectly correlated with ground truth).

In these experiments, we vary one parameter (n , k , or m) and keep the rest fixed at the default values shown in Table 5.9. We average the recall values over 25 independently generated synthetic data sets. Each data set has 1000 reviewers and 1000 objects. We have conducted separate experiments with larger number of reviewers/objects with similar results. We only use a data set if each object has at least 3 reviewers, and each reviewer has at least 3 objects. For *LQ*, although $\alpha = 1$ is used during data generation, we do not use this information, and instead set $\alpha = 0.5$ as done in Section 5.4.

5.5.2 Varying Number of Reviewers per Object n

Recall

We study how the number of reviewers assigned to each object n affects recall. Figure 5.5.a plots quality recall at different values of n . It shows that recall increases with n , and that *R-Ranked* and *A-Ranked* generally outperform *Naive* or *Riggs*. The random assignment of reviewers to objects means that objects may have different compositions of reviewers in terms of leniency.

- For small n , it is more likely for an object to be assigned mainly *lenient* or mainly *strict* reviewers, which significantly distorts its rating scores. *LQ* solutions take the leniency of each reviewer into account, better compensating for the distortion, resulting in higher quality recall.
- As n increases, statistically the assignment gets more even, and more objects will share a similar composition of reviewers, which is the underlying distribution of

CHAPTER 5. QUALITY AND LENIENCY IN COLLABORATIVE RATING NETWORKS

Figure 5.5: Vary n : Quality Recall and Leniency Recall

reviewers in terms of leniency. As a result, the variance due to reviewers' leniency will get less important, as most objects are affected similarly. It gets easier to separate the two classes of objects, and all solutions move towards perfect recall.

Figure 5.5.b plots leniency recall for *R-Ranked* and *A-Ranked*, which shows the same trend of increasing recall with n . In general, this is expected, as leniency and quality are mutually dependent. Hence, in most cases, showing quality recall is sufficient.

Distribution of Leniency Classes among An Object's Reviewers

To show that as n increases, objects get a more similar composition of reviewers, we look at how the distribution of leniency classes among an object's reviewers changes with n .

Each object o_j has a distribution vector $\mathbf{d}_j = [d_-, d_0, d_+]$, where $\mathbf{d}_j.d_-$, $\mathbf{d}_j.d_0$, $\mathbf{d}_j.d_+$ are the percentages of o_j 's reviewers who are *strict*, *neutral*, and *lenient* (according to ground truth) respectively. Based on the input parameters in Table 5.9, the expected distribution vector is $\mathbf{d}_J = [30\%, 40\%, 30\%]$. However, due to the random assignment of reviewers to objects, $\mathbf{d}_j$ may deviate from $\mathbf{d}_J$. The *distribution error* of an object

Figure 5.6: Vary n : Leniency Class Distribution Error

o_j is defined as the Euclidean distance from the actual distribution $\mathbf{d}_j$ to the expected distribution $\mathbf{d}_J$ as shown in Eq. 5.16.

$$dist(\mathbf{d}_j, \mathbf{d}_J) = \sqrt{(\mathbf{d}_j.d_- - \mathbf{d}_J.d_-)^2 + (\mathbf{d}_j.d_0 - \mathbf{d}_J.d_0)^2 + (\mathbf{d}_j.d_+ - \mathbf{d}_J.d_+)^2} \quad (\text{Eq. 5.16})$$

Figure 5.6 plots the median and the standard deviation of the distribution errors ($dist(\mathbf{d}_j, \mathbf{d}_J)$ values) across all o_j 's. As n increases, both median and standard deviation decrease, which means all objects uniformly approach the expected distribution $\mathbf{d}_J$.

5.5.3 Varying Proportion of Non-*neutral* Classes k

We study how the proportion of non-*neutral* reviewers k affects quality recall. Figure 5.7.a plots quality recall for different values of k . It shows that recall decreases with k , and that *R-Ranked* and *A-Ranked* generally outperform *Naive* and *Riggs*. For *R-Ranked*, Figure 5.7.b shows quality recall curves at different values of n . The same trend of decreasing recall applies to other values of n , with lower recall figures for lower n .

The reason for decreasing recall with increasing k is the greater likelihood for an object to get an imbalanced distribution of reviewers. Table 5.11 shows, for varying k , the

CHAPTER 5. QUALITY AND LENIENCY IN COLLABORATIVE RATING NETWORKS

Figure 5.7: Vary k : Quality RecallTable 5.11: Vary k : Percentages of Objects with High $\mathbf{d}_j.d_-$ or $\mathbf{d}_j.d_+$

k	$\mathbf{d}_j.d_-$			$\mathbf{d}_j.d_+$		
	$\geq 60\%$	$\geq 70\%$	$\geq 80\%$	$\geq 60\%$	$\geq 70\%$	$\geq 80\%$
10%	0	0	0	0	0	0
30%	0	0	0	0	0	0
50%	2	0	0	2	0	0
70%	10	3	1	10	3	0
90%	27	10	3	26	10	3

percentage of objects assigned mainly *strict* reviewers ($\mathbf{d}_j.d_- \geq 60\%$, 70%, or 80%) and the percentage of objects assigned mainly *lenient* reviewers ($\mathbf{d}_j.d_+ \geq 60\%$, 70%, or 80%). It shows that the percentages of both types of objects are higher when k is higher. At $k = 10\%$, there is no object with $\mathbf{d}_j.d_- \geq 60\%$ or $\mathbf{d}_j.d_+ \geq 60\%$. At $k = 90\%$, 27% of objects has $\mathbf{d}_j.d_- \geq 60\%$ and 26% of objects has $\mathbf{d}_j.d_+ \geq 60\%$. Since misclassification occurs when a *low-quality* object with mainly *lenient* reviewers is confused with a *high-quality* object with mainly *strict* reviewers, it follows that as the number of objects with imbalanced distribution of reviewers rises, the misclassification rate also increases.

CHAPTER 5. QUALITY AND LENIENCY IN COLLABORATIVE RATING NETWORKS

Figure 5.8: Vary m : Quality RecallTable 5.12: Vary m : Percentages of Objects with High $\mathbf{d}_j.d_-$ or $\mathbf{d}_j.d_+$

m	$\mathbf{d}_j.d_-$			$\mathbf{d}_j.d_+$		
	$\geq 60\%$	$\geq 70\%$	$\geq 80\%$	$\geq 60\%$	$\geq 70\%$	$\geq 80\%$
10%	48	24	9	0	0	0
30%	20	7	2	0	0	0
50%	5	1	0	4	1	0
70%	0	0	0	20	7	2
90%	0	0	0	48	24	9

5.5.4 Varying Proportion of *lenient* within Non-*neutral* Classes m

We study how varying the proportion of *lenient* within non-*neutral* reviewers m affects quality recall. Figure 5.8.a plots quality recall for different values of m . Again, *R-Ranked* and *A-Ranked* outperform *Naive* and *Riggs*. As m increases, recall initially decreases, reaches a trough in $40\% \leq m \leq 60\%$ range, and then increases. Figure 5.8.b shows quality recall curve for *R-Ranked* at different values of n . It shows the same trend in quality recall, at lower recall values for lower n .

To see why recall is lowest in $40\% \leq m \leq 60\%$ range, we again compare the percentage of objects assigned mainly *strict* reviewers with the percentage of objects assigned mainly *lenient* reviewers. Table 5.12 shows that the percentage of objects with mainly *strict* reviewers and the percentage of objects with mainly *lenient* reviewers are inversely

Figure 5.9: Vary α : Quality Recall

related. When m is low, the former is high, but the latter is low. Since misclassification requires the confusion of a *low-quality* object with mainly *lenient* reviewers with a *high-quality* object with mainly *strict* reviewers, it follows that misclassification rate is highest (recall is lowest) when there is a sizeable number of *both* types of objects, which is when m is within the 40% to 60% range.

5.5.5 Varying Compensation Factor α

In previous sections, we set $\alpha = 0.5$. Here, we study the effect of the compensation factor α on quality recall. Figure 5.9 plots quality recall at various α . For *R-Ranked* and *A-Ranked*, quality recall increases as α approaches the value used in the data generation process ($\alpha = 1$). To achieve high performance, it is not necessary to use exactly the same α setting as used in data generation. For $\alpha \geq 0.4$, *R-Ranked* and *A-Ranked* achieve perfect recall. For $\alpha \geq 0.1$, *R-Ranked* and *A-Ranked* outperform *Naive* and *Riggs*.

5.6 Discussion

Summary

We propose the *Leniency-aware Quality* or *LQ* model to address the score summarization problem, by mining the leniency of reviewers from the rating scores and using it to adjust the quality scores of objects correspondingly. We show that the *LQ* model accomodates two compensation modes (*Relative* or *Absolute*), and generates two solution types (*Exact* or *Ranked*), giving rise to four possible *LQ* solutions (see Table 5.3). However, our experiments on both real-life and synthetic data show that these different solutions result in largely similar outcomes, which suggest that it is the underlying principle of using leniency to derive quality (and vice versa) that largely determines the result. The experiments on real-life data show that *LQ* results in a significant correction of *Naive*'s quality ranking (more so than *Riggs*). *LQ* also results in a finer differentiation of the very competitive objects at the top ranks. The experiments on synthetic data show that *LQ* consistently achieves a higher recall of the pre-determined ground truth than either *Naive* or *Riggs* under different parameter settings.

Related Work

Our score summarization research is quite related to the research on standardizing the score ranges of different reviewers. For instance, z-score normalization [WMMY02] may be used to center each reviewer's range at his/her own mean. The normalized scores could be re-calibrated to a common standard [Ark03]. For this to work, all reviewers must rate all objects, or at least a large subset of common objects. This condition is not easily satisfied, especially on the Web, where rating is usually voluntary and interest-driven.

Another approach to the score summarization problem is to weigh the contributions (e_{ij} scores) of reviewers differently. Existing weighting mechanisms focus on the notion of "reputation" [CS01, RW01]. Reputation-based models cannot solve the leniency problem, as even a reviewer skilled at discriminating the quality of objects may still be lenient, i.e., the reviewer may use a higher range of scores compared to other reviewers. Thus, our proposed model that is designed to factor the leniency of reviewers in collaborative rating networks is more applicable to the problem addressed in this chapter.

Chapter 6

Rating Dependencies in Collaborative Rating Networks

6.1 Overview

Multi-criteria rating of objects in social media has become very common as users often rate objects based on different rating criteria, derived from important object features. For example, to evaluate a digital camera on a product review site, a reviewer may give scores to different camera features (e.g., ease of use, battery life, memory size), in addition to an overall score. By observing the correlation between criterion-level scores and overall scores, or between scores of two different criteria, we can derive insights about the dependency behavior of reviewers and objects.

In this chapter, we seek to determine the *reviewer dependency* of every reviewer, and the *object dependency* of every object from the rating data.

- *Reviewer Dependency.* A reviewer is said to have high reviewer dependency between a pair of criteria when his or her ratings on objects based on the two criteria exhibit strong correlation. For example, a camera reviewer may value a camera's memory size so much so that s/he assigns high overall scores to cameras with high memory size scores, and low overall scores to cameras with low memory size scores. By

CHAPTER 6. RATING DEPENDENCIES IN COLLABORATIVE RATING NETWORKS

recognizing this dependency, we can decipher the reviewer's interest on the memory size criterion.

- *Object Dependency.* For an object, its object dependency represents the extent to which the object shows correlation in ratings between two criteria. For example, a well-written conference paper may introduce an idea that is so unique that its reviewers either like it due to the idea being novel (high novelty and overall scores), or dislike it as the idea does not look novel to them (low novelty and overall scores). In this case, we say that the conference paper has a high object dependency between novelty and overall scores.

Since reviewer dependency and object dependency are behavioral concepts that apply on a pair of criteria, they are inherently different from bias/controversy (see Chapter 4) and leniency/quality (see Chapter 5), which are defined for a single rating criterion.

A *Naive* approach to address this problem is to simply apply a standard correlation measure, such as *linear correlation* or *mutual information* (which we will review in Section 6.3). In this approach, a reviewer dependency is equated to the correlation between his/her scores on the first criterion and his/her scores on the second criterion.

However, *Naive* fails to recognize the relationship between reviewer dependency and object dependency. Highly correlated rating scores may be due to reviewer dependency or object dependency. We therefore propose an approach that attributes the correlation in scores to reviewer dependency, only if that correlation has been shown on objects with low object dependency. Since such objects are not expected to exhibit high correlation, we can be more confident that any correlation detected from the scores is likely due to reviewer dependency.

The rest of this chapter is organized as follows. Section 6.2 describes our proposed model. Section 6.3 discusses two correlation measures to be used in conjunction with the

Table 6.1: Table of Notations

<i>Notation</i>	<i>Description</i>
ea_{ij}	rating score by r_i on o_j for the first criterion
eb_{ij}	rating score by r_i on o_j for the second criterion
$\mathbf{ea}_{r_i}$	r_i 's vector of ea_{ij} scores on various o_j 's
$\mathbf{eb}_{r_i}$	r_i 's vector of eb_{ij} scores on various o_j 's
$\mathbf{ea}_{o_j}$	o_j 's vector of ea_{ij} scores by various r_i 's
$\mathbf{eb}_{o_j}$	o_j 's vector of eb_{ij} scores by various r_i 's
rd_i	r_i 's reviewer dependency between ea_{ij} and eb_{ij} scores
od_j	o_j 's object dependency between ea_{ij} and eb_{ij} scores

model. Section 6.4 describes the two solution types derivable from the model. Section 6.5 presents our experimental setup and the results. Section 6.6 concludes this chapter.

6.2 Rating Dependencies (RD) Model

The notations used in the following discussion are listed in Table 6.1. A reviewer r_i may assign to an object o_j two rating scores $ea_{ij}, eb_{ij} \in [0, 1]$ based on two different criteria ea and eb . Without any loss of generality, the overall score is treated as a criterion. We want to determine each r_i 's *reviewer dependency* between ea and eb denoted by $rd_i(ea, eb) \in [0, 1]$, and each o_j 's *object dependency* between ea and eb denoted by $od_j(ea, eb) \in [0, 1]$. When ea and eb are implicit, we may further simplify the reviewer dependency and object dependency notations as rd_i and od_j respectively.

Since reviewer dependency and object dependency are related quantities, we propose the *Rating Dependencies* or *RD* model based on the following mutual dependency principle.

- A reviewer has *high reviewer dependency* on a pair of rating criteria when his or her rating scores on objects based on the two criteria exhibit *high correlation*, and these rated objects exhibit *low object dependency* on the same two criteria.
- An object has *high object dependency* on a pair of rating criteria when the rating scores it receives from reviewers based on the two criteria exhibit *high correlation*,

CHAPTER 6. RATING DEPENDENCIES IN COLLABORATIVE RATING NETWORKS

and these reviewers exhibit *low reviewer dependency* on the same two criteria.

Our proposed *RD* model consists of a pair of equations: Eq. 6.1 to determine reviewer dependency rd_i and Eq. 6.2 to determine object dependency od_j . To determine rd_i with Eq. 6.1, we compute the correlation observed on r_i 's scores ($\mathcal{F}(\mathbf{ea}_{r_i}, \mathbf{eb}_{r_i})$), and reduce it by the average object dependency of r_i 's objects ($\mathcal{Avg}_j (1 - od_j)$). $\mathcal{F}$ is a correlation measure, such as *linear correlation* and *mutual information*, which we will review in Section 6.3. $\mathbf{ea}_{r_i}$ and $\mathbf{eb}_{r_i}$ are the vectors of r_i 's ea_{ij} and eb_{ij} scores across different o_j 's rated by r_i . $\mathcal{F}(\mathbf{ea}_{r_i}, \mathbf{eb}_{r_i})$ is higher when there is greater correlation between r_i 's ea_{ij} and eb_{ij} scores. $\mathcal{Avg}_j (1 - od_j)$ is higher when r_i 's reviewed objects generally have low dependency. Thus, a reviewer has high dependency rd_i if she assigns highly correlated scores on objects with low dependency od_j .

$$rd_i = \mathcal{F}(\mathbf{ea}_{r_i}, \mathbf{eb}_{r_i}) \times \mathcal{Avg}_j (1 - od_j) \quad (\text{Eq. 6.1})$$

$$od_j = \mathcal{F}(\mathbf{ea}_{o_j}, \mathbf{eb}_{o_j}) \times \mathcal{Avg}_i (1 - rd_i) \quad (\text{Eq. 6.2})$$

Object dependency is determined using Eq. 6.2, which is symmetrical to Eq. 6.1. An object has high dependency od_j if it exhibits highly correlated scores by reviewers with low dependency rd_i . In both Equations Eq. 6.1 and Eq. 6.2, we require $rd_i, od_j \in [0, 1]$ and $\mathcal{F}(\mathbf{ea}_{r_i}, \mathbf{eb}_{r_i}), \mathcal{F}(\mathbf{ea}_{o_j}, \mathbf{eb}_{o_j}) \in [0, 1]$.

The above equations are based on the principle of mutual dependency between rd_i and od_j . To determine rd_i , we need to know od_j (vice versa). Moreover, rd_i and od_j are inversely related, with rd_i higher when od_j is lower. The reviewer dependency of various reviewers are related by co-rating common objects, and the object dependency of various objects are related by having common reviewers. As reviewers and objects are inter-connected within the rating data, the mutual dependency extends to all reviewers and objects.

The *RD* model determines the dependency of individual r_i 's and o_j 's. However, taken in aggregate, these individual dependencies could also reveal more general criteria-level dependencies. For example, if the rd_i (or od_j) values are generally high for most reviewers (or objects), it may suggest a general dependency between the two criteria ea and eb .

Although we have modeled dependency between exactly two criteria ea and eb , the model could still apply to rating data with $k > 2$ criteria. For example, high co-dependency among a subset of three or more criteria may be determined by investigating whether there is always a high pairwise dependency between any and all criteria pairings.

Naive Model as a Special Case

A suitable baseline alternative to the *RD* model should be the basic model which does not incorporate the mutual dependency between rd_i and od_j . This baseline model, called the *Naive* model, consists of the pair of Eq. 6.3 to determine rd_i and Eq. 6.4 to determine od_j . *Naive*'s equations compute rd_i and od_j respectively using only the correlation measure $\mathcal{F}(\mathbf{ea}, \mathbf{eb})$. We further observe that *RD* is the more general model that would degenerate into *Naive* when all od_j 's (or rd_i 's) are uniform, in which case *RD*'s Eq. 6.1 (or Eq. 6.2) practically degenerates into *Naive*'s Eq. 6.3 (or Eq. 6.4).

$$rd_i = \mathcal{F}(\mathbf{ea}_{r_i}, \mathbf{eb}_{r_i}) \quad (\text{Eq. 6.3})$$

$$od_j = \mathcal{F}(\mathbf{ea}_{o_j}, \mathbf{eb}_{o_j}) \quad (\text{Eq. 6.4})$$

6.3 Correlation Measures

We review two established measures of correlation: *Linear Correlation* and (non-linear) *Mutual Information*, which can be used as the $\mathcal{F}$ function in the *RD* model. They represent the standard statistical and probabilistic approaches to measure correlation between two sets of scores.

Table 6.2: Comparative Scenarios: LC vs. MI

Case	$\mathbf{ea}$	$\mathbf{eb}$	$LC(\mathbf{ea}, \mathbf{eb})$	$MI(\mathbf{ea}, \mathbf{eb})$
1	[1, 1, 1, 1]	[1, 1, 1, 1]	0	0
2	[1, 0, 1, 0]	[1, 1, 0, 0]	0	0
3	[1, 1, 0, 0]	[1, 1, 0, 0]	1	1
4	[1, 1, 0, 0]	[0, 0, 1, 1]	1	1
5	[1, 1, 1, 0]	[1, 1, 1, 0]	1	0.8

6.3.1 Linear Correlation (LC)

Linear correlation measures the strength in which two variables are linearly related. The measure we use is a slightly modified version of the *Pearson product-moment correlation coefficient* [WMMY02]. For two vectors $\mathbf{ea}$ and $\mathbf{eb}$ of N real-valued elements each, the correlation coefficient $LC(\mathbf{ea}, \mathbf{eb}) \in [0, 1]$ is determined by Eq. 6.5. $\mathbf{ea}[n]$ is the n^{th} element of $\mathbf{ea}$, while $\mu_{\mathbf{ea}}$ and $\sigma_{\mathbf{ea}}$ are the mean and the standard deviation of $\mathbf{ea}$'s elements respectively. Similar notations apply to $\mathbf{eb}$. Higher $LC(\mathbf{ea}, \mathbf{eb})$ value indicates greater correlation between $\mathbf{ea}$ and $\mathbf{eb}$. This measure is also symmetric, i.e., $LC(\mathbf{ea}, \mathbf{eb}) = LC(\mathbf{eb}, \mathbf{ea})$.

$$LC(\mathbf{ea}, \mathbf{eb}) = \left| \frac{\sum_n^N (\mathbf{ea}[n] - \mu_{\mathbf{ea}})(\mathbf{eb}[n] - \mu_{\mathbf{eb}})}{(N - 1) \times \sigma_{\mathbf{ea}} \times \sigma_{\mathbf{eb}}} \right| \quad (\text{Eq. 6.5})$$

This measure is slightly different from the original *Pearson correlation*. While *Pearson correlation* ranges from -1 to 1, Eq. 6.5 ignores the sign, thus confining the range to 0 to 1. Secondly, *Pearson correlation* is undefined for $\sigma_{\mathbf{ea}} = 0$ or $\sigma_{\mathbf{eb}} = 0$. However, we choose to define $LC(\mathbf{ea}, \mathbf{eb}) = 0$ for these cases. $\sigma_{\mathbf{ea}} = 0$ or $\sigma_{\mathbf{eb}} = 0$ happens when either $\mathbf{ea}$ or $\mathbf{eb}$ has uniform elements, in which case the two vectors can be considered independent.

Consider the cases in Table 6.2. $LC(\mathbf{ea}, \mathbf{eb})$ is high when $\mathbf{ea}$ and $\mathbf{eb}$'s elements rise/fall together (Cases 3–5), and low when they do not vary (Case 1) or vary independently (Case 2).

6.3.2 Mutual Information (MI)

Mutual information could potentially detect non-linear relationship between two variables [DMM04]. Intuitively, it measures the degree to which two variables α and β share information, i.e., to what extent knowing the distribution of α would tell us about the distribution of β (vice versa). It compares the observed joint probability distribution of α and β to what the joint distribution would be if α and β are independent.

We employ a dichotomized version of the *mutual information* function on two vectors $\mathbf{ea}$ and $\mathbf{eb}$ (each containing N real-valued elements). For $\mathbf{ea}$ and $\mathbf{eb}$ respectively, we define two corresponding vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, whose elements are either 0 or 1 as defined by Equations Eq. 6.6 and Eq. 6.7 respectively. We then derive the probability distribution of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ by counting the frequency of 0's and 1's within each vector. For example, $P(\alpha = 1) = (\sum_n \boldsymbol{\alpha}[n]) \div N$. The joint probability distribution is similarly derived. For example, $P(\alpha = 1 \wedge \beta = 1) = (\sum_n \boldsymbol{\alpha}[n] \times \boldsymbol{\beta}[n]) \div N$, and $P(\alpha = 1 \wedge \beta = 0) = (\sum_n \boldsymbol{\alpha}[n] \times |\boldsymbol{\beta}[n] - 1|) \div N$, and so on.

$$\boldsymbol{\alpha}[n] = \begin{cases} 1 & \text{if } \mathbf{ea}[n] \geq \mu_{\mathbf{ea}} \\ 0 & \text{otherwise} \end{cases} \quad (\text{Eq. 6.6})$$

$$\boldsymbol{\beta}[n] = \begin{cases} 1 & \text{if } \mathbf{eb}[n] \geq \mu_{\mathbf{eb}} \\ 0 & \text{otherwise} \end{cases} \quad (\text{Eq. 6.7})$$

The mutual information $MI(\mathbf{ea}, \mathbf{eb}) \in [0, 1]$ is thus defined by Eq. 6.8. To avoid divide-by-zero cases, we define $P(\alpha \wedge \beta) \log_2 \frac{P(\alpha \wedge \beta)}{P(\alpha)P(\beta)} = 0$ for the cases where $P(\alpha) = 0$ or $P(\beta) = 0$. The MI measure is also symmetric, i.e., $MI(\mathbf{ea}, \mathbf{eb}) = MI(\mathbf{eb}, \mathbf{ea})$.

$$MI(\mathbf{ea}, \mathbf{eb}) = \sum_{\alpha \in \{0,1\}} \sum_{\beta \in \{0,1\}} P(\alpha \wedge \beta) \log_2 \frac{P(\alpha \wedge \beta)}{P(\alpha)P(\beta)} \quad (\text{Eq. 6.8})$$

The dichotomized MI is preferred to the original continuous function due to the small number of scores associated with most reviewers/objects in online rating systems. For

the data used in the experiments (see Section 6.5), we have an average of 11.4 scores per reviewer and 6.3 scores per object. The rating scale has 5 possible scores, resulting in 25 ($\alpha \wedge \beta$) combinations. With so few scores per reviewer/object and so many ($\alpha \wedge \beta$) combinations, we would not be able to obtain stable $P(\alpha \wedge \beta)$ values. Dichotomization reduces the number of ($\alpha \wedge \beta$) combinations to 4, resulting in more stable $P(\alpha \wedge \beta)$ values.

The behavior of MI has some similarity, but is not identical to, LC . As shown in Table 6.2, MI is similar to LC for (Cases 1–4) and is different for Case 5. One property of MI is that when both α and β are identical, MI would result in the entropy of α (or β). This entropy is lower when the probability distribution of α (or β) is skewed (e.g., many more 1's than 0's) as in Case 5. Hence, LC and MI do not necessarily produce identical outcomes, as they are based on different types of relationships (linear vs. non-linear).

6.4 Solution Types

Solving the *Naive* model is straightforward, given a correlation measure. On the other hand, the *RD* model also needs to consider the mutual dependency between reviewer dependency and object dependency, and the inter-connectivity among reviewers and objects. We identify two types of solution for the *RD* model: *Exact*, which produces exact values of reviewer dependency and object dependency, and *Ranked*, which produces rankings by reviewer dependency and by object dependency. The two solution types are independent and might not result in identical outcome. *Ranked* solution is still applicable as at times we are only interested in which reviewers (objects) are the most/least dependent.

Before we describe each solution type in more detail, we first develop the equations to be solved. We have an instance of Eq. 6.1 for each r_i and an instance of Eq. 6.2 for each o_j . This system of equations can be more compactly represented in terms of matrices, as given in Eq. 6.9 and Eq. 6.10. For m reviewers and n objects, R is $m \times 1$ vector of

rd_i values; O is $n \times 1$ vector of od_j values. U is $m \times n$ matrix, whose each element $u_{ij} = \mathcal{F}(\mathbf{ea}_{r_i}, \mathbf{eb}_{r_i}) \div n_i$. u_{ij} is the same across all j 's. n_i is the number of objects rated by r_i . V is $m \times n$ matrix, whose each element $v_{ij} = \mathcal{F}(\mathbf{ea}_{o_j}, \mathbf{eb}_{o_j}) \div m_j$. v_{ij} is the same across all i 's. m_j is the number of reviewers of o_j . $\mathbf{1}$ is a vector of appropriate length, whose elements are all 1's.

$$R = U (\mathbf{1} - O) \quad (\text{Eq. 6.9})$$

$$O = V^T (\mathbf{1} - R) \quad (\text{Eq. 6.10})$$

Substituting Eq. 6.10 into Eq. 6.9 results in the recursive equation in terms of R given in Eq. 6.11. A simpler form is given in Eq. 6.12, where X and Y are constants and R is the variable. Once solved, R could then be used to determine O with Eq. 6.10.

$$R = U\mathbf{1} - UV^T\mathbf{1} + UV^T R \quad (\text{Eq. 6.11})$$

$$R = X + YR \quad (\text{Eq. 6.12})$$

Exact Solution

The *Exact* solution is the unique value of R (and the corresponding O) satisfying Eq. 6.12. It is derived from solving Eq. 6.12 as a linear system of equations (in terms of various rd_i 's). The solution is to evaluate $R = (I - Y)^{-1}X$, where I is the identity matrix. However, a valid *Exact* solution exists if and only if $\det(I - Y) \neq 0$, in which case the system of equations has a unique solution.

Ranked Solution

The *Ranked* solution is the rankings of reviewers by rd_i and objects by od_j , which satisfy Eq. 6.13. This equation is modified from Eq. 6.12, by adding a non-zero, real-valued scalar variable λ . As $X + YR$ results in a rescaling of R (by λ), the relative ratio (or ranking) among R 's elements is preserved. Once λ is known, we can then rescale λR back to R .

$$\lambda R = X + YR \quad (\text{Eq. 6.13})$$

Factorizing R out from the right-hand side of Eq. 6.13 results in the eigenvector equation Eq. 6.14. X^m represents a matrix obtained by replicating the $m \times 1$ vector across m columns. γ is the inverse of the sum of elements of R , i.e., $\gamma = (\sum_i rd_i)^{-1}$. Hence, the R we are interested in is the principal eigenvector of $(\gamma X^m + Y)$.

$$\lambda R = (\gamma X^m + Y)R \quad (\text{Eq. 6.14})$$

The principal eigenvector of $(\gamma X^m + Y)$ can be derived using iterative methods [AR87, GVL96]. The iterative form of Eq. 6.14 is $R_{k+1} = (\gamma X^m + Y)R_k$. λ is removed by normalizing R after each iteration. In our case, normalization returns R to the state where the maximum element is always 1. Subject to the assumption that $(\gamma X^m + Y)$ has linearly-independent eigenvectors and a uniquely largest eigenvalue [GVL96], the iterations would converge to the principal eigenvector almost independently of the initial R_0 . The converged R and O form the *Ranked* solution.

In summary, we have described two ways to derive a solution for the *RD* model. *Exact* solution produces exact values, which can be used to produce rankings. *Ranked* solution produces only rankings. The two solution types are independent, and are based on slightly different equations (Eq. 6.12 for *Exact* and Eq. 6.13 for *Ranked*).

Table 6.3: Data Size

	<i>Reviewers</i>	<i>Objects</i>	<i>Overall Scores</i>	<i>Weight Scores</i>
<i>Original</i>	781	428	2288	2087
<i>Filtered</i>	102	184	1163	1163

6.5 Experiments

Our experimental objective is to verify the effectiveness of the *RD* model on a real-life data. We compare the reviewer dependency and object dependency ranks produced by *Naive* and *RD* (both *Exact* and *Ranked* solutions). Next, we highlight several case examples that shed more light on how factoring the mutual dependency principle makes the *RD* model more effective.

6.5.1 Data

Our data were obtained from *Epinions*, a product review Web site. We crawled the site over three days (April 20-23, 2007), beginning from a seed page¹. The collected data consisted of a subset of all products, reviewers, and scores.

Each product belongs to a category and each category has its specific rating criteria. Each reviewer may assign one or more scores based on these criteria. These scores, which ranged from 1 to 5, were normalized to a range from 0.2 to 1 by a simple division by 5. We chose to focus on the *Beers* category, as within the collected data, this category was relatively large and had more active reviewers and more actively-rated objects. There are four rating criteria for *Beers* category: *Overall*, *Weight*, *Flavor*, and *Complexity*. In the following experiments, we mainly focus on the (*Overall*, *Weight*) pairing. We have also separately carried out other experiments with (*Overall*, *Flavor*) and (*Overall*, *Complexity*) pairings with similar results. For the (*Overall*, *Weight*) pairing, the original data size collected is shown in the first row of Table 6.3.

¹http://www.epinions.com/rest-Restaurants-All/ViewAll_~1/Grp_~14529

We pre-processed the data as follows. First, we considered a reviewer as having rated an object only if the reviewer had assigned *both Overall* and *Weight* scores. Otherwise, the assigned score was ignored. What matters is the correspondence between the two rating criteria, and not the individual scores. Second, we ensured that each reviewer must have at least 3 objects and each object must have at least 3 reviewers, by iteratively removing reviewers and objects not meeting the condition until the condition was met. The final data size after filtering is shown in the second row of Table 6.3.

We ran *Naive*, as well as the two solution types (*Exact* and *Ranked*) of the *RD* model on this data. Subsequently, we refer to the respective solutions as *Naive*, *Exact*, and *Ranked*. For each solution, we try both *LC* and *MI* correlation measures.

6.5.2 Comparison of Ranked Lists

This part of the experiment compares the ranked lists produced by *Naive*, *Exact*, and *Ranked* solutions for the (*Overall*, *Weight*) pairing. We construct a ranked list of reviewers and a ranked list of objects for each solution. Reviewers and objects are ranked in decreasing order of rd_i and od_j respectively. The highest rd_i or od_j value is given rank 1. Same values share the same rank.

Naive vs. Exact

First, we compare the reviewer dependency ranks assigned by *Exact* and *Naive*. Figures 6.1.a and 6.1.b are scatterplots of reviewer dependency ranks based on the respective measures of *LC* and *MI*. A point in each scatterplot represents a reviewer, with the x -value being the rank assigned by *Exact* and the y -value the rank assigned by *Naive*. From these two figures, we make the following observations.

- The reviewer dependency ranks assigned by *Exact* and *Naive* are positively correlated in general, but *are not identical*. The positive correlation is expected as

CHAPTER 6. RATING DEPENDENCIES IN COLLABORATIVE RATING NETWORKS

Figure 6.1: *Naive* vs. *Exact*: Reviewer Dependency

both are based on the same $\mathcal{F}$ function (LC or MI). However, *Exact* also takes into account object dependency, resulting in a different ranking. Figure 6.1.a and Figure 6.1.b show large variances around the diagonal, indicating that *Exact* and *Naive*'s rankings are quite different. For example, there are instances where a significant number of reviewers are tied according to *Naive*, but are differentiated by *Exact*. In Figure 6.1.a, 6 reviewers share rank 1 by *Naive*, but are given ranks ranging from 1 to 22 by *Exact*. In Figure 6.1.b, 12 reviewers share rank 5 by *Naive*, but are given ranks ranging from 1 to 17 by *Exact*.

- There are more rank differences among the top ranks than among the bottom ranks. In Figure 6.1.a and Figure 6.1.b, the variances are largest in the top ranks and generally decrease as we move to the lower ranks. This suggests that if a reviewer has low correlation in the first place, taking into account the object dependency of rated objects would not have much effect (see Eq. 6.1). This applies especially to the lowest ranks, where the LC or MI value is likely zero (or a very small value).
- The ranks computed using LC and MI are not identical as shown by the different patterns on Figure 6.1.a and Figure 6.1.b. This is not unexpected, given that LC

Figure 6.2: *Naive* vs. *Exact*: Object Dependency

and *MI* capture different forms of relationships as mentioned in Section 6.3. More importantly, for both *LC* and *MI*, the *RD* model's taking into account the mutual dependency principle results in significant differences from the *Naive* model.

Figure 6.2.a and Figure 6.2.b are the scatterplots for object dependency ranks. As they show much resemblance to the earlier Figure 6.1.a and Figure 6.1.b, similar observations as for reviewer dependency ranks can also be made here. This itself is an important observation, for it highlights the mutual dependency principle. If *Exact*'s object dependency ranks were identical to *Naive*'s, so would the reviewer dependency ranks be. Instead, the variance in reviewer dependency ranks results from the variance in object dependency ranks (and vice versa).

Ranked vs. Exact

Next, we compare the dependency ranks assigned to reviewers by *Exact* and *Ranked*. Figure 6.3.a and Figure 6.3.b are the corresponding scatterplots of reviewer dependency ranks. In either scatterplot, there is much less variance around the diagonal, which

Figure 6.3: *Ranked* vs. *Exact*: Reviewer Dependency

implies that *Exact* and *Ranked* produce very similar rankings. Due to the mutual dependency principle, the two solutions also produce very similar object dependency rankings. The scatterplots for object dependency are not shown here, as they are very similar to Figures 6.3.a and 6.3.b. The comparison of *Naive* vs. *Ranked* largely follows that of *Naive* vs. *Exact*, and thus is not shown here for space consideration.

6.5.3 Case Examples

We show three examples to illustrate the effectiveness of *RD*. The first two concern an object and a reviewer, for whom *RD* and *Naive* disagree on the (*Overall*, *Weight*) dependency values and ranks. The last example shows how by inspecting the results for the different criteria pairings $\{(\textit{Overall}, \textit{Weight}), (\textit{Overall}, \textit{Flavor}), (\textit{Overall}, \textit{Complexity})\}$, we can identify the dominant criteria for a reviewer. We use only the *Exact* solution to represent *RD*, as we have noted the similarity between *Exact* and *Ranked* solutions in the previous section.

Table 6.4: Profile of Object *boulevard_dry_stout*

Object			Object Dependency (Rank)			
			with <i>LC</i>		with <i>MI</i>	
			<i>Naive</i>	<i>Exact</i>	<i>Naive</i>	<i>Exact</i>
boulevard_dry_stout			1.00 (1)	0.33 (34)	0.92 (5)	0.33 (20)
Reviewers	Score <i>Overall</i> <i>Weight</i>		Reviewer Dependency (Rank)			
			with <i>LC</i>		with <i>MI</i>	
			<i>Naive</i>	<i>Exact</i>	<i>Naive</i>	<i>Exact</i>
lafeet	1.0	1.0	0.83 (25)	0.69 (19)	0.47 (18)	0.41 (19)
wingdman	1.0	1.0	0.91 (12)	0.71 (17)	1.00 (1)	0.78 (13)
impydykiechick	0.8	0.8	0.84 (23)	0.60 (34)	0.92 (5)	0.73 (15)

Table 6.5: Objects reviewed by *wingdman*

Objects	Score	
	<i>Overall</i>	<i>Weight</i>
beamish_irish_stout	1.0	1.0
boulevard_bully__porter	1.0	1.0
boulevard_dry_stout	1.0	1.0
dixie_blackened_voodoo_lager	1.0	1.0
sierra_nevada_stout	1.0	1.0
grolsch_premium_lager	0.8	0.6
hacker-pschorr_weisse	0.6	0.6
murphy_s_irish_amber	0.6	0.6
samuel_adams_summer_ale	0.6	0.2
tequiza_beer	0.6	0.2

Case Example I

Object *boulevard_dry_stout*'s profile is given in Table 6.4. For *LC* and *MI* respectively, *Naive* assigns this object high object dependency of 1.00 (rank 1) and 0.92 (rank 5). *Exact* assigns it much lower object dependency of 0.33 (rank 34) and 0.33 (rank 20) respectively. Its rating scores alone indeed suggest high object dependency. However, Table 6.4 also shows that *boulevard_dry_stout*'s three reviewers (*lafeet*, *wingdman*, *impydykiechick*) all have high reviewer dependency. For instance, *wingdman* has high reviewer dependency of 0.71 (rank 17) for *LC* and 0.78 (rank 13) for *MI* respectively.

We take a more detailed look at *wingdman*'s rating scores in Table 6.5. Visual inspection reveals a trend whereby *wingdman*'s *Overall* score is high (or low) when his *Weight* score is high (or low). Given the high reviewer dependency of its reviewers, it is expected

Table 6.6: Profile of Reviewer *adamldemarco*

Reviewer			Reviewer Dependency (Rank)			
			with <i>LC</i>		with <i>MI</i>	
			<i>Naive</i>	<i>Exact</i>	<i>Naive</i>	<i>Exact</i>
adamldemarco			0.87 (16)	0.79 (8)	0.92 (5)	0.91 (2)
Objects	Score <i>Overall</i> <i>Weight</i>		Object Dependency (Rank)			
			with <i>LC</i>		with <i>MI</i>	
			<i>Naive</i>	<i>Exact</i>	<i>Naive</i>	<i>Exact</i>
orval_trappist_ale	0.8	0.8	0.29 (94)	0.12 (105)	0.00 (134)	0.00 (134)
hoegaarden_original_ white_beer	0.8	0.6	0.03 (134)	0.02 (134)	0.02 (112)	0.02 (113)
george_killians_irish_ red	0.6	0.4	0.32 (87)	0.13 (99)	0.00 (134)	0.00 (134)

Table 6.7: Reviewers rating *hoegaarden_original_ white_beer*

Reviewers	Score	
	<i>Overall</i>	<i>Weight</i>
andaryl	1.0	0.8
fuche_bu	1.0	0.8
average_joe	1.0	0.6
mrkstvns	1.0	0.6
latakiahaze	1.0	0.4
beerbuff	1.0	0.2
chuckv	1.0	0.2
adamldemarco	0.8	0.6
proxam	0.8	0.6
xpander007	0.8	0.6
tsomes	0.8	0.2

that *boulevard_dry_stout* would show high correlation between its *Overall* and *Weight* scores. However, this correlation should be attributed more to its reviewers. Thus, *RD* justifiably assigns lower object dependency (and ranks) to *boulevard_dry_stout*.

Case Example II

Reviewer *adamldemarco*'s profile is given in Table 6.6. For *LC* and *MI* respectively, *Naive* assigns reviewer dependency ranks of 16 and 5, and *Exact* assigns higher ranks of 8 and 2. *Naive*'s reviewer dependency values are actually higher, but in relative terms (ranking) *adamldemarco*'s reviewer dependency has been promoted by *Exact* because the objects have very low object dependency ranks. In particular, *hoegaarden_original_ white_beer* is ranked 134 for *LC* and 113 for *MI* by *Exact* (out of 184 objects).

Table 6.8: Profile of Reviewer *loraxc*

Reviewer	Reviewer Dependency (Rank) – Exact with LC		
	(Overall, Weight)	(Overall, Flavor)	(Overall, Complexity)
loraxc	0.97 (1)	0.86 (4)	0.00 (84)

Reviewer	Score			
	Overall	Weight	Flavor	Complexity
anchor_steam_beer	0.8	0.6	0.2	0.2
bert_grant_s_imperial_stout	1.0	1.0	1.0	0.2
samuel_smith_s_oatmeal_stout	1.0	1.0	1.0	0.2

hoegaarden_original_white_beer's scores by different reviewers are shown in Table 6.7. Evidently, this object's *Overall* scores are not affected by its *Weight* scores. Since such objects are not expected to exhibit high correlation, any correlation is likely due to the reviewer. *Exact* gives a higher reviewer dependency rank to *adamldemarco*, to reflect this higher likelihood of *adamldemarco*'s dependency.

Case Example III

We now turn to the comparison among different criteria pairings. In Table 6.8, we show the profile of reviewer *loraxc*, which has a very low reviewer dependency of 0 (rank 84) for (*Overall*, *Complexity*) and very high reviewer dependency of 0.97 (rank 1) and 0.86 (rank 4) for (*Overall*, *Weight*) and (*Overall*, *Flavor*) respectively. These ranks are based on *Exact* solution with *LC* measure. *loraxc*'s reviewer dependency ranks suggest that the dominant criteria affecting *loraxc*'s scores are *Weight* and *Flavor*, while *Complexity* does not have much significance. This is also evident from *loraxc*'s scores on different objects, with uniform scores for *Complexity*, while the *Weight* and *Flavor* scores rise and fall with the *Overall* scores. It is unlikely that three different objects deserve identical *Complexity* scores. It is more likely that *loraxc* simply assigns a default score for *Complexity* and uses the other criteria to arrive at the *Overall* scores. From that we can infer that *loraxc* considers mainly *Weight* and *Flavor* to arrive at the *Overall* score.

6.6 Discussion

Summary

In this chapter, we address the problem of determining reviewer dependency and object dependency with respect to two rating criteria. The key principle in our *RD* model is modeling the mutual dependency between reviewer dependency and object dependency. Experiments on real-life data show that there are significant differences between *Naive* and *RD* models due to *RD*'s taking into account the mutual dependency principle. *RD* is also more effective, able to differentiate reviewers and objects considered equivalent by *Naive*. Further inspection of specific case examples yields the same conclusion. Although our modeling is based on two given rating criteria, without loss of generality, it can be applied in a pairwise manner for multiple criteria, for instance in learning of the dominant rating criteria affecting the overall scores.

Related Work

Prior work on rating dependency has mainly focused on *systematic* dependency, using correlation analysis [CCWA03] to test the dependency between a given factor and the assigned scores. For example, [GG97] studies whether students tend to assign higher scores to instructors who have given them higher grades; [FGHH06] shows that venture capitalists' evaluation of start-up teams is dependent on their similarity to these teams. Systematic dependency affects all reviewers and objects in general, and thus cannot tell us the dependency of an individual reviewer or object. Our proposed model measures the *specific* dependency of individual reviewers and objects, as reviewers and objects may exhibit varying dependency with respect to two rating criteria.

Chapter 7

Conclusion

This dissertation concerns the discovery and analysis of social networks based on user activities online. Online activities often mirror offline ones; essentially the Internet and mobile technologies allow people to do things online that they use to do offline. The main difference is that online activities often leave electronic trail (i.e., data can be collected) and may potentially involve a much more massive number of people than their offline equivalents (e.g., MMORPG ¹ games, online product reviews, online auctions). In this dissertation, we identify and focus our research on two types of online data that do not readily feed into existing social network discovery and analysis techniques, namely: *spatio-temporal data* and *collaborative rating data*.

In Chapter 2, we give an overview of social network research, which can be broadly classified into discovery, analysis, and application. *Discovery* deals with how to construct a social network from some data, based on some criterion. The four most commonly used criteria are self-reported, communication, similarity, and co-occurrence. Our interest in discovery concerns social network discovery from spatio-temporal data based on co-occurrences in space and time. *Analysis* deals with how to analyze a social network for useful insights about actors, paths, and subgroups. Our interest in analysis concerns

¹Massively Multi-player Online Role Playing Games

CHAPTER 7. CONCLUSION

mining rating-related behaviors of actors (which can be reviewers or objects). *Application* deals with using the social networks and related insights in different types of applications.

In Chapter 3, we address the problem of social network discovery from spatio-temporal data. We adopt spatio-temporal co-occurrence as a basis to infer social associations among users, and develop the *STEvent* model that mines events from spatio-temporal data and infers links from users' common participation in events. We describe a two-phase algorithm to implement the *STEvent* model, which we show theoretically and empirically to be efficient, with a time complexity that is approximately linear to the data size. The effectiveness of this model is tested on two sets of real-life data: *Cyber Location Data* (a log of Web pages accessed by users at different times) and *Physical Location Data* (a log of physical locations from where users connect to the campus wireless network over time). The experiments show that the top links discovered by *STEvent* demonstrate a relatively high degree of similarity, which is a well-recognized predictor of social association [Fel81, Car91]. Moreover, the results for *Physical Location Data* are significantly better than for *Cyber Location Data*, hinting that co-occurrences in physical locations may be a better predictor of social associations among users with similar demographic attributes. On the other hand, wide deployment of the *STEvent* framework may require first resolving privacy issues that may surface, as some individuals may be concerned about his/her movement data being mined for associations. It is also likely that only large institutions (governments, universities, mobile phone companies) would have access to such data and apply this research work.

In Chapter 4, we investigate one set of behaviors from collaborative rating data related to score deviations, namely: reviewer's bias and object's controversy. Score deviation, whereby a reviewer differs from co-reviewers when assigning rating score to an object, may be attributed to the bias of the reviewer or to the controversy of the object. We observe that bias and controversy are mutually dependent in an inversely reinforcing

CHAPTER 7. CONCLUSION

manner. A reviewer has high bias if s/he exhibits high score deviations on objects with low controversy (and vice versa). Thus, we propose a framework consisting of the *Inverse Reinforcement* or *IR* model to determine bias and controversy, and the *Evidence* model to measure the degree of confidence with which each bias/controversy outcome is derived. We show how these proposed models may be formulated into eigenvector equations and implemented efficiently using iterative methods. We discuss the convergence properties of the iterative method implementation, and empirically observe that such convergence is usually quite rapid (generally in a matter of seconds in our experiments). Extensive experiments on real-life and synthetic data verify that the proposed model is more effective than the comparative *Naive* model, which ignores the mutual dependency between bias and controversy. The main limitation to this approach is it is currently a look-back approach, detecting bias and controversy which have already happened in order to do a better evaluation analysis. We have made no attempt to investigate the possible causes of bias and controversy, which would have helped in preventing them in the future.

In Chapter 5, we investigate how the leniency of reviewers (as shown by score inflations or deflations by reviewers) may affect the quality of objects (as aggregated from rating scores assigned by reviewers). Leniency refers to the tendency of a reviewer to assign higher scores than the rated object may deserve (as determined by the object's quality). We observe that leniency can only be determined in the context of the network (instead of in isolation for each reviewer), and that knowing leniency requires knowing quality (and vice versa). We therefore propose the *Leniency-aware Quality* or *LQ* model, which mines the leniency of reviewers and uses it to adjust the quality scores simultaneously. Experiments show that the *LQ* model is more effective than the comparative *Naive* (which relies on simple averaging) and *Riggs* (which relies on weighted averaging) models. On real-life data, *LQ* results in a more significant correction of *Naive* than *Riggs* does. On synthetic data, *LQ* achieves consistently higher quality recall than either *Naive* or *Riggs*.

CHAPTER 7. CONCLUSION

However, it also needs to be noted that our approach to leniency and quality works best for the rating scenarios where reviewers generally make few ratings and objects receive few ratings. Given a large enough number of ratings per object, classical approaches such as simple averaging would work as well with lower time complexity.

In Chapter 6, we study reviewer dependency and object dependency behaviors by analyzing score correlations in rating data. A reviewer is said to have high reviewer dependency if s/he exhibits high score correlations on a pair of rating criteria, provided that the score correlations have been exhibited on objects with low object dependencies between the same pair of criteria (vice versa). We develop the *Rating Dependencies* or *RD* model that factors this relationship between reviewer dependency and object dependency, and describes two correlation measures that can be used in conjunction with the model, namely linear correlation (*LC*) and mutual information (*MI*). While *LC* and *MI* measures linear and non-linear correlations respectively, our experiments show that using the *RD* model with either measure yields largely similar outcomes. The experiments on real-life data show that the *RD* model when used with either measure is more effective than *Naive*, which ignores the mutual dependency between reviewer dependency and object dependency.

While our dissertation has mainly focused on online activities, many of the techniques that we propose here can be applied in offline contexts as well. For example, offline rating activities such as sports judging or educational testing may benefit from similar analysis of rating-related behaviors.

7.1 Future Work

The work in this dissertation constitutes pioneering research on social network discovery and analysis from online activities, especially concerning spatio-temporal data and col-

CHAPTER 7. CONCLUSION

laborative rating data. As such, there is a wide scope for further research. Below, we identify several specific future research directions in this area.

Integration of Various Types of Social Networks

Diverse types of social associations may be mined from heterogenous data such as emails, co-citation, or co-authorship. We have focused our attention on two: spatio-temporal and collaborative rating data. One interesting direction is to integrate the analysis of social networks discovered from these heterogeneous data in order to discover richer forms of associations. For instance, the study of conflict of interest may benefit from such an integrated approach. Suppose that we have a rating network of academicians rating papers, and another network of friendship/co-authorship among academicians. A conflict of interest may exist when an academicians rates a paper by a co-author or a friend. By comparing an academician's rating scores on papers with conflict of interest versus papers without, we can derive insights on whether the academician tends to act (i.e., assign higher scores to papers by friends) or not act (i.e., assign scores objectively) on conflicts of interest.

Application of Rating-related Behaviors

Integrating the analysis results into specific applications will help to enhance the effectiveness of the applications. For example, recommender system allows a user to rate objects (e.g., movies), and recommends other highly-rated objects that the system believes the user may like. Awareness of rating-related behaviors can enhance the recommendations. The identification of high-quality recommendable objects will benefit from taking into account the leniency of reviewers. Given the same overall score, a controversial object is less recommendable than a non-controversial one that attracts consensus among its reviewers. Reviewer dependency helps to reveal the preferences of a user; a user interested

CHAPTER 7. CONCLUSION

in cameras, who shows high rating dependency between the memory criterion and the overall score criterion, should be recommended cameras with high memory scores.

Incorporating Non-rating Information for Mining Rating-related Behaviors

We have not taken into account other types of information than rating data that may be available in specific applications. Online trading communities may retain some information about a user's past transactions. Analysis of peer review networks may also benefit from additional information on academicians, such as past topics, co-authors, etc. Integrating rating and non-rating data in the analysis may help to answer such interesting questions as whether a reviewer is biased or dependent only on some (and not all) classes of objects (e.g., products of certain categories, papers of certain topics), or whether there is a link between leniency and some demographic attribute like age or experience.

Appendix A

List of Publications

The work discussed in this dissertation has resulted in 6 international publications. We list these and other publications of the author during his Ph.D. candidature below.

Conference and Workshop Publications

- (a) Hady W. Lauw, Ee-Peng Lim, Teck-Tim Tan, and HweeHwa Pang, “Mining Social Network from Spatio-Temporal Events”, *Workshop on Link Analysis, Counterterrorism and Security in conjunction with SIAM Data Mining Conference (SDM2005)*, Newport Beach, California, April 2005.
- (b) Hady W. Lauw, Ee-Peng Lim, and Ke Wang, “Bias and Controversy: Beyond the Statistical Deviation”, *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD2006)*, Philadelphia, Pennsylvania, August 2006.
- (c) Hady W. Lauw, Ee-Peng Lim, and Ke Wang, “Summarizing Review Scores of Unequal Reviewers”, *SIAM International Conference on Data Mining (SDM2007)*, Minneapolis, Minnesota, April 2007.
- (d) Hady W. Lauw and Ee-Peng Lim, “A Multitude of Opinions: Mining Online Rating Data”, *NSF Symposium on Next-Generation Data Mining & Cyber-Enabled Discovery for Innovation (NGDM2007)*, Baltimore, Maryland, October 2007.

CHAPTER A. LIST OF PUBLICATIONS

Journal Publications

- (a) Hady W. Lauw, Ee-Peng Lim, HweeHwa Pang, and Teck-Tim Tan, “Social Network Discovery by Mining Spatio-Temporal Events”, *Computational & Mathematical Organization Theory*, Vol. 11, No. 2, 2005, pp. 97-118.
- (b) Hady W. Lauw, Ee-Peng Lim, and Ke Wang, “Bias and Controversy in Evaluation Systems”, *IEEE Transactions on Knowledge and Data Engineering (TKDE)*. To appear, 2008.

Other Publications

- (a) Ee-Peng Lim, Ba-Quy Vuong, Hady W. Lauw, and Aixin Sun, “Measuring Qualities of Articles Contributed by Online Communities”, **IEEE/WIC/ACM International Conference on Web Intelligence (WI2006)**, Hong Kong, December 2006.
- (b) Hady W. Lauw, Ee-Peng Lim, and HweeHwa Pang, “TUBE (Text-cUBE) for Discovering Documentary Evidence of Associations among Entities”, **ACM Symposium on Applied Computing (SAC2007)**, Seoul, Korea, March 2007.
- (c) Meiqun Hu, Ee-Peng Lim, Aixin Sun, Hady W. Lauw, and Ba-Quy Vuong, “Measuring Article Quality in Wikipedia: Models and Evaluation”, **ACM Conference on Information and Knowledge Management (CIKM2007)**, Lisboa, Portugal, November 2007.
- (d) Meiqun Hu, Ee-Peng Lim, Aixin Sun, Hady W. Lauw, and Ba-Quy Vuong, “On Improving Wikipedia Search Using Article Quality”, **ACM International Workshop on Web Information and Data Management (WIDM2007)**, Lisboa, Portugal, November 2007.

CHAPTER A. LIST OF PUBLICATIONS

- (e) Ba-Quy Vuong, Ee-Peng Lim, Aixin Sun, Minh-Tam Le, Hady W. Lauw, and Kuiyu Chang, “On Ranking Controversies in Wikipedia: Models and Evaluation”, **ACM International Conference on Web Search and Data Mining (WSDM2008)**, Stanford, California, February 2008.
- (f) Young Ae Kim, Minh-Tam Le, Hady W. Lauw, Ee-Peng Lim, Haifeng Liu, and Jaideep Srivastava, “Building a Web of Trust without Explicit Trust Ratings”, **Workshop on Data Engineering for Blogs, Social Media, and Web 2.0 in conjunction with IEEE International Conference on Data Engineering (ICDE2008)**, Cancun, Mexico, April 2008.
- (g) Haifeng Liu, Ee-Peng Lim, Hady W. Lauw, Minh-Tam Le, Aixin Sun, Jaideep Srivastava, and Young Ae Kim, “Predicting Trusts among Users of Online Communities - An Epinions Case Stud”, **ACM Conference on Electronic Commerce (EC2008)**, Chicago, Illinois, July 2008.

References

- [AA03] L. A. Adamic and E. Adar. Friends and neighbors on the Web. *Social Networks*, 25(3):211–230, July 2003.
- [AAR96] A. Arning, R. Agrawal, and P. Raghavan. A linear method for deviation detection in large databases. In *Proceedings of the 2nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 164–169, New York, 1996. ACM.
- [AIS93] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Proceedings of the ACM International Conference on Management of Data*, pages 207–216, Washington, USA, 1993.
- [AR87] H. Anton and C. Rorres. *Elementary Linear Algebra with Applications*. John Wiley & Sons, Inc., 1987.
- [Ark03] H. R. Arkes. The nonuse of psychological research at two federal agencies. *Psychological Science*, 14(1):1–6, 2003.
- [ARSX03] R. Agrawal, S. Rajagopalan, R. Srikant, and Y. Xu. Mining newsgroups using networks arising from social behavior. In *Proceedings of the 12th International World Wide Web Conference*, pages 688–703, Budapest, Hungary, 2003.
- [AS94] R. Agrawal and R. Srikant. Fast algorithm for mining association rules. In *Proceedings of the 20th International Conference on Very Large Databases*, pages 487–499, Santiago, Chile, 1994.
- [AT05] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions.

REFERENCES

- IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, 2005.
- [AT06] M. A. Ahmad and A. Teredesai. Modeling spread of ideas in online social networks. In *Proceedings of the 5th Australasian Conference on Data mining and Analytics*, pages 185–190, Darlinghurst, Australia, 2006. Australian Computer Society, Inc.
- [Baz90] M. H. Bazerman. *Judgment in Managerial Decision-Making*. Wiley, 2nd edition, 1990.
- [BCMS04] C. Borgs, J. Chayes, M. Mahdian, and A. Saberi. Exploring the community structure of newsgroups. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 783–787, New York, NY, USA, 2004. ACM.
- [BH83] J. A. Barnes and F. Harary. Graph theory in network analysis. *Social Networks*, 5:235–244, 1983.
- [BH06] J. L. Blackburn and M. D. Hakel. An examination of sources of peer-review bias. *Psychological Science*, 17(5):378–382, 2006.
- [BHKL06] L. Backstrom, D. Huttenlocher, J. Kleinberg, and X. Lan. Group formation in large social networks: Membership, growth, and evolution. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 44–54, New York, NY, USA, 2006. ACM.
- [BL96] L. Busenitz and C. Lau. A cross-cultural cognitive model of new venture creation. *Entrepreneurship: Theory and Practice*, 20(4):25–39, 1996.
- [Bon72] P. Bonacich. Technique for analyzing overlapping memberships. *Sociological Methodology*, 4:176–185, 1972.
- [Bon91] P. Bonacich. Simultaneous group and individual centralities. *Social Networks*, 13(2):155–168, 1991.
- [Boy04] D. M. Boyd. Friendster and publicly articulated social networking. In *Extended abstracts of the Conference on Human Factors and Computing Systems*, pages 1279–1282, Vienna, Austria, 2004.

REFERENCES

- [BRRT05] A. Borodin, G. O. Roberts, J. S. Rosenthal, and P. Tsaparas. Link analysis ranking: Algorithms, theory, and experiments. *ACM Transactions on Internet Technology*, 5(1):231–297, 2005.
- [Car91] K. Carley. A theory of group stability. *American Sociological Review*, 56(3):331–354, 1991.
- [CC99] C. Chen and L. Carr. Trailblazing the literature of hypertext: Author co-citation analysis (1989-1998). In *Proceedings of the 10th ACM Conference on Hypertext and Hypermedia*, pages 51–60, New York, NY, USA, 1999. ACM.
- [CCWA03] J. Cohen, P. Cohen, S. G. West, and L. S. Aiken. *Applied Multiple Regression/Correlation Analysis for the Behavioral Sciences*. Lawrence Erlbaum Associates, Inc., Hillsdale, New Jersey, 3rd edition, 2003.
- [CF06] D. Chakrabarti and C. Faloutsos. Graph mining: Laws, generators, and algorithms. *ACM Computing Surveys*, 38(1):2, 2006.
- [CH00] D. J. Cook and L. B. Holder. Graph-based data mining. *IEEE Intelligent Systems*, 15(2):32–41, March 2000.
- [CKY05] A. Chapanond, M. S. Krishnamoorthy, and B. Yener. Graph theoretic and spectral analysis of Enron email data. In *Proceedings of the Workshop on Link Analysis, Counterterrorism, and Security (in conj. with SIAM International Conference on Data Mining)*, pages 15–22, Newport Beach, CA, USA, 2005.
- [CP03] T. Choudhury and A. Pentland. Sensing and modeling human networks using the Sociometer. In *Proceedings of the 7th IEEE International Symposium on Wearable Computing*, pages 216–222, Washington, DC, 2003. IEEE Computer Society.
- [CS01] M. Chen and J. P. Singh. Computing and using reputations for internet ratings. In *Proceedings of the 3rd ACM Conference on Electronic Commerce*, pages 154–162, New York, 2001. ACM.

REFERENCES

- [Cul86] M.J. Culnan. The intellectual development of management information systems, 1972-1982: A co-citation analysis. *Management Science*, 32(2):156–172, 1986.
- [Dav68] J. A. Davis. Statistical analysis of pair relationships. *Sociometry*, 31(1):102–119, 1968.
- [DC05] J. Diesner and K. M. Carley. Exploration of communication networks from the Enron email corpus. In *Proceedings of the Workshop on Link Analysis, Counterterrorism, and Security (in conj. with SIAM International Conference on Data Mining)*, pages 3–4, Newport Beach, CA, USA, 2005.
- [DKNS01] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the Web. In *Proceedings of the 10th International Conference on World Wide Web*, pages 613–622, New York, 2001. ACM.
- [DMM04] A. Dionisio, R. Menezes, and D. A. Mendes. Mutual information: A measure of dependency for nonlinear time series. *Physica A*, 344:326–329, 2004.
- [DN92] S. Dumais and J. Nielsen. Automating the assignment of submitted manuscripts to reviewers. In *Proceedings of the 15th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 233–244, 1992.
- [DR01] P. Domingos and M. Richardson. Mining the network value of customers. In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 57–66, San Francisco, CA, USA, 2001.
- [EP06] N. Eagle and A. Pentland. Reality Mining: Sensing complex social systems. *Personal and Ubiquitous Computing*, 10(4):255–268, 2006.
- [Fel81] S. L. Feld. The focused organization of social ties. *American Journal of Sociology*, 86:1015–1035, 1981.
- [FGHH06] N. Franke, M. Gruber, D. Harhoff, and J. Henkel. What you are is what you like – similarity biases in venture capitalists’ evaluations of start-up teams. *Journal of Business Venturing*, 21(6):802–826, 2006.

REFERENCES

- [FKS03] R. Fagin, R. Kumar, and D. Sivakumar. Comparing top k lists. *SIAM Journal on Discrete Mathematics*, 17(1):134–160, 2003.
- [FLG00] G. W. Flake, S. Lawrence, and C. L. Giles. Efficient identification of web communities. In *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 150–160, New York, NY, USA, 2000. ACM.
- [FLGC02] G. W. Flake, S. Lawrence, C. L. Giles, and F. M. Coetzee. Self-organization and identification of Web communities. *Computer*, 35(3):66–71, 2002.
- [FMT04] C. Faloutsos, K. S. McCurley, and A. Tomkins. Fast discovery of connection subgraphs. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 118–127, Seattle, WA, USA, 2004.
- [Fre92] L. C. Freeman. The sociological concept of “group”: An empirical test of two models. *American Journal of Sociology*, 98:152–166, 1992.
- [Fre96] L. C. Freeman. Cliques, galois lattices, and the structure of human social groups. *Social Networks*, 18(3):173–187, 1996.
- [GG97] A. G. Greenwald and G. M. Gillmore. Grading leniency is a removable contaminant of student ratings. *American Psychologist*, 52(11):1209–1217, 1997.
- [GH06] J. Golbeck and J. Hendler. Inferring binary trust relationships in Web-based social networks. *ACM Transactions on Internet Technology*, 6(4):497–529, 2006.
- [GHW97] L. Garton, C. Haythornthwaite, and B. Wellman. Studying online social networks. *Journal of Computer-Mediated Communication*, 3(1), June 1997.
- [GKRT04] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins. Propagation of trust and distrust. In *Proceedings of the 13th International Conference on World Wide Web*, pages 403–412, New York, NY, USA, 2004. ACM.
- [GLM01] J. Goldenberg, B. Libai, and E. Muller. Talk of the network: A complex systems look at the underlying process of word-of-mouth. *Marketing Letters*, 12(3):211–223, 2001.

REFERENCES

- [Gra87] M. Granovetter. Threshold models of collective behavior. *American Journal of Sociology*, 83(6):1420–1443, 1987.
- [GS82] G. R. Grimmett and D. R. Stirzaker. *Probability and Random Processes*. Oxford University Press, 1982.
- [GS97] J. Geller and R. Scherl. Challenge: Technology for automated reviewer selection. In *Proceedings of the International Joint Conferences on Artificial Intelligence*, pages 55–61, 1997.
- [GVL96] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 3rd edition, 1996.
- [Hav03] T. H. Haveliwala. Topic-sensitive PageRank: A context-sensitive ranking algorithm for Web search. *IEEE Transactions on Knowledge and Data Engineering*, 15(4):784–796, 2003.
- [HKTR04] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, 22(1):5–53, 2004.
- [HLS⁺07] M. Hu, E.-P. Lim, A. Sun, H. W. Lauw, and B.-Q. Vuong. Measuring article quality in Wikipedia: Models and evaluation. In *Proceedings of the 16th ACM Conference on Conference on Information and Knowledge Management*, pages 243–252, New York, NY, USA, 2007. ACM.
- [HP06] S. Hettich and M. J. Pazzani. Mining for proposal reviewers: Lessons learned at the National Science Foundation. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 862–871, 2006.
- [IWM00] A. Inokuchi, T. Washio, and H. Motoda. An apriori-based algorithm for mining frequent substructures from graph data. In *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery*, pages 13–23, London, UK, 2000. Springer-Verlag.
- [Kes63] M. Kessler. Bibliographic coupling between scientific papers. *American Documentation*, 14:10–25, 1963.

REFERENCES

- [KK01] M. Kuramochi and G. Karypis. Frequent subgraph discovery. In *Proceedings of the 2001 IEEE International Conference on Data Mining*, pages 313–320, Washington, DC, USA, 2001. IEEE Computer Society.
- [KK07] M. Kuramochi and G. Karypis. Discovering frequent geometric subgraphs. *Information Systems*, 32(8):1101–1120, 2007.
- [KKT03] D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 137–146, Washington, D.C, USA, 2003.
- [Kle99] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.
- [KN97] E. Knorr and R. Ng. A unified notion of outliers: Properties and computation. In *Proceedings of the 3rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 219–222, 1997.
- [KNRT04] R. Kumar, J. Novak, P. Raghavan, and A. Tomkins. Structure and evolution of blogspace. *Communications of the ACM*, 47(12):35–39, 2004.
- [KNV06] Y. Koren, S. C. North, and C. Volinsky. Measuring and extracting proximity in networks. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 245–255, New York, NY, USA, 2006. ACM.
- [Kre02] V. E. Krebs. Mapping networks of terrorist cells. *Connections*, 24(3):43–52, 2002.
- [KS05] P. S. Keila and D. B. Skillicorn. Structure in the Enron email dataset. In *Proceedings of the Workshop on Link Analysis, Counterterrorism, and Security (in conj. with SIAM International Conference on Data Mining)*, pages 55–64, Newport Beach, CA, USA, 2005.
- [KSGM03] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina. The Eigentrust algorithm for reputation management in P2P networks. In *Proceedings of the 12th International Conference on World Wide Web*, pages 640–651, New York, NY, USA, 2003. ACM.

REFERENCES

- [KSS97] H. Kautz, B. Selman, and M. Shah. Referralweb: Combining social networks and collaborative filtering. *Communications of the ACM*, 40(3):63–65, 1997.
- [LAH07] J. Leskovec, L. A. Adamic, and B. A. Huberman. The dynamics of viral marketing. *ACM Transactions on the Web*, 1(1):5, 2007.
- [LC03] S. Lin and H. Chalupsky. Unsupervised link discovery in multi-relational data via rarity analysis. In *Proceedings of the 3rd IEEE International Conference on Data Mining*, pages 171–178, Melbourne, FL, USA, 2003.
- [LC04] S. Lin and H. Chalupsky. Issues of verification for unsupervised discovery systems. In *Workshop on Link Analysis and Group Detection (in conj. with the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining)*, Seattle, WA, USA, 2004.
- [LES06] C. Lampe, N. Ellison, and C. Steinfield. A Face(book) in the crowd: Social searching vs. social browsing. In *Proceedings of the 20th ACM Conference on Computer Supported Cooperative Work*, pages 167–170, New York, 2006. ACM.
- [McC06] A. McCook. Is peer review broken? *The Scientist*, 20(2):26, 2006.
- [MH04] M. Mukherjee and L. B. Holder. Graph-based data mining on social networks. In *Workshop on Link Analysis and Group Detection (in conj. with the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining)*, Seattle, WA, USA, 2004.
- [ONL02] K. Ong, W. Ng, and E. Lim. Mining relationship graphs for effective business objectives. In *Proceedings of the 6th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 561–565, Taipei, Taiwan, 2002.
- [PBMW98] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: Bringing order to the Web. In *Stanford Digital Library Technologies Project*, 1998.
- [PL05] S. Patil and J. Lai. Who gets to know what when: Configuring privacy permissions in an awareness application. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 101–110, New York, NY, USA, 2005. ACM.

REFERENCES

- [Pre97] Jon Anderson Preston. Evaluation software: Improving consistency and reliability of performance rating. In *The Supplemental Proceedings of the Conference on Integrating Technology into Computer Science Education: Working Group Reports and Supplemental Proceedings*, pages 132–134, New York, 1997. ACM.
- [RD02] M. Richardson and P. Domingos. Mining knowledge-sharing sites for viral marketing. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 61–70, Edmonton, Alberta, Canada, 2002.
- [RDHT04] J. Resig, S. Dawara, C. M. Homan, and A. Teredesai. Extracting social networks from instant messaging populations. In *Workshop on Link Analysis and Group Detection (in conj. with the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining)*, Seattle, WA, USA, 2004.
- [RKZF00] P. Resnick, K. Kuwabara, R. Zeckhauser, and E. Friedman. Reputation systems. *Communications of the ACM*, 43(12):45–48, 2000.
- [Rot02] R. Rothenberg. From whole cloth: Making up the terrorist network. *Connections*, 24(3):36–42, 2002.
- [RQC⁺04] E. Reid, J. Qin, W. Chung, J. Xu, Y. Zhou, R. Schumaker, M. Sageman, and H. Chen. Terrorism knowledge discovery project: A knowledge discovery approach to addressing the threats of terrorism. In *Proceedings of the 2nd Symposium on Intelligence and Security Informatics*, pages 125–145, Tucson, AZ, USA, 2004.
- [RT04] J. Resig and A. Teredesai. A framework for mining instant messaging services. In *Workshop on Link Analysis, Counterterrorism, and Privacy (in conj. with SIAM International Conference on Data Mining)*, Lake Buena Vista, FA, USA, 2004.
- [RV97] P. Resnick and H. R. Varian. Recommender systems. *Communications of the ACM*, 40(3):56–58, 1997.

REFERENCES

- [RW01] Tracy Riggs and Robert Wilensky. An algorithm for automated rating of reviewers. In *Proceedings of the 1st ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 381–387, New York, 2001. ACM.
- [SCHT07] X. Song, Y. Chi, K. Hino, and B. Tseng. Identifying opinion leaders in the blogosphere. In *Proceedings of the 16th ACM Conference on Conference on Information and Knowledge Management*, pages 971–974, New York, NY, USA, 2007. ACM.
- [SH01] S. Shekhar and Y. Huang. Discovering spatial co-location patterns: A summary of results. In *Proceedings of the 7th International Symposium on Spatial and Temporal Databases*, pages 236–256, Redondo Beach, CA, USA, 2001.
- [SHA99] M. Simon, S. M. Houghton, and K. Aquino. Cognitive biases, risk perception, and venture formation: How individuals decide to start companies. *Journal of Business Venturing*, 15:113–134, 1999.
- [Sma73] H. Small. Co-citation in the scientific literature: A new measure of the relationship between two documents. *Journal of the American Society for Information Science*, 24(4):265–269, 1973.
- [Sma77] H. G. Small. A co-citation model of a scientific specialty: A longitudinal study of collagen research. *Social Studies of Science*, 7:139–166, 1977.
- [SQCF05] J. Sun, H. Qu, D. Chakrabarti, and C. Faloutsos. Neighborhood formation and anomaly detection in bipartite graphs. In *ICDM*, pages 418–425, 2005.
- [SW93] M. F. Schwartz and D. C. M. Wood. Discovering shared interests using graph analysis. *Communications of the ACM*, 36(8):78–89, 1993.
- [SWB97] D. Sullivan, D.H. White, and E.J. Barboni. Co-citation analyses of science: An evaluation. *Social Studies of Science*, 7:223–240, 1997.
- [TF06] H. Tong and C. Faloutsos. Center-piece subgraphs: Problem definition and fast solutions. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 404–413, New York, NY, USA, 2006. ACM.

REFERENCES

- [TMRL02] M. Terry, E.D. Mynatt, K. Ryall, and D. Leigh. Social Net: Using patterns of physical proximity over time to infer shared interests. In *Extended Abstracts of the ACM Conference on Human Factors in Computing Systems*, pages 816–817, New York, 2002. ACM.
- [WF94] S. Wasserman and K. Faust. *Social Network Analysis: Methods and Applications*. Cambridge University Press, 1994.
- [WM98] H.D. White and K.W. McCain. Visualizing a discipline: An author co-citation analysis of information science, 1972-1995. *Journal of American Society of Information Science and Technology*, 49(4):327–355, 1998.
- [WMM02] Henry M. Walker, Weichao Ma, and Dorene Mboya. Variability of referees’ ratings of conference papers. In *Proceedings of the 7th Annual Conference on Innovation and Technology in Computer Science Education*, pages 178–182, New York, 2002. ACM.
- [WMMY02] R. E. Walpole, R. H. Myers, S. L. Myers, and K. Ye. *Probability & Statistics for Engineers & Scientists*. Prentice-Hall, Inc., 7th edition, 2002.
- [XC03] J. Xu and H. Chen. Untangling criminal networks: A case study. In *Proceedings of the 1st Symposium on Intelligence and Security Informatics*, pages 232–248, Tucson, AZ, USA, 2003.
- [XC04] J. Xu and H. Chen. Fighting organized crimes: Using shortest-path algorithms to identify associations in criminal networks. *Decision Support Systems*, 38(3):473–487, 2004.
- [XC05] J. J. Xu and H. Chen. CrimeNet explorer: a framework for criminal network knowledge discovery. *ACM Transactions on Information Systems*, 23(2):201–226, 2005.
- [XL04] L. Xiong and L. Liu. Peertrust: Supporting reputation-based trust in peer-to-peer communities. *IEEE Transactions on Knowledge and Data Engineering*, 16(7):843–857, 2004.
- [XMKC04] J. Xu, B. Marshall, S. Kaza, and H. Chen. Analyzing and visualizing criminal network dynamics: A case study. In *Proceedings of the 2nd Symposium on Intelligence and Security Informatics*, pages 359–377, Tucson, AZ, USA, 2004.

REFERENCES

- [YCL07] B. Yang, W. Cheung, and J. Liu. Community mining from signed social networks. *IEEE Transactions on Knowledge and Data Engineering*, 19(10):1333–1348, 2007.
- [YH02] Xifeng Yan and Jiawei Han. gSpan: Graph-based substructure pattern mining. In *Proceedings of the 2002 IEEE International Conference on Data Mining*, page 721, Washington, DC, USA, 2002. IEEE Computer Society.
- [YH03] X. Yan and J. Han. CloseGraph: Mining closed frequent graph patterns. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 286–295, 2003.
- [YS03] B. Yu and M. P. Singh. Searching social networks. In *Proceedings of the 2nd Joint Conference on Autonomous Agents and Multiagent Systems*, pages 65–72, New York, 2003. ACM.
- [ZvA04] J. Zhang and M. van Alstyne. SWIM: Fostering social network based information search. In *Extended Abstracts on Human Factors in Computing Systems*, pages 1568–1568, New York, 2004. ACM.