Trust and Credibility in Online Social Networks

Hao Xue

Submitted to the graduate degree program in Department of Electrical Engineering and Computer Science and the Graduate Faculty of the University of Kansas in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Committee members

Prof. Fengjun Li, Chairperson

Prof. Prasad Kulkarni

Prof. Bo Luo

Prof. Cuncong Zhong

Prof. Mei Liu

Prof. Hyunjin Seo

Date defended: July 12, 2019

The Dissertation Committee for Hao Xue certifies
that this is the approved version of the following dissertation :

Trust and Credibility in Online Social Networks

Prof. Fengjun Li, Chairperson

Date approved: July 12, 2019

Abstract

Increasing portions of people's social and communicative activities now take place in the digital world. The growth and popularity of online social networks (OSNs) have tremendously facilitated online interaction and information exchange. As OSNs enable people to communicate more effectively, a large volume of user-generated content (UGC) is produced daily. As UGC contains valuable information, more people now turn to OSNs for news, opinions, and social networking. Besides users, companies and business owners also benefit from UGC as they utilize OSNs as the platforms for communicating with customers and marketing activities. Hence, UGC has a powerful impact on users' opinions and decisions. However, the openness of OSNs also brings concerns about trust and credibility online. The freedom and ease of publishing information online could lead to UGC with problematic quality. It has been observed that professional spammers are hired to insert deceptive content and promote harmful information in OSNs. It is known as the spamming problem, which jeopardizes the ecosystems of OSNs.

The severity of the spamming problem has attracted the attention of researchers and many detection approaches have been proposed. However, most existing approaches are based on behavioral patterns. As spammers evolve to evade being detected by faking normal behaviors, these detection approaches may fail.

In this dissertation, we present our work of detecting spammers by extracting behavioral patterns that are difficult to be manipulated in OSNs. We focus on two scenarios, review spamming and social bots. We first identify that the rating deviations and opinion deviations are invariant patterns in review spamming activities since the goal of review spamming is to insert deceptive reviews. We utilize the two kinds of deviations as clues for trust propagation and propose our detection mechanisms. For social bots detection, we identify the behavioral patterns among users in a neighborhood is difficult to be manipulated for a social bot and propose a neighborhood-based

detection scheme. Our work shows that the trustworthiness of a user can be reflected in social relations and opinions expressed in the review content. Besides, our proposed features extracted from the neighborhood are useful for social bot detection.

Acknowledgements

In this section, I would like to express my gratitude to my advisor, my committee members, my colleagues, and my family for their guidance, encouragement, and support along the road of my PhD study.

First of all, I would like to thank my advisor Dr. Fengjun Li for her support, guidance, patience, and contributions during my PhD study. When I began my PhD, she led me to the field of security and patiently helped me grow as a PhD student. She has been very enthusiastic in all our discussion and is always willing to provide insightful comments. I will always be grateful for her patience in guiding me to approach research questions and address issues during research.

I am also very thankful to my committee members: Dr. Bo Luo, Dr. Prasad Kulkarni, Dr. Cuncong Zhong, Dr. Mei Liu, and Dr. Hyunjin Seo. They offer professional suggestions on my proposal and dissertation. Without their help, I cannot finish the entire course of my PhD.

I would like to thank all my colleagues in University of Kansas: Dr. Lei Yang, Dr. Yuhao Yang, Dr. Abdulmalik Humayed, Dr. Chao Lan, Qiaozhi Wang, Pegah Nokhiz, Chris Seasholts, Yingying Ma, Sana Awan, and Sohaib Kiani. They have provided a lot of support in my study. Working together with them is fun and memorable.

Last but not least, I want to thank my family, not just my parents but also my in-laws. My parents give me endless love and support throughout my life. Despite the long trip from China to the US, they have visited me several times and supported me in every way they could during their stay. Without them, I would not be able to have the chance to begin my study in the US. My in-laws have been supporting me with caring, kindness, encouragement, and understanding. I'm blessed to have a family like this. My beloved wife, the one I am so lucky and grateful to have, has been super supportive through this journey. She has brought so much love, joy, encouragement, and positivity to my life. Without her, I would never become the person I am.

Contents

1 Introduction **1**

1.1 Rise of Social Networks 1

1.2 Spamming Problem in Social Networks 2

1.3 Countermeasures 3

1.3.1 Existing Detection 3

1.3.2 Challenges and Issues 3

1.4 Contribution 4

1.5 Organization 5

2 Background and Related Work **6**

2.1 Information Credibility in Online Social Networks 6

2.1.1 Review Spamming in Online Review Systems 6

2.1.2 Trustworthy Recommendations in Recommender Systems 8

2.2 Social Bots Detection 9

2.3 Trust Propagation in Online Social Networks 10

2.3.1 Trust Propagation 11

2.3.2 Modeling Proximity with Random walk 12

2.4 Opinion Mining and Sentiment Analysis 12

2.5 Social Circle Detection 14

3 Trust-Aware Spam Review Detection **15**

3.1 Introduction 15

3.2 Overview of the Problem and Solution 16

3.2.1 Review Spamming Problem . . . 16
3.2.2 Solution Overview . . . 17
3.3 Trust-Aware Spam Detection . . . 18
3.3.1 Trust Inference from Social Relations . . . 18
3.3.2 Trust-based Rating Prediction . . . 21
3.3.3 Trust-Aware Detection based on Rating Deviation . . . 22
3.4 Experiments and Evaluations . . . 24
3.4.1 Data Collection and Dataset . . . 24
3.4.2 Experiment Results and Evaluations . . . 24
3.5 Discussions . . . 31
3.6 Summary . . . 32

4 Content-based Detection through Three-Layer Trust Propagation 33
4.1 Introduction . . . 33
4.2 Overview Problem and Our Solution . . . 36
4.2.1 Problem Overview . . . 36
4.2.2 Overview of Our Solution . . . 37
4.3 Aspect Category Specific Opinion Indicator . . . 38
4.3.1 Aspect-based Opinion Extraction . . . 39
4.3.2 Supervised Opinion Extraction . . . 41
4.3.3 Unsupervised Opinion Extraction . . . 42
4.3.4 Opinion Vector and Quality Vector . . . 45
4.4 Content-based Trust Computation . . . 47
4.4.1 The Three-Layer Trust Relationships . . . 47
4.4.2 Trust Propagation and Trust Scores . . . 49
4.4.3 The Computational Framework . . . 54
4.5 Experiments and Evaluations . . . 55
4.5.1 Dataset . . . 55

4.5.2 Supervised Opinion Extraction . . . 56
4.5.3 Unsupervised Opinion Extraction . . . 59
4.5.4 Trust Propagation with Supervised Opinion Extraction . . . 59
4.5.5 Trust Propagation with Unsupervised Opinion Extraction . . . 65
4.5.6 Evaluation on the Dataset with Yelp Flagged Reviews . . . 68
4.6 Summary . . . 72

5 Neighborhood-based Social Bot Detection 74
5.1 Introduction . . . 74
5.2 Problem Formulation and Overview of Solution . . . 75
5.2.1 Problem Formulation . . . 75
5.2.2 Solution Overview . . . 79
5.3 Neighborhood Modeling . . . 82
5.3.1 Model Neighborhood with Multiple Views . . . 83
5.3.2 View Definition . . . 84
5.3.3 View Construction . . . 86
5.4 Feature Extraction from Views . . . 87
5.4.1 Feature Extraction with Graph Metrics . . . 88
5.5 Social Circle Detection with Multi-View Clustering . . . 89
5.5.1 Motivation . . . 90
5.5.2 Preliminaries of Multi-View Clustering . . . 90
5.6 Social Bot Detection . . . 94
5.6.1 Definition and Summarization of Feature Sets . . . 95
5.6.2 Methods for Social Bot Detection . . . 96
5.7 Experiments and Evaluations . . . 97
5.7.1 Dataset . . . 97
5.7.2 Social Circle Detection with SCSC . . . 98
5.7.3 Content Analysis . . . 106

5.7.4 Social Bot Detection 110
5.7.5 Discussions 114
5.8 Summary 117

6 Conclusion **118**

List of Figures

3.1 Example graph with friendship and compliment relationships and its proximity matrix 20

3.2 Comparing prediction accuracy of the proposed model using three social graphs and the baseline CF approach 25

3.3 Average MAE and rating deviation distribution 27

3.4 CDF plot of the overall trustworthiness score 27

3.5 The relationship between the overall trustworthiness score and the number of friends when t_u is larger or smaller than 0.5 29

3.6 The relationship between the overall trustworthiness score and the number of friends when t_u is larger or smaller than 0.9 30

4.2 Distributions under the aggregate opinion setting: (a) the distribution of honesty score for users; (b) the distribution of faithfulness score for reviews; (c) the distribution of truthfulness score for statements. 60

4.3 Distributions under the positive opinion setting: (a) the distribution of honesty score for users; (b) the distribution of faithfulness score for reviews; (c) the distribution of truthfulness score for statements. 61

4.4 Trust scores distributions with synthetic data: (a) the distribution of honesty score for users; (b) the distribution of faithfulness score for reviews; (c) the distribution of truthfulness score for statements. 61

4.5 Distributions under the aggregate opinion setting: (a) the distribution of honesty scores for users; (b) the distribution of faithfulness scores for reviews; (c) the distribution of truthfulness scores for statements. 65

4.6 Distributions under the positive opinion setting: (a) the distribution of honesty scores for users; (b) the distribution of faithfulness scores for reviews; (c) the distribution of truthfulness scores for statements. 66

4.7 Trust score distributions with synthetic data: (a) the distribution of honesty scores for users; (b) the distribution of faithfulness scores for reviews; (c) the distribution of truthfulness scores for statements. 66

5.1 An example of users follow back a social bot on Twitter 77

5.2 An example of power-law distribution $P(d) \sim d^{-\gamma}$ 77

5.3 Degree distributions from nine randomly chosen bots. In each figure, the x-axis represents the value of degree d and y-axis represents $P(d)$, the fraction of nodes with degree d 78

5.4 Degree distribution from nine randomly chosen legitimate users. In each figure, the x-axis represents the value of degree d and y-axis represents $P(d)$, the fraction of nodes with degree d 78

5.5 A simplified example of the neighborhood of a seed user within a larger network . 80

5.6 A simplified representation of the connections and interactions among onn-users in the neighborhood of a seed user. 81

5.9 Votes for optimal value of k for bots and legitimate users 100

5.21 Overlap ratio of top N frequent words used by neighborhoods of social bots and legitimate users 108

List of Tables

3.1 Comparing prediction accuracy of four approaches using MAE and MAUE 26

3.2 Human Evaluation Agreement on Suspicious Users 31

3.3 Human Evaluation Agreement on Normal Users 31

4.1 Notations of terms and concepts. 38

4.2 The performance of aspect category classification. 57

4.3 The classification performance of category-based sentiment polarities. 58

4.4 The aspect categories extracted from aspect clustering. 60

4.5 The average truthfulness scores of the statements under two initialization settings. . 63

4.6 Statistics of honesty scores using supervised opinion extraction and synthetic data. 63

4.7 The agreements between our model and human evaluators. 65

4.8 Statistics of honesty scores using unsupervised opinion extraction with synthetic data. 67

4.9 The agreements between our model and the evaluators. 68

4.10 The evaluation results of the selected suspicious users for supervised and unsupervised opinion mining 70

5.1 Summarization of Feature Sets 96

5.2 Average TSR for all users, bots, and legitimate users 99

5.3 Top 20 frequent words used in the neighborhoods of bots and legitimate users . . . 107

5.4 Top 3 topics for bots and legitimate users 111

5.5 Classification performances for different feature sets using Random Forest 112

5.6 Classification performances for different feature sets combined with baseline features using Random Forest 113

5.7 Clustering performances for different feature sets using Agglomerative Clustering . 114
5.8 Feature importance of GL feature set with Random Forest 115

Chapter 1

Introduction

1.1 Rise of Social Networks

The advancement of the Internet has tremendously changed the way people live. One of the most influential creation in the information age is online social network (OSN) and it has been increasingly important in our daily life. A huge amount of information is exchanged and shared every day as millions of users engage in a diverse range of routine activities on OSNs such as media sharing, message communication, and content curation.

As the popularity of OSNs grows, the influence of online information is also growing. Online review systems, as the representative of digital media-facilitated social collaborative networks, become one of the most popular platforms people visit for information reference and getting suggestions. For example, TripAdvisor.com, which specializes in travel-related services, has reached 315 million unique monthly visitors and over 200 million reviews, and Yelp.com, which is known for restaurant reviews, has a total of 71 million reviews of businesses and a monthly average of 135 million unique visitors to the site. With a large volume of information and opinions regarding products or services available on these platforms, many people tend to read the reviews before making purchasing decisions and their behaviors and decisions may be significantly affected by others' opinions. A recent survey [12] shows that 97% of consumers read online reviews for local businesses and 85% of consumers trust online reviews as much as personal recommendations. Compared to online review networks, traditional relationship-based OSNs, such as Facebook and Twitter, have also attracted huge amounts of users and daily visits as these platforms help people and organizations connect online and share information efficiently. As of the end of the year 2018,

Twitter has 1.3 billion accounts and 328 million monthly active users with 500 million Tweets sent each day [121]. Such numerous amounts of active users and information exchanged lead to the great value of search and advertising. For instance, it is reported that more than 70 million businesses own a Facebook page and 65.8% of US companies with 100+ employees use Twitter for marketing [121].

As users constantly contribute content to these online platforms, there has been a wealth of peer-to-peer information or user-generated content (UGC) available online. A recent report shows [120] that consumers and marketers disagree on the content authenticity and consumers are more likely to believe UGC is authentic compared to brand-created content. Therefore, UGC could be powerful tools for marketing or sources of opinions.

1.2 Spamming Problem in Social Networks

However, the relative ease of posting online could lead to a significant impact on the overall quality of information accessible to the users. Thus, the credibility of UGC can be problematic. For example, a recent study on Yelp shows that about 16% restaurant reviews are considered suspicious and rejected by Yelp internally [72]. Researches [45, 65, 133, 24] show that individual or professional spammers have been hired to plant deceptive reviews into online review systems to mislead people in making unwise decisions. This is known as *online review spamming*, which was first identified in [45]. The spamming problem does not limit to review spamming. On OSNs like Twitter, automatic accounts known as *social bots* are deployed to maliciously spread harmful information such as malware and fake news. For example, during the 2010 U.S. midterm elections, social bots were employed to support certain candidates and smear their opponents with tweets pointing to websites with fake news [111].

Thus, driven by different incentives, including financial and political ones, a non-negligible portion of UGC on OSNs is untrustworthy, which raises concerns of trust and credibility in OSNs.

1.3 Countermeasures

In this section, we overview the existing detection approaches and provide an analysis of the challenges and issues.

1.3.1 Existing Detection

The spamming problem is destructive to the trust and credibility of OSNs. Its severity has attracted people from various field to engage in combating spammers.

In industry, many deterrence-based and reputation-based approaches have been adopted to help maintain the functionality of OSNs. For example, Amazon and Yelp have adopted the "review of the review" mechanism, which allows users to vote for the review's "helpfulness" and use the ratings and vote counts as a measure to assess the quality and trustworthiness of the review. Twitter allows users to report suspicious content. Many feature-based schemes have been proposed by researchers to identify review spammers, including textual features [45, 96, 88], temporal features [24, 145], individual or group behavior patterns of spammers [133], and sentiment inconsistency [65, 90]. Similarly, behavioral features [129, 60, 144, 18, 151, 17] have been popular in detecting social bots.

1.3.2 Challenges and Issues

Although many approaches have been proposed to address this problem, there are some issues in these approaches.

- **Spammers will mimic normal behaviors**. Changes in spammers' behaviors have been observed indicating spammers are evolving to avoid being detected, which makes rule-based and behavior-based methods less effective. Methods based on textual similarity would easily fail when well-written fake reviews carefully avoid duplication. Spammers may imitate the behavioral pattern of regular users to reduce their levels of suspicious. What's worse, multiple spamming accounts can be controlled by advanced algorithms to conduct large

scale of spamming activities in a well-organized way and even interact with other users as humans do.

- **Ground truth of accounts or content is hard to acquire**. It is difficult to determine whether an account is a spammer or not, especially for review spammers. Labeling spammers by human evaluators is not only labor-intensive and time-consuming. The deterrence-based and reputation-based approaches adopted by the big companies can be considered as a type of crowdsourcing method. Without property incentives, users' participation may be inadequate. Overall, the judgments are usually made based on observations of behavior patterns, which can be subjective and biased.

1.4 Contribution

In this dissertation, we focus on two scenarios in spamming problem, review spamming and social bots. To address the aforementioned challenges, we propose our work of the trust and credibility in OSNs by studying the invariant features that are hard to be manipulated by spammers. More specifically, we observed that the rating and opinion deviations are usually invariant in review spamming. Since inserting deceptive reviews is the main purpose of review spammers, avoiding rating and opinion deviations is against the goal of spammers. On the other hand, a social bot can fake human behaviors to evade being detected, but the behaviors of users in its network are difficult to be manipulated. Thus, the properties of the surrounding network are invariant. We summarize the main contributions of this work as:

- **Utilizing the social relations provided by online review systems for review spammers detection.** We proposed a structure-based framework that detects review spammers using rating deviations and contextual social relationships. We first present a trust-based rating prediction algorithm using local proximity derived from social relationships, such as friendships and complements relationships, using the random walk with restart. We then incorporated the predicted ratings into a pseudo user-item matrix to overcome the sparsity problem

and compute the overall trustworthiness score for every user in the system, which is used as the spamicity indicator.

- **Inferring trustworthiness from the content of online reviews.** We applied opinion-mining techniques to extract opinions that are expressed in the reviews. Then, we integrated the opinions to obtain opinion vectors for individual reviews and statements. Finally, we developed an iterative content-based computational model to compute honesty scores for users, reviews, and statements. According to the results, there exist differences of statement truthfulness across different categories. Our model shows that the trustworthiness of a user is closely related to the content of his/her reviews.

- **Detecting social bots with the neighborhoods.** We proposed a novel detection mechanism to detect social bots by studying the neighborhood formed around the users. We modeled the neighborhoods of users using the ego-network model with multiple representations. Besides the global properties of the networks, we also applied a multi-view clustering approach to detect hidden social circles within the neighborhoods of users. Then we trained bot detectors with features extracted from the users' neighborhoods. The results indicated there exist distinguishable differences between the properties of neighborhoods in legitimate users and social bots. Experimental results show that our proposed neighborhood-based features could help with social bot detection.

1.5 Organization

The rest of the dissertation is organized as follows. Chapter 2 presents the background and related work about this dissertation. Chapter 3 proposes a rating-deviation-based framework that detects review spammers using contextual social relationships. Chapter 4 investigates the problem of inferring trustworthiness from review content. In Chapter 5, we present our neighborhood-based bot detection approach. Finally, Chapter 6 concludes the dissertation.

Chapter 2

Background and Related Work

In this chapter, we present the background and the related work of our research. We first introduce the credibility issues and existing approaches in Section 2.1. In Section 2.2, we review the related work of social bot detection. Then we discuss the related work of trust propagation in OSNs in Section 2.3. In Section 2.4, we present the related work of opinion mining and sentiment analysis, which we adopt in the work of Chapter 4. Finally, we introduce the related work of discovering social circles in Section 2.5, which serves as an auxiliary step for our neighborhood-based approach introduced in Chapter 5.

2.1 Information Credibility in Online Social Networks

Generally, people share and exchange information on social networks. One problem is that the quality and reliability of the user-generated content cannot be guaranteed. On the other hand, the individual a user is communicating is sometimes anonymous, thus the user may be concerned about the trustworthiness of the information provided by that individual. In this section, we identify and introduce two information credibility problems that are relevant to our research problems.

2.1.1 Review Spamming in Online Review Systems

Review spamming, also known as opinion spamming, usually refers to "illegal" activities (i.e., posting fake reviews) that try to mislead readers or game the review systems by giving undeserving positive/negative opinions to some target entities in order to promote/demote the entities. Opinion spamming has many forms, for instance, fake reviews, fake comments, fake social network

postings, and deceptive messages, etc. Usually, it is quite hard for people to manually spot spam reviews, which make it critical to deploy some defensive mechanism against review spamming behaviors.

The problem of spam review detection was first studied in [45], which mainly focused on detecting duplicate spam reviews based on content similarity. Later on, various approaches have been proposed as the review spamming problem attracts more attention. One popular direction is to design machine learning methods based on behavioral patterns, such as rating behavior [65, 26], unexpectedness [46] , temporal and spatial patterns [62, 24, 145], group behaviors [88, 63], and inconsistency [90], etc. These behavioral features are usually manually selected based on heuristics and observations. One drawback of behavioral-based methods is that spammers' behavioral can change as they learn to disguise themselves as normal users. On the other hand, our approach focuses directly on opinions expressed in the review, which is more straightforward and does not require crafting the behavioral features. So, aspects other than behavior should be taken into account. In [133], an iterative structure-based model was proposed by calculating three intertwined scores for reviewer, review, and store respectively. However, they treated all links equally and didn't consider features on the link. Our approach, on the other hand, not only takes users' behaviors like rating variance into consideration but also uses relationship strengths as the weights on the links for spamming detection. Besides, we adopt the rating prediction technique in recommendation systems to incorporate trust into our detection framework.

With the development of deep neural networks, various deep learning based methods have been proposed [63, 155, 64, 113]. These methods mainly constructed deep neural networks to learn the dense representation of the reviews. The advantage is that no deliberately crafted features needed. However, these methods are more complex and take much more time to train.

The third type of approaches is content-based. [96] proposed a method based on linguistic features such as n-gram, while [25] focused on the syntactic rules. These work mainly utilized the language patterns of the reviews rather than the actual opinions as our model does. [95, 102] conducted sentiment analysis on the review. However, these work only analyzed the sentiment of

each review, while we focus on finer-grained aspect-specific opinions. Compared to aforementioned work, [27] is the closest to ours. They analyzed the aspect-specific profile of the products and compared the compatibility with the reviews'. Compared to our proposed method, their work only considered each review individually, while our method integrated the aspect-specific opinions into the trust propagation model and computed the credibility of users as well.

2.1.2 Trustworthy Recommendations in Recommender Systems

Recommender Systems are widely used online (e.g., Amazon.com, Netflix, YouTube, etc) to provide suggestions about suitable items for users based on their previous preferences and rating behaviors. Among the various models developed, collaborative filtering (CF) [114] has been the most successful one. Typically, CF methods can be divided into two categories, model-based and memory-based. In model-based methods, machine learning techniques [153, 128] are applied on data related to users' behaviors to learn models for predictions. Memory-based models work under the assumption that users who agree in the past will also agree in the future. Similarities are usually calculated using cosine similarity or Pearson Correlation. For a particular user, a missing rating is calculated by aggregating ratings from k most similar users.

However, the standard CF method suffers from the sparsity problem and performs poorly for cold-start users, who newly joined the system and have few review history. Also, standard CF method is not attack-resistant. Because of the existence of malicious users and spamming activities, various approaches seek to incorporate trust into the recommender system to improve the quality and trustworthiness of predictions. Although recommender systems have been comprehensively analyzed, the study of social-based or trust-based recommender systems is relatively new. Trust and reputation were first integrated in [125], but its trust and reputation purely rely on ratings, which limits the effectiveness. Trust propagation was taken into account to make trustworthy predictions in some of the early work [79, 76, 77]. However, these methods still rely on user-user similarity and thus cannot yield accurate results for cold-start users. More recent work [85, 61, 32] have proposed different ways to address the cold start problem. In this proposed work, our

method overcomes the sparsity problem by taking social relationships into account to measure users' closenesses. For users without much review history, our approach is still able to make trustworthy predictions as long as social relationships exist.

2.2 Social Bots Detection

Social bots are accounts created by human or automated scripts and controlled by algorithms to perform certain tasks [11, 28, 18]. While some benign social bots are adopted by companies for purposes such as marketing and customer care, malicious social bots deployed by attackers can endanger the ecosystem of OSNs. In this work, we focus on the detection of malicious social bots. Compared to review spammers we discussed previously, social bots can be considered as automated spammers with more diverse intentions, including promoting harmful information such as malware and rumors [10, 28], harvesting users' privacy data [11], and infiltrating OSNs [11]. To prevent the harm brought by social bots, researchers have been engaging in the design of bot detection mechanisms.

One type of defense systems is based on human detection. The human effort is from either crowdsourcing [126, 131] or hired experts [122]. This type of systems assumes bot detection is a feasible and simple task for human and trust the judgment of the annotators. While human detection may be effective [28], the drawbacks are also obvious. The human labor is usually slow and the standards of different annotators may be different. Hence, human-involved detection is not our focus in this work. In the rest of this dissertation, we refer social bot detection to automatic approaches.

One popular type of automatic detection is feature-based [129, 60, 144, 18, 151, 17] by treating bot detection as a machine learning problem. The adopted features are commonly generated from the different perspectives of users' behavioral patterns to capture orthogonal dimensions of characteristics. The categories of features can be summarized as following: user meta-data (e.g. length of user name, number of friends, etc), user's network (e.g. distribution of in-degree and out-degree), the content of the user posted (e.g. number of words and pos-tags, etc), sentiment,

and timing. The major problem of this existing feature-based approaches is that the adopted features are largely dependent of users' behaviors such as the number of tweets and sentiment in the tweet, or descriptive information of users such as length of the username. As bots are evolving and changing to mimic human behaviors, these methods may fail.

Another type of detection is graph-based. These approaches usually make strong assumptions of the structures of the OSNs [28]. For example, [14] assumes that a user who interacts with a legitimate user is considered legitimate himself. [146, 100] rely on the assumption that the structure of an OSN alone is able to separate social bots from legitimate users. These assumptions sound plausible but have been proved wrong. Various experiments [11, 20, 123, 60] have shown that it is possible for bots and legitimate users to connect or interact with each other.

With the development of deep learning techniques, recent work has made attempts to apply deep neural networks (DNNs) [56, 75] to help with bot detection. These work adopted the content users posted as the major input and let the DNNs to generate and select the features automatically. However, training DNNs are usually expensive. Besides, the results of DNNs are usually harder to interpret, which gives limited hints of how bots are distinguished from legitimate users.

In this work, we approach this problem by taking advantage of the users' surrounding networks. Our experimental results show that the behaviors of surrounding users are good predictors of whether a user is legitimate or not.

2.3 Trust Propagation in Online Social Networks

Social networks encourage users to connect to each other and form their own online social circles. When a user turns to social networks for information, it is intuitive for him to believe the statements from a trusted (social-connected) user than from a stranger. Recursively, since a trusted source will also trust the beliefs of his friends, trusts may propagate (with appropriate discounting) through the relationship network.

2.3.1 Trust Propagation

Trust and trust propagation have been extensively studied in the literature. The general idea of reinforcement based on graph link information has been proved effective. HITS [52] and PageRank [97] are successful examples in link-based ranking computation. Trust propagation has been widely applied in recommender systems to make trustworthy predictions [77, 85, 61, 32]. In these work, trust is transmitted in a network of neighbors. The nodes in the networks are usually of the same type, for example, users. [133] built a heterogeneous graph to compute trustworthiness scores for users, reviews, and stores. Besides trust, various work also modeled the propagation of distrust [124, 159]. In our model, we do not model distrust explicitly. Instead, we added the penalty to the propagated trust when a deviation from the majority happens. Compared to our work, these approaches only considered link-based information provided by the system but did not consider content information.

Trust not only can be determined by surrounding neighbors in a network but also can rely on the context when it comes to content. [156] proposed a topic-based model to estimate the trustworthiness of users and tweets on Twitter. Compared to our model, their model evaluates trust based on topic similarities, while our approach models trust based on the deviations of aspect-specific opinions from majority opinions. The work in [9] modeled trust of users by comparing the content of social network posts to actually happened events. Similar to our model, this work models trust by deviations. However, our proposed model utilized deviations in finer-grained dimensions. [130] proposed a content-driven framework for computing trust of sources, evidence, and claims. The difference between this model and ours is that we extract more fine-grained information from content, while the model in [130] mainly used the similarities between content in general. Also, in [130], the inter-evidence similarity plays an important role to make sure that similar evidence gets similar scores. However, the consensus opinions used in our model already represent such similarity, so we did not add the inter-evidence similarity. Besides, we also redefined the computational rules in the context of our problem.

2.3.2 Modeling Proximity with Random walk

The formalization of a sequence of some random steps on a graph or a web is a random walk. The relations among items and users can be represented using graphs where objects and relationships are represented as nodes and weighted edges (directed or undirected) respectively. Thus, similarities and closenesses of two nodes can be measured using transition probabilities by applying random walks on graphs [71]. Several approaches applied this idea to recommender systems. [153] proposed a random walk method to capture the transitive similarities along the item-item matrix to alleviate the sparsity problem in CF. However, since the type of items may vary, it is arbitrary to say that the captured transitive similarity would be accurate. A trust-based and item-based model for recommender system was proposed in [42]. It used the ratings of connected nodes directly as recommendations, which introduced bias into the recommendations. A random walk with restarts (RWR) method was proposed in [55] to measure the closeness between among users, music tracks, and tags for the collaborative recommendation. This work showed the effectiveness of modeling closenesses among nodes, but compared with our work, we strengthened the connections among nodes by incorporating multiple relationships. Also, our model runs more efficiently by focusing on only a partition that contains the starting point and achieves good local approximation.

2.4 Opinion Mining and Sentiment Analysis

Opinion mining and sentiment analysis is the field of study that analyzes people's opinions, sentiments, evaluations, attitudes, and emotions from written language [66]. It is also one of the most active research areas in natural language processing (NLP), data mining, and text mining. The growing importance of opinion mining and sentiment analysis coincides with the growth of social media such as online review systems, blogs, and social networks.

Note that opinion mining and sentiment analysis sounds like two different areas, but they are highly related and overlapped. The process of opinion mining is usually twofold, opinion aspect extraction and aspect-based sentiment prediction. Aspect extraction aims to extract product

features from the opinionated text. The words that represent desired aspects are often nouns or noun phrases that can be captured using syntactic patterns. Thus, various methods that utilized dependency-based rules were proposed [39, 67, 105, 107, 109, 149, 143, 68, 69]. With advancements of deep learning, deep neural network can also achieve decent performance [135, 106]. Usually, extracted aspects are usually words that represent specific aspects. Thus, grouping them into high-level concepts are usually required for further analysis. Lexical tools like WordNet [83] are often used. Besides, topic modeling-based approaches are also very popular[134, 16, 54, 13, 47, 127], as they are able to extract and group aspects simultaneously.

On the other hand, the goal of sentiment analysis is to analyze the polarity orientation of the sentiment words towards a feature or a topic of the product. One common way is to use some sentiment tools directly, such as MPQA Subjectivity Lexicon [142] and SentiWordNet [4]. With such tools developed by other researchers, various work has been proposed [22, 35, 115]. Another common practice is to infer the polarity of target words using a small group of seed terms with known polarity [21, 44]. In addition, supervised learning algorithms are often applied in previous work [98, 116, 48], as well as deep learning based approaches [118, 19].

In this work, we adopted two methods for aspect extraction. One is supervised-based method with Support Vector Machine (SVM). This method does not output specific aspect words but assigns the high-level category and corresponding sentiment polarity directly. The other one is to use a dependency parser. We applied K-means clustering to cluster the aspect words together. For each aspect, we used a sentiment to determine its sentiment. Note that opinion mining is not the main focus and contribution of this work. The difference between our goal and typical opinion mining work is that we are not trying to improve the performance of extracting opinions. Instead, our purpose of applying opinion technique is to use the extracted aspects as deviation indicators for trustworthiness analysis.

2.5 Social Circle Detection

Social circles in social networks, also referred as online communities, aims to group people who share similar or have more connections together. Identifying social circle has been an important task in social network analysis and privacy-preserving research. In this dissertation, we apply social circle detection as an auxiliary step of social bot detection. We only focus on automatic social circles detection as manually assigning circles is labor-intensive and hard to consider the latent information provided by social networks.

Since the structure of a social network is naturally a graph, many graph-based approaches that rely on the topological structures of the networks are proposed [29]. Examples of are graph-partition [49, 5], hierarchical clustering [30], spectral clustering [93], divisive algorithms [33, 110], modularity-based methods [92], and non-negative matrix factorization [108]. Another type of work utilizes the content information and adopted models like generative Bayesian model [157, 158, 70, 15] and tag network [136].

In our problem, we have multiple sources of data about the users' networks, including topological structures, direct and indirect interactions, and content information. To integrates data from different sources, we propose to apply a multi-view clustering model called Selective Co-trained Spectral Clustering. The circles are detected by achieving agreement across different views.

Chapter 3

Trust-Aware Spam Review Detection

3.1 Introduction

Many online review systems encourage interactions among their users, for example, Yelp.com and Last.fm allow registered users to form friendships; Amazon supports sending "helpfulness" tags to reviews that a user finds useful; Epinions and Ciao allow a user to add others in a trust list. So, online review systems can be viewed as a giant social graph. Inspired by the structural analysis [53, 97], we propose to utilize the social relationships among users in the review systems. We argue that spam reviewers, while trying their best to pretend as genuine users, still behave abnormally in large-scale social interactions in general. Recruited with monetary incentives, they are reluctant to devote a large amount of effort that is typically required in social interactions. As a result, spammers tend to be more isolated in social graphs than regular active users. Moreover, users tend to trust those who are socially connected with them more than strangers, indicating a correlation between trust and social relationship strength among users. Since the primary objective of review spamming is rating manipulation, rating variance metrics seem to be effective in distinguishing spam reviews from regular reviews. However, due to the data sparsity problem that is typical in review systems, the straightforward adoption of rating deviation based detection mechanisms perform poorly. From this consideration, we propose to first utilize social relations to predict "trustworthy" ratings for items that a user has not yet reviewed. Then, the predicted ratings will be incorporated into a model that evaluates the quality of reviews according to rating variances.

Rating prediction is a typical problem in recommender systems. For instance, collaborative filtering methods are commonly used to predict ratings or preferences that a user would give to

target items. Moreover, trust propagation and trust networks are used in making trustworthy predictions [76], while trust, reputation, and similarity are combined in [125] to improve the quality of recommendations. Recent trust-aware recommendation approaches take users' trust relations into account and incorporate social relationships among users to improve traditional recommendation systems [6, 73, 55, 74]. While a few systems let users to explicitly express the perceived trust about other users, e.g., *Epinions* allows a user to add another user to her trust list if she likes or agrees with the review issued by this user, most of them provide indirect mechanisms for inferring the trust, e.g., the "helpfulness" votes that a user gives to reviews when the content is considered as useful. Our goal is different from these approaches since we define the concept of trust as the belief of a user in a review that it is not a spam. Therefore, we rely on social ties that indicate strong trust relationships than review quality or prediction accuracy.

3.2 Overview of the Problem and Solution

In this section, we first overview the problem review spamming problem. Then we present the idea of our proposed detection approach.

3.2.1 Review Spamming Problem

Nowadays, many review systems provide more rich functionalities than merely rating and review content. Results from an analysis of Yelp.com's reviews showed that when evaluating the usefulness of online reviews, profile information and reputations of users would influence the perceived usefulness [117]. Attributes of users are significantly associated with how reviews are evaluated. Our preliminary study shows that a review should be detailed, in a suitable length, balanced and consistent in order to be perceived as helpful. In terms of credibility, more factors of the user should be taken into account. A credible user should be someone who uses the real name and has a real profile picture. On the profile, the number of friends, the number of compliments received, and whether the user is an "Elite" would affect the judgment. Our study also showed that when

assessing reviews, the ones written by a friend would be considered more trustworthy than the ones from a stranger. In this work, we argue that users with less or none social interactions will be more suspicious to be review spammers than users who are socially active.

3.2.2 Solution Overview

In this paper, we focus our work on Yelp.com, but our algorithm can be extended to any social review system with complex user interactions. Yelp aims at building a community rather than merely being a rating platform. It provides rich functionalities to its users. Besides writing text reviews and assigning ratings, users can also upload photos with their reviews or use mobile App to "check-in". Popular and active users can be promoted as "Yelp Elite" as recognition of role models on and off-site. Besides, Yelp encourages users to form social relationships and interact with each other in various means, including following other users, sending compliments to other users, sending a private message to other users, and tagging other users' reviews (e.g., cool, funny, and useful), etc. Typically, users of Yelp.com would take efforts to maintain a positive image online. Besides writing faithful reviews, they would also devote time to form social relationships with other users. On the other hand, driven by financial motivation, review spammers are hired to promote or demote their target items. It is natural to think that they devote most of their time into posting fake reviews and would not take much effort interacting with other users online. Thus, they behave very differently from normal users in the aspect of social interactions. However, existing work on spamming detection didn't take this into account.

In this work, we propose an approach that integrates trust-based rating prediction using random walk with restart and rating deviation based iterative model for spam detection. In the problem of spam detection, the rating is an important factor. A review's rating about an item reflects its opinion [133]. Nowadays, spam reviews can hardly dominate the system, so the opinions of the majority should reflect the actual quality of an item at a certain extent. Conceptually, if a large portion of reviews of a user deviates much from the majority's views, it is reasonable for us to consider this user as a suspicious spammer. Rating deviation has been used as a major feature for spam detection

[86, 65, 133]. However, review data on most review systems is sparse: for many users, the number of reviews is not large enough to derive stable credibility. In the Yelp.com dataset that we used in our experiment, about 80.59% users wrote less or equal to 2 reviews and about 93.37% users wrote less than 5 reviews. Under this circumstance, it's necessary to find reliable methods to make trustworthy predictions to fill the missing entries in the sparse user-item rating matrix. An effective solution would be applying random walks on graphs, [153, 42] proved the effectiveness, but their models have limitations, which are discussed in the previous chapter.

3.3 Trust-Aware Spam Detection

In this section, we describe an iterative model to calculate the overall trustworthiness of all the reviewers in the system and use it as an indicator to determine the likelihood of being a review spammer. More specifically, we first introduce a random walk with restart approach to infer the perceived trustworthiness of one user for another user based on the social relations between them and then present our trust-based rating prediction model to derive proximity-based predictions to overcome the data sparsity problem. Finally, we present the design of the iterative model to compute an overall trustworthiness score for each user as the spamicity indicator.

3.3.1 Trust Inference from Social Relations

The goal of this work is to detect suspicious content and actions in online review systems with third-party user-generated content (UGC). It is conceivable that social relations among users can be utilized to measure the trustworthiness of a user perceived by others and extend it to the UGC he submits.

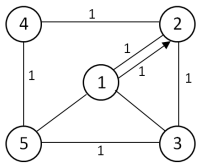
Trust-aware recommendation systems or social collaborative recommendation systems are developed based on the assumption that users have similar tastes with other users they trust or connect to. However, this hypothesis may not always be true in the real world. For example, one's friends may have variant opinions about the same item, and thus social regularization is introduced to treat

friends differently based on how similar a friend is with the target user [74]. From this aspect, our goal is different from these approaches since we define the concept of trust as the belief of a user in the UGC that it is submitted by a legal reviewer but not a suspicious spammer. Such belief can be generated from the interactions among the users. In particular, we consider two relationships available in our dataset collected from *Yelp.com*: the social *friendship* that often reflects a strong tie between users with mutual and cooperative interactions, and the unilateral *compliment* relationship (similar to up-votes, helpfulness votes, etc. in other online review systems) that does not require a confirmation in the reverse direction. Under our definition of trust, for a target user, the one-way compliment relationship represents an equally trustful relationship as the two-way friendship to other users since it indicates a subjective perception of trust. Based on these considerations, we propose to represent the inherent relational structure among the users in a graph G and model the trustworthiness that a user i gives to other users as the *proximity* from node i to any other nodes in G.

Definition 1. Given a graph $G = (V, E)$, a **random walk** on the graph refers to the process of traversing the graph according to the probability transition matrix S.

Among various proximity measures, we adopt the random walk with restart (RWR) model to measure the distance between two nodes since RWR explores the geometry of the graph and takes all the possible paths into account. Moreover, RWR can model the multi-faceted relationship between two nodes, which makes it an ideal proximity measurement for the problem we study in this work. RWR model starts a random walk at node i and computes the proximity of every other node to it. The RWR proximity from node i to node j is the probability that a random walk starting from i reaches j after infinite time, considering that at any transition the random walk will restart at i with a probability α ($0 < \alpha < 1$) or randomly move to another node along the link with a probability (1 - α). From an initial state (denoted as a column vector $\mathbf{q}$), the state of node i at step $k+1$ can be calculated as [55]

$$\mathbf{p}_i^{(k+1)} = (1-\alpha)\mathbf{S}\mathbf{p}_i^{(k)} + \alpha\mathbf{q} \tag{3.1}$$



(a) An example graph with 5 nodes

	Node 1	Node 2	Node 3	Node4	Node 5
Node 1	0.5671	0.1678	0.1184	0.0486	0.0979
Node 2	0.1248	0.5845	0.1197	0.1030	0.0678
Node 3	0.1379	0.1357	0.5621	0.0453	0.1189
Node 4	0.0624	0.1672	0.0598	0.5515	0.1589
Node 5	0.1248	0.0845	0.1197	0.1030	0.5678

(b) Converged proximity matrix

Figure 3.1: Example graph with friendship and compliment relationships and its proximity matrix

where $\mathbf{p}_i^k$ is the *proximity vector* of node i at step k with $p_i(j)^k$ denoting the probability at step k that the random walk is at node j, and $\mathbf{S}$ represents the column normalized transition probability matrix for all nodes in the graph with $S_{i,j}$ denoting the transition probability of moving from a current state at node i to the next state at node j. From an initial state, we recursively apply equation (3.1) until it converges after l steps. Then, the steady-state transition probability $p_i^l(j)$ represents the proximity from node j to the target user i.

Example 1. Figure 3.1 illustrates a toy graph of 5 nodes and the proximity matrix computed from the graph. The link between two nodes denotes the bidirectional friendship relationship and a directed arrow points to the receiver of a compliment. In the example, the closenesses with user 1 is user 2 $>$ user 3 $>$ user 5 $>$ user 4.

Here, we treat the strengths of friendship and compliment equally and set them both to be 1. The distribution of $\mathbf{p}_i$ is highly skewed such that the calculated proximity drops exponentially with the increase in the distance between two nodes. To speed up the computation of the proximity, it is suggested to perform RWR only on the partition that contains the starting point and iteratively approximate a local estimation. To achieve a desirable (near) real-time response, we define the *neighborhood* of a target node as the partition with nodes of a maximum distance of m and restrict the iteration number by l steps. Moreover, as different types of relations may indicate different levels of trust, we further define *link strength* for each type of relationship links and take them into account when column normalizing the transition matrix $\mathbf{S}$.

3.3.2 Trust-based Rating Prediction

The proximity measured from the friendship and compliment relations in the RWR model demonstrates the perceived trustworthiness of a user to others who are socially connected with him, and thus can be used as a trustworthiness weight in the user-based collaborative filtering approach to weight each user's contribution to the rating prediction. In particular, in an online review system with $|\mathcal{U}|$ users and $|\mathcal{I}|$ items, we model the trustworthiness of the target user a to another user u in the system as a function of $p_{a,u} = \mathbf{p}_a(u)$ and adopt Resnick's standard prediction formula [114] to calculate the predicted rating of user a to any item i that a has not yet reviewed:

$$\hat{r}_{a,i} = \bar{r_a} + \frac{\sum_{u \in \mathcal{U}_N(a), u \neq a} (r_{u,i} - \bar{r_u}) \omega_{a,u}}{\sum_{u \in \mathcal{U}, u \neq a} \omega_{a,u}} \tag{3.2}$$

where $\bar{r_a}$ and $\bar{r_u}$ are the average ratings of user a and u, respectively, $r_{u,i}$ is the rating of user u to item i, and $\omega_{a,u} = f(p_{a,u})$. $f(\cdot)$ is a linear trust function to relate the perceived trustworthiness with the relationship-based closeness. For simplicity, we consider $\omega_{a,u} = p_{a,u}$ in this work. The prediction is based on the ratings to the item i from all the users in the neighborhood of the target a (i.e., $\mathcal{U}_N(a)$) who has reviewed i. This predicted rating aggregates the contributions of users who are considered trustworthy by the target user a but neglects the contributions from users with common preference judgments in the past, which makes it different from traditional user-based CF approaches. This is because our goal is to find a trusted prediction of the rating whose value falls into a reasonable range (with non-suspicious rating variance) but not the most accurate prediction of the rating. From this consideration, our model is more tolerant to small inaccuracies in rating predictions than the CF model and its variants and thus can support several relaxations for better efficiency. Social relations are employed in our model to overcome the data sparsity problem, however, it should be noted that for users with no social interactions, it is still impossible to predict the ratings for items that they have not reviewed. Finally, with the trust-based predictions, we form a "pseudo user-item rating matrix" of $|\mathcal{U}|$ users and $|\mathcal{I}|$ items with three types of elements, the original ratings $r_{u,i}$, the predicted ratings $\hat{r}_{u,i}$ and empty ratings "-".

3.3.3 Trust-Aware Detection based on Rating Deviation

In recommendation systems, rating deviation that is inversely related to the recommendation accuracy is often considered as a confidence measurement [58]. Hybrid recommendation approaches have been proposed to first adopt any existing CF algorithm as a "black box" to predict ratings of unrated items, and recommend the top-N items by filtering out the ones with rating variances larger than a deviation threshold, which can be user-specified or item-specific standard deviation. Based on the observation that the accuracy of predictions monotonically decreases with the increase of rating variance [1], we propose to calculate a *quality score*, q_i, for every item i based on all the ratings (with both original and predicted ones) received on i, denoted as $R_{u,i}$, as well as the item-specific rating variance. In particular, $R_{u,i}$ is from the pseudo user-item rating matrix, which is either $r_{u,i}$ or $\hat{r}_{u,i}$. A straightforward approach is illustrated in an iterative model as below:

$$q_i = \frac{\sum_{u=1}^{|\mathcal{U}|} R_{u,i} v_{u,i}}{\sum_{u=1}^{|\mathcal{U}|} v_{u,i}} \tag{3.3}$$

where $v_{u,i}$ is a rating-deviation-based vote defined as:

$$v_{u,i} = \begin{cases} 1, & |R_{u,i} - q_i| \leq \Delta \\ 0, & |R_{u,i} - q_i| > \Delta \end{cases} \tag{3.4}$$

Here, Δ is the deviation threshold defining the maximum acceptable rating deviation for any trust-based rating predication that is considered accurate. The value of Δ can be selected arbitrarily from a suggested range or defined as the standard deviation to model the worst case scenario. With a large Δ, ratings of any arbitrary reviews are more likely to be included in the quality estimation of an item. On the contrary, when the Δ is small, the quality estimation is more likely to be biased. In hybrid recommendation systems, it is commonly suggested to select Δ in a range from 0.8 to 1.2 [1]. As discussed in Section 3.3.2, our model is less sensitive to rating inaccuracy than CF based approaches, therefore, we suggest to tolerate a reasonably larger inaccuracy with a large Δ in order to incorporate more trusted ratings. In the experiment, we learned the value of Δ (=2.011)

from a small set of labeled data.

The total number of votes received by a user reflects the trustworthiness of the user by taking into account both social relational structure with all the neighboring nodes and the rating deviations of all the items rated by the user and his neighbors. Therefore, it is natural to derive an *overall trustworthiness score*, $t_u \in [0,1]$, for any user u in the system:

$$t_u = \frac{\frac{\sum_{i=1}^{n} v_{u,i}}{\#ofreviews(u)}}{max_{a \in \mathcal{U}}(\frac{\sum_{i=1}^{n} v_{a,i}}{\#ofreviews(a)})} \tag{3.5}$$

where t_u is defined as the per-review vote count normalized by the maximal per-review vote count among all the users. This is to limit the impact (and the potential bias) of less active users who only reviewed a small number of items on the estimated item quality. The overall trustworthiness score t_u is updated iteratively each round according to the deviation-based votes. To incorporate the trustworthiness into item quality estimation defined in Equation (3.3), we re-define it as the weighted quality score:

$$q_i = \frac{\sum_{u=1}^{|\mathcal{U}|} R_{u,i} t_u}{\sum_{u=1}^{|\mathcal{U}|} t_u} \tag{3.6}$$

We describe the process of the iterative model in Algorithm 1, which can be considered as a two-layer trust propagation model between items and users. Finally, the overall trustworthiness scores for all users calculated from the iterative model are used as indicators to distinguish regular reviewers and the potential spam reviewers. In particular, users with $t_u \leq \tau_L$ is considered as spammers and users with $t_u \geq \tau_U$ is considered non-spammers, where τ_L and τ_U are pre-selected lower and upper thresholds. In the next section, we will discuss the detection results based on experiments over a dataset of 50,304 reviews collected from a representative online review system, *Yelp.com*, and evaluate the performance of our trust-aware iterative detection model.

Algorithm 1: Framework for Two-Layer Trust Propagation between Items and Users

Input : Sets of items $\mathcal{I}$ and users $\mathcal{U}$
Initial t_u for all users in $\mathcal{U}$
Rating deviation threshold Δ
Output: Item quality scores q_i for all items in $\mathcal{I}$
Overall trustworthiness scores t_u for all users in $\mathcal{U}$
repeat
  Compute the quality scores for all items using 3.6
  Count trust votes and compute per-review vote counts for all users using 3.4
  Update the overall trustworthiness score using 3.5
until *converged*;

3.4 Experiments and Evaluations

In this section, we first introduce the dataset we use in the experiments, and then present the experiment results for evaluations.

3.4.1 Data Collection and Dataset

In the proposed iterative model, we consider social relationships among the users and calculate the proximity to a target user as an indicator of the perceived trust. Since there is no publicly available dataset that includes both reviews and social relationships (e.g., friendships and other uni- or bi-directional relationships), we have collected data from Yelp.com, a widely used online review system known mainly for restaurant reviews, for experimentation. We have crawled approximately 9 million (9,314,945) reviews submitted by 1,246,453 reviewers for 125,815 stores in 12 cities in the United States between 2004 and 2013, and completed the entire data collection process by March 2013. In this work, we extracted a smaller dataset of the city of Palo Alto, CA, with 300 stores, 22,877 users, and 50,304 reviews, and conducted experiments over this dataset.

3.4.2 Experiment Results and Evaluations

In this section, we explain the details about the experiments we conducted. We first analyze the performance of the proposed trust-based rating prediction. Then we evaluated the trustworthiness

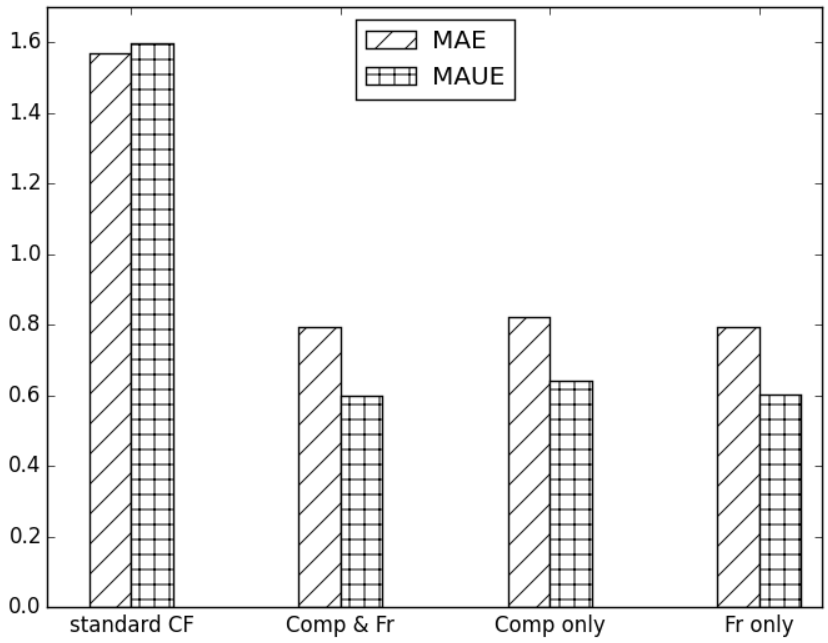


Figure 3.2: Comparing prediction accuracy of the proposed model using three social graphs and the baseline CF approach

score computed with our iterative model.

Trust-based Rating Predictions. We applied our trust-based rating prediction with RWR algorithm on three social graphs that include compliment relationship only, friendship only, and the two-faceted relationship, respectively, and compare the resulted prediction accuracy with a baseline approach that adopts the user-based collaborative filtering model. For performance evaluation, we consider two metrics, the Mean Absolute Error (MAE) and the Mean Absolute User Error (MAUE), defined as below:

$$MAE = \frac{\sum_{u=1}^{|\mathcal{U}|} \sum_{i=1}^{|\mathcal{I}_u|} |\hat{r}_{u,i} - r_{u,i}|}{\sum_{u=1}^{|\mathcal{U}|} |\mathcal{I}_u|} \tag{3.7}$$

and

$$MAUE = \frac{\sum_{u=1}^{|\mathcal{U}|} (\sum_{i=1}^{|\mathcal{I}_u|} (\hat{r}_{u,i} - r_{u,i}) / |\mathcal{I}_u|)}{|\mathcal{U}|} \tag{3.8}$$

where $\mathcal{I}_u$ is the set of items on which user u has both actual and predicted ratings, denoted as $r_{u,i}$ and

$\hat{r}_{u,i}$, respectively. From definitions, MAE measures the average absolute deviation between users' predicted ratings and actual ratings on the items in the evaluation set [38]. Different from MAE, MAUE denotes the average of the mean errors of all users [76]. We compare the performance of four approaches in terms of their MAEs and MAUEs and show the results in Table 3.1 and Figure 3.2. Obviously, our proposed method outperformed the baseline CF method under both metrics. Let us further compare the performance of the proposed trust-based rating prediction using RWR in three social graphs. While all three groups of relationships yield very close MAEs and MAUEs denoting similar performances in prediction accuracy, the two-faceted relationship that combines compliments and friendships achieves the best performance, while the compliments-only approach performed worst among the three. This is consistent with our expectations since friendship is a stronger relationship due to bilateral agreements from both sides. The results also showed that the predicted ratings are accurate estimations of user's opinions that are derived in a collaborative means.

Metrics / Methods	MAE	MAUE
Standard CF	1.5672	1.5956
Compliments-only	0.8232	0.6423
Friendships-only	0.7934	0.6033
Two-faceted	0.7921	0.5985

Table 3.1: Comparing prediction accuracy of four approaches using MAE and MAUE

Figure 3.3 shows further details about the distribution of rating variances and the average MAE. As shown in the figure, we divided the rating deviation into 9 ranges and compare the prediction accuracy of standard CF and RWR on compliments-and-friendships graph. It is obvious that our method outperformed standard CF since a large number of predicated ratings fall into the ranges with smaller deviations. The potential causes of why the standard CF did not yield a satisfying performance will be discussed in Section VI.

Overall Trustworthiness Scores. In the experiment, the initial trustworthiness score for all users are set to 0.5. The model iteratively follows the process shown in Algorithm 1. The algorithm is considered converged when the summation of the differences of all trustworthiness scores is less

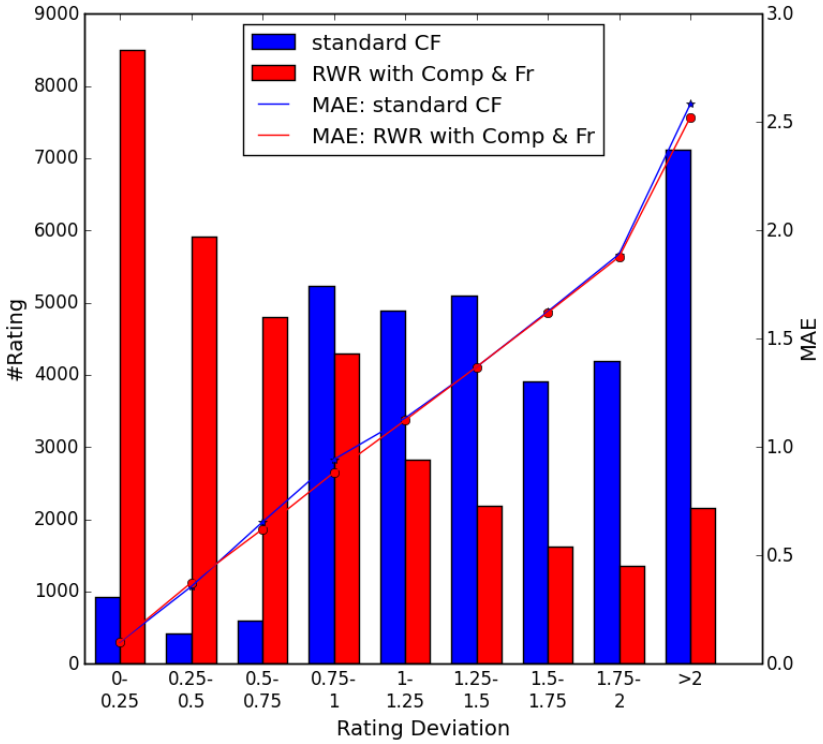


Figure 3.3: Average MAE and rating deviation distribution

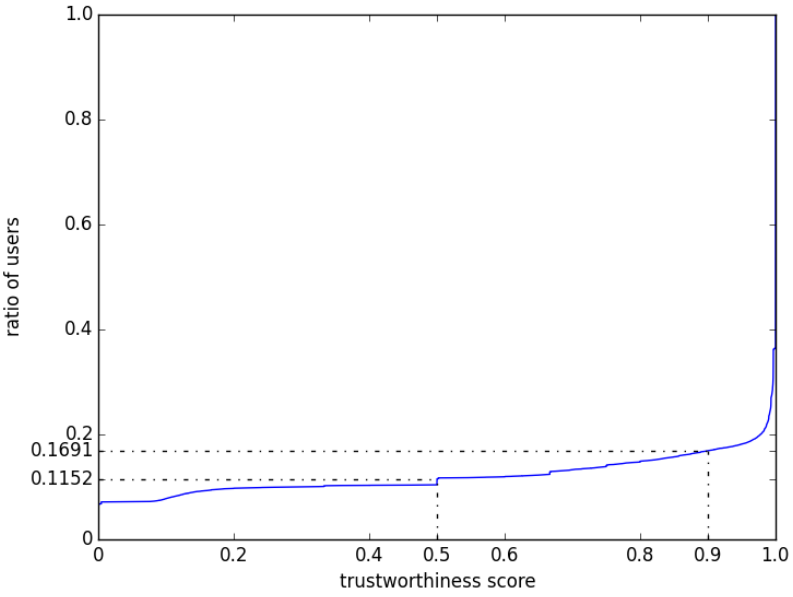
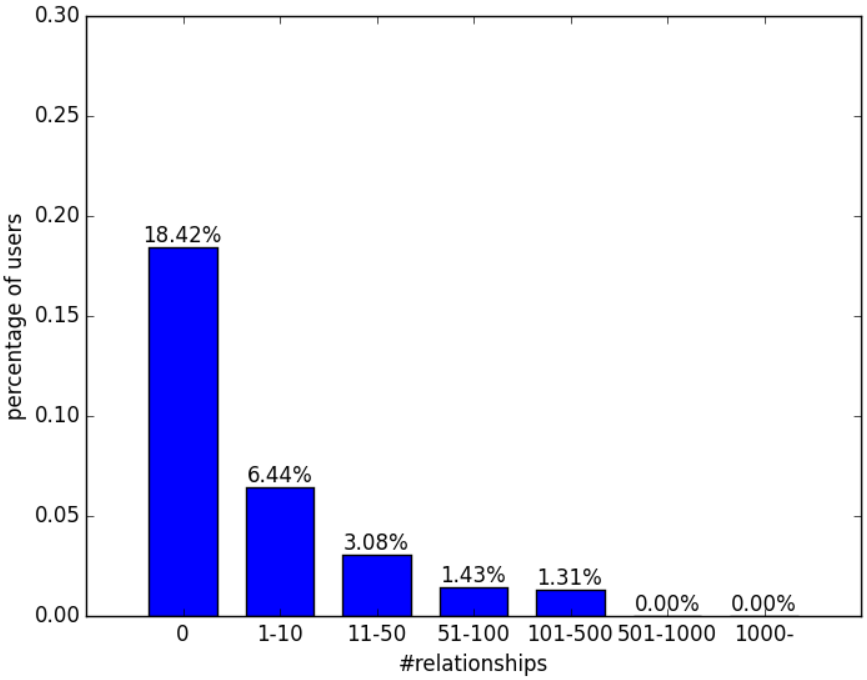


Figure 3.4: CDF plot of the overall trustworthiness score

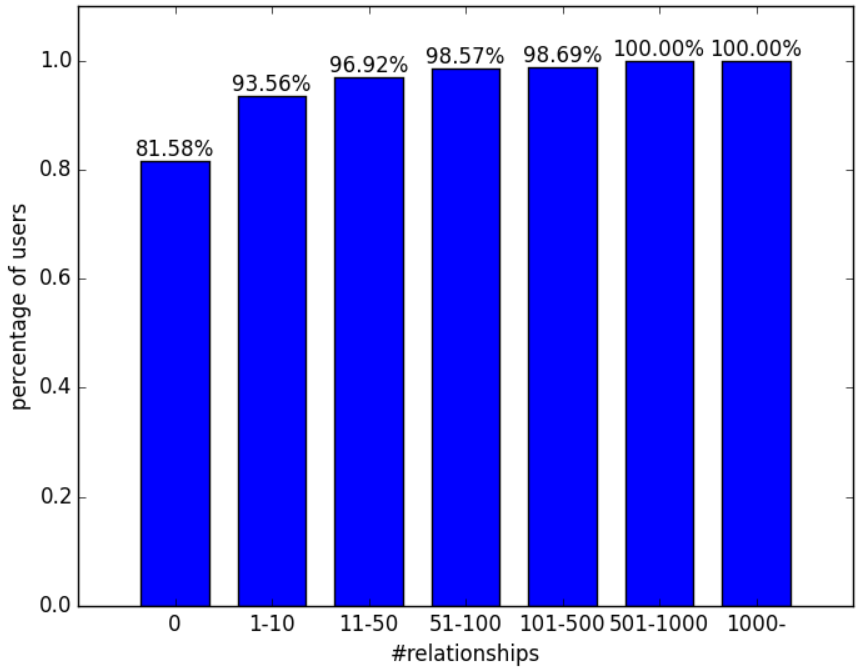
than or equal to $\varepsilon = 0.05$. The deviation threshold Δ is learned from a small set of labeled data using a discretization method, Recursive Minimal Entropy Partitioning (RMEP) [23]. The small dataset was manually labeled by three graduate students independently. The threshold learned was 2.011. The cumulative distribution function (CDF) of the overall trustworthiness scores is shown in Figure 3.4. From the figure, we see that about 11.52% of the users have trustworthiness scores less or equal to 0.5; 16.91% of the users have trustworthiness scores less or equal to 0.9. So about 80% of the users have high trustworthiness scores larger than 0.9. The results seem reasonable since most of the users in the system should be normal users rather than suspicious spammers, no matter how subjective their ratings are.

In order to show the correlation between social relationships and trustworthiness scores, we divide the number of relationships (both friendship and compliment) into seven consecutive ranges in the ascending order. The relationship between the number of relationships and the percentage of users, with trustworthiness scores below or above 0.5 and 0.9, respectively, is shown in Figure 3.5 and Figure 3.6. It is obvious that the ratios of users with high trustworthiness scores increase along with the increase in the number of relation links. It supports the conclusion that users who are more socially-active have higher overall trustworthiness scores.

Evaluations. For evaluating the experiment results, we adopt the idea presented in [65]. We first rank all users based on their trustworthiness scores in descending order and selected the top-40 users and bottom-40 users. Then, we mix all the selected users together so that the results to be evaluated demonstrate no relationship between the order and the trustworthiness scores. All related reviews of the selected users were retrieved from the dataset for evaluation. Two human evaluators were recruited for evaluation. They were all instructed with the background of review spam detection and the evaluation criteria. The content of this task was to read the reviews and manually assign a binary label of whether a user is suspicious or not based on their best judgments. The process was conducted independently between the two evaluators. The agreement among evaluators and results of the model are shown in Table 3.2 and Table 3.3. For example, in Table 3.2, evaluator 1 identified 20 suspicious users out of all 100 users (our model identified 40) and

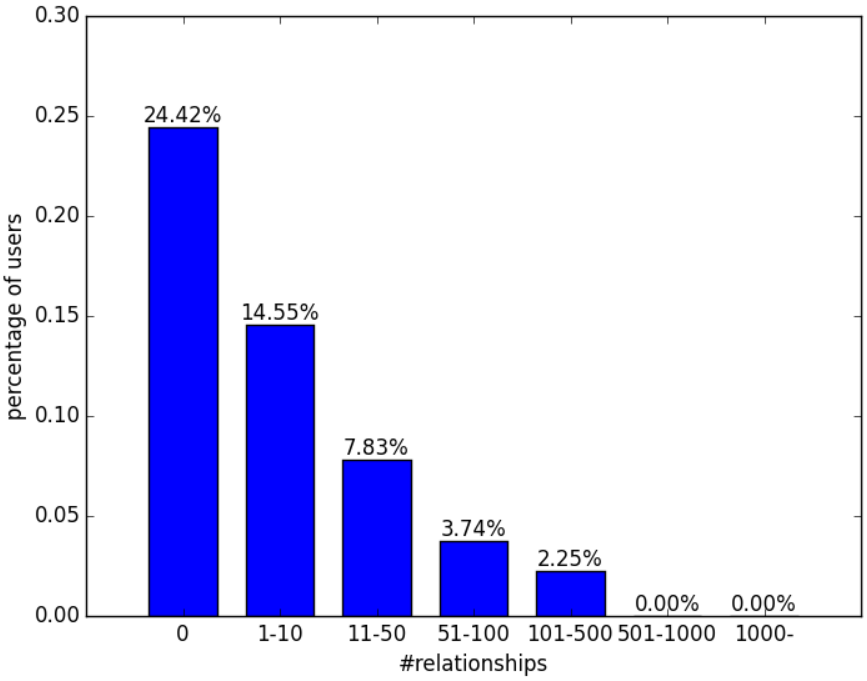


(a) % of users with $t_u < 0.5$

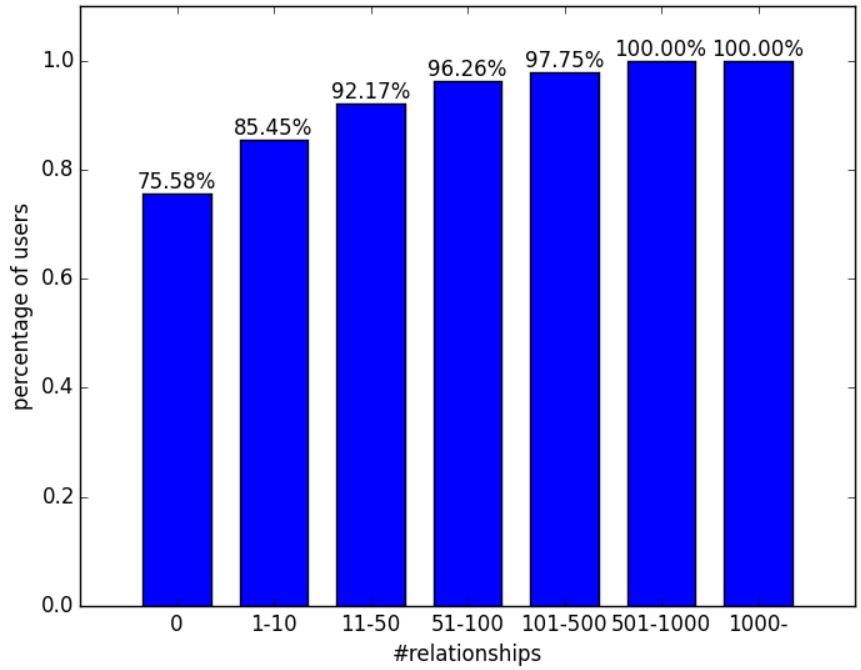


(b) % of users with $t_u > 0.5$

Figure 3.5: The relationship between the overall trustworthiness score and the number of friends when t_u is larger or smaller than 0.5



(a) % of users with $t_u < 0.9$



(b) % of users with $t_u > 0.9$

Figure 3.6: The relationship between the overall trustworthiness score and the number of friends when t_u is larger or smaller than 0.9

overlapping agreement with our model is 20. Results showed that both evaluators detected less suspicious users but more normal users than the model did. The agreement between evaluators was higher than the agreement between the evaluators and the model.

Table 3.2: Human Evaluation Agreement on Suspicious Users

	Evaluator 1	Evaluator 2	Model
Evaluator 1	20	17	20
Evaluator 2	-	23	23
Model	-	-	40

Table 3.3: Human Evaluation Agreement on Normal Users

	Evaluator 1	Evaluator 2	Model
Evaluator 1	60	54	40
Evaluator 2	-	57	40
Model	-	-	40

3.5 Discussions

Our experiment results show that the standard CF method is not as effective as expected, with various possible reasons. First, one of the weaknesses of CF is that it is not attack-resistant [76, 78]. When review spammers post fake reviews on the system, the ratings of fake reviews significantly affect the overall rating of the target items and mislead other users. As a result, it is difficult for the CF model to achieve the expected accuracy. Moreover, these ratings deviate largely from the majority. For their aspect, most of the users have a negative similarity with them. Therefore, the rating predicted for them is not accurate, which further affects the overall performance of the CF model. Our model, on the other hand, works under the assumption that review spammers tend to be socially inactive. Many of them would be isolated or barely connected with other users in the system. Our prediction model only aggregates the ratings from trusted users, which potentially filters out the influence of spammers.

The second possible reason is that CF cannot address cold-start users, which are users just joining the system with little review history. Relying on similarities to make predictions, CF is

not effective for cold-start users because the similarities for them are not reliable. However, on Yelp.com, users are notified when their friends on Facebook or other social networks are registered. As a result, cold-start users without much review histories may have many social relationships to support our model. In other approaches of spam detection, data of these users are typically removed from datasets since it would be difficult to judge whether they are spammers or not. However, our model can still effectively take them into account as long as social relationships exist.

Interestingly in our experiment results, some users with no social relationship still achieved a high trustworthiness score. This is because we assume spammers have less social relationships, but not vice versa. It does not necessarily mean that users with limited social relationships are suspicious. If a socially-lazy user whose opinions on different items always agree with the majority, he should be considered as not suspicious. While other models either remove these cases from their detection or perform poorly, our model detects the socially-lazy user with high accuracy.

3.6 Summary

In this chapter, we study the problem of measuring users' trustworthiness using contextual social relationships that are available in several online review systems. We first present a trust-based rating prediction algorithm using local proximity derived from social relationships, such as friendships and complements relationships, using the random walk with restart. We then incorporate the predicted ratings into a pseudo user-item matrix to overcome the sparsity problem and compute the overall trustworthiness score for every user in the system, which is used as the spamicity indicator. Experiments on the collected Yelp dataset show that the proposed trust-based prediction achieves higher accuracy than standard CF method. Results also show a strong correlation between social relationships and the computed trustworthiness scores.

Chapter 4

Content-based Detection through Three-Layer Trust Propagation

4.1 Introduction

In the previous chapter, we present our work of detecting review spammers using social relations and rating deviations. Compared to numerical ratings, the content of a review obviously contains more details and thus reflects the user's opinion about the target business better. Therefore, studying the content of the reviews definitely could help with the detection of review spammers.

However, utilizing the review content could be tough. The content of spam reviews does not contain clear clues about email spamming, such as a malicious URL or a malicious attachment. Therefore, it is difficult to establish the "ground truth" that distinguishes spam reviews from normal reviews [132, 133, 86, 96]. Some studies recruited people from Amazon Mechanical Turk to create synthetic spam reviews. However, these approaches were criticized because the quality of the tasks completed on crowd-sourcing platforms is often lower than expected and the selected Turkers' behavior may not resemble spammers'.

Many online review systems adopt deterrence-based or reputation-based detection approaches to measure the quality of the reviews. For example, the "Verified Purchase" mechanism of *Amazon.com* allows only customers who actually made the purchase to post reviews. A more generally adopted approach is the "review of the review", which allows users to rate a review or vote for its "helpfulness", and uses the ratings or vote counts as a measure to assess review quality. While these mechanisms provide additional information about how helpful or trustworthy a review is,

their limitations are also obvious. Similar to reviews, the "review of review" is a subjective judgment and itself can also be faked. Moreover, it often suffers from inadequate user participation. Surveys show that only a small portion of users provides reviews online. Furthermore, among thousands of views of a review, only a few readers provide feedbacks. As a result, a large amount of reviews has no helpfulness or usefulness rating at all.

On the other hand, detection-based mechanisms are considered more suitable since spamming activities often demonstrate certain patterns. Many learning-based schemes have been proposed to identify deceptive reviews and reviewers from textual features [45, 96, 88], temporal features [24, 145], spammers' individual or group behavior patterns [133, 88, 63], and sentiment inconsistency [90]. The rationale behind these approaches is two-fold. Along the first direction, detection models rely on the deviation in rating behaviors. Since the objective of opinion spammers is to alter users' perception of the quality of the target, they often generate a large number of reviews with extreme ratings using different accounts. In this way, spammers can significantly distort the mean rating. So, the detection focuses on rating-based features that reflect the deviation from the aggregated rating (or the majority view) [65, 2] and other rating behaviors (e.g., change of average rating over time, change of average ratings across groups of users, etc.). While these approaches have been used with success, they can be easily gamed by spammers who evolve to avoid extreme rating behavior. Along the second direction, detection schemes rely on text features, such as duplicated texts [45, 88], psycholinguistic deceptive characteristics [96], etc. These approaches use spam reviews that are manually identified or created to train the classifiers, which introduce expensive labeling costs. To make things worse, automated spam review generators based on deep learning have been proposed recently [152], which leverage text features used in spam detection for spam generation.

In this chapter, we propose to use the inconsistency in opinions expressed in one's review and the consensus opinions shared by a group of anonymous users to determine the trustworthiness of a review and its reviewer. This is inspired by the approaches that identify opinion spams whose ratings are inconsistent with the opinions expressed in the review text [65, 90]. Moreover, we target

spams that are carefully crafted to evade existing detection mechanisms, rather than the ones with simple duplication or obvious deceptive cues. An important assumption in our approach is that in a healthy review system, benign users should always outnumber spammers, which is also the assumption of any reputation-based systems. Therefore, the aggregated opinion from a majority vote should reflect the actual quality of service/product on every aspect.

Definition 2. An **aspect** in a review refers to the term that expresses a ratable feature of the target entity. The group of aspects with the same semantic concept is called a **aspect category**.

Although the majority views have some limitations in extreme cases, such as inertia against sudden change of quality, we assume that in most cases the majority view reflects the facts effectively.

Our scheme takes multiple deviation indicators into consideration and integrates the deviations across all aspects to obtain an overall deviation. In particular, we present two approaches to extract aspect-specific opinions. In the first approach, we trained a classifier with a small labeled dataset using supervised learning. While achieving high accuracy, it is costly due to data labeling and rigid as the aspect categories need to be pre-determined. To overcome the limitations, we developed an unsupervised approach to extract aspects from review content. To effectively integrate the aspect-specific indicators, we adopt a three-layer trust propagation framework, which was first proposed in [130], to calculate trust scores for users, reviews, and the statements.

Definition 3. In our model, we define a **statement** as the aspect category-specific opinion of a specific target.

Using the extracted opinions as input, the three-layer trust propagation framework iteratively computes the trust scores and propagates them among users, reviews, and statements. The converged trust score reflects the overall deviation of a user from the aggregated opinion across all the aspects and all the reviewed targets, which we believe is a strong indicator of trust to distinguish regular users and spammers.

The contributions of this chapter can be summarized in three aspects:

1. We propose a novel aspect category-specific opinion indicator as a content-based measure to quantify the quality and trustworthiness of review content. We focus on detecting carefully composed opinion-wise deviated spam reviews, rather than typical spam reviews that can be easily detecting using existing behavior-based or rating-based methods.

2. We propose a three-layer trust propagation model based on the intertwined relationships between three types of nodes, users, reviews, and statements. Compared to other link-based approaches [133] that utilized behavioral features such as rating deviation, our system takes opinions of multiple aspect categories extracted from review content into account, which captures the opinions expressed by users directly.

3. We develop an effective iterative computational framework for three concepts that model the level of trustworthiness of the three types of nodes, namely, the honesty of users, the faithfulness of reviews, and the truthfulness of statements. Compared to iterative frameworks like [133, 130], our framework is not only based on the reinforcements between the three concepts but also based on the level of consensus between individual opinion and aggregated opinion.

4.2 Overview Problem and Our Solution

In this section, we first explain the review spamming problem with opinions. Then we present the overview of our proposed content-based trust propagation framework.

4.2.1 Problem Overview

Rating deviation is widely used in previous review spamming detection approaches. It is a strong indicator of spamming since the primary goal of the spammers is to promote or demote the target entity by increasing or decreasing its average rating dramatically. However, rating deviation-based detection scheme can be evaded. For example, the spammers can regulate their behavior by avoiding inserting extreme ratings within a short period to change their temporal rating patterns. To

tackle this problem, we propose an opinion deviation-based trust indicator as a new detection feature, which is robust to detection evasion.

Obviously, in an online review, the opinion of users is expressed not only in the rating of the review but also in the review content. Some previous approaches compare the sentiment extracted from the content of a review with its rating to look for inconsistent opinions and use it to detect spam reviews. However, since conflicting opinions may be expressed in the reviews, these approaches are limited due to the poor performance of sentiment analysis especially when conflicts exist.

4.2.2 Overview of Our Solution

For a specific review, only the words that contain expressions of opinions are valuable for analysis. Therefore, in this work, we propose to treat a review as a set of ratable aspects with corresponding sentiments. Since conflicting opinions in a review often regard different aspects of the target entity, by dividing the review into sets of words regarding different aspects, our approach can effectively address this problem. To do so, we first conduct opinion mining on the content of each review to extract the opinions on a set of aspects. In particular, we propose a supervised learning based approach and an unsupervised-learning based approach to extract opinions across a set of aspects from a dataset of reviews collected from Yelp.com. Since the extracted opinions are on multiple aspects, we can group opinions from different users on the same aspect of the target and compute the aggregated opinion, which highly likely reflects the "fact" (e.g., the actual quality of the reviewed entity on this aspect). Therefore, the deviation of a user's opinion from the aggregated opinion may reflect her trustworthiness.

To measure the influences on users' trustworthiness due to opinion deviation, we propose a three-layer trust propagation framework to compute the overall influences across multiple entities and aspects. The higher the score is in the final output, the more trustworthy an entity is (user, review, or statement), the details are discussed in Section 4.4. As shown in Figure 4.1, u_i represents a user, r_i represents a review, and e_i represents an entity (e.g., a restaurant). For ease of

Table 4.1: Notations of terms and concepts.

Notation	Definition
u	a user
r	a review
s	a statement
e	an entity
a_i	the i-th aspect word
ac_i	the i-th aspect category
l_i	the i-th sentiment label
$ao_{u,e}$	the aggregated opinion of user u to entity e
o	an opinion vector
os	an opinion status vector
$h^{(n)}(u_i)$	honesty score of user u_i in round n
$f^{(n)}(r_i)$	faithfulness score of review r_i in round n
$t^{(n)}(s_i)$	truthfulness score of statement s_i in round n
$\delta^{(n)}(u_i)$	opinion-based deviation of user u_i in round n
$\xi(u_i, r_j)$	the confidence of u_i on r_i about the target entity
$\rho(r_i, s_i)$	the relevant of r_i to s_i

presentation, we list the notations of terms and their meanings in Table 4.1. Given an application domain, users may be interested in only a few abstract *aspects categories* about the targets. For example, in restaurant reviews, users are more interested in the food flavor and quality, price, service, atmosphere, etc. In product reviews, users more concern about quality, lifetime, price, etc. Each aspect category may consist of more specific aspects, which can be directly extracted by our opinion mining algorithm. We use $ac_{i,j}$ to represent the j-th aspect category of entity e_i. Then, we construct *statements* for each entity. Each statement is an opinion on a particular aspect category of the entity, for example, "restaurant1-food-positive" is a statement that expresses the "positive" opinion toward the aspect category "food" of entity "restaurant1".

4.3 Aspect Category Specific Opinion Indicator

Existing content-based detection approaches take textual content of a review as input, which often uses word-level features (e.g., n-grams) and known lexicons (e.g., WordNet[83] or psycholinguistic lexicon [65]) to learn classifiers that identify a review as spam or non-spam. To train the classifier,

costly and time-consuming manually labeling of reviews is required. Due to the subjectiveness of human judgment and personal preferences, there is no readily available ground truth of opinions. Therefore, a high-quality labeled dataset is difficult to obtain. Some existing work adopts crowdsourcing platforms such as Amazon Mechanical Turk to recruit human labeler, however, it is pointed out the quality of the labeled data is very poor. Different from these approaches, our opinion spam detection scheme utilizes the deviation of the majority opinion. Although biased opinions always exist in UGC, we argue that a majority of users may be *biased but honest*, instead of maliciously deceptive. This is based on an overarching assumption regarding reviewer behaviors – that is the majority of reviews are posted by honest reviewers, as recognized by many existing researches on opinion spam detection [65, 90, 2]. A news report [91] states that it is estimated that among online hotel reviews, between 1% to 6% of positive reviews are fake. A recent study shows that about 16% of the restaurant reviews on Yelp are filtered about Yelp's filter [72]. The fact that the functionality of the online review systems depends on the well-being of systems themselves makes it barely possible for spammers to dominate the review systems. We argue that if this assumption does not hold, online peer review systems will be completely broken and useless. Furthermore, hiring a huge number of spammers to dominate the opinions of the review systems is very costly and infeasible. As a result, we propose to use the majority opinions as the "ground truth".

4.3.1 Aspect-based Opinion Extraction

Existing work [80, 87, 99, 39, 31] on opinion mining studies opinions and sentiments expressed in review text at document, sentence or word/phrase levels. Typically, the overall sentiment or subjectiveness of a review (document-level) or a sentence of a review is classified and used as a text-based feature in spam detection. However, we consider these opinions are either too coarse or too fine-grained. For example, opposite opinions are commonly expressed in an individual review. It may be positive about one aspect of the target entity but negative about another. This is difficult to capture using the document-level sentiment analysis. Therefore, the derived review-level majority

opinion is inaccurate and problematic.

Another direction of approaches proposes to use opinion features that associate opinions expressed in a review with specific aspects of the target entity [36, 90]. Intuitively, these opinion features are nouns or noun phrases that typically are the subjects or objects of a review sentence.

Example 2. In the review below, the underlined words/phrases can be extracted as opinion features.

"*This place is the bomb for milkshakes, ice cream sundaes, etc. Onion rings, fries, and all other "basics" are also fantastic. Tuna melt is great, so are the burgers. Classic old school diner ambiance. Service is friendly and fast. Definitely come here if you are in the area ...*"

In some cases, users may comment on a large number of specific aspects about the target entity. The derived opinion features are too specific and too fine-grained to form a majority opinion on each feature, since other reviews about the same target may not comment on these specific features. However, from the above example, we can see that opinion features such as "milkshakes", "fries", and "burgers" are all related to an aspect category "food". If we define a set of aspect categories, opinion features about a same or a similar high-level concept can be grouped together.

Consider a set of reviews (**R**), which are written by a group of users (**U**) about a set of entities (**E**). Each review $r \in \mathbf{R}$ consists of a sequence of words $\{w_1, w_2, ..., w_{n_r}\}$. Then, we can define a set of m aspect categories $ac = \{ac_1, ac_2, ...ac_m\}$, each aspect category ac_i covers a set of aspects a_i that represent the same concept. For each aspect, correspondingly there is a sentiment polarity label $l = \{l_1, l_2, ...l_k\}$. As a result, for each review r, we can extract a set of aspect-sentiment tuple $< a_i, l_i >$. By aggregating the tuples that belong to the same aspect category together, we get the the aggregated opinion for each aspect category $< ac_i, l_i >$. Combining aggregated opinions for all commented aspect category, a final aggregated opinion $ao_{u,e} = \{< ac_i, l_i >\}$ is used to represent the aspect category-specific opinions of a user u towards a target entity e.

Typical sentiment polarity labels include "positive", "negative", "neutral", and "conflict" [31, 104]. The "conflict" label captures inconsistencies within a review but does not provide any information about the inter-review consistency, therefore we do not consider this label in our model.

In this work, we use Yelp reviews as our dataset to study the credibility of users and their reviews. We present two approaches for opinion extraction, supervised learning based and unsupervised learning based. We first started with supervised learning methods as it is fast and simple. With data labeled with aspect categories and corresponding sentiment, identifying the aspect-sentiment tuples is quite straightforward. However, its limitations are also obvious. First, getting data labeled requires human labor and is time-consuming. Second, the number and type of categories are predefined due to the labeled data and the type of data can be applied is also limited. The effort of adding a new category is often costly as it usually requires re-labeling the data. On the other hand, with unsupervised learning, we gain the flexibility of identifying opinions expressed in more aspect categories and the entire opinion mining process can be automated with minimal human intervention. Although the unsupervised method may cause some loss of accuracy in terms of aspect category compared to the supervised method, we can tolerate some inaccurate classifications as our proposed framework doesn't rely on output from a single category. Note the outputs of these two types of methods are different, thus the results of the two approaches cannot be combined.

4.3.2 Supervised Opinion Extraction

The study in [31] identifies six categories for restaurant reviews, *food, price, service, ambience, anecdotes* and *miscellaneous* for review classification. We followed up the idea and define a small set of aspect categories including four meaningful aspects *food, price, service*, and *ambience* for restaurant reviews. We treat the opinion extraction as a classification problem and adopt the Support Vector Machine (SVM) supervised learning model for opinion extraction. We use the SemEval dataset [104], which is a decent-sized set of labeled data for restaurant reviews, to train our classifier (see more details in Section 4.5). Our goal is to identify an adequate number of aspects that are commonly expressed in reviews so that we can construct a credibility indicator from the aggregated opinions. In fact, too many over-specific aspects complicate the credibility computing model instead of improving it.

Example 3. In the review below, the words with underline are aspects that belong to the category "food".

"*This place is the bomb for milkshakes, ice cream sundaes, etc. Onion rings, fries, and all other "basics" are also fantastic. Tuna melt is great, so are the burgers. Classic old school diner ambiance. Service is friendly and fast. Definitely come here if you are in the area ...*"

If we consider every single ratable aspect word as an aspect category, there will exist too many categories. It is helpful to construct the aspect-specific indicators if we consider aspects that represent the same concept separately. Therefore, we mainly focus on the four high-level aspect categories rather than more specific aspect categories.

Besides the four major categories, we combine all other aspect category labels as a fifth category *miscellaneous*. Also, using *miscellaneous* also helps improve the quality of classifications of the first four categories (i.e., *food, price, service* and *ambience*) as the classifier will not misclassify some irrelevant aspects into those four categories. We first applied the trained classifier on reviews with sentence-level to identify the aspect categories each sentence is about. Note it is possible that a sentence is about multiple aspect categories. For example, the sentence *The burger here is pretty good but the price is a bit expensive* talks about two aspect categories, *food* and *price*.

Next, we conduct the aspect-specific sentiment classification upon the classified aspect-specific sentences to obtain aspect-specific sentiment polarities. For each category, the classification is conducted independently. For example, to determine the sentiment polarities of the "food" category, we conduct a sentiment classification upon all sentences that have been classified into the category "food", and determine the aspect-sentiment tuples: "food-positive", "food-negative", and "food-neutral".

4.3.3 Unsupervised Opinion Extraction

The advantages of applying supervised learning methods for opinion extraction is fast and simple. With data labeled with aspect categories and corresponding sentiment, identifying the aspect-sentiment tuples is quite straightforward. However, its limitations are also obvious. First, getting

data labeled requires human labor and is time-consuming. Second, the number and type of categories are predefined due to the labeled data and the type of data can be applied is also limited. The effort of adding a new category is often costly as it usually requires re-labeling the data. With unsupervised learning, we gain the flexibility of identifying opinions expressed in more aspect categories and the entire opinion mining process can be automated with minimal human intervention. However, abstract aspect categories are more difficult to define since they are domain-specific and thus need to be carefully tuned for a given domain.

While the supervised approach can directly identify the aspect categories from a sentence or a review, the approach using unsupervised learning consists of several steps.

Opinion Extraction with Dependency Relations. Rather than directly assigning classification labels to a review, the aspect-specific opinions need to be explicitly identified. As mentioned earlier, the objects (i.e., the aspect a_i in the tuple) of an opinion feature are usually expressed as nouns or noun phrases. On the other hand, the modifiers (i.e., the sentiment label l_i in the tuple) are usually expressed as adjectives, adverbs, or verbs with sentiment orientations. Thus, the aspect-sentiment tuple can be identified to search for certain patterns of POS tags. However, the object and corresponding modifier may not necessarily occur close to each other in a sentence. Manually defining POS tag patterns can be costly and inaccurate. Thus, in this work, we used dependency relations [81] to parse and extract the qualified expressions.

A dependency relation is an asymmetric binary relationship between a term called head or governor and another term called modifier or dependent [81]. As suggested in [143, 149], we decided to use three types of dependencies, including "nsubj", "amod", and "dobj", in the opinion extraction process, denoting subject-predicate relations, adjectival modifying relations, and verb-object relations, respectively.

Aspect Categorization. Unlike the supervised method, labels of aspect categories are directly assigned by the classifier, the unsupervised method requires to group the extracted aspects that are semantically similar to aspect categories. There are many ontology-based lexical tools like WordNet [83] provides the synonyms of words. However, the limitation of these tools is that

the concept-based synonyms provided are predefined and fixed. Also, a word usually has more than one "senses" (treated as polysemy), calculating semantic similarity needs to find the correct sense manually, which cannot be processed in an automatic manner. Another difficulty is that the semantic similarities sometimes only exist in a certain context, for example, "steak" and "egg" are semantically similar only in the context of restaurant reviews. Without the context, "steak" is more related with meat while "egg" is more related with "bird". In online reviews, this is often the case with ratable aspects. Thus, the semantic similarity cannot be acquired from some predefined tools but needs to be learned directly from the reviews. Topic modeling approaches are able to group words in topics based on textual data, but the topics are usually not coherent enough to be used as aspect categories.

In the field of Natural Language Processing (NLP), the existing work that studies the semantic representation of words usually work under the assumption that words occurring within similar contexts are semantically similar [37]. Vector-based models have been successful in representing the semantic relationships among words. Conceptually, mapping words or phrases to vectors are known as word embeddings. Models such as word2vec [82] and GloVe [103] have proved their effectiveness and outperformed simple models in many NLP tasks.

In this work, we used the word2vec model to learn the word embeddings of all words occurring in the reviews. With vector representations of aspects learned from the review, it is much easier to compute the similarities. Intuitively, aspects that occur in similar context will have similar embeddings. To group aspects into categories, we applied K-means, a widely-used clustering algorithm is able to partition the data into k clusters based on feature similarity. We used the learned word embeddings of all aspects as the features for K-means as the word embeddings capture the semantic similarities well enough. The output clusters represent the aspect categories learned based on the semantic similarities captured from the reviews.

Sentiment Orientation of Modifiers. After aspects categorized, the next task is to assign a sentiment label (i.e., l_i) to the corresponding modifiers extracted from the dependency relations. Although we also obtained the word embeddings of the modifiers using word2vec, we cannot apply a

clustering algorithm to group them into different sentiment labels. When word2vec is trained, the sentiment orientation is not embedded in the training process, thus the obtained word vectors do not contain information about the sentiment polarity. Actually, modifiers with opposite sentiment polarities will both co-occur with certain target words frequently. For example, "expensive" and "cheap" are of opposite sentiment polarities in terms of price. However, they may both occur in a similar context. As a result, the similarity between their word vectors is high. If we apply K-means on the word embeddings of the modifiers, "expensive" and "cheap" will be grouped in the same cluster even though they have the opposite sentiment polarity for price.

Instead of utilizing the word embeddings with K-means, we used a widely-adopted opinion lexicon [39] to identify positive and negative modifiers. The lexicon contains two lists of words, one with 2006 positive words and the other with 4,783 negative words. For each extracted modifiers, if it is included in one of the lists, we assign the corresponding sentiment label to it. If the modifier is not included in either of the lists, we ignore it. Note there is no list of words for the sentiment orientation "neutral", we will only have two types of sentiment labels in this unsupervised-based setting, "positive" and "negative".

4.3.4 Opinion Vector and Quality Vector

With opinion mining approaches, we can focus on the the extracted opinions and ignore the rest of the words in the review. However, the opinions are still expressed in words, which is difficult to be used in computation. Thus, to utilize the extracted opinions for further analysis, we quantify the extracted opinions as an opinion vector.

Definition 4. An **opinion vector** $\mathbf{o} = [o_1, ..., o_n]$ is a vector that captures the aspect category-specific opinions and their corresponding sentiment polarities. n is the number of predefined aspect categories.

Sentiment polarities are represented by element values, where a positive sentiment is denoted by "+1", a negative sentiment is denoted by "-1", and neutral is denoted by "0" (if provided). Since

a statement may not necessary to express an opinion about an aspect, we distinguish no opinion expressed from a neutral opinion by defining a corresponding opinion status vector **os** to record the observed status of each aspect category-specific opinion.

With the opinion vectors, we can aggregate the opinions on multiple aspects from all reviewers of an entity to form four aspect-specific aggregated opinions. While aspect-specific opinions are subject judgments and thus can be biased, the aggregated aspect-specific sentiments are highly likely to reflect the true quality of the entity from a specific aspect. This is because individual biases are typically smaller aspect level than at document-level, which is more affected by the weights subjectively assigned by individuals to multiple aspects. In this sense, aspect-level bias can be corrected by the majority view if the review amount is adequate. Furthermore, comparing with rating, aspect category-specific opinions are more difficult to be tampered by opinion spammers, whose review text are likely to be pointless, wrongly focused, or brief. Finally, the aggregated sentiments are robust to correct the inaccuracy introduced by opinion mining models. Opinion mining often suffers from precision problems, but our goal is to decide if the overall aspect-specific opinion is positive, neutral, or negative. Although each individual input incurs a small uncertainty, the chance to affect the overall value is very small. Based on these considerations, we derive the aggregated aspect category-specific opinion vectors as $\mathbf{o_{agg}} = [o_{agg_1}, ..., o_{agg_5}]$ and $\mathbf{os_{agg}} = [os_{agg_1}, ..., os_{agg_5}]$, where

$$o_{agg_i} = \begin{cases} 1, & avg_{i \in A_i}(o_i) > \theta_p \\ -1, & avg_{i \in A_i}(o_i) < \theta_n \\ 0, & \text{otherwise.} \end{cases} \tag{4.1}$$

In this way, the aggregated sentiment polarity of each aspect is mapped to the positive, neutral (not included with opinions extracted using unsupervised method), and negative labels based on the averages. The aggregated aspect category-specific opinion vector is considered a *quality vector*.

Example 4. The **quality vector** is the aggregation of the opinion vectors of a specific target.

The quality vector can be used to determine the credibility of a user statement. Intuitively, a statement is more credible and of higher quality, if it expresses a consistent opinion with the aggregated opinion about one aspect of the target entity, and thus the reviewer is considered more honest and trustworthy.

4.4 Content-based Trust Computation

We compute the aggregated aspect-specific opinion vector as a quality measure and use individual aspect-specific opinion vector as a credibility (or trust) measure. To integrate trust measures across multiple users and multiple entities, trust propagation models are commonly used [154, 130]. In this work, we model the relationships and inter-dependencies between user, review, and the entity being reviewed, and adopt a three-layer trust propagation model to compute the trust-related scores for users, reviews, and aspect-specific statements iteratively.

4.4.1 The Three-Layer Trust Relationships

The three-layer trust propagation model was first introduced in [130] to measure the trustworthiness of online claims and their sources, especially when conflicting information is provided by multiple sources. Traditionally, this problem was modeled as a trust propagation problem using the bipartite graph consisting of *sources* and *evidences* that support a same claim. In particular, the trustworthiness of a source relies on the confidence of all the evidences that it provides to support its claims, and the confidence of a claim depends on the trustworthiness of all sources that provide evidences to it. Different from the bipartite graph-based two-layer models, the three-layer model introduces an additional intermediate layer to represent the influence on one evidence due to another evidence of the same claim. As explained in [130], the inter-evidence interactions can be used to model the similarity between two evidences so that similar evidences receive similar trustworthiness scores.

We adopt the three-layer architecture to model the relationships among three types of nodes

(i.e., *users*, *reviews*, and *statements*) in our review system, in which the inter-dependencies between nodes and on the links are completely different from the ones in [130] and thus need to be re-defined. The key idea of adopting the three-layer architecture is to use the intermediate layer to model the inter-review interactions and the influence on one review due to the other reviews about the same entity (or more precisely the same aspect of that entity).

Nodes. As shown in Figure 4.1, we denote a *user*, a *review* of a user, and a *statement* about a target entity as u_i, r_j, and s_k, respectively. Here, the statement is defined as an opinion expressed on a particular aspect of the target entity. For example, a statement $restaurant_1 - food - positive$ expresses a positive opinion about the food of $restaurant_1$.

Definition 5. We define three types of trust scores for the three types of nodes respectively. The **honesty** score, denoted as $h(u_i)$, is assigned to each user node. The **faithfulness** score, denoted as $f(r_i)$, is assigned to each review node. The **truthfulness** score, denoted as $t(s_i)$, is assigned to each statement node. The three types of trust scores take value between 0 and 1.

Edges. The edge from a user node to a review node means the user posts this review, and the edge from a review node to a statement node represents the review supports this statement. As shown in Figure 4.1, a user can post one review regarding multiple statements of the target entity (e.g., u_2 posts r_3 on s_2 and s_4 of entity e_2) or multiple reviews about multiple entities (e.g., u_1 posts r_1 about e_1 and r_2 about e_2), and multiple users can post reviews on the same aspect of a target entity (e.g., u_1 and u_2 post r_2 and r_3 on s_2 of e_2, respectively). Finally, we use the double-arrow lines to denote the inter-review influences on one review due to other reviews.

Interactions on Edges. There are four main types of interactions in our three-layer trust propagation model: (1) The user-review edges represent the confidence of the user on the review about the target entity, denoted as $\xi(u_i, r_j)$. Therefore, the faithfulness of a review can be calculated as $f(r_i) = h(u_i)\xi(u_i, r_j)$, where $\xi(u_i, r_j)$ is normalized by the maximal $\xi(u_i, r_j)$ of all reviews posted by u_i. It measures the relative reliability of a particular review among all the reviews posted by a same user who reviews multiple different entities. (2) The review-statement edges represent the relevance of a review to a statement of the target entity, denoted as $\rho(r_i, s_i)$. Opinions expressed in

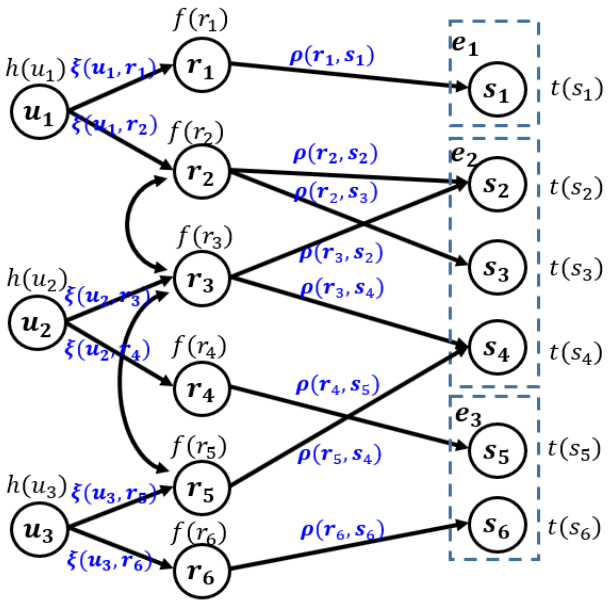


Figure 4.1: The three-layer trust propagation model.

all reviews about a target entity regard multiple aspects of the entity. Given a statement s_i about a particular aspect of the target, $\rho(r_i, s_i)$ measures how relevant the review r_i is about the statement and how strong the sentiment expressed in the review support the statement. (3) The review-review edges in the inter-user context group reviews from different users on the same aspect of a target entity. This leads to a voting strategy to aggregate specific opinions, in terms of positive, negative and neutral, on each aspect of the reviewed entity. Thus, the deviation between the individual opinion and the aggregated opinion provides useful information to distinguish normal users and spammers. This is because benign users may express different opinions in some cases due to individual experiences or subjectivity, only spammers continuously express an opposite opinion against the fact. Therefore, in the inter-user context, the influences on a review due to other reviews about the same statement of the target can be measured by its opinion deviation.

4.4.2 Trust Propagation and Trust Scores

As described in Section 4.4.1, three types of trust scores are defined for user, review, and statement nodes. Similar to the approach presented in [130], the three trust scores can be computed iteratively

over the trust framework following the interactions between the three types of nodes.

Basic Trust Propagation Model. In the basic model, we consider only the influences on the trust score of each node due to trust scores of other connected nodes. We start with the *faithfulness* score for the review, which represents the confidence in the trustworthiness of the review. Initially, the *faithfulness* score of review r_i is $f^{(0)}(r_i)$. It can be set to 1, denoting an equal confidence in the trustworthiness of all reviews, or estimated by other spamming detection algorithms based on content, structure, or behavior-based features. We then update the faithfulness scores for all reviews following the below equation:

$$f^{(n+1)}(r_i) = \mu f^{(n)}(r_i) + (1-\mu)h^{(n)}(u(r_i)) \tag{4.2}$$

where $\mu \in (0,1)$ is an interpolation co-efficient that controls the bias of prior knowledge of $f(r)$ on future estimates of the review faithfulness.

Similarly, the *truthfulness* score of a statement can be calculate as:

$$t^{(n+1)}(s_i) = \frac{\sum_{r_j \in \mathcal{R}(s_i)} [f^{(n)}(r_j) \times h^{(n)}(u(r_j))]}{|\mathcal{R}(s_i)|} \tag{4.3}$$

where $\mathcal{R}(s_i)$ is the collection of all the reviews about statement s_i (on a particular aspect of the target entity). From equation 4.3, we see that the truthfulness score of a statement s_i is relevant to the average trustworthiness of all the reviews regarding the aspect expressed in the statement, and the honesty scores of the users whose reviews are relevant to this statement.

Finally, the *honesty* score of a user depends on the average trustworthiness of all the statements that he supports. Therefore, it can be calculate as:

$$h^{(n+1)}(u_i) = \frac{\sum_{s_j \in \mathcal{S}(u_i)} t^{(n)}(s_j)}{|\mathcal{S}(u_i)|} \tag{4.4}$$

where $\mathcal{S}(u_i)$ is the set of statements about which the user u_i posts reviews.

From the equations above, it is obvious that three types of trust scores are influenced by each

other following the edges that connect them and thus can be computed iteratively from an initial setting.

Enhanced Model with User-Review and Review-Statement Interactions. We improve the basic trust propagation model by integrating the confidence and relevance factors onto the user-review and review-statement edges, respectively. In particular, we update equations 4.2 and 4.3 as bellow:

$$f^{(n+1)}(r_i) = \mu f^{(n)}(r_i) + (1-\mu)[h^{(n)}(u(r_i)) \times \xi(u_i, r_j)] \tag{4.5}$$

$$t^{(n+1)}(s_i) = \frac{\sum_{r_j \in \mathcal{R}(s_i)} [f^{(n)}(r_j) \times h^{(n)}(u(r_j)) \times \rho(r_j, s_i)]}{|\mathcal{R}(s_i)|} \tag{4.6}$$

In the basic model, all reviews from the same user can be considered equally reliable. However, reviews posted by the same user can have different $\xi(u_i, r_j)$, if we consider factors such as review length or the number of aspects covered in the review. This allows us to integrate detection features adopted in other approaches into the trust propagation framework. If we assume all reviews are written by users with unified confidence, $\xi(u_i, r_j)$ can be set to a constant, e.g., 1. In general, $\rho(r_j, s_i)$ measures the relevance (i.e., support) of review r_j to statement s_i, which can be computed using any sentiment analysis scheme that returns the confidence that a review expresses the same opinion as the statement does. In this work, we extract the opinions as aspect-based vectors. Therefore, $\rho(r_j, s_i)$ is set to 1 if r_j has an non-zero value for the aspect element that the statement is about, or 0 if otherwise.

It is worth noting that both models can be degenerated to the two-layer model by replacing $f^{(n)}(r_j)$ in equation 4.3 (or 4.6) with the corresponding form in equation 4.2 (or 4.5). This is because these models do not capture the influences due to the interactions on the inter-review edges.

Enhanced Model with Inter-Review Interactions. Finally, we integrate the inter-review interactions into the enhanced model to build the complete three-layer trust propagation model that captures all types of influences due to node connectivity and interactions on these edges.

For a review r_i, based on the deviation between its opinion about the aspect category of a relevant statement s_j (denoted as $o(r_{u_i}(s_j), s_j)$) and the aggregated opinion of statement s_j (denoted as $ao(s_j)$), we define a deviation function $\Delta(o(r_i, s_j), ao(s_j))$ as:

$$\Delta(o(r_i, s_j), ao(s_j)) = \begin{cases} 0, & o(r_i, s_j) = ao(s_j) \\ 1, & o(r_i, s_j) = -ao(s_j) \\ 0.5, & otherwise \end{cases} \tag{4.7}$$

Corresponding, a sentiment support from a review to a statement can be defined based on the consistency between $o(r_{u_i}(s_j), s_j))$ and $ao(s_j)$. We define the support function as

$$supp(o(r_i, s_j), ao(s_j)) = 1 - \Delta(o(r_i, s_j), ao(s_j)) \tag{4.8}$$

Here, if the sentiment polarity expressed in the review on a specific aspect category is the same as the sentiment polarity of the aggregated opinion of the statement ($o(r_i, s_j) = ao(s_j)$), then we say the review fully supports the statement. On the other hand, if the sentiment polarities between a review and a statement are totally opposite ($o(r_i, s_j) = -ao(s_j)$), i.e. positive and negative, or negative and positive, we say the review rejects the statement. For all other cases, we say the reviews partially support the statement.

The influence from the review-review edges in the inter-user context on the honesty score of a user is caused by whether the aspect category-specific opinions towards a statement (i.e, $o(r_i, s_j)$) and the corresponding aggregated opinions (i.e., $ao(s_j)$) is consistent or not. If a user's review supports the aggregated opinion and the statement has high truthfulness scores, this user will be rewarded for being consistent with a highly trusted statement. However, being inconsistent with the statement does not necessarily mean the user is dishonest. If a user's review expresses an opinion that rejects a statement with a low truthfulness score, this user should not be penalized but rewarded. The influence of an individual deviation may not always lead to the correct penalty or reward, however, based on our assumption that the benign users outnumber the spammers, the

overall influence introduced by all deviations across multiple entities and aspects should reflect the changes of trust scores correctly. Therefore, we consider the overall deviation for each user, and use the average deviation in the computing:

$$\delta^{(n+1)}(u_i) = \frac{\sum_{s_j \in \mathcal{S}(u_i)} d(t^{(n)}(s_j), supp(o(r_{u_i}(s_j), s_j), ao(s_j)))}{|\mathcal{S}(u_i)|} \tag{4.9}$$

where the function d models the deviation caused by the opinion difference expressed between a user's review and a statement and the trustworthiness of the statement. In particular, we defined d for the supervised-learning based approach in equations 4.10 and equations 4.11 for the unsupervised-learning based approach. $2x-1$ in both equations maps the score $t(s_j)$ from [0, 1] to [-1, 1].

$$d(x,y) = -y * ln(\frac{e^{2(2x-1)}}{1+e^{2(2x-1)}}) - (1-y) * ln(\frac{1}{1+e^{2(2x-1)}}) \tag{4.10}$$

$$d(x,y) = -y * ln(\frac{e^{2x-1}}{1+e^{2x-1}}) - (1-y) * ln(\frac{1}{1+e^{2x-1}}) \tag{4.11}$$

The only difference between equations 4.10 and 4.11 is the coefficient used before $(2x-1)$, which acts like an amplifier for the score $t(s_j)$. We can see that with more coarse-grained opinions, adding an amplifier makes the deviation achieve similar results as in unsupervised setting. Based on experimental exploration, we set the coefficient as 2.

We define the function d in this form so that the deviation decreases when the trust score $t(s_j)$ is small, even when the support between r_{u_i} and s_j is 0. When the sentiment support is 1, the deviation decreases when the score $t(s_j)$ is large. On the other hand, When the sentiment support is 0.5, the deviation increases in both directions as $t(s_j)$ increases, but at a slower speed. Finally, we compute the honesty scores for the users to reflect the influence of the overall deviation as below:

$$h^{(n+1)}(u_i) = \frac{\beta + 1}{\beta + e^{\delta^{(n+1)}(u_i)}} \tag{4.12}$$

The parameter β here controls the extent the score of a user u_i is affect the his/her deviation $\delta(u_i)$. With smaller value of β, the score drops quicker as $\delta(u_i)$ increases. The measurement of trust is propagated along the structural connections. For example, a user's honesty score is dependent on the trustworthiness of the statements in his reviews, thus the trust is propagated from his statements to the user himself, and further propagates to his reviews and back to his statements. Each type of score gets feedbacks from the other two, which allows reinforcement based on the connections among the nodes.

4.4.3 The Computational Framework

The scores of users, reviews, and statements are computed in an iterative computational framework, as shown in Algorithm 2. In the beginning, the nodes of users, reviews, and statements are generated from reviews data. The textual content of reviews are processed for extracting the aspect category-based opinions. In each round, all nodes update their scores accordingly and do normalization after all scores are updated. After the model converges, the final result is output as the measurement of modeled trust for users, reviews, and statements.

For the convergence of the reinforcement-based iterative model, the two-layer model like HITS [52] and PageRank [97] are proved to be converged using eigenvectors in the original papers. As for three-layers, to the best of our knowledge, there is no mathematical proof in models with similar structure [154, 130] as ours, since the three-layers models cannot be rewritten in matrix form. During experiments, we observe that our three-layer model always converges. With opinions extracted in the supervised setting, the model empirically converges in around 100 iterations. With unsupervised setting, the model empirically converges in around 30 iterations. We infer the reason that the model with opinions extracted from unsupervised methods converges faster is that with finer grained opinions, the model is able to aggregate more information to compute the trust

scores, and the changes of the propagated trust between two consecutive rounds are larger than the ones in the supervised approach.

Algorithm 2: Framework for Three-Layer Trust Propagation

Input : Collections of users $\mathcal{U}$, reviews $\mathcal{R}$, and statements $\mathcal{S}$

Initial sentiment polarities for all statements in $\mathcal{S}$

Parameters μ, β

Output: Honesty scores $h(u)$ for all users in $\mathcal{U}$

Faithfulness scores $f(r)$ for all reviews in $\mathcal{R}$

Truthfulness scores $t(s)$ for all statements in $\mathcal{S}$

repeat

Compute the faithfulness scores for all reviews using Equation 4.5

Compute the truthfulness scores for all statements using Equation 4.6

Compute the honesty scores for all users using Equation 4.12

Normalize each type of score with the largest as 1.0

until *converged*;

4.5 Experiments and Evaluations

We conduct several experiments on three different datasets of the Yelp reviews. In this section, we will explain the design of the experiments and evaluate the performance of our proposed models with experiment results and a case study.

4.5.1 Dataset

We used three datasets in the experiments. The first one is the SemEval-2014 dataset [104] published in the Semantic Evaluation series, the second is a dataset that we crawled from Yelp.com in 2013, and the last one was shared by authors of [89].

The SemEval dataset contains 3,041 sentences from restaurant reviews, which we used to train our classifier. In SemEval, each sentence is labeled with one or multiple aspect categories (i.e., food, service, price, ambience, and anecdotes/miscellaneous) and the corresponding sentiment

polarities (i.e., positive, neutral, negative, and conflict). As discussed in Section 4.3, the "conflict" sentiment category is not considered in our model. We then split this dataset in 4:1 ratio with a training dataset and a testing dataset of 2,432 and 609 labeled sentences, respectively.

The Yelp dataset that we have crawled from Yelp.com in 2013 contains 9,314,945 reviews about 125,815 restaurants in 12 U.S. cities, which were input by 1,246,453 users between 2004 and 2013. We extracted the data for the city of Palo Alto, California to test our content-aware trust propagation models.It contains 128,361 reviews about 1,144 restaurants from 45,180 users. To build the graph of our three-layer model, we conducted data cleaning to discard reviews that do not contain aspect-specific opinion indicators. We also ignored the statements with less than three related reviews and the users with less than three expressed statements and then adjusted the corresponding relationships. After cleaning, the dataset used in the supervised approach contains 46,652 reviews from 2,184 users for 1,071 restaurants. The dataset used in the unsupervised approach contains 40,064 reviews written by 2,182 users for 1,034 restaurants. Although our datasets contain rich information about the reviewers, such as the total number of reviews, average ratings, social relationships, etc., we only used the review content in this study.

The Yelp dataset with flagged reviews collected by authors of [89] is also used to evaluate the performance of our trust propagation model. It contains 788,471 reviews about 250,078 restaurants from 35,430 users.

4.5.2 Supervised Opinion Extraction

We first train a Support Vector Machine (SVM) classifier for opinion extraction. For feature extractions, we used bag-of-words and extracted the tf-idf weights as features. The classifiers for aspect categories and sentiment polarities were trained separately at the sentence level. A single sentence may contain multiple aspect categories. Since SVM is a binary classifier, a trained SVM classifier only classifies a sentence into one category, but cannot determine if it contains multiple categories. So, we trained five binary one-vs-all SVM classifiers independently, one for each aspect category, as suggested by [50]. Once we had the classified aspect categories, we ap-

Table 4.2: The performance of aspect category classification.

Label	Precision	Recall	F1-score	Support	Accuracy
food	0.81	0.78	0.80	238	0.844
not_food	0.86	0.88	0.87	371	
avg / total	0.84	0.84	0.84	609	
price	0.91	0.62	0.73	65	0.952
not_price	0.96	0.99	0.97	544	
avg / total	0.95	0.95	0.95	609	
service	0.82	0.69	0.75	122	0.906
not_service	0.92	0.96	0.94	487	
avg / total	0.90	0.91	0.90	609	
ambience	0.83	0.52	0.64	84	0.920
not_ambience	0.93	0.98	0.95	525	
avg / total	0.91	0.92	0.91	609	
anecdotes/miscellaneous	0.77	0.70	0.73	243	0.796
not_anecdotes/miscellaneous	0.81	0.86	0.84	366	
avg / total	0.79	0.80	0.79	609	

plied the trained classifiers of sentiment on each category to obtain the category-based sentiment polarities. As a results, one review may contain opinions about several aspect categories, such as "food,positive", "price,neutral", "service,negative", which are called *aspect-specific opinions*. Therefore, the extracted opinion of a review consists of a set of aspect-specific opinions.

Extracted Aspect Categories. The results of aspect category classification are shown in Table 4.2, in which "avg" means the average of precision, recall, and f1-score, and "total" denotes the total support of each category.

Among the five categories, the "anecdotes/miscellaneous" category has the worst precisions and recalls, which is reasonable, since this category contains all aspects that cannot be classified into any other categories. It is easier to determine what does not belong to anecdotes/miscellaneous than to determine what does. As a result, the precision, recall, and f1-score of not_anecdotes/miscellaneous are higher than anecdotes/miscellaneous itself. The "food" category is the most popular aspect category in restaurant reviews, interestingly, it has the second-worst performance among the five categories. This is partially due to the fact that there are too many different terms and aspects representing food types. When using the tf-idf weights as features, it is difficult to have a unified

Table 4.3: The classification performance of category-based sentiment polarities.

Label	Precision	Recall	F1-score	Support	Accuracy
food,negative	0.39	0.36	0.38	33	
food,neutral	0.50	0.04	0.07	25	
food,positive	0.80	0.92	0.85	169	0.740
avg / total	0.71	0.74	0.70	227	
price,negative	0.55	0.44	0.49	25	
price,neutral	0.00	0.00	0.00	2	
price,positive	0.67	0.81	0.73	36	0.635
avg / total	0.60	0.63	0.61	63	
service,negative	0.66	0.69	0.67	48	
service,neutral	0.00	0.00	0.00	7	
service,positive	0.73	0.79	0.76	61	0.698
avg / total	0.66	0.70	0.68	116	
ambience,negative	0.64	0.30	0.41	23	
ambience,neutral	0.00	0.00	0.00	5	
ambience,positive	0.68	0.92	0.78	49	0.675
avg / total	0.62	0.68	0.62	77	
anecdotes/miscellaneous,negative	0.11	0.10	0.10	31	
anecdotes/miscellaneous,neutral	0.60	0.49	0.54	96	
anecdotes/miscellaneous,positive	0.60	0.73	0.66	107	0.547
avg / total	0.54	0.55	0.54	234	

representation of the category. So, it is difficult to train an effective classifier for the food category than the price or service categories.

Extracted Aspect-Specific Opinions. We present the results of sentiment classification in Table 4.3. It shows only the performance of using SVM as the classifier for opinion mining, which is not the evaluation of the performance of opinion mining on the Yelp dataset. Comparing to the performance of aspect category classification, the performance of category-based sentiment polarity classification is worse. This may because bag-of-words captures representative features for categories better than capturing sentiment polarities. Sometimes, the sentiment polarities are implicit and context-dependent. Moreover, since the category-based sentiment analysis takes the classification results for aspect categories as the input, mistakes in the previous classification will be amplified and affect the overall performance. It is worth noting that, despite all these issues, our classification performance of sentiment polarity is still better than or comparable to the baseline

and some approaches in the SemEval 14 contest [104].

4.5.3 Unsupervised Opinion Extraction

The trust propagation model takes the classification results of aspect categories and category-based sentiment polarities as input. As discussed in Section 4.5.2, SVM does not yield the best results. So, we examine the unsupervised classification models to improve the overall performance of our model.

In unsupervised opinion extraction, we first parse the reviews using the Stanford PCFG parser [51] to capture the dependency relations in parsing. Then, we computed the word embeddings to include the semantic meanings of words. In particular, we used Word2Vec [82] with the skip-gram loss function to train our word embedding model, and used the Python library gensim [112] to implement it. We chose skip-gram instead of CBoW since it performs better in semantic tasks [82]. Finally, we used K-means for aspect clustering and set the cluster number to 9 in the following experiments. Among a few values we tried for the cluster number, it generated the most reasonable results on our data.

Extracted Aspect Categories. In Table 4.4, we show the top-7 words of the nine clusters identified in the unsupervised aspect extraction approach. From the table, we can see that the word2vec model captures the semantic relationships between words and clusters them into meaningful aspect categories. For example, the top-7 words in cluster 7 are spot, patio, location, space, area, etc., which represent a category about the "environment". Similarly, words in cluster 9 such as music, ambiance, atmosphere, interior, etc. represent a semantic category of "ambience". As a result, we used these most frequent words of each cluster to represent an aspect category.

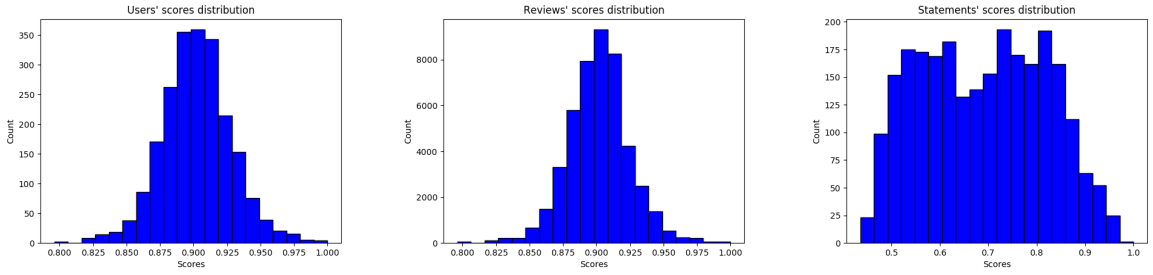
4.5.4 Trust Propagation with Supervised Opinion Extraction

In supervised opinion extraction, we classified the reviews into five aspect categories. However, in the experiments, we only used four categories in trust propagation, while the miscellaneous category is ignored. The miscellaneous category contains a mix of opinions on multiple aspects that

Table 4.4: The aspect categories extracted from aspect clustering.

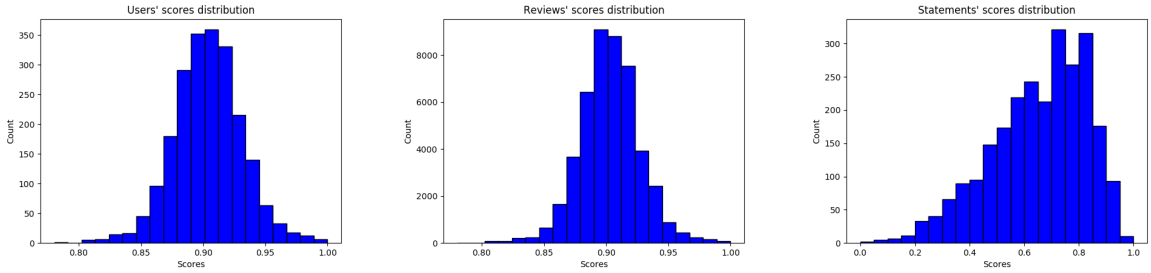
1	visit, favorite, experience, star, rating, review, trip
2	drink, dish, food, juice, entree, appetizer, meal
3	veggie, rice, fish, wrap, roll, steak, burger
4	date, evening, night, event, party, occasion, birthday
5	yogurt, cream, dessert, cupcake, ice-cream, cake, pie
6	bartender, waitress, waiter, server, table, complaint, job
7	spot, patio, location, space, area, room, parking
8	choice, selection, size, amount, variety, type, combination
9	music, decor, service, ambiance, atmosphere, vibe, interior

cannot be classified into the other four categories. Therefore, it is often the case that one user's opinion classified as "miscellaneous" is about an entirely different aspect from another user's opinion on "miscellaneous". As a result, this category cannot contribute much to trust measurement. However, it is necessary to keep the miscellaneous category in opinion extraction so as to improve the precision of the other four categories.



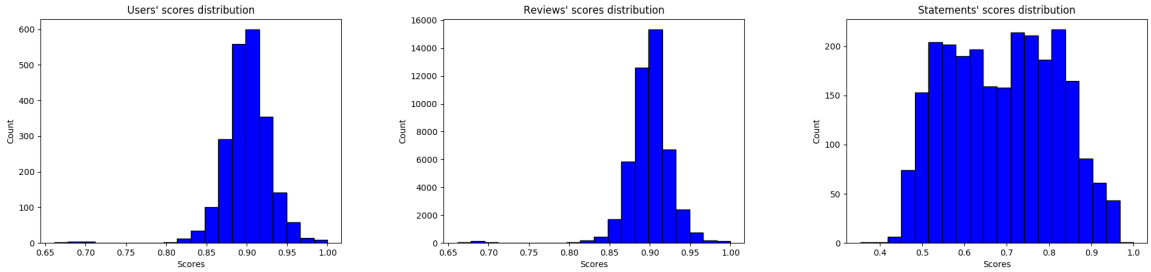
(a) (b) (c)

Figure 4.2: Distributions under the aggregate opinion setting: (a) the distribution of honesty score for users; (b) the distribution of faithfulness score for reviews; (c) the distribution of truthfulness score for statements.



(a) (b) (c)

Figure 4.3: Distributions under the positive opinion setting: (a) the distribution of honesty score for users; (b) the distribution of faithfulness score for reviews; (c) the distribution of truthfulness score for statements.



(a) (b) (c)

Figure 4.4: Trust scores distributions with synthetic data: (a) the distribution of honesty score for users; (b) the distribution of faithfulness score for reviews; (c) the distribution of truthfulness score for statements.

In the experiments, we adopt two initialization settings for the statements. The three-layer model is constructed based on the structural relationships among users, reviews, and statements. A statement is an aspect-specific opinion expressed in the review of a user about a restaurant. For example, user u_1 writes a review that a restaurant provides good services, from which we can extract a "service, positive" statement, while user u_2 feels opposite so his review expresses a "service, negative" statement. Obviously, with three sentiment polarities, there could be three statements for each restaurant on each aspect category. To reduce the complexity, we keep at most one statement for each restaurant on each aspect category in the framework and remove the other two. This could be done by considering the support from a review to a statement. For

example, if the "service, positive" statement is selected, the support from a review expressing "service, negative" will be set to 0. To determine which statements remain in the trust propagation framework, we considered two different settings in the experiments. In the first setting, we kept only the statements that express the aggregate opinions (i.e. the opinions shared by the majority of the users) for each aspect category. In the second setting, we initiate all the statements with the positive polarity, which means all the positive statements are kept in the framework. Finally, for μ and β, we set their values to 0.5 and 1.0, respectively.

Trust Scores. We calculate the three types of trust scores under two initialization settings of the statements. Firstly, we studied the distributions of trust scores under two settings. As shown in Figure 4.2(a) and (b), both the honesty scores for users and the faithfulness scores for reviews follow the normal distribution with the mean around 0.75. This implies that some users may be biased or dishonest, but most of the users and their reviews are trustworthy. The distribution of the truthfulness scores for statements is shown in Figure 4.2(c), which is somehow skewed and pushed towards 1. This means most of the claims are highly truthful, which is reasonable since they are initialized under the aggregate opinion setting, representing the opinions of the majority.

On the contrary, when we initialize all the polarities of the statements as positive, we intend to include some false statements in the framework. Intuitively, they should receive much lower support from truthful users, and they are expected to have low truthfulness scores. This is proved by the experiment as shown in Figure 4.2(f). Comparing to 4.2(c), quite a few statements have truthfulness scores lower than 0.5, while the statements with positive aggregate opinions still receive high truthfulness scores. It is worth noting that the changes in statement trust scores do not mean our model is sensitive to initialization. It simply shows that our trust propagation model is effective in capturing and penalizing the untruthful statements by giving low scores to them.

From 4.2(d) and (e), we can see that the honesty and faithfulness scores still demonstrate the normal distribution under the second setting. Comparing to 4.2(a) and (b), the absolute scores of individual users and reviews change slightly, since users are affected by the false statements at different levels. However, the shape of the distribution and the relative ranking do not change

Table 4.5: The average truthfulness scores of the statements under two initialization settings.

Category	Initialized with aggregate opinions	Initialized with all positive opinions
Food	0.771	0.796
Price	0.575	0.501
Service	0.600	0.588
Ambience	0.684	0.703

Table 4.6: Statistics of honesty scores using supervised opinion extraction and synthetic data.

Synthetic type	Min	Average	Median	Max
Support	0.916	0.947	0.950	0.985
Reject	0.661	0.691	0.692	0.718

much, which indicates our model is still robust to initialization settings.

Then, we compared the average truthfulness scores of four aspect categories to understand the influence of initialization settings. As shown in Table 4.5, in both settings, the statements about the food category received the highest average scores. Among the four categories, the scores of the "price" and "service" categories under the second initiation setting are smaller than the ones under the first setting. We observed quite a few positive statements about these two categories received lower trustfulness than positive statements about the other categories, which indicates the opinions on "price" and "service" are more controversial and subjective. In the second setting, these categories are more likely to be penalized more.

Evaluations. To evaluate the performance of our model, we did experiments with synthetic data and human evaluators under the first initialization setting. We first created a small synthetic dataset of 20 users, who were randomly chosen from our dataset. We manually changed their reviews so that 10 users' reviews fully support the statements relevant to their reviews, and the other 10 users' reviews fully reject all the statements. With this synthetic dataset, we aim to see if the model correctly rewards the "truthful" users who agree with the majority of the others and penalizes "untruthful" users who always deviate from the majority opinions.

From Figure 4.4(a) and (b), we can see two obvious clusters based on the trust scores for users and reviews. As expected, the 10 users who fully rejected the statements obtained smaller trust scores than others. Meanwhile, comparing to Figure 4.2(a), the scores of the remaining users,

including the other 10 user with synthetic data and the users from the original dataset, increased obviously. This is a side effect introduced by the 10 users who fully supported the statements. This indicates when there are more truthful users in the system, the model could better distinguish the truthful users from the spammers. We further show the average as well as the maximum and minimum honesty scores of the two synthetic groups in Table 4.6, which demonstrates the distinction between two groups as expected.

Next, we recruited three human evaluators to test the performance of our model. To generate the evaluation dataset, we randomly selected 20 users from our dataset and randomly selected 8 reviews for each user. In each test, we presented two users and their reviews together as a test set to the evaluators. In each evaluation, we gave the evaluators 20 tests, which were randomly selected from a total of 190 test sets. The human evaluators were asked to read all the reviews of the two users and rank them based on their relative honesty. Then, we compared evaluators' judgments with the user honesty scores calculated by our model. We considered a judge agrees with another if both ranked the two users in the same way. Finally, we measured the agreements between every two evaluators and between each evaluator and our model and presented the results of one evaluation in Table 4.7.

Overall, the agreement between our model and the human evaluators are low, with an average of 53.3%. This is because the evaluators had difficulties in telling if one user is more honest than the other only from the 16 reviews. As we can see, the agreements between human evaluators are almost the same, which indicates the evaluators did not reach a consistent judgment among themselves. This is not very surprising since we have observed that human judgment is not reliable, when the content of the reviews is carefully crafted to avoid obvious content-based spamming cues. In fact, the reviews used in the evaluation has already been filtered by the Yelp Filter, which is a proprietary filter developed by Yelp.com [41, 40]. While Yelp does not reveal its filtering mechanism, some study [89] shows that the Yelp filter is likely to use behavioral features such as review length, rating deviation, percentage of positive reviews, etc. to filter out suspicious reviews. Therefore, the reviews we presented to human evaluators had few behavioral features to help them

Table 4.7: The agreements between our model and human evaluators.

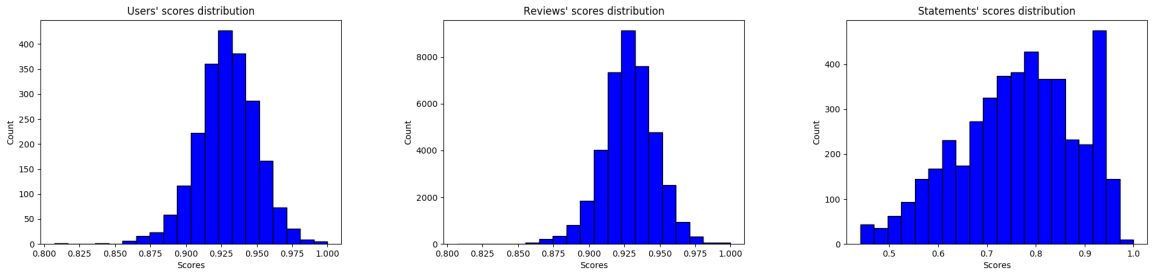
	Our model	Evaluator 1	Evaluator 2	Evaluator 3
Our model	–	12	10	10
Evaluator 1	–	–	10	10
Evaluator 2	–	–	–	12

in the judgment. This is also the reason that we propose to develop the model based on the deviation in aspect-specific opinions instead of the content-based features used in previous approaches.

4.5.5 Trust Propagation with Unsupervised Opinion Extraction

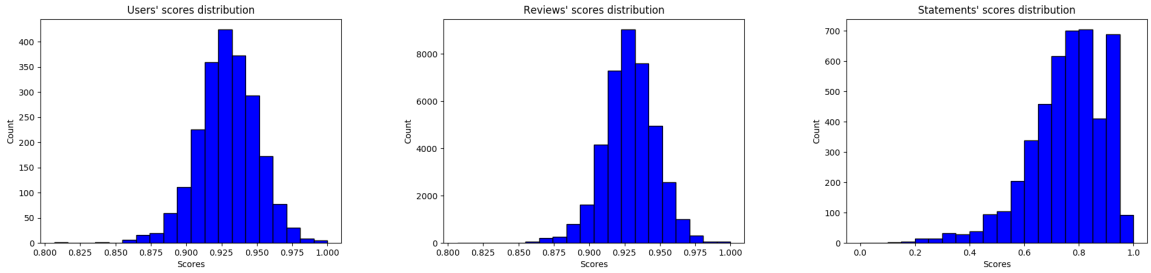
In this experiment, the opinions extracted from the unsupervised-based approach were input into the trust propagation model to calculate the opinion vectors and quality vectors. We used the 9 aspect categories as shown in Table 4.4, and set the values of μ and β to 0.5 and 1.0, respectively. Similar to the experiment in Section 4.5.4, we also adopted two initialization settings, one with statements of the majority opinions and the other with statements of positive opinions.

Trust Scores. We compute the trust scores for users, reviews and statements under two settings. Similar to the experiments with supervised opinion extraction, the honesty scores and faithfulness scores under two settings tend to be normally distributed, as shown in Figure 4.5 (a) and (b) as well as in 4.6 (d) and (e). Comparing to Figure 4.2, the mean honesty and faithfulness scores with unsupervised opinion extraction are larger than the ones with supervised opinion extraction.



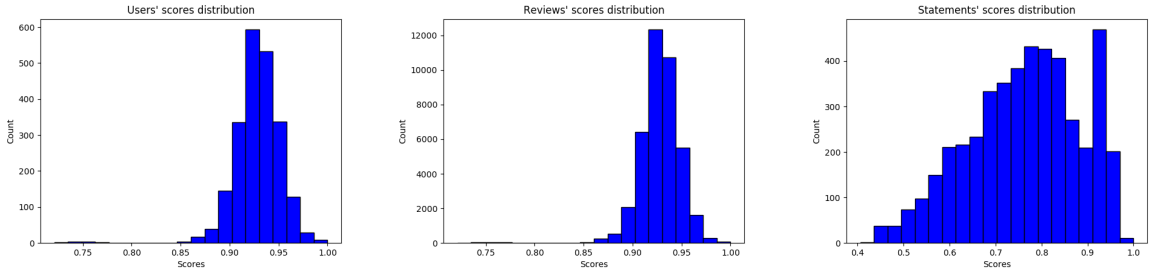
(a) (b) (c)

Figure 4.5: Distributions under the aggregate opinion setting: (a) the distribution of honesty scores for users; (b) the distribution of faithfulness scores for reviews; (c) the distribution of truthfulness scores for statements.



(a) (b) (c)

Figure 4.6: Distributions under the positive opinion setting: (a) the distribution of honesty scores for users; (b) the distribution of faithfulness scores for reviews; (c) the distribution of truthfulness scores for statements.



(a) (b) (c)

Figure 4.7: Trust score distributions with synthetic data: (a) the distribution of honesty scores for users; (b) the distribution of faithfulness scores for reviews; (c) the distribution of truthfulness scores for statements.

Evaluations. Similarly to the evaluations of the model with supervised opinion extraction, we conducted two evaluations, one with synthetic data and the other with human evaluators.

We take the same process as described in Section 4.5.4 to prepare the synthetic data. The distributions of the honesty, faithfulness and trustfulness scores for users, reviews, and statements are shown in Figure 4.7. The honesty and faithfulness scores also demonstrated the cluster effect – the 10 users who fully rejected the statements obtained clearly lower scores than others. Table 4.8 shows the average, median, maximum and minimum scores of the two groups of users with synthetic data. From the table, we see that all the users supporting the aggregate opinions obtained very high honesty scores, while all the users rejecting the aggregate opinions were penalized much

Table 4.8: Statistics of honesty scores using unsupervised opinion extraction with synthetic data.

Synthetic type	Min	Average	Median	Max
Support	0.940	0.966	0.965	0.982
Reject	0.733	0.751	0.749	0.780

more. The results show a similar tendency as the results with supervised opinion extraction.

Five human evaluators were recruited to evaluate the model with unsupervised opinion extraction under the setting with aggregate opinions. We provided a short training to the human evaluators with information about how to determine a review is trustful from its content. Then, we randomly selected 10 restaurants from our dataset, and randomly selected 4 reviews for each restaurant. In particular, the four reviews included three reviews with high faithfulness scores and one review with a low faithfulness score. In each test, we presented one restaurant and its four reviews to the evaluator and asked them to select the one that appeared to be the most suspicious review to them. The four reviews were displayed in a random order to eliminate the correlation between the review score and the display order.

Each evaluator completed 10 tests and thus selected 10 most suspicious reviews with her best judgments. We measured the agreements between our model and each evaluator, as well as the agreements between every two evaluators. As shown in Table 4.9, our model is consistent with the judgments of human evaluators and achieves an overall agreement ratio of 72%. Meanwhile, the evaluators agreed with each other in deciding the most suspicious review for each restaurant and achieved an overall agreement of 69%. The purpose of this evaluation is to verify whether humans agree with our model's results about reviews' trustworthiness by reading the content of reviews and checking the expressed opinions. The evaluation result shows that our model achieves quite good agreements with human evaluators. In addition, the agreements between our model and human evaluators are quite comparable with the agreements among evaluators themselves.

Table 4.9: The agreements between our model and the evaluators.

	Our model	Evaluator 1	Evaluator 2	Evaluator 3	Evaluator 4	Evaluator 5
Our model	–	8	7	9	5	7
Evaluator 1	–	–	9	8	6	8
Evaluator 2	–	–	–	7	6	7
Evaluator 3	–	–	–	–	5	7
Evaluator 4	–	–	–	–	–	6

4.5.6 Evaluation on the Dataset with Yelp Flagged Reviews

To further evaluate our model, we performed experiments on a separate dataset with flagged Yelp reviews, which was collected and published by authors of [89].

Dataset. This dataset contains 788,471 reviews about 250,078 restaurants from 35,430 users. Among the 788,471 reviews, 8,303 reviews were "flagged" by the Yelp filter [41]. These reviews were written by 7,118 users about 98 restaurants in Chicago, Illinois. Therefore, to build the dataset for evaluation, we used the 8,303 reviews as the "seeds" to extract all the restaurants reviewed by the 7,118 users as well as all the reviews and reviewers of these restaurants. The final evaluation dataset contains 100,290 reviews about 16,782 restaurants, which were submitted by 34,785 users.

The Behavior-based Detector. The authors in [89] proposed a detection approach that combined n-gram with behavioral features such as *maximum number of reviews*, *percentage of positive reviews*, *review length*, *reviewer rating deviation*, and *maximum content similarity* to identify spamming reviews [89]. They used the aforementioned dataset and treated the "flagged" reviews as the "ground truth" for spamming reviews. Their results are shown in Figure 4.8 (b), where P, R, F1, and A denote Precision, Recall, F1-score, and Accuracy, respectively. According to their results, bigrams combined with behavior features achieved the highest accuracy and F1 score of 85.7% and 86.1%, respectively.

This approach and several others following this direction treat spam detection as a classification problem, in which the performance of the classifiers highly depends on the quality of the training data. However, in our study of the flagged reviews, which were labeled or detected as spam in [89], we find several reviews are currently "unflagged" by the Yelp filter and publicly visible. These

Feature Setting	P	R	F1	A
Unigrams	62.9	76.6	68.9	65.6
Bigrams	61.1	79.9	69.2	64.4
Behavior Feat.(BF)	81.9	84.6	**83.2**	**83.2**
Unigrams + BF	83.2	80.6	81.9	83.6
Bigrams + BF	86.7	82.5	**84.5**	**84.8**

(a): Hotel

P	R	F1	A
64.3	76.3	69.7	66.9
64.5	79.3	71.1	67.8
82.1	87.9	**84.9**	**82.8**
83.4	87.1	85.2	84.1
84.1	87.3	**85.7**	**86.1**

(b): Restaurant

Figure 4.8: The performance of the behavior-based detector in [89].

reviews receive medium to high ranks in our system, while they were marked as spam in [89]. In fact, reviews flagged by the Yelp filter may not only include spam reviews [41]. Moreover, from the above behavior features, we can see that the detector in [89] could be evaded by sophisticated spammers who write lengthy reviews, avoid simple duplication and extreme rating behaviors. If such advanced spam reviews exist in the dataset, we are not sure how many of them were correctly flagged by the Yelp filter in 2013.

Evaluation Metrics. As explained above, the flagged reviews cannot be used directly as the ground truth. So, we select a smaller set of more suspicious users with higher confidence to be the ground truth for evaluation. For each flagged review in the evaluation, we calculate the *flagged ratio* of its reviewer as the percentage of flagged reviews among all his/her reviews. This ratio reflects how suspicious a user is from the view of the Yelp filter. On the other hand, our model measures the degrees of trustworthiness of the reviews, users, and the aspect-specific review statements with three types of trust scores. So, we define another metric to evaluate users' honesty scores calculated by our model. We define the *deviation ratio* for the user as the percentage of reviews with aspect category-specific deviations among all his/her reviews. This ratio reflects the overall deviation level of a user from the view of opinion mining. Then, we rank the users based on their honesty scores in the descending order and study the bottom-100 set, which contains the most suspicious spammers identified by our model. We define the *bottom-100 ratio* as the percentage of users who fall into the bottom-100 set. This ratio reflects the overall suspicious level of a certain group of users from the view of our trust propagation model.

Evaluation Results. In this evaluation, we want to study the users who are likely to be the spam-

mer. So, we target the set of users whose reviews were flagged by the Yelp filter. However, besides spamming, the reviews may be flagged due to other illegitimate content. Also, some may not contain aspect-specific opinions, for example, empty or very short reviews with 5-star ratings. Therefore, we identified a small set of users whose flagged reviews are meaningful, reflecting opinions on one or multiple aspects. In particular, we extracted the aspects from the reviews and filtered out the ones whose content does not cover any aspect. We further compared the opinions in each review with the aggregated opinions, and keep the ones deviating from the aggregated opinions. This reduced the set to 123 flagged reviews from 121 users with the supervised method and 82 flagged reviews from 77 users with the unsupervised method. These users are more suspicious than others in the evaluation dataset viewed by both the Yelp filter and our model. Then, we calculated the deviation ratio and the flagged ratio for each user. We use the average values of these two ratios and the bottom-100 ratio as the measurements of the selected users. The results are shown in Table 4.10. We can see that the model with unsupervised opinion mining got larger flagged ratio and deviation ratio than the supervised one. As a result, the bottom-100 ratio with unsupervised opinion mining is much higher. The reason is that the opinion mining with unsupervised method considers more aspects than the supervised method. When determining if there is aspect-wise deviation, the unsupervised method is more fine-grained.

Table 4.10: The evaluation results of the selected suspicious users for supervised and unsupervised opinion mining

	Avg. Deviation Ratio	Avg. Flagged Ratio	Bottom-100 Ratio
Supervised	0.2297	0.0526	0.0248
Unsupervised	0.5769	0.2426	0.2727

Case Study. We further investigate the other users who were ranked to the bottom 100 by our model to see if they are suspicious spammers and how suspicious they are.

Since there is no ground truth, we analyze all the information that we can find about the user, including user profiles, interactions with other users in Yelp, the reviews that are not included in the evaluation dataset, etc. Some users have weak personal profiles with little personal information. Most of these users have no friends and never interact with others. Some of them demonstrate

extreme rating patterns. These are indicators of spamming accounts as recognized by other works. We show two weak user profiles in Figure 4.9 and Figure 4.10 respectively. For privacy reasons, we removed user names, emails, and IDs of the profiles. Both profiles use the default profile picture and have no friends nor photos, indicating the least effort in maintaining their online images. Moreover, the user in Figure 4.9 has reviewed 32 restaurants. 81.3% of his reviews have either 5-star ratings or 1-star ratings, which demonstrates extreme rating behavior.

Among the flagged reviews, we further find that several of them are now publicly visible on Yelp, which means they are no longer flagged by the Yelp filter. Figure 4.11 shows two examples of such reviews. Both reviews were previously flagged, but now can be found via content search in the restaurant pages. From the review content as well as the information about user profile and activities, we fail to find any cue indicating spamming. This shows that not all the flagged reviews used in [89] are spam reviews and the approach of using the flagged reviews as the labeled ground truth is problematic.

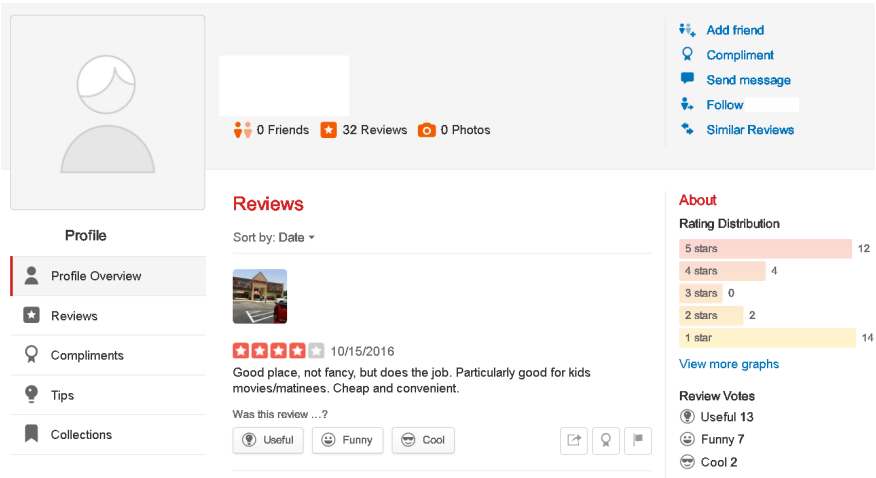


Figure 4.9: Case study: example of weak Yelp profile 1.

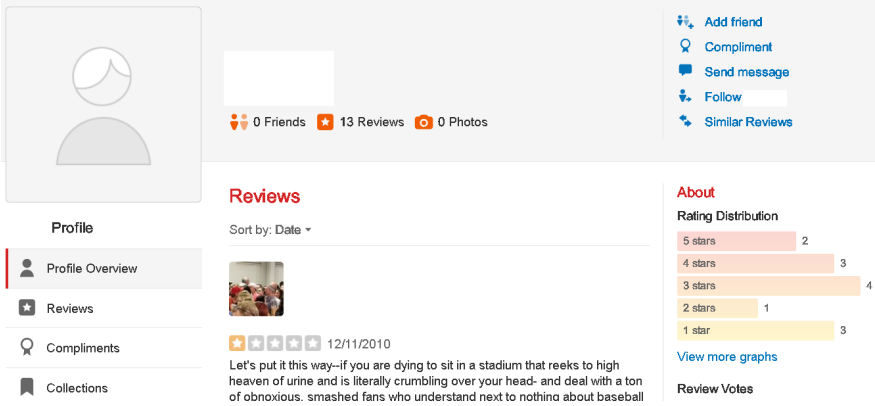
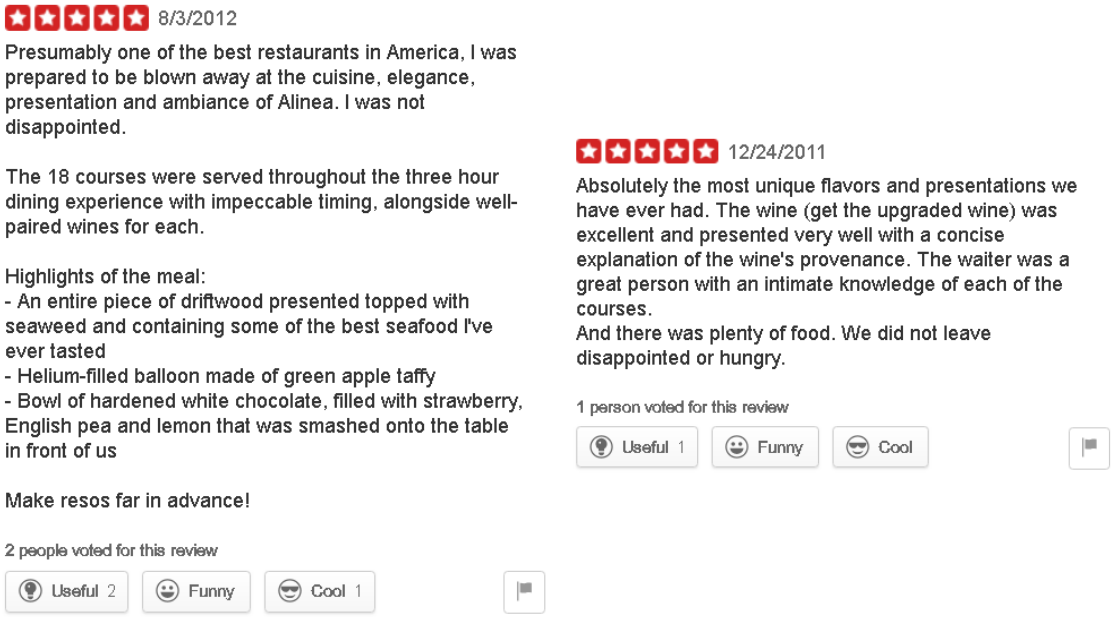


Figure 4.10: Case study: example of weak Yelp profile 2.



(a) (b)

Figure 4.11: Case study: reviews previously flagged by the Yelp filter but publicly visible now.

4.6 Summary

In this chapter, we study the problem of inferring trustworthiness from the content of online reviews. We first apply opinion-mining techniques using both supervised learning and unsupervised learning algorithms to extract aspect category-specific opinions expressed in the reviews. Then, we integrate the opinions to obtain opinion vectors for individual reviews and statements. Finally,

we develop an iterative content-based computational model to compute honesty scores for users, reviews, and statements. According to the results, there exist differences of statement truthfulness across different categories. Our model shows that the trustworthiness of a user is closely related to the content of his/her reviews.

Chapter 5

Neighborhood-based Social Bot Detection

5.1 Introduction

In Chapter 3 and 4, we present our study of measuring trust and credibility using rating deviation and textual content respectively for online review systems like Yelp. In this chapter, we introduce our work of social bot detection in OSNs like Facebook and Twitter. These platforms are also easy targets for attacks as they provide convenient platforms for users to share content freely online and have attracted huge numbers of users.

Social bots are accounts created by human or automated scripts and controlled by algorithms to generate content and establish interactions [11, 28, 18]. The existence and activities of social bots have been observed in social networks in the past few years [11, 60]. Some social bots are benign and helpful, they are usually adopted by companies for marketing purpose and customer care. In this work, we mainly focus on the bots that are created for malicious intents. This kind of bots are essentially content polluters and their presence can endanger the online ecosystems. Social bots are usually deployed to promote harmful information [10, 28], harvest users' privacy data [11], and infiltrate OSNs [11]. The problem of social bots is not platform-specific and happens across different culture. Twitter has a long-lasting bot problem. The famous Chinese microblogging service Weibo has devoted numerous resources into fighting against the "water army".

The existence and activities of social bots tamper the welfare of OSNs, thus the detection of social bots is an important endeavor and has attracted the attention of both industry and academia. For example, Twitter has updated its reporting process to let users to report suspicious tweet and suspended millions of accounts for violating its policies on malicious and spamming behavior

[126]. While it is beneficial to encourage users to get involved in the protection of the ecosystems, the reporting mechanism heavily relies on the participation of users and requires manual labor. On the other hand, automatic methods mainly utilize the behaviors of the users themselves. However, as mentioned previously, the spammers are capable of learning and changing the behaviors to evade the detection models. It is the same with social bots and it is not hard for social bots to change the behavioral patterns as the activities of bots are controlled by algorithms.

Since social bots are capable of faking human behaviors, it is desirable to build a detection scheme that takes factors other than individual behaviors into account. For a specific user, the interactions among other users in his neighborhood are not controlled by the user himself. Thus, the patterns of interactions in the neighborhood are difficult to fake for social bots. In this chapter, we propose our social bot detection scheme that utilizes the neighborhoods formed around users. We extract features by studying the properties of users' neighborhoods. Through experiments with both supervised and unsupervised methods, we show that our proposed neighborhood-based features are helpful for social bots detection.

5.2 Problem Formulation and Overview of Solution

In this section, we first formulate the problem of bot detection by overviewing the existing solutions. Then we present the high-level idea of our solution.

5.2.1 Problem Formulation

Existing social bot detection techniques mainly focus on behavioral patterns of bots. A large number of proposed methods aim to automatically detect bots based on machine learning systems that utilize features that are extracted from users' descriptive information [34, 123, 60, 129] such as length of description, number of friends, number of tweets, etc; or abnormality in behavioral patterns, including reading and writing actions [122], time between two consecutive tweets [129], and link creation timestamps [151]. The problem of these systems is that the features or abnormalities

are dependent on the behaviors of the individuals and can be manipulated. Bots are continuously evolving to be deceptive [28] as they are designed to appeal "normal" [11] and learn to mimic the human behaviors to confuse the existing detection approaches [43].

Another type of approaches is based on graph-theoretic techniques and relies on examining the structure of the social graph [14, 146, 100]. These approaches usually assume that legitimate users usually refuse to connect with unknown users or a user who connects with a legitimate user is considered to be legitimate itself [28]. However, it is proven to be wrong from both sides. From the side of social bots, it is beneficial for them to establish large numbers of connections to mimic human behavior [11]. On the other hand, experiments have shown that legitimate user would connect with strangers or accept friendship requests indiscriminately [11, 20, 123, 60]. For instance, the authors of [60] set up several Twitter-based social honeypots to attract social bots for research purposes. Among all the "harvested" users, they found that not all of them are spammers or malicious promoters, some of the attracted users were legitimate users who took advantages of the reciprocity in following relationships on Twitter. Reciprocity means if a user A follows another user B, then user B may follow user A back as a courtesy. Some previous work [84, 137] have shown that reciprocity is not uncommon in online social networks in terms of forming connections, indicating that the graph structure would not be effective as an indicator for distinguishing bots and legitimate users.

Given the context that social bot would fake human behaviors and the reciprocity of forming relationships exist in OSNs, we conducted a preliminary study on Twitter to study the behaviors of social bots. We have acquired a list of social bots and legitimate users. The detail of how we obtained the list is explained in Section 5.7. We first discovered a similar phenomenon of reciprocity between legitimate users and social bots. We searched the usernames of several social bots on Twitter and found out that legitimate users would follow social bots back.

Example 5. Figure 5.1 shows an example that legitimate users tweeting or replying a social bot about following back. For privacy reasons, we have removed all the profile pictures and usernames.

Figure 5.1: An example of users follow back a social bot on Twitter

Besides discovering that legitimate users sometimes would connect with social bots on OSNs, we also conducted a structural analysis to show that social bots may have similar properties as legitimate users. In terms of degree distributions in OSNs, study [3] have shown that OSNs like Twitter can be considered scale-free networks.

Definition 6. A **scale-free network** refers to a network that its degree distribution follows a power-law [141], meaning fraction of nodes with degree d follows a power-law $d^{-\gamma}$.

Figure 5.2 shows a curve that follows a power-law distribution. The trend shown in the curve is that as the number of degrees (i.e., d) increases, the portion of nodes with such number (i.e., $P(d)$) of degree decreases drastically. This heavy-tailed degree distribution reveals one of the most notable properties of a scale-free network that the portion of nodes with smaller degree greatly exceeds the nodes with a much larger degree.

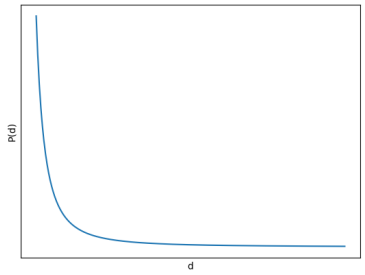


Figure 5.2: An example of power-law distribution $P(d) \sim d^{-\gamma}$

For a specific user, his personal network formed by other users who are connected to him can be considered a subgraph of the entire social graph. To study and compare the overall structural

features of the networks between legitimate users and social bots, we conducted an analysis using degree distribution. We randomly chose nine social bots and nine legitimate users on Twitter for this preliminary study. For each user, we formed his network by considering the users who follow and are also followed by this user. The relationship between the value of degree d and the fraction of nodes with degree d (i.e., $P(d)$) are shown in Figure 5.3 and Figure 5.4 for bots and legitimate users respectively.

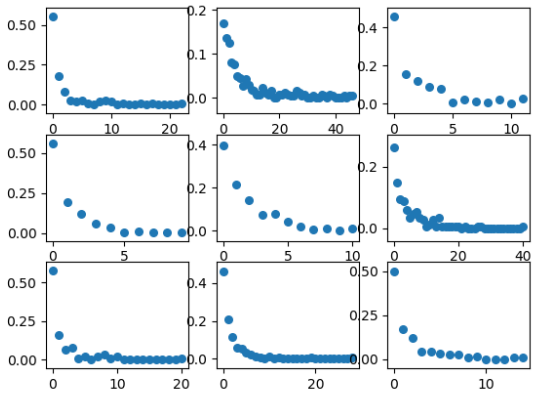


Figure 5.3: Degree distributions from nine randomly chosen bots. In each figure, the x-axis represents the value of degree d and y-axis represents $P(d)$, the fraction of nodes with degree d

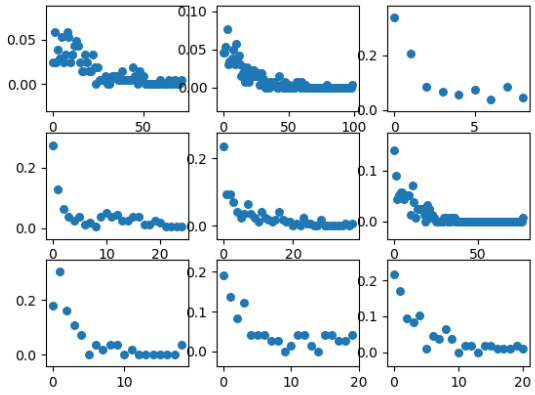


Figure 5.4: Degree distribution from nine randomly chosen legitimate users. In each figure, the x-axis represents the value of degree d and y-axis represents $P(d)$, the fraction of nodes with degree d

The shapes of scatter plots show that the degree distributions of networks of both bots and

legitimate users roughly follow the power-law distribution, meaning that the networks formed by bots and legitimate users are all close to scale-free networks.

From the discussions presented above, a social bot detection scheme that does not rely on behavior features that are easy to be manipulated is highly desirable.

5.2.2 Solution Overview

To prevent the influence of manipulated behavioral patterns, we need to take the factors that are hard to be controlled by individuals into account. In this work, we identify that behavioral patterns of other users in a given user's network are difficult to be manipulated or faked. Thus, for a specific user in the social network, we focus on the users who are connected to the user instead of the user himself. Before presenting our approach, some preliminary concepts need to be defined and introduced.

Ego Network and Neighborhood. Ego network is widely used in social network analysis. Compared to ordinary social networks that focus on the global structures, it takes an ego-centric view on the surrounding network for a specific user.

Definition 7. An **ego network** refers to the network formed by a focal node, known as the **ego**, and the nodes, known as the **alters**, that are directly connected to the ego plus the connections, if any, among the alters.

Definition 8. The **connection** between two users refer to the basic link formed between them.

On most OSNs, the connections refer to the basic *friendship* relationship. In an ego network, the connections could either be directed or undirected. For example, on Facebook, two users are connected if they are *friends* (undirected). While on Twitter, the connections refer to the *following/follower* relationship (directed). In this work, we also call the ego network of a user as his *neighborhood*.

Definition 9. For a given user, his **neighborhood** refers his ego network. More specifically, we call the user himself (i.e., ego) as the **seed user** and the users inside his neighborhood (i.e., alters)

as the **one-neighborhood-user** (onn-user).

Example 6. Figure 5.5 demonstrates a simplified example of the neighborhood of a seed user within a larger network. The node filled in black represents the seed user. The onn-users and users outside the seed user's neighborhood are represented as nodes filled in grey and white represent respectively. The lines represent the connections between nodes.

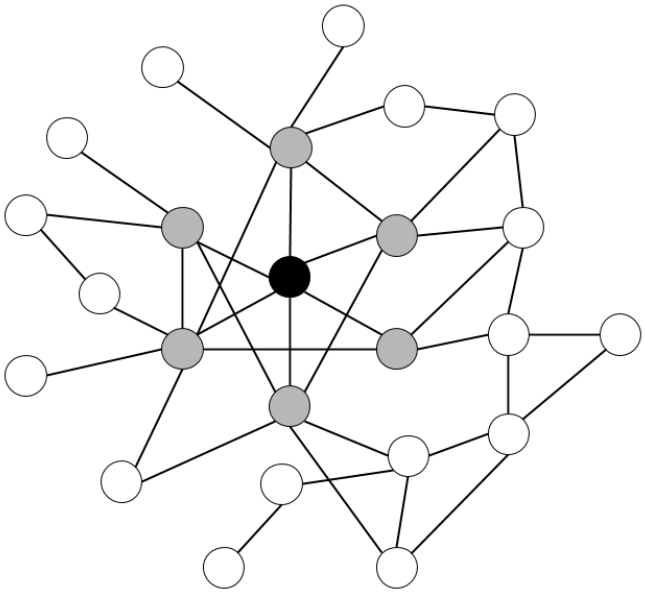


Figure 5.5: A simplified example of the neighborhood of a seed user within a larger network

Social Bot Detection via Neighborhood. On social networks, the users will interact with each other in various ways that the social network supports. For example, on Twitter, a user can interact with another user in ways including like, reply, and retweet.

Definition 10. The **interactions** between two users refer to the activities of communications (except building connections) between them in the way the OSN supports.

Besides the interactions between the seed user and his neighborhood, users within the neighborhood may also interact with each other. They may know other directly or indirectly and interact just like the seed user and his connections. If we exclude the seed user and only consider the connections and interactions inside the neighborhood, it still makes a valid graph.

Example 7. Figure 5.6 demonstrates a simplified example of the connections and interactions among onn-users in a neighborhood of size six. Nodes marked with numbers *1* to *6* represent onn-users in the neighborhood. The solid line represents connections and dashed line represents any kinds of interactions (e.g., like, reply, etc).

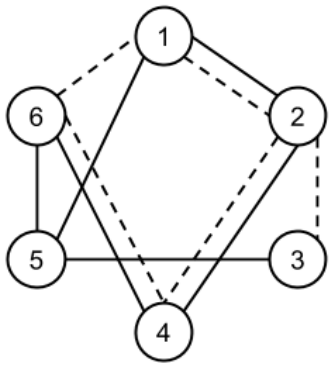


Figure 5.6: A simplified representation of the connections and interactions among onn-users in the neighborhood of a seed user.

Usually, the behaviors of bots are largely automated by algorithms [11, 119]. To mimic human behaviors, they are programmed to build connections with other users and post content. However, interactions are more likely to happen between users who are familiar with each other rather than strangers. Thus, even though legitimate users may exist in the neighborhood of a social bot, it is natural to assume that due to the limitation of the controlling algorithm, the overall communications within the neighborhood would be different with the neighborhood of a legitimate user. Although bots can fake human behaviors, we argue that it is hard for bots to fake a legitimate "society". For a bot account, he may post content and form social relationships like legitimate users, but it is usually hard, if not impossible, for him to control his entire neighborhood to behave like a legitimate user's network. Thus, taking the property of users' neighborhoods into account would be helpful for social bot detection.

On the other hand, the relationships between users in the neighborhood could be various. For example, they can be the user's family, classmates, colleagues, or just people met online with shared interests. As a result, there will be social circles naturally formed in the user's neighbor-

hood. Thus, there will be patterns reflected in terms of interactions and connectivity for users inside and outside the social circles. For example, users who are classmates of the user may also be classmates of each other, so it is likely that they are connected and have some shared friends. Based on their connectivity, they may interact more with each other than other users in the neighborhood. On the other hand, the social circles formed in the bots' networks will be different than the circles naturally formed in the legitimate users' networks.

In this work, we introduce a novel bot detecting mechanism by studying the property of users' neighborhoods. We focus on Twitter, the famous online social networking service. To take advantage of various sources of interactions among users available on Twitter, we model the neighborhoods formed around the users with multiple data representations including connectivity, interactions, and content similarity. To utilize the hidden social circles within the neighborhoods, we propose to detect social circles with a multi-view clustering approach [150]. We trained our social bot detection models with features extracted from the overall neighborhoods as well as the discovered social circles.

The rest of this chapter is organized as follows. In Section 5.3, we present how we model a user's neighborhood with multiple views. Then in Section 5.4, we explain the details about the feature extraction. In Section 5.5, we introduce how we discover hidden social circles within a user's neighborhood with multi-view clustering. In Section 5.7, we experimentally evaluate our proposed neighborhood-based features with both supervised and unsupervised learning. Finally, we summarize this chapter in Section 5.8.

5.3 Neighborhood Modeling

In this section, we introduce how we model the neighborhoods of Twitter users. Since the following/follower relationship is unidirectional on Twitter, we give our definition of friends on Twitter. Hence, in our problem, the neighborhood of a seed user is composed of his friends.

Definition 11. We define two users as **friends** if they follow each other on Twitter.

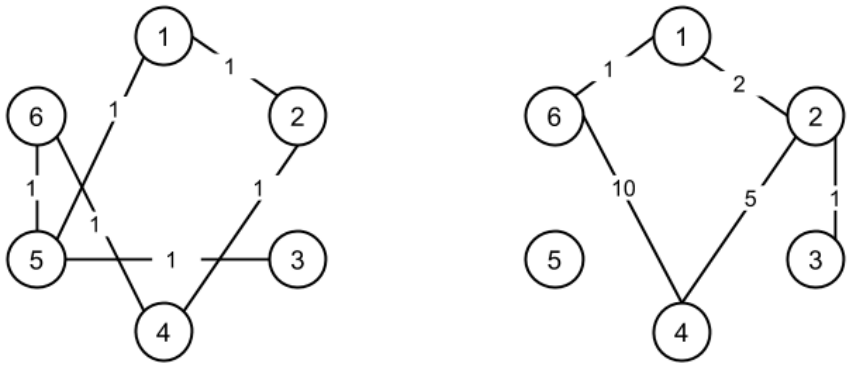
On Twitter, users are able to communicate with each other in various ways. For example, a user can directly reply to another user's tweet with comments. If a user finds a tweet interesting, he can retweet the tweet. Thus, for a given neighborhood of a seed user, there exist multiple different representations.

5.3.1 Model Neighborhood with Multiple Views

We borrow the term *view* from multi-view learning to model the neighborhood with different types of data. Multiple views are essentially the different representations obtained from multiple sources of the same data. For example, an image can be represented by its color or texture, which can be seen as different features extracted from the image.

Here, we use the concept *view* not to deal with multi-view learning yet but for consistency with later sections, we still use the term *view*. With different views, the actual onn-users in the neighborhood are the same, the differences are just the edges and weights. Thus, from now on, for a seed user, we use the term *neighborhood* not just to represent the network where links are friendship relations, but to represent the general network formed by the seed user's onn-users.

Example 8. Figure 5.7 shows a simplified toy example of the neighborhood with two different views.



(a) Friendship (b) Reply

Figure 5.7: A simplified example of the neighborhood of a user with two different views

One of the advantages of considering multiple views is that different views would contain complementary information and thus provide a comprehensive understanding of the actual properties of the neighborhoods. For example, a social bot may pretend to be legitimate by forming a large neighborhood. However, the onn-users in his neighborhood may not know each other at all and thus barely have any interactions with each other. In this work, we consider views in three major aspects, connectivity, interaction, and content.

5.3.2 View Definition

Inspired by the work in [150], we adopt the six views defined in [150] and added one additional view. Thus, we defined in total seven views ($K^{(1)}$, $K^{(2)}$, ..., $K^{(7)}$) for different representations of a seed user's neighborhood. For a neighborhood with size n, each view K is essential an n-by-n symmetric similarity matrix. The views are defined as below.

1. **Friendship view** $K^{(1)}$. This view is simply the adjacency matrix of the friendship between onn-users, where $K^{(1)}_{ij} = 1$ if onn-user i and j follow each other on Twitter and $K^{(1)}_{ij} = 0$ otherwise. It is widely used as a structural feature for social circle detection.

2. **Shared friends view** $K^{(2)}$. This view represents the shared friends between onn-users. For example, $K^{(2)}_{ij} = m$ means onn-user i and j have m shared friends (excluding the seed user).

3. **Reply view** $K^{(3)}$. This view represents the total number of replies happened between onn-users on Twitter. For example, $K^{(3)}_{ij} = m$ means onn-user i and j replied to each other m times in total (i replies j and j replies i both count).

4. **Retweet view** $K^{(4)}$. This view represents the total number of retweets happened between onn-users on Twitter. For example, $K^{(4)}_{ij} = m$ means onn-user i and j retweeted tweets from each other m times in total (i retweets j and j retweets i both count).

5. **Co-reply view** $K^{(5)}$. This view represents the total number of tweets onn-users both replied to on Twitter. For example, $K^{(5)}_{ij} = m$ means onn-user i and j both replied to m tweets.

6. **Co-retweet view** $K^{(6)}$. This view represents the total number of tweets onn-users both retweeted on Twitter. For example, $K^{(6)}_{ij} = m$ means onn-user i and j both retweeted m tweets.

7. **Content similarity view** $K^{(7)}$. This view represents the content similarity between the tweets onn-users posted on Twitter. There are various ways to obtain the content similarity between the tweets. The authors in [150] employed TagMe [1], a text annotation service with topics in Wikipedia, with TF-IDF weight. One of the drawbacks of their approach is that the annotation accuracy and coverage are not ideal as TagMe only annotates text chunks that have Wikipedia pages. The semantic relations between similar concepts cannot be reflected from the annotations (e.g., cheeseburger and hamburger have similar meanings but will be annotated as two different entities). Another problem is that using TF-IDF for vectorization may assign lower weights to words that are frequently mentioned. We are not interested in measuring the importance of words. Instead, in this work, frequently appeared words should be a hint for higher content similarity. We are looking for a method that is capable to capture the semantic meaning of words and give us good vectorization results. We propose to use Latent Dirichlet allocation (LDA) [7], a popular topic modeling model. We treated the tweets posted by an onn-user as a document. The documents of all onn-users of all seed users make up a corpus. We trained a topic modeling model using the corpus and obtained the topic distribution vector for each onn-user. The content similarity between two onn-users i and j is computed as

$$sim(i, j) = 1 - Hellinger(v_i, v_j) \tag{5.1}$$

where $hellinger(v_i, v_j)$ is the Hellinger Distance [139] computed for topic distribution vector v_i and v_j. For two discrete distributions $P = (p_1, ..., p_n)$ and $Q = (q_1, ..., q_n)$, the Hellinger Distance is between P and Q is computed as:

[1] https://tagme.d4science.org/tagme/

$$Hellinger(P,Q) = \frac{1}{\sqrt{2}}\sqrt{\sum_{i=1}^{n}(\sqrt{p_i}-\sqrt{q_i})^2} \tag{5.2}$$

In this view, $K_{ij}^{(7)} = m$ means the topic distribution similarity between onn-user i and j is m.

Each view represents an independent relationship between the onn-users in the neighborhood. More specifically, the first two views (i.e., friendship and shared friends) represent the direct and indirect connectivity among onn-users. The following four views (i.e., reply, retweet, co-reply, and co-retweet) captures the strengths of different interactions among onn-users. The last view models the topic-wise similarity within the neighborhood.

With these defined views, we can obtain seven different graphs with corresponding representations of the neighborhood of a seed user.

5.3.3 View Construction

Each view is represented as an n-by-n symmetric matrix (assuming the size of the neighborhood is n). The entries in the matrix represent the modeled similarity/strength between a pair of users. In this work, we do not consider self-similarity. As a result, the diagonal of each similarity matrix is all zeros.

The constructions for the first two views are fairly intuitive. As defined previously, we consider two users friends if they follow each other. The relevant information can be accessed from the following and follower list of each onn-user. It is worth noting when constructing the shared friends view, the seed user is excluded from the count of shared friends since the seed user is a friend of every onn-user.

The data for the constructions of the four interaction views (i.e., reply, retweet, co-reply, and co-retweet) is embedded in users' timelines. During the construction, we noticed that Twitter adopted two types of retweeting activity. Besides the normal retweet action, Twitter allows users to add a comment when retweeting a tweet. This activity is marked as "quote" in the crawled data.

For the *retweet* and *co-retweet* view, we considered both types of retweeting activities. In this way, we would have a more accurate and comprehensive representation of the retweeting interactions.

The construction of the content similarity view is relatively complex. In this work, we only consider the content of the tweets users posted and ignore the content of in the retweeted tweets. The problem is that a user may have posted a considerable amount of content since he created his Twitter account. It would be both time-consuming and computationally expensive to process all the content of all users have generated. On the other hand, the actual content a user has posted may change over time. Hence, we are more interested in the content users recently posted. As suggested in [60, 94], we only consider the most recent 300 English tweets. We employ the famous topic modeling model Latent Dirichlet Allocation (LDA) for computing content similarity. To prevent the influence of noise, we conducted basic preprocessing as in most NLP tasks before feeding the tweets into LDA. We first cleaned out all mentions (e.g., @xxx), hashtags (e.g., #xxx), emojis, and URLs. These expressions are commonly used in tweets but contain little information for content similarity. After data cleaning, we conduct tokenization to convert tweets into individual tokens and removed all stopwords. The output of LDA is a trained topic model that contains topic information inferred from the input corpus. For each document in the corpus, the trained LDA model would return its topic distribution as a vector with a length of the number of topics.

5.4 Feature Extraction from Views

We have defined seven views in aspects of connectivity (topology), interaction, and content similarity. As mentioned before, there is no explicit feature matrix defined for the views in the way we modeled the views. On the other hand, the characteristic of the neighborhood within a specific context is encoded within the corresponding view. Thus, to capture the property of the neighborhood from different perspectives, we just need to extract features from the views themselves.

5.4.1 Feature Extraction with Graph Metrics

As mentioned in Section 5.3, each view is represented by an n-by-n symmetric matrix (assuming the size of the neighborhood is n). The entries of the view represent the strength of the edges between the onn-users. Therefore, for each seed user, each view can be considered as the adjacency matrix for a weighted social graph of his neighborhood with a different context. Hence, to extract the features from the views, we adopt several graph metrics, including:

- **Density** of a graph measures the ratio between the actual number of edges and the maximal possible number of edges. For a graph G, its density is defined as

$$density(G) = \frac{2m}{n(n-1)} \tag{5.3}$$

  where m is the number of edges and n is the number of nodes in graph G. The formula 5.3 shows that the density of a certain graph is directly proportional to the number of edges formed in the graph. Therefore, a graph is more *dense* if there are more edges in it.

- **Closeness centrality** of a node measures its average farness (reciprocal of distance) to all other nodes. The higher the closeness centrality of a node is, the shorter the distances it has to all other nodes. Closeness centrality is a type of measurement of centrality and can be used for detecting nodes that are capable of spreading information efficiently through a graph. For a node u, its closeness centrality defined as

$$close\_central(u) = \frac{n-1}{\sum_{v=1}^{n-1} d(v,u)} \tag{5.4}$$

  where $d(v,u)$ is the distance between v and u. n is the number of nodes. For a graph G, we calculated the average closeness centrality of all nodes as the metric for G.

- **Clustering coefficient** of a node measures how close its neighbors are to being a clique. Study shows that in social networks, nodes tend to create tightly knit groups characterized

by a relatively high density of ties [138]. For a node u in an unweighted graph, its clustering coefficient is defined as

$$clust\_coef(u) = \frac{2T(u)}{deg(u)(deg(u)-1)} \tag{5.5}$$

and for a node u in a weighted graph, its clustering coefficient is defined as

$$clust\_coef(u) = \frac{1}{deg(u)(deg(u)-1)} \sum_{uv} (\hat{w_{uv}} \hat{w_{uw}} \hat{w_{vw}})^{1/3} \tag{5.6}$$

where $T(u)$ is the number of triangles through node u and $deg(u)$ is the degree of u. For a weighted graph, $\hat{w_{uv}}$ is the weight normalized by the maximum weight in the graph. The above definition is also referred to as *local clustering coefficient*. For a graph G, we calculated the average clustering coefficient of all nodes as the metric for G. The clustering coefficient is similar to density in the sense that it is also related to the number of edges in the graph. Compared to graph density, clustering coefficient takes a personal view as it is defined for a node instead of a general view for a graph.

In the previous preliminary study shown in Section 5.2, we have discovered that the general feature like degree distribution cannot capture the differences between bots and legitimate users well. We choose to use graph metrics for feature extraction to capture the latent characteristics expressed in different views.

5.5 Social Circle Detection with Multi-View Clustering

In OSNs, users usually will have more interactions with familiar people or like-minded users who share similar interests. For example, family members may have frequent communication with each other. Thus, hidden social circles will be constructed in terms of connections and interactions within the large social graph.

Definition 12. A **social circle** refers to a group of people with certain type of social intimacy [59].

To utilize the views we have defined previously, we applied a multi-view clustering approach to detect the hidden social circles. In this section, we first present the motivation and observations of conducting social circle detection. Then we give a brief introduction of the multi-view clustering method we applied.

5.5.1 Motivation

Social circle detection has been studied as a fundamental task in social network analysis, as it has many potential applications such as content recommendation and privacy management. In this work, we take social circle detection as an auxiliary step for social bots detection.

With different views defined in Section 5.3, a user's neighborhood can be examined with different representations. However, besides the global characteristics of a user's neighborhood, we are also interested in checking the properties of the smaller and more clustered communities. As mentioned in [150], users in the same circles usually have behaviors described as below:

- Users in the same circle are more likely to be connected and have many shared friends.
- Users in the same circle tend to share similar interests on the same content.
- Users in the same circle are more likely to interact with each other.

Thus, within the neighborhoods of legitimate users, social circles are often naturally formed by different relationships. On the other hand, for bots, their neighborhoods are usually deliberately constructed. The activities of bots are usually controlled by algorithms with automation, hence, the individual behavior of bots will be different as humans. As a result, the social circles formed within the neighborhoods of bots, if any, will have different properties as legitimate users.

5.5.2 Preliminaries of Multi-View Clustering

Twitter does not provide the functionality to present the explicit social circles, thus the social circles are hidden on Twitter. Based on the aforementioned observations, we propose to use a multi-view

clustering method proposed in [150] called *Selected Co-Trained Spectral Clustering* (SCSC) for our social circle detection. To help the readers better under the model we applied, we will explain the preliminaries of multi-view learning.

Multi-View Learning. When training conventional machine learning algorithms, such as Support Vector Machines (SVM) and Spectral Clustering (SC), multiple views need to be concatenated into one single view (a longer feature vector). The concatenation is not physically meaningful because each view has a specific statistical property. What's more, it may cause overfitting when applying the model on a small size training sample. On the other hand, multi-view learning improves the learning performance by introducing one function to model a particular view and jointly optimizes all the functions to exploit the redundant views [147]. Co-training [8] is one of the earliest schemes for multi-view learning. The training is conducted alternately to maximize the agreement on two distinct views [147]. The *Selected Co-Trained Spectral Clustering* (SCSC) method we used for social circle detection falls within this category.

Detecting Social Circles with Multi-view Clustering. We applied a multi-view clustering method to discover the hidden social circles. The SCSC method can effectively integrate structural, interaction, and content features together to improve the clustering performance [150]. Note that designing and improving multi-view clustering is not our focus, so we would not go deep into the mathematical details. For technical details, the reader is advised to refer to [150, 59].

Spectral Clustering [93] (SC) is a well-known graph clustering algorithm. The input for SC is the similarity matrix of the data. The model makes use of the discriminative information in the eigenvectors of the graph Laplacian. When multiple views are available, Co-train Spectral Clustering (CSC) alternately refines the graph Laplacian of one view based on the clustering result suggested by other views. CSC is designed to assume all views are completely observed and transfer the complete graph information across views. The drawback of CSC is that when many views are extremely sparse (partially observed), enforcing a complete agreement between the sparse views and other views will mis-refine the graph Laplacians and result in degradation of clustering performance [150]. In our data, we have observed that for bots, many interaction

views are sparse. This is indeed a problem if we apply multi-view clustering for social circle detection. Thus, we propose to apply the Selected Co-Trained Spectral Clustering (SCSC) method that handles the sparsity problem properly.

The work in [150] applied and evaluated several network clustering methods, including SCAN [148], Spectral Clustering (SC) [93], and Co-trained Spectral Clustering (CSC) [57]. The authors found that SCSC outperformed all these models in terms of compactness.

In our problem, SCSC is the most appropriate one. SCAN is purely based on the topology information and will output hubs and outliers in the results, which we do not need. SC also works under single-view. Comparing the CSC, the SCSC method handles the sparsity problem by only transferring the clustering results on observed edges from partially observed views with a selective process [150, 59]. Here, for convenience of description, we repeat the design of the SCSC method as described in [59].

Algorithm 3: Selective Co-Trained Spectral Clustering [59]

Input : Similarity matrices of T views $K_1, K_2, ..., K_T$

Output : Final cluster matrix C by applying K-means on the concatenated feature matrix $U = [U^{(k_1)}, ..., U^{(k_l)}]$ of dominant views $k_1, ..., k_l$

Initialize: $\forall t \in [T], U^{(t)} = eig(K^{(t)}, k), C^{(t)} = clust(U^{(t)}, k)$

for *i=1* **to** *rounds* **do**

  for $t \in [T]$ **do**

    Refine similarity matrix $K^{(t)} = update(t) \circ K^{(t)}$

    Update $U^{(t)} = eig(K^{(t)}, k)$

    Update $C^{(t)} = clust(U^{(t)}, k)$

  end

end

In Algorithm 3, the operator $eig(K,k)$ returns an n-by-k matrix with columns are the k principle eigen-vectors of the normalized Laplacian of matrix K. The operator $clust(U,k)$ essentially applies k-means clustering with k clusters on matrix U. To determine whether a view is partially observed, a sparse rate ρ_j is defined:

$$\rho_t = \frac{\text{\# zeros in } K^{(t)}}{\text{\# all elements in } K^{(t)}} \tag{5.7}$$

A view $K^{(t)}$ is partially observed if ρ_t is below a pre-defined threshold ρ_{thre}. In Algorithm 3, $update(t)$ is the selective process. It is implemented by comparing the sparse rate of other views with ρ_{thre}. The final detected social circles are essentially the clustering results from k-means clustering. So, the social circles we detected using SCSC are non-overlapping circles. In other words, each onn-user can only belong to one social circle. It is worth noting that for a given neighborhood with T views $K_1, ..., K_T$, SCSC will only output one assignment for the social circles (which users belong to which social circle).

Evaluation of Detected Social Circles. There are two main challenges in evaluating the quality of the detected social circles. The first one is the number of social circles to be detected. As described in Algorithm 3, the final circles are obtained by running k-means clustering with the parameter k. Thus, the number of social circles is determined by our input. The second challenge is that there is no ground truth of how users should be clustered in circles and no feature matrix is defined in the views. Thus, the clustered results cannot be evaluated by external evaluation metrics such as Rand index (requires ground truth of clusters) or internal evaluation metrics such as Davies–Bouldin index (requires feature matrices). For the social circles detected for a seed user, we adopt the same evaluation metrics *Normalized Similarity Ratio* (NSR) and *Total Similarity Ratio* (TSR) defined in [150, 59]. We first introduce two auxiliary metrics, namely *within-circle similarity*:

$$S_w^{(t)} = \frac{\sum_{(i,j)} K_{ij}^{(t)} \cdot 1\{C_{ij} > 0\}}{\sum_{(i,j)} 1\{C_{ij} > 0\}} \tag{5.8}$$

and *between-circle similarity*:

$$S_b^{(t)} = \frac{\sum_{(i,j)} K_{ij}^{(t)} \cdot 1\{C_{ij} < 0\}}{\sum_{(i,j)} 1\{C_{ij} < 0\}} \tag{5.9}$$

where $t \in |T|$ (assumming there are T views in total) and $(i, j) \in [n] \times [n]$ (assuming there are n onn-users in the seed user's neighborhood). The matrix C represents the clustering results.

$C_{ij} = 1$ stands for onn-user i and j are clustered into the same circle while $C_{ij} = -1$ means i and j are clustered into different circles. In this way, the within-circle similarity $S_w^{(t)}$ computes the average similarity for onn-users who are clustered into same circles. Similarly, the between-circle similarity $S_b^{(t)}$ measures the average similarity for onn-users who are clustered into different circles. For a single, we define its *Normalized Similarity Ratio* (NSR) as the ratio of the aforementioned two metrics:

$$NSR^{(t)} = \frac{S_w^{(t)}}{S_b^{(t)} + \alpha} \tag{5.10}$$

Here, α is a small constant to be added in the denominator when $S_b^{(t)} = 0$. $NSR^{(t)}$ measures the quality of the detected circles regards to a specific view $K^{(t)}$. The larger $NSR^{(t)}$ is, the more meaningful the circles are for the view $K^{(t)}$. Note when view $K^{(t)}$ is sparse, it is possible that the value of $NSR^{(t)}$ gets very large.

Similarly, the *Total Similarity Ratio* (TSR) measures the overall quality of the detected circles across all views. It is defined as:

$$TSR = \frac{\sum_t S_w^{(t)}}{\sum_t S_b^{(t)}} \tag{5.11}$$

Hence, for good clustering results, it is natural to expect higher TSRs as onn-users within the same social circle should have higher similarities than onn-users outside the circle. Thus, the TSR can be considered as a way of computing the compactness of the detected circles.

5.6 Social Bot Detection

In this section, we present the details of how we utilize the neighborhood and social circles for social bot detection.

5.6.1 Definition and Summarization of Feature Sets

Recall that we have defined seven views in Section 5.3. For a given user, these views are essentially the different graph representations of the user's neighborhood. Without considering the discovered social circles, we can apply the graph metrics directly on the views to obtain the features of a seed user's entire neighborhood. We refer to these features as the *global features* (GL) since they capture the characteristics of the entire graph from a global view without considering social circles.

On the other hand, for a given seed user, the discovered social circles can be considered as subgraphs within the neighborhood. It is worth repeating that although we have defined multiple views, there only exists one assignment of the social circles across different views for a given user's neighborhood. Thus, the features we defined in Section 5.4 can also be applied to the discovered social circles. For each social circle, we can obtain three graph metrics as features. However, given the nature of clustering, these circles do not have orders. On the contrary, the clusters from the clustering method should have a parallel relationship. Thus, in the feature space, these features cannot be aligned like normal features. For example, for a seed user A, the density computed from the first social circle of his friendship view does not necessarily correspond to the density of the first social circle computed using the friendship view of another seed user B. To resolve this issue, we use the average and standard deviation of the graph metrics computed from social circles as the features extracted from social circles. We refer to these features as the *circle features* (CC) since they capture the characteristics within each social circle.

Besides the features extracted using graph metrics, we can also obtain the NSR calculated with each view from the social circles. As defined in Section 5.5, NSR captures the ratio between *within-circle similarity* and *between-circle similarity* for a specific view. We consider the NSRs as additional features obtained from social circles and refer to them as *NSR features* (NSR).

Overall, we have defined three feature sets and we refer to them as the *neighborhood-based features*. The three features can be summarized in Table 5.1.

Table 5.1: Summarization of Feature Sets

	density	closeness centrality	clustering coefficient	
Graphs of view 1 - 7	GL			
Circles with view 1 - 7	CC			NSR

5.6.2 Methods for Social Bot Detection

We treat social bot detection as a feature-based detection problem. With the three feature sets defined above, we encode the patterns that are difficult to fake into our neighborhood-based features. The next step is to choose the appropriate machine learning methods as our bot detectors.

As mentioned previously, we have acquired a list of social bots and legitimate users (the detail is explained in Section 5.7), which means that we have the ground truth of whether a user account is a social bot or not. Applying supervised-learning methods is a natural choice when ground truth is available. We have compared several popular supervised models, including support vector machine (SVM), logistic regression (LR), Random Forest, etc. Among all the methods, we choose Random Forest as the classifier for our supervised bot detection. Random Forest is an ensemble learning method that works by building a multitude of decision trees and applying bootstrap aggregating to the tree models during training. The classification output is determined by taking the majority vote of the tree models. Compared to other supervised models like SVM and LR, one of the advantages of the Random Forest model is that the bootstrapping procedure applied during training decreases the variance and leads to better model performance. Thus, Random Forest is capable to prevent overfitting and can be generalized better.

In real-world scenarios, it is not always the case that the ground truth is available. Thus, to emulate the scenario that ground truth is unavailable, we propose to apply an unsupervised approach for social bot detection. K-means is a widely-used clustering method as it is fast and can be used for most unsupervised tasks. However, its drawback is also obvious. The performance

of K-means relies on the initialization of the cluster centroids. With different initial centroids, K-means may output very different cluster assignments. Although repeating multiple iterations of K-means would help improve the stability of the clustering output, we would prefer a method that is capable to generate stable results. We choose Agglomerative Clustering for our unsupervised bot detection. Agglomerative Clustering is a bottom-up hierarchical clustering model. It starts by treating each data sample as a separate cluster. Then, the model recursively merges the pair of clusters with the minimum distance until the predefined number of clusters is reached. Unlike K-means, the performance of Agglomerative Clustering is stable and does not rely on initialization.

5.7 Experiments and Evaluations

In this section, we present the details about the experiments we have conducted and the evaluations of the experimental results.

5.7.1 Dataset

In this work, we focus on Twitter. Fortunately, some researchers have made the Twitter dataset they have crafted publicly accessible. We choose to use one of the published datasets as the ground truth of seed users. From the dataset of seed users, we developed our crawler to crawl the data of related onn-users.

Original Dataset of Seed Users. We obtained the list of seed users from the famous TAMU Social Honeypots Dataset [2]. This dataset was originally collected from December 30, 2009 to August 2, 2010 on Twitter and introduced in [60]. The original dataset contains 22,223 bot accounts and 19,276 legitimate users.

Data Collection. In this work, we focus on ordinary users whose primary language is English. We consider a user to be an *ordinary* if his account is "non-verified" and has less than 10,000 followings and 10,000 followers. Twitter has its own defense mechanism and would suspend users

[2] http://infolab.tamu.edu/data/

who violated the platform's term of service. The data of suspended users cannot be accessed, so we also filter out them. With the aforementioned filtering conditions, we obtained a list of our seed users, which contains 9,669 legitimate users and 12,532 bot accounts. The original dataset was collected in the year of 2010. However, to get the most recent data about the users, we have developed our own scripts to crawl the data. The API we used for data collection is called Tweepy [3]. For each user, our scripts collected data including the user's profile, the following and follower list, and the timeline (e.g., tweets, retweet, etc) of the user. The data collection process is divided into two rounds. In the first round, we use our scripts to collect the data of the seed users using the aforementioned obtained list. The data collection process for seed users started in November 2018 and finished in January 2019. After we finished collecting the data of the seed users, we went on to collect the data of their onn-users. As of June 2019, we managed to collect the data of related onn-users for over 6,000 seed users combined. In this dissertation, we used a dataset which contains 560 seed users (280 bots and 280 legitimate users) and 72,750 onn-users for experiments.

5.7.2 Social Circle Detection with SCSC

We apply the SCSC method with constructed views to detect the hidden social circles within the neighborhoods. As mentioned in Algorithm 3, the social circles are determined by applying K-means clustering in the final step, the number of social circles k is an input parameter for SCSC. Although we have obtained the information of users' neighborhoods, we do not have prior knowledge about how many circles exist for each user. Theoretically, the optimal number of social circles for each seed user may be different. However, assigning different numbers of circles for each user will introduce inconsistency when extracted features from social circles. Hence, in this work, we set the number of circles the same for all seed users when applying SCSC as authors of [150, 59] did. The experiments are conducted with different k values from 3 to 6 and set the value of the threshold of sparse rate ρ_t to be 0.1. For the discovered social circles, we mainly conduct the evaluation using the previously defined metrics NSR and TSR as we do not have ground truth for

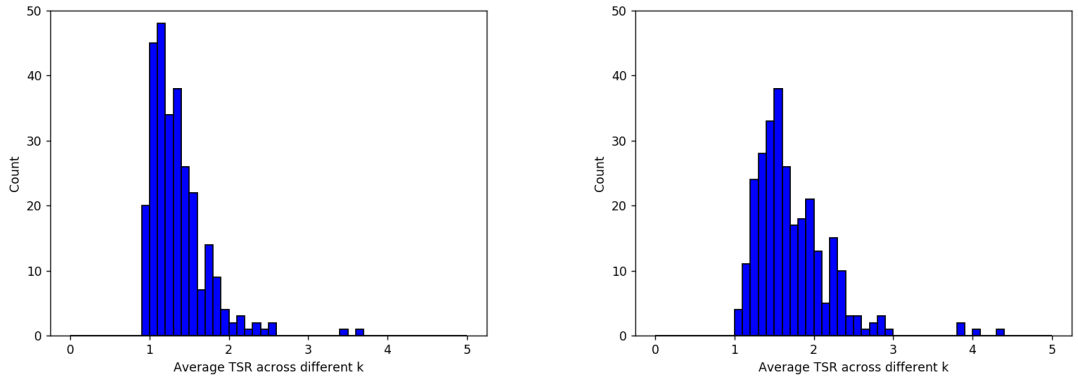
[3] http://www.tweepy.org/

social circles with Twitter.

In the experiments, we apply SCSC with multiple different values of k and obtained the clustering results with different numbers of circles. We first compute the average TSR as the overall measurement of the social circles within each user's neighborhood. The comparisons are shown in Table 5.2 and Figure 5.8. While the differences of distributions of average TSR is not obvious between bots and legitimate users, the results in Table 5.2 shows that the average TSR for legitimate users is higher than bots, indicating that the overall quality of discovered social circles of legitimate users is better than bots.

Table 5.2: Average TSR for all users, bots, and legitimate users

	Bots	Legitimate users	All users
Average TSR	1.3576	1.7304	1.5440



(a) (b)

Figure 5.8: Distribution of TSR with seven views for (a) bots and (b) legitimate users

Before diving into further analysis, we need to determine the value of the number of circles k. We count each onn-user "vote" for their optimal value of based on the largest TSR. Based on the results in Figure 5.9, we chose to pick 6 for our further analysis and present the experimental results when k (number of social circles) $= 6$.

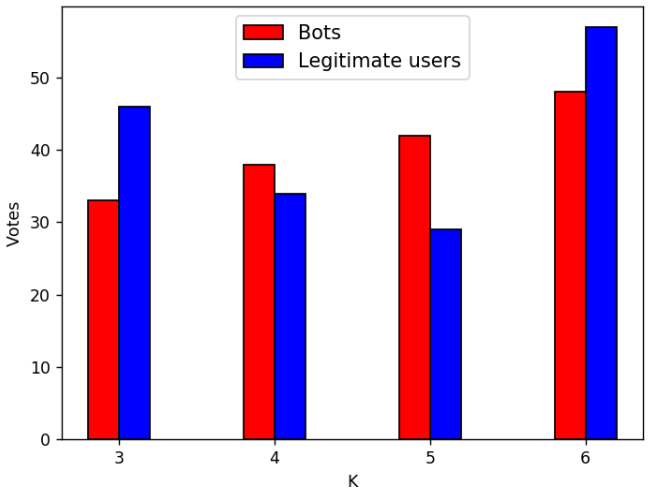
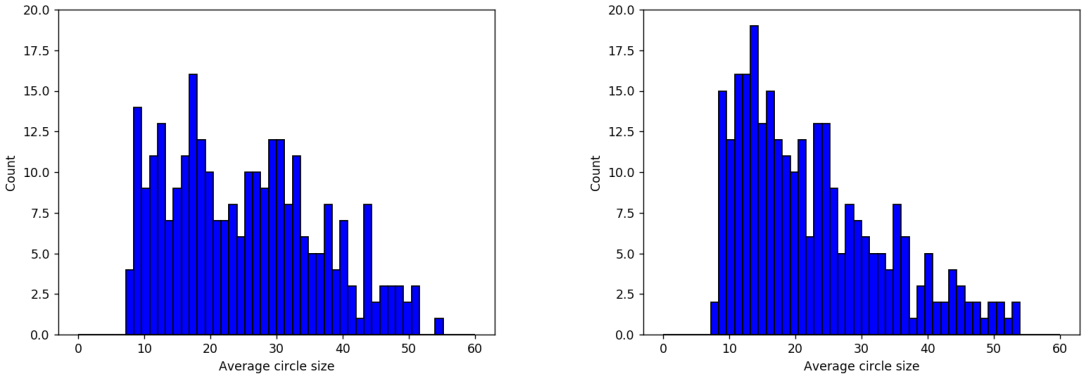


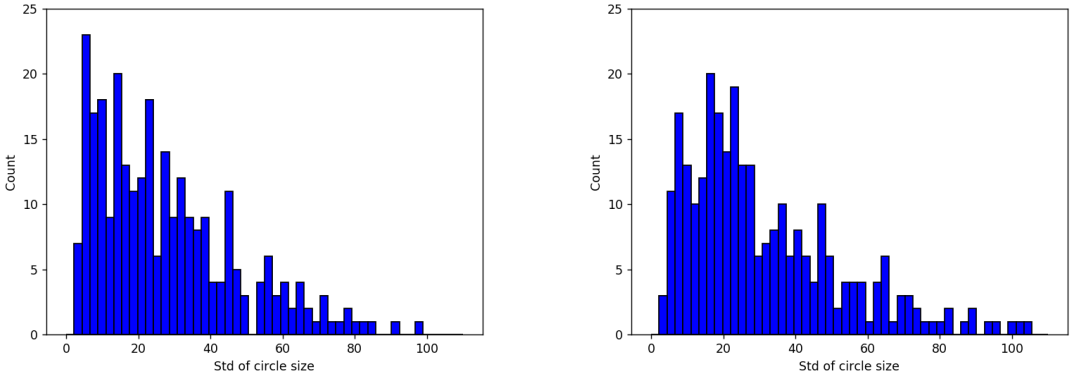
Figure 5.9: Votes for optimal value of k for bots and legitimate users

In the output results of SCSC, there is no guarantee whether the sizes of discovered circles would be evenly distributed or not. Here the size of a social circle refers to the number of onn-users that are assigned to the specific circle. Figure 5.10 and 5.11 compared the circles of bots and legitimate users with average and the standard deviation of the sizes. From the distributions, we observe no obvious differences between bots and legitimate users in terms of circle sizes. The previous preliminary study presented in Section 5.4 has shown that the composition of the neighborhoods of bots and legitimate users may share some similarity. The comparisons of sizes of discovered circles further show that the general property of the neighborhoods formed around social bots and legitimate users are rather similar than different.



(a) (b)

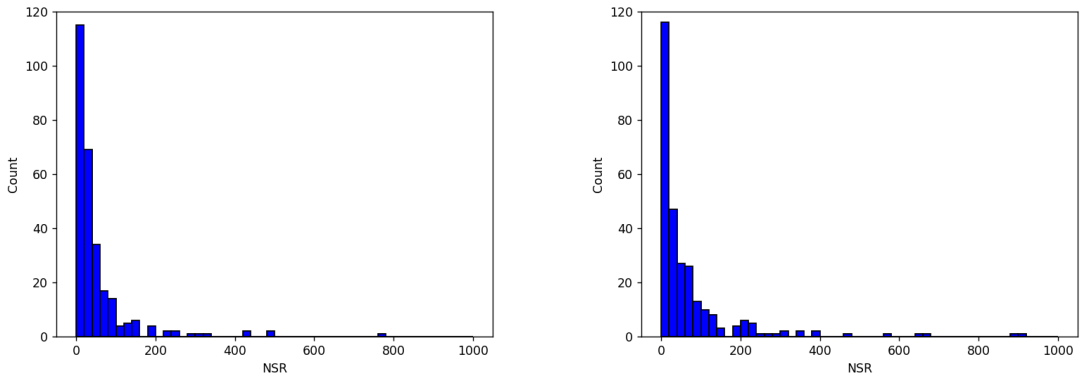
Figure 5.10: Distribution of average social circle size for (a) bots and (b) legitimate users



(a) (b)

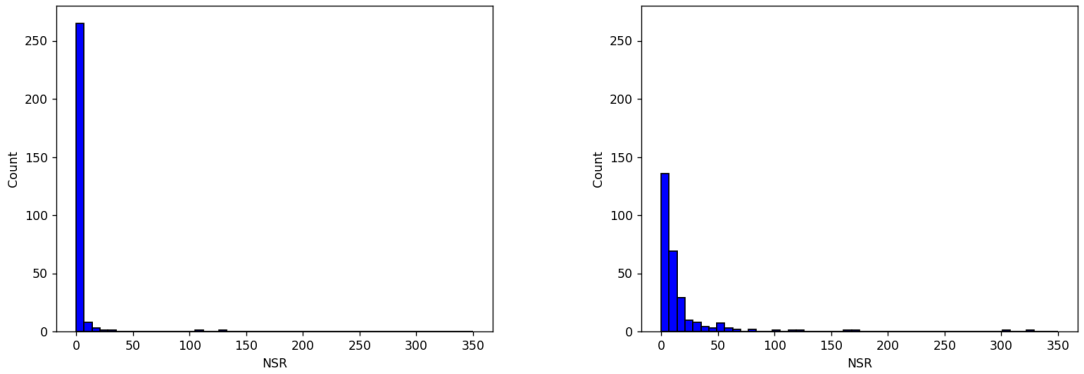
Figure 5.11: Distribution of std of social circle size for (a) bots and (b) legitimate users

To further compare the results of discovered circles, we report the $NSR^{(t)}$ between legitimate users and bots for all seven views from Figure 5.12 to Figure 5.18 respectively.



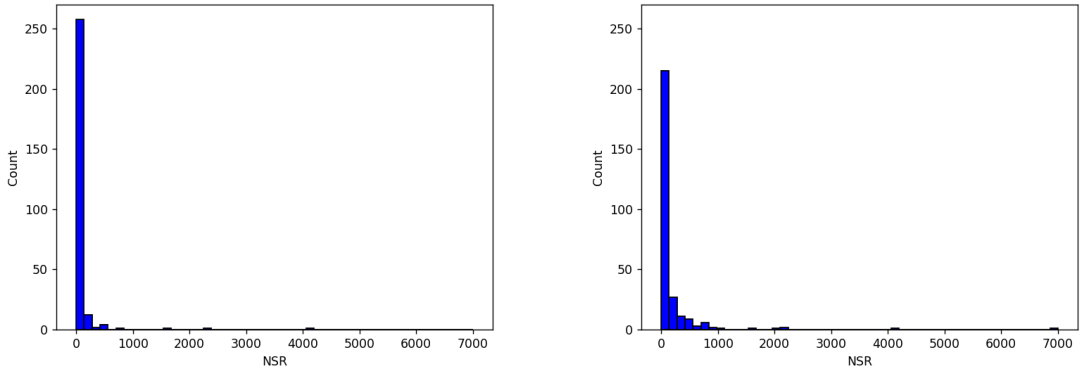
(a) (b)

Figure 5.12: Distribution of NSR for friendship view for (a) bots and (b) legitimate users



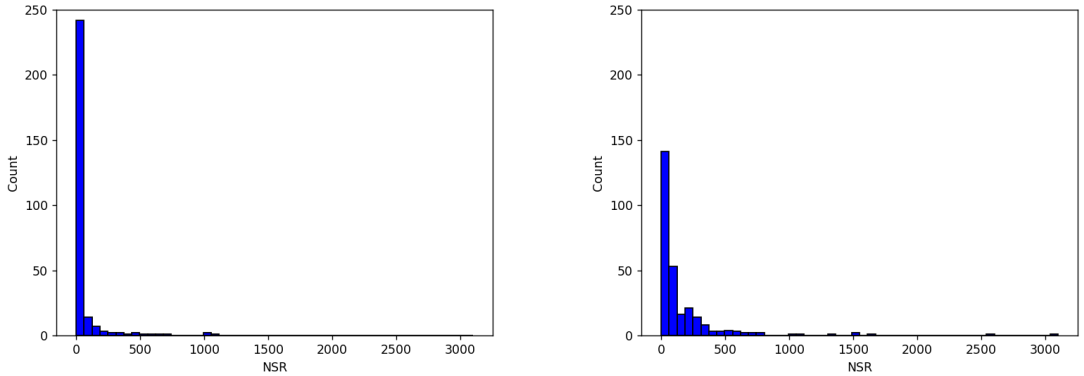
(a) (b)

Figure 5.13: Distribution of NSR for share friends view for (a) bots and (b) legitimate users



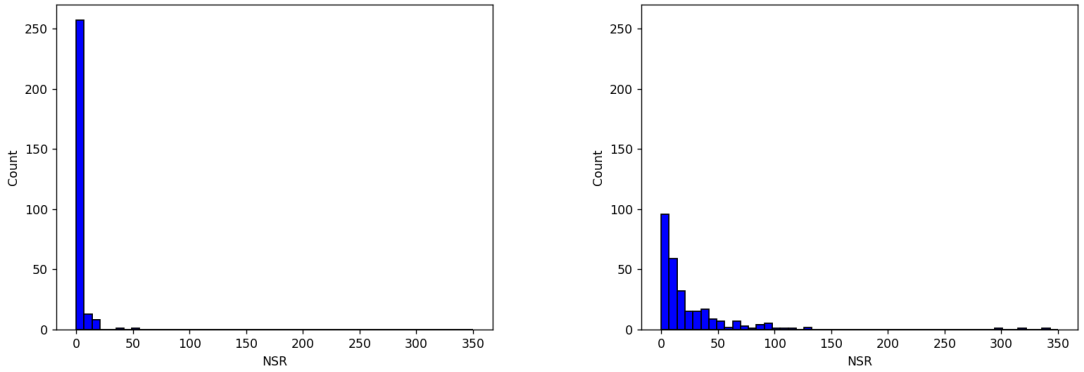
(a) (b)

Figure 5.14: Distribution of NSR for reply view for (a) bots and (b) legitimate users



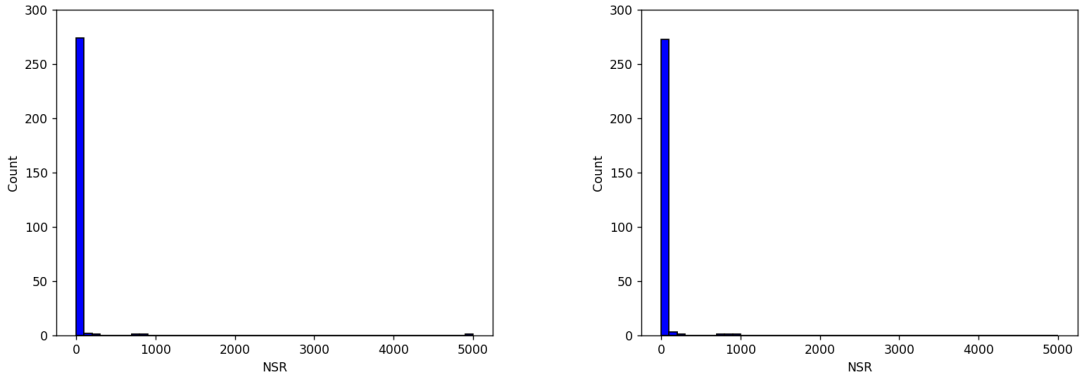
(a) (b)

Figure 5.15: Distribution of NSR for retweet view for (a) bots and (b) legitimate users



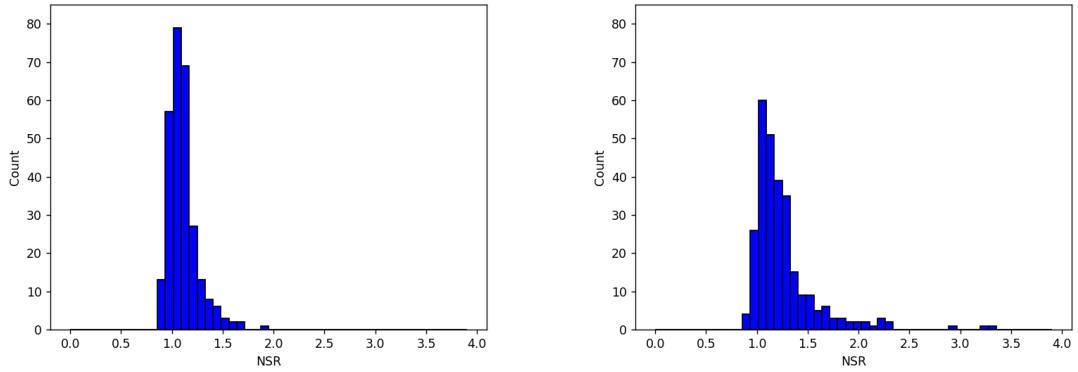
(a) (b)

Figure 5.16: Distribution of NSR for co-reply view for (a) bots and (b) legitimate users



(a) (b)

Figure 5.17: Distribution of NSR for co-retweet view for (a) bots and (b) legitimate users



(a) (b)

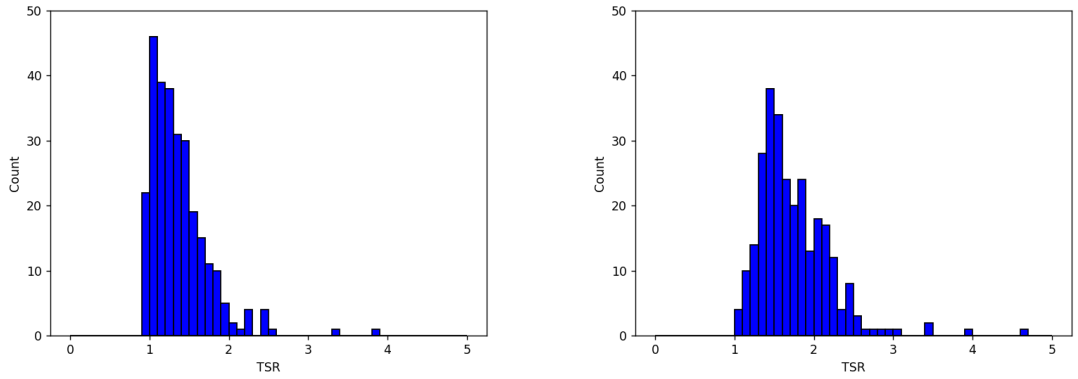
Figure 5.18: Distribution of NSR for content similarity view for (a) bots and (b) legitimate users

The comparisons show that the NSR of legitimate users is generally larger than bots. This phenomenon is more obvious in the interaction views. The reason behind it is fairly straightforward. Usually, a user will only interact with other users that he is familiar with. For a bot account, it is hard for a bot to pretend the interactions in his neighborhood. Thus, the constructed interaction views of bots are usually sparser than legitimate users.

Figure 5.18 shows that the values of NSR are much smaller than other views. The reason for the differences is that the matrices $K^{(7)}$ are generally much denser than the similarity matrices of other views. Based on the definition of views, the content similarity view is computed based on the Hellinger Distance of topics. On Twitter, a user may not have interactions with other users, especially strangers, but it is unlikely that he does not post any tweets. As long as a user posts something, content similarity will exist.

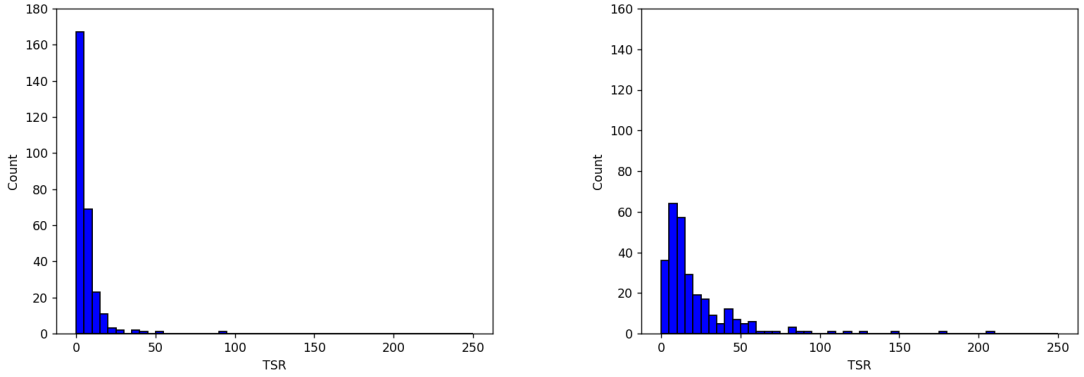
During experiments, we observe that the last view, content similarity view, has a large effect on the computed TSR. Thus, we compared the TSR computed with all seven views and the TSR only with the first six views as shown in Figure 5.19 and Figure 5.20 respectively. The comparison shows that the differences in TSR between bots and legitimate users are more obvious when the influence of content similarity view is excluded. This phenomenon indicates that the content similarity is less effective than interaction related feature in terms of distinguishing social bots and

legitimate users.



(a) (b)

Figure 5.19: Distribution of TSR with seven views for (a) bots and (b) legitimate users



(a) (b)

Figure 5.20: Distribution of TSR with first six views for (a) bots and (b) legitimate users

5.7.3 Content Analysis

From Figure 5.18, we observed that the differences between the neighborhoods of bots and legitimate users are smaller than we expected. To further understand the content-wise property of neighborhoods of both side (bots and legitimate users), we conducted further content analysis.

We model the content similarity as the topic distribution similarity. In topic modeling, a topic

Table 5.3: Top 20 frequent words used in the neighborhoods of bots and legitimate users

Bots	Legitimate Users
new	like
get	one
free	new
one	love
day	get
like	good
video	day
love	amp
amp	today
make	time
check	happy
good	know
today	got
great	see
time	u
us	go
best	people
know	thanks
online	great
people	back

is modeled as a distribution of words. We start our analysis by first studying the words used by the neighborhood of both sides. We extract the tokens as well as the corresponding frequencies in the corpus. Then we sort the tokens in the descending order of the frequencies to get the frequent words used in posted tweets. As an example, Table 5.3 shows the top 20 frequent words used in the neighborhoods of bots and legitimate users.

We see a large overlap exists from words in the two columns. To measure how many words are overlapped in the frequent words, we define an overlap ratio for N as the number of overlapped words in the top N frequent words on both sides. We calculate the overlap ratio for N from 2,000 to 50,000 with the step length of 2,000. The results are shown in Figure 5.21.

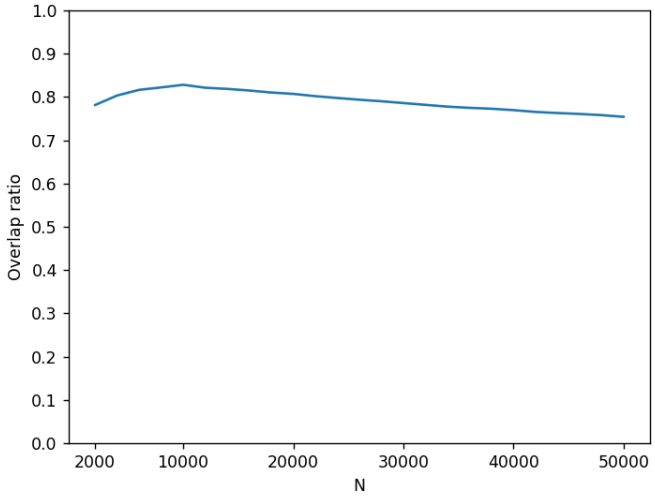
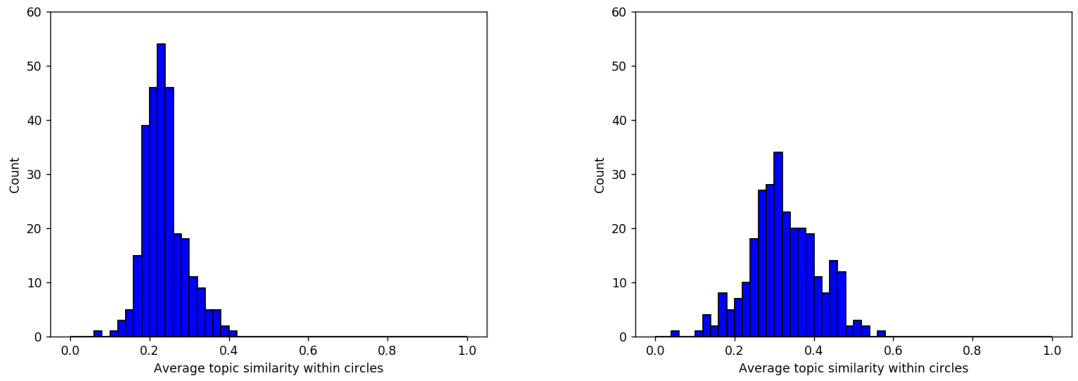


Figure 5.21: Overlap ratio of top N frequent words used by neighborhoods of social bots and legitimate users

The overlap ratios between neighborhoods of bots and legitimate users are above 70% from top 2,000 to top 50,000 frequent words. The highly overlapped frequent words indicate that the topics from social bots and legitimate users are likely to be similar since topics are inferred based on the co-occurrences of words. We conduct further analysis of the inferred topics based on the clustering results of social circles. We compute the pairwise topic similarity within each circle. For example, if a circle consists of 5 users, then the number of pairs for topic similarity is $\binom{5}{2} = 10$. For each circle, we use the average of all the pairwise topic similarities to represent the content similarity of that circle. The distribution of the average content similarity per circle is shown in Figure 5.22.

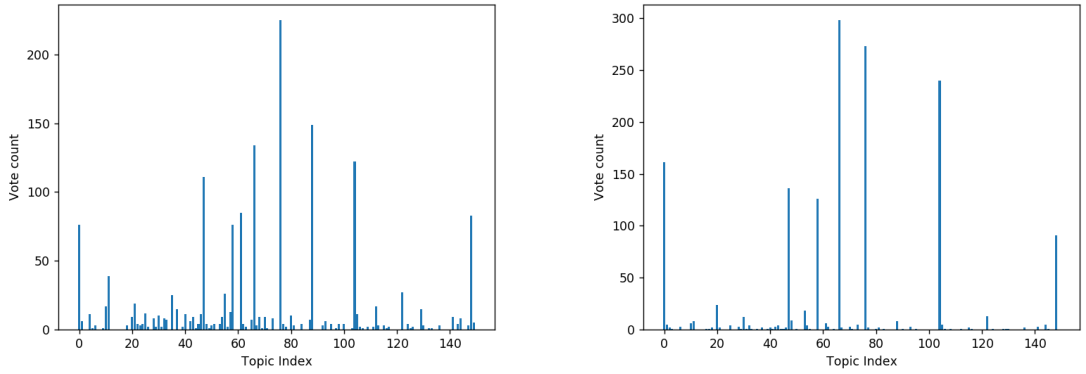


(a) (b)

Figure 5.22: Distribution of average content similarity per circle for (a) bots and (b) legitimate users

Figure 5.22 shows that the shapes of both of the topic similarity distributions are close to the "bell shape", indicating both distributions roughly follow normal distributions. Compared to bots, the circle-wise content similarity within the neighborhoods of legitimate users has a larger standard deviation.

Lastly, we study the actual topics inferred from the tweets. We conduct this analysis on circle level and determined the topics for each circle by majority vote. Based on the topic distribution vectors, we let the users in each circle "vote" for the topics with the largest probability. For each circle, the topic with the most votes is considered as the topic for this circle. By aggregating the votes from all circles, we can get the "popularity" of the topics, as shown in Figure 5.23.



(a) (b)

Figure 5.23: Circle-level topic vote for (a) bots and (b) legitimate users

The figures show that the popular topics (the high bars) on both sides are fairly similar. We can see that the topic votes of neighborhoods in bots are more scattered than legitimate users. Combined with the comparison in Figure 5.22, we can conclude that in the discovered circles of legitimate users' neighborhoods, the distributions of topics are more condensed toward the dominate topics, which explained the higher standard deviation observed in Figure 5.22. In other words, the dominance of popular topics in the circles of bots' neighborhood is less obvious, thus it is more likely for users within the same circle have similar topic similarities.

We choose the top three topics from each side to examine what the popular topics represent. Table 5.4 display each topic with the top ten frequent words. The topics shown in the table are not really about concrete concepts, but the words that users usually use when posting Tweets. We can see that these popular topics are consistent with the aforementioned frequently used words. Thus, the content analysis indicates that the topics inferred from tweets from the neighborhoods of both sides are similar.

5.7.4 Social Bot Detection

We conduct experiments of social bot detection using both supervised (classification) and unsupervised learning (clustering) methods. During our experiments, we observe that the features extracted

Table 5.4: Top 3 topics for bots and legitimate users

Bots			Legitimate Users		
Topic 76	Topic 88	Topic 66	Topic 66	Topic 76	Topic 104
today	need	like	like	today	like
day	help	get	get	day	one
good	know	people	people	good	think
time	use	know	know	time	would
love	make	cant	cant	love	well
morning	work	really	really	morning	good
work	learn	one	one	work	thats
get	article	never	never	get	ive
back	find	want	want	back	get
one	get	even	even	one	really

from the content similarity view show little contribution to the performances. Thus, we do not include the features extracted from the content similarity view. The implementations of the machine learning models we apply are from the Python library scikit-learn [101].

Baseline Features for Comparison. We adopt some of the simple user-level features proposed in [60] as the baseline. The authors defined several account-level features that can be divided into four categories, *User Demographics* (UD), *User Friendship Networks* (UFN), *User Content* (UC), and *User History* (UH). For comparison, we only adopt the UD and UFN features as our baseline. The UD features are features extracted from the descriptive information about a user account, including the length of the screen name, length of description, and the longevity of the account. The UFN features are features extracted from friendship information, including the number of following and followers, the ratio between the number of following and followers, the percentage of bidirectional friends, and the standard deviation of numerical IDs of following and followers. As a result, there are ten features in our baseline. The details of these features can be found in [60]. We refer to these baseline features as UDUFN.

Supervised Learning with Random Forest. As mentioned in Section 5.6, we choose Random Forest as the classifier for our supervised learning experiments. The evaluation metrics we adopt for evaluating the classification results are *accuracy* (ACC), *precision* (P), *recall* (R), *F-measure* (here we use F_1 score), and *area under the curve* (AUC). Each classification experiment is con-

ducted with 5-fold cross-validation (CV). To neutralize the effect of randomness, each CV process is repeated 20 times and the average performance is used as the final evaluation result for each metric. All experiments are conducted with the same classification setting to ensure fairness. The results are shown in Table 5.5.

Table 5.5: Classification performances for different feature sets using Random Forest

Feature Set	ACC	P	R	F_1	AUC
GL	0.9201	0.9177	0.9230	0.9203	0.9531
CC	0.9148	0.9181	0.9125	0.9145	0.9517
NSR	0.8938	0.8889	0.9003	0.8936	0.9445
GL+CC	0.9159	0.9155	0.9162	0.9151	0.9582
GL+NSR	0.9186	**0.9191**	0.9176	0.9179	0.9615
CC+NSR	0.9086	0.9027	0.9161	0.9087	0.9532
GL+CC+NSR	0.9150	0.9136	0.9183	0.9152	0.9601
UDUFN	**0.9307**	0.8939	**0.9794**	**0.9339**	**0.9680**

We can see that the baseline features still slightly outperforms our neighborhood-based features in metrics of ACC, recall, F_1 score, and AUC. Among all the features we have proposed, GL had the best performance in terms of ACC while GL combined with NSR performed best in AUC. The reason that our features do not beat the performance of the baseline features is that usually a neighborhood of a seed user may be mixed with legitimate users and bots.

Our proposed features are generated from the users' neighborhoods while the baseline features are extracted from seed users themselves. Theoretically, our proposed features should provide complementary information for the baseline features. Thus, we further conduct experiments by combining our proposed features with the baseline features. The settings for the classifications are the same. The results are shown in Table 5.6.

Table 5.6: Classification performances for different feature sets combined with baseline features using Random Forest

Feature Set	ACC	P	R	F_1	AUC
UDUFN+GL	**0.9642**	0.9776	0.9500	**0.9632**	0.9915
UDUFN+CC	0.9575	0.9716	0.9434	0.9569	0.9914
UDUFN+NSR	0.9586	0.9581	0.9602	0.9585	**0.9937**
UDUFN+GL+CC	0.9596	0.9751	0.9447	0.9592	0.9893
UDUFN+GL+NSR	0.9636	**0.9778**	0.9488	0.9627	0.9920
UDUFN+CC+NSR	0.9570	0.9732	0.9399	0.9558	0.9914
UDUFN+GL+CC+NSR	0.9593	0.9728	0.9450	0.9583	0.9901
UDUFN	0.9307	0.8939	**0.9794**	0.9339	0.9680

From the results, we can observe improved performance. Among all the combinations, the one combined with GL achieved the best ACC while the one combined with NSR achieved the best AUC. The classification results show that our proposed features indeed provide complementary information for bot detection as the combined features achieve better performance.

Unsupervised Learning with Agglomerative Clustering. We choose Agglomerative Clustering for unsupervised learning experiments. In each step, Agglomerative Clustering merges the two clusters based on the computed affinity among clusters. We use Euclidean Distance as the metric for affinity calculation, which is the default setting in the implementation we apply. During our experiments, we observe that the *NSR features* do not work well, so we exclude the entire NSR feature set from the results. With ground truth of the class labels available, we use some external evaluation metrics, including *accuracy* (ACC), *F-measure* (here we use F_1 score), *rand index* (RI), *adjusted rand index* (ARI), and *normalized mutual information* (NMI). Among all the metrics, ACC and F_1 are calculated just like in supervised learning. The metric RI measures the similarity between two clustering assignments (e.g., the prediction and the ground truth of labels) and is related to *accuracy*. The range of RI is $[0, 1.0]$. The ARI is the corrected-for-chance version for RI

[140] and takes randomness into account. The range of ARI is $[-1.0, 1.0]$ and a random labeling has an ARI close to 0. NMI also measures the similarity between the two clustering assignments with a range of $[0, 1.0]$. Note in the unsupervised learning experiments, since the output result of Agglomerative Clustering is stable, we do not need to repeat each experiment multiple iterations and get the average performance. The results of clustering performance are shown in Table 5.7.

Table 5.7: Clustering performances for different feature sets using Agglomerative Clustering

Feature Set	ACC	F_1	RI	ARI	NMI
GL	0.8036	0.7755	0.6837	0.3676	0.3233
CC	0.7250	0.6638	0.6005	0.2013	0.1913
GL+CC	**0.8196**	**0.8000**	**0.7038**	**0.4077**	**0.3453**
UDUFN	0.7446	0.6801	0.6190	0.2383	0.2464

The results show that only the classifier trained with circle features (CC) is outperformed by the baseline. The combined feature set (GL with CC) achieves the best performance. Unlike in the experiments with supervised learning methods, the circle features are capable of boosting the performance of global features, meaning the discovered of social circles provide complementary information for computing the affinity between seed users. In addition, the better performance of our proposed global feature with unsupervised learning method indicates the properties of users' neighborhoods are effective indicators of the users themselves.

5.7.5 Discussions

From the results shown in the classification experiments, we observe that the classifiers trained with features defined in the GL category always achieve the best ACC. When combining with other features, the results do not show improvement in performance. On the other hand, the classifiers trained without the GL features never outperformed the one trained singly using GL features. Thus, we can conclude that in classification, the features directly extracted from the graphs (GL features) are capable to capture the differences in neighborhoods between bots and legitimate users better

Table 5.8: Feature importance of GL feature set with Random Forest

View name	Graph metric	Feature importance
friendship	closeness centrality	0.0170
	clustering coefficient	0.0313
	density	0.0235
shared friends	closeness centrality	0.0178
	clustering coefficient	0.0122
	density	0.0150
reply	closeness centrality	0.1359
	clustering coefficient	0.0710
	density	0.1796
retweet	closeness centrality	0.0923
	clustering coefficient	0.0244
	density	0.1252
co-reply	closeness centrality	0.0970
	clustering coefficient	0.0072
	density	0.0883
co-retweet	closeness centrality	0.0265
	clustering coefficient	0.0099
	density	0.0250

than features extracted in other ways. Also, the classification results indicate that the discovered social circles did not show significant improvement in the performance of detecting social bots.

Interpretation of Feature Importance. During our experiments, we also discover that the graph metrics we adopted worked differently with different graphs. There is no single graph metric that is optimal across different graphs. We output the feature importance of the Random Forest model fitted using the GL features in the experiment. Note that the GL features include 18 features in total (3 graph metrics $\times$ 6 types of graphs). The results are shown in Table 5.8.

From the importance of the features, we see different trends in different types of graphs. For friendship view, the clustering coefficient is more important than the other two. Thus, whether nodes are showing cliques is more important in the simple topology. On the other hand, in the four interaction-based views, the clustering coefficient is the least important one. For a single node in a network, closeness centrality measures how centralized the node is. The higher the centrality is, the easier for it to spread the information. As for the density of a network, it measures how connected the nodes in the network are. These two metrics reflects the extent of interactions better than the

clustering coefficient. We also notice that the importance of the indirect interactions (co-reply and co-retweet) is generally lower than direct interactions (reply and retweet).

Feature Robustness. The features adopted by existing bot detection methods are mainly about the behaviors or descriptive information of the individual users, for example, the number of connections or the strength of the activities. These features rely on the fact that differences exist between the behaviors of bots and legitimate users. However, the bots are capable of evolving and learning to imitate the behaviors of human. In other words, the user-level features can be easily manipulated to trick the behavior-based detectors.

On the other hand, the features we proposed in this work exclude the influence of the seed users themselves in the very beginning. Instead of focusing on the characteristics of the users, we study the properties of the neighborhoods of the users. Note in the real world, it is much harder to manipulate and change the behaviors of a large network, especially when the majority of users in the network are not in control. Thus, the neighborhood-based features we have proposed are more resilient.

Effectiveness of Social Circles in Social Bot Detection We have introduced three major sets of features, global (GL), circle (CC), and NSR. The circle features and NSR rely on the discovered social circles. From the results of the experiments, we have observed that the circle features only show a contribution to the performance with unsupervised learning methods. On the other hand, the classification results show that combining NSR helps improve AUC with a slight cost of lower recall (R). The area under the curve is computed with the true positive rate (TPR, a.k.a recall) and the false positive rate (FPR). With lower TPR, the improved AUC means reduced FPR, which is equivalent to improved true negative rate (TNR). Thus, introducing NSR helps the detection of social bots with slightly cost of misclassification of legitimate users. Overall, the experiments show that the hidden social circles are valuable in social bot detection since the ground truth are not always available.

5.8 Summary

In this chapter, we study the problem of detecting social bots by utilizing the properties of neighborhoods and proposed a novel bot detection scheme. We discover that legitimate users could get involved with the neighborhoods of social bots. The comparisons of the degree distributions indicate the similarity of the general topology of the neighborhoods between bots and legitimate users. To capture the latent differences, we consider seven different views to represent the users' neighborhood and adopted several graph metrics as features. Furthermore, we assume that there will be natural social circles formed within the neighborhood of legitimate users. While for bots, since their neighborhoods are usually constructed by algorithms deliberately, the hidden social circles within their neighborhoods would be different as legitimate users. To discover these hidden social circles, we apply a multi-view clustering method SCSC that is capable to integrate data from multiple sources.

We have conducted thorough experiments to study how our proposed neighborhood-based features would help with bot detection. Experimental results in classification show that our proposed features are capable of providing complementary information for the simple user-level features. We also observe that neighborhood-based features outperform the baseline features in experiments with clustering, indicating that our proposed neighborhood-based feature are more effective in capturing the distinguishable differences between social bots and legitimate users. Overall, we discover that the neighborhoods of users are valuable for distinguishing social bots and legitimate users.

Chapter 6

Conclusion

The increasing popularity of OSNs has not only encouraged more users to communicate online but also attracted malicious spammers to disseminate harmful information. In this dissertation, we propose a thorough study on measuring the credibility of user-generated content and detecting spammers in OSNs.

First, we study the problem of measuring the trustworthiness of users with rating deviations. In online review systems like Yelp, the numerical rating is a straightforward reflection of the quality of the target business. We assume that spammers tend to be lazy in maintaining their social images and have less social relations than legitimate users. We propose an approach that integrates the strengths of social relations and rating deviations to compute the overall trustworthiness score for users. The experimental results indicate a strong correlation between social relations and the trustworthiness of users.

Second, we observe that the review content carries fine-grained information of users' opinions toward a target business than numerical ratings. We study the problem of inferring trustworthiness from the review content. We model the opinions of a user as tuples of sentiment toward a specific category. We apply both supervised and unsupervised methods to extract users' opinions. Then we propose a trust propagation method that is able to measure the trustworthiness of users, reviews, and statements at the same time. We show that the trustworthiness of a user is closely related to the content he posted.

Lastly, we study the problem of detecting social bots by examining the neighborhoods formed around the users, with the assumption that the behavioral patterns inside the neighborhood are difficult for social bots to fake. Besides the global properties of neighborhoods, we take a closer

look at the hidden social circles formed within the neighborhoods and apply a multi-view clustering method to discover them. We extract features from the global graphs and social circles and conduct experiments using both supervised and unsupervised models. The experimental results show that our proposed neighborhood-based features are useful for social bot detection.

In conclusion, we study the problem of measuring the credibility of UGC and detecting spammers in OSNs in several different aspects, in the goal of helping users understand the trustworthiness of online information spread on social networks.

References

[1] Adomavicius, G., Kamireddy, S., & Kwon, Y. (2007). Towards more confident recommendations: Improving recommender systems using filtering approach based on rating variance. In *Proc. of the 17th Workshop on Information Technology and Systems* (pp. 152–157).

[2] Akoglu, L., Chandy, R., & Faloutsos, C. (2013). Opinion fraud detection in online reviews by network effects. In *ICWSM*.

[3] Aparicio, S., Villazón-Terrazas, J., & Álvarez, G. (2015). A model for scale-free networks: application to twitter. *Entropy*, 17(8), 5848–5867.

[4] Baccianella, S., Esuli, A., & Sebastiani, F. (2010). Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *LREC*, volume 10 (pp. 2200–2204).

[5] Barnes, E. R. (1982). An algorithm for partitioning the nodes of a graph. *SIAM Journal on Algebraic Discrete Methods*, 3(4), 541–550.

[6] Bedi, P., Kaur, H., & Marwaha, S. (2007). Trust based recommender system for the semantic web. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, IJCAI'07 (pp. 2677–2682).

[7] Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *the Journal of machine Learning research*, 3, 993–1022.

[8] Blum, A. & Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory* (pp. 92–100).: ACM.

[9] Bodnar, T., Tucker, C., Hopkinson, K., & Bilén, S. G. (2014). Increasing the veracity of event detection on social media networks through user trust modeling. In *Big Data (Big Data), 2014 IEEE International Conference on* (pp. 636–643).: IEEE.

[10] Boshmaf, Y., Muslukhov, I., Beznosov, K., & Ripeanu, M. (2011). The socialbot network: when bots socialize for fame and money. In *Proceedings of the 27th annual computer security applications conference* (pp. 93–102).: ACM.

[11] Boshmaf, Y., Muslukhov, I., Beznosov, K., & Ripeanu, M. (2013). Design and analysis of a social botnet. *Computer Networks*, 57(2), 556–578.

[12] BrightLocal (2017). Brightlocal local consumer review survey 2017. `https://www.brightlocal.com/learn/local-consumer-review-survey/`.

[13] Brody, S. & Elhadad, N. (2010). An unsupervised aspect-sentiment model for online reviews. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics* (pp. 804–812).: Association for Computational Linguistics.

[14] Cao, Q., Sirivianos, M., Yang, X., & Pregueiro, T. (2012). Aiding the detection of fake accounts in large scale social online services. In *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation* (pp. 15–15).: USENIX Association.

[15] Chang, J., Boyd-Graber, J., & Blei, D. M. (2009). Connections between the lines: augmenting social networks with text. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 169–178).: ACM.

[16] Chen, Z., Mukherjee, A., & Liu, B. (2014). Aspect extraction with automated prior knowledge learning. In *ACL (1)* (pp. 347–358).

[17] David, I., Siordia, O. S., & Moctezuma, D. (2016). Features combination for the detection of

malicious twitter accounts. In *2016 IEEE International Autumn Meeting on Power, Electronics and Computing (ROPEC)* (pp. 1–6).: IEEE.

[18] Davis, C. A., Varol, O., Ferrara, E., Flammini, A., & Menczer, F. (2016). Botornot: A system to evaluate social bots. In *Proceedings of the 25th International Conference Companion on World Wide Web* (pp. 273–274).: International World Wide Web Conferences Steering Committee.

[19] dos Santos, C. & Gatti, M. (2014). Deep convolutional neural networks for sentiment analysis of short texts. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers* (pp. 69–78).

[20] Elyashar, A., Fire, M., Kagan, D., & Elovici, Y. (2013). Homing socialbots: intrusion on a specific organization's employee using socialbots. In *2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2013)* (pp. 1358–1365).: IEEE.

[21] Fahrni, A. & Klenner, M. (2008). Old wine or warm beer: Target-specific sentiment analysis of adjectives. In *Proc. of the Symposium on Affective Language in Human and Machine, AISB* (pp. 60–63).

[22] Fang, X. & Zhan, J. (2015). Sentiment analysis using product review data. *Journal of Big Data*, 2(1), 5.

[23] Fayyad, U. & Irani, K. (1993). Multi-interval discretization of continuous-valued attributes for classification learning.

[24] Fei, G., Mukherjee, A., Liu, B., Hsu, M., Castellanos, M., & Ghosh, R. (2013). Exploiting burstiness in reviews for review spammer detection. In *ICWSM*: Citeseer.

[25] Feng, S., Banerjee, R., & Choi, Y. (2012a). Syntactic stylometry for deception detection. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2* (pp. 171–175).: Association for Computational Linguistics.

[26] Feng, S., Xing, L., Gogar, A., & Choi, Y. (2012b). Distributional footprints of deceptive product reviews. In *Sixth International AAAI Conference on Weblogs and Social Media*.

[27] Feng, V. W. & Hirst, G. (2013). Detecting deceptive opinions with profile compatibility. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing* (pp. 338–346).

[28] Ferrara, E., Varol, O., Davis, C., Menczer, F., & Flammini, A. (2016). The rise of social bots. *Communications of the ACM*, 59(7), 96–104.

[29] Fortunato, S. (2010). Community detection in graphs. *Physics reports*, 486(3-5), 75–174.

[30] Friedman, J., Hastie, T., & Tibshirani, R. (2001). *The elements of statistical learning*, volume 1. Springer series in statistics New York.

[31] Ganu, G., Elhadad, N., & Marian, A. (2009). Beyond the stars: Improving rating predictions using review text content. In *WebDB*, volume 9 (pp. 1–6).

[32] Gao, P., Miao, H., Baras, J. S., & Golbeck, J. (2016). Star: Semiring trust inference for trust-aware social recommenders. In *Proceedings of the 10th ACM Conference on Recommender Systems* (pp. 301–308).: ACM.

[33] Girvan, M. & Newman, M. E. (2002). Community structure in social and biological networks. *Proceedings of the national academy of sciences*, 99(12), 7821–7826.

[34] Grier, C., Thomas, K., Paxson, V., & Zhang, M. (2010). @ spam: the underground on 140 characters or less. In *Proceedings of the 17th ACM conference on Computer and communications security* (pp. 27–37).: ACM.

[35] Guzman, E. & Maalej, W. (2014). How do users like this feature? a fine grained sentiment analysis of app reviews. In *Requirements Engineering Conference (RE), 2014 IEEE 22nd International* (pp. 153–162).: IEEE.

[36] Hai, Z., Chang, K., Kim, J.-J., & Yang, C. C. (2013). Identifying features in opinion mining via intrinsic and extrinsic domain relevance. *IEEE Transactions on Knowledge and Data Engineering*, 26(3), 623–634.

[37] Harris, Z. (1968). Mathematical structures of language. *Interscience tracts in pure and applied mathematics*.

[38] Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1), 5–53.

[39] Hu, M. & Liu, B. (2004). Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 168–177).: ACM.

[40] Inc., Y. (2009). Why yelp has a review filter. `https://www.yelpblog.com/2009/10/why-yelp-has-a-review-filter`.

[41] Inc., Y. (2010). Yelp's recommendation software explained. `https://www.yelpblog.com/2010/03/yelp-review-filter-explained`.

[42] Jamali, M. & Ester, M. (2009). Trustwalker: A random walk model for combining trust-based and item-based recommendation. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09 (pp. 397–406). New York, NY, USA: ACM.

[43] Ji, Y., He, Y., Jiang, X., Cao, J., & Li, Q. (2016). Combating the evasion mechanisms of social bots. *computers & security*, 58, 230–249.

[44] Jijkoun, V. & Hofmann, K. (2009). Generating a non-english subjectivity lexicon: relations that matter. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics* (pp. 398–405).: Association for Computational Linguistics.

[45] Jindal, N. & Liu, B. (2008). Opinion spam and analysis. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*, WSDM '08 (pp. 219–230). New York, NY, USA: ACM.

[46] Jindal, N., Liu, B., & Lim, E.-P. (2010). Finding unusual review patterns using unexpected rules. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10 (pp. 1549–1552). New York, NY, USA: ACM.

[47] Jo, Y. & Oh, A. H. (2011). Aspect and sentiment unification model for online review analysis. In *Proceedings of the fourth ACM international conference on Web search and data mining* (pp. 815–824).: ACM.

[48] Kennedy, A. & Inkpen, D. (2006). Sentiment classification of movie reviews using contextual valence shifters. *Computational intelligence*, 22(2), 110–125.

[49] Kernighan, B. W. & Lin, S. (1970). An efficient heuristic procedure for partitioning graphs. *Bell system technical journal*, 49(2), 291–307.

[50] Kiritchenko, S., Zhu, X., Cherry, C., & Mohammad, S. (2014). Nrc-canada-2014: Detecting aspects and sentiment in customer reviews. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)* (pp. 437–442).

[51] Klein, D. & Manning, C. D. (2003). Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1* (pp. 423–430).: Association for Computational Linguistics.

[52] Kleinberg, J. M. (1999a). Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5), 604–632.

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

[54] Konishi, T., Tezuka, T., Kimura, F., & Maeda, A. (2012). Estimating aspects in online reviews using topic model with 2-level learning. In *Proceedings of the International MultiConference of Engineers and Computer Scientists*, volume 1 (pp. 120–126).

[55] Konstas, I., Stathopoulos, V., & Jose, J. M. (2009). On social networks and collaborative recommendation. In *Proceedings of the 32Nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09 (pp. 195–202).

[56] Kudugunta, S. & Ferrara, E. (2018). Deep neural networks for bot detection. *Information Sciences*, 467, 312–322.

[57] Kumar, A. & Daumé, H. (2011). A co-training approach for multi-view spectral clustering. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)* (pp. 393–400).

[58] Kwon, Y. (2008). Improving top-n recommendation techniques using rating variance. In *Proceedings of the 2008 ACM Conference on Recommender Systems*, RecSys '08 (pp. 307–310).

[59] Lan, C., Yang, Y., Li, X., Luo, B., & Huan, J. (2017). Learning social circles in ego-networks based on multi-view network structure. *IEEE Transactions on Knowledge and Data Engineering*, 29(8), 1681–1694.

[60] Lee, K., Eoff, B. D., & Caverlee, J. (2011). Seven months with the devils: A long-term study of content polluters on twitter. In *Fifth International AAAI Conference on Weblogs and Social Media*.

[61] Lee, W.-P. & Ma, C.-Y. (2016). Enhancing collaborative recommendation performance by combining user preference and trust-distrust propagation in social networks. *Knowledge-Based Systems*, 106, 125–134.

[62] Li, H., Chen, Z., Mukherjee, A., Liu, B., & Shao, J. (2015). Analyzing and detecting opinion spam on a large-scale dataset via temporal and spatial patterns. In *ninth international AAAI conference on web and social Media*.

[63] Li, H., Fei, G., Wang, S., Liu, B., Shao, W., Mukherjee, A., & Shao, J. (2017a). Bimodal distribution and co-bursting in review spam detection. In *Proceedings of the 26th International Conference on World Wide Web* (pp. 1063–1072).: International World Wide Web Conferences Steering Committee.

[64] Li, L., Qin, B., Ren, W., & Liu, T. (2017b). Document representation and feature combination for deceptive spam review detection. *Neurocomputing*, 254, 33–41.

[65] Lim, E.-P., Nguyen, V.-A., Jindal, N., Liu, B., & Lauw, H. W. (2010). Detecting product review spammers using rating behaviors. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10 (pp. 939–948). New York, NY, USA: ACM.

[66] Liu, B. (2012). Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1), 1–167.

[67] Liu, Q., Gao, Z., Liu, B., & Zhang, Y. (2013). A logic programming approach to aspect extraction in opinion mining. In *Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2013 IEEE/WIC/ACM International Joint Conferences on*, volume 1 (pp. 276–283).: IEEE.

[68] Liu, Q., Gao, Z., Liu, B., & Zhang, Y. (2015). Automated rule selection for aspect extraction in opinion mining. In *IJCAI*, volume 15 (pp. 1291–1297).

[69] Liu, Q., Liu, B., Zhang, Y., Kim, D. S., & Gao, Z. (2016). Improving opinion aspect extraction using semantic similarity and aspect associations. In *AAAI* (pp. 2986–2992).

[70] Liu, Y., Niculescu-Mizil, A., & Gryc, W. (2009). Topic-link lda: joint models of topic and

author community. In *proceedings of the 26th annual international conference on machine learning* (pp. 665–672).: ACM.

[71] Lovász, L. (1993). Random walks on graphs: A survey. *Combinatorics, Paul erdos is eighty*, 2(1), 1–46.

[72] Luca, M. & Zervas, G. (2016). Fake it till you make it: Reputation, competition, and yelp review fraud. *Management Science*, 62(12), 3412–3427.

[73] Ma, H., Yang, H., Lyu, M. R., & King, I. (2008). Sorec: Social recommendation using probabilistic matrix factorization. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, CIKM '08 (pp. 931–940).

[74] Ma, H., Zhou, D., Liu, C., Lyu, M. R., & King, I. (2011). Recommender systems with social regularization. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, WSDM '11 (pp. 287–296).

[75] Madisetty, S. & Desarkar, M. S. (2018). A neural network-based ensemble approach for spam detection in twitter. *IEEE Transactions on Computational Social Systems*, 5(4), 973–984.

[76] Massa, P. & Avesani, P. (2004). Trust-aware collaborative filtering for recommender systems. In *On the Move to Meaningful Internet Systems 2004: CoopIS, DOA, and ODBASE* (pp. 492–508). Springer.

[77] Massa, P. & Avesani, P. (2007a). Trust-aware recommender systems. In *Proceedings of the 2007 ACM conference on Recommender systems* (pp. 17–24).: ACM.

[78] Massa, P. & Avesani, P. (2007b). Trust-aware recommender systems. In *Proceedings of the 2007 ACM Conference on Recommender Systems*, RecSys '07 (pp. 17–24). New York, NY, USA: ACM.

[79] Massa, P. & Bhattacharjee, B. (2004). Using trust in recommender systems: An experimental

analysis. In *Trust Management*, volume 2995 of *Lecture Notes in Computer Science* (pp. 221–235). Springer Berlin Heidelberg.

[80] McAuley, J., Leskovec, J., & Jurafsky, D. (2012). Learning attitudes and attributes from multi-aspect reviews. In *2012 IEEE 12th International Conference on Data Mining* (pp. 1020–1025).: IEEE.

[81] Mel'cuk, I. A. et al. (1988). *Dependency syntax: theory and practice*. SUNY press.

[82] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

[83] Miller, G. A. (1995). Wordnet: a lexical database for english. *Communications of the ACM*, 38(11), 39–41.

[84] Mislove, A., Marcon, M., Gummadi, K. P., Druschel, P., & Bhattacharjee, B. (2007). Measurement and analysis of online social networks. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement* (pp. 29–42).: ACM.

[85] Moradi, P. & Ahmadian, S. (2015). A reliability-based recommendation method to improve trust-aware recommender systems. *Expert Systems with Applications*, 42(21), 7386–7398.

[86] Mukherjee, A., Kumar, A., Liu, B., Wang, J., Hsu, M., Castellanos, M., & Ghosh, R. (2013a). Spotting opinion spammers using behavioral footprints. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 632–640).: ACM.

[87] Mukherjee, A. & Liu, B. (2012). Aspect extraction through semi-supervised modeling. In *Proceedings of the 50th annual meeting of the association for computational linguistics: Long papers-volume 1* (pp. 339–348).: Association for Computational Linguistics.

[88] Mukherjee, A., Liu, B., Wang, J., Glance, N., & Jindal, N. (2011). Detecting group review

spam. In *Proceedings of the 20th International Conference Companion on World Wide Web*, WWW '11 (pp. 93–94).

[89] Mukherjee, A., Venkataraman, V., Liu, B., & Glance, N. S. (2013b). What yelp fake review filter might be doing? In *ICWSM*.

[90] Mukherjee, S., Dutta, S., & Weikum, G. (2016). Credible review detection with limited information using consistency features. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (pp. 195–213).: Springer.

[91] nbcnews (2014). Be wary of awesome and scathing online reviews. `https://www.nbcnews.com/business/consumer/be-wary-awesome-scathing-online-reviews-n72116`.

[92] Newman, M. E. (2004). Fast algorithm for detecting community structure in networks. *Physical review E*, 69(6), 066133.

[93] Ng, A. Y., Jordan, M. I., & Weiss, Y. (2002). On spectral clustering: Analysis and an algorithm. In *Advances in neural information processing systems* (pp. 849–856).

[94] Nilizadeh, S., Labrèche, F., Sedighian, A., Zand, A., Fernandez, J., Kruegel, C., Stringhini, G., & Vigna, G. (2017). Poised: Spotting twitter spam off the beaten paths. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (pp. 1159–1174).: ACM.

[95] Ott, M., Cardie, C., & Hancock, J. T. (2013). Negative deceptive opinion spam. In *Proceedings of the 2013 conference of the north american chapter of the association for computational linguistics: human language technologies* (pp. 497–501).

[96] Ott, M., Choi, Y., Cardie, C., & Hancock, J. T. (2011). Finding deceptive opinion spam by any stretch of the imagination. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1* (pp. 309–319).: Association for Computational Linguistics.

[97] Page, L., Brin, S., Motwani, R., & Winograd, T. (1999). *The PageRank Citation Ranking: Bringing Order to the Web.* Technical Report 1999-66, Stanford InfoLab. Previous number = SIDL-WP-1999-0120.

[98] Pang, B. & Lee, L. (2004). A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd annual meeting on Association for Computational Linguistics* (pp. 271).: Association for Computational Linguistics.

[99] Pang, B., Lee, L., et al. (2008). Opinion mining and sentiment analysis. *Foundations and Trends® in Information Retrieval*, 2(1–2), 1–135.

[100] Paradise, A., Puzis, R., & Shabtai, A. (2014). Anti-reconnaissance tools: Detecting targeted socialbots. *IEEE Internet Computing*, 18(5), 11–19.

[101] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.

[102] Peng, Q. & Zhong, M. (2014). Detecting spam review through sentiment analysis. *JSW*, 9(8), 2065–2072.

[103] Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (pp. 1532–1543).

[104] Pontiki, M., Galanis, D., Pavlopoulos, J., Papageorgiou, H., Androutsopoulos, I., & Manandhar, S. (2014). Semeval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)* (pp. 27–35).

[105] Popescu, A.-M. & Etzioni, O. (2007). Extracting product features and opinions from reviews. In *Natural language processing and text mining* (pp. 9–28). Springer.

[106] Poria, S., Cambria, E., & Gelbukh, A. (2016). Aspect extraction for opinion mining with a deep convolutional neural network. *Knowledge-Based Systems*, 108, 42–49.

[107] Poria, S., Cambria, E., Ku, L.-W., Gui, C., & Gelbukh, A. (2014). A rule-based approach to aspect extraction from product reviews. In *Proceedings of the Second Workshop on Natural Language Processing for Social Media (SocialNLP)* (pp. 28–37).

[108] Psorakis, I., Roberts, S., Ebden, M., & Sheldon, B. (2011). Overlapping community detection using bayesian non-negative matrix factorization. *Physical Review E*, 83(6), 066114.

[109] Qiu, G., Liu, B., Bu, J., & Chen, C. (2011). Opinion word expansion and target extraction through double propagation. *Computational linguistics*, 37(1), 9–27.

[110] Radicchi, F., Castellano, C., Cecconi, F., Loreto, V., & Parisi, D. (2004). Defining and identifying communities in networks. *Proceedings of the National Academy of Sciences*, 101(9), 2658–2663.

[111] Ratkiewicz, J., Conover, M. D., Meiss, M., Gonçalves, B., Flammini, A., & Menczer, F. M. (2011). Detecting and tracking political abuse in social media. In *Fifth international AAAI conference on weblogs and social media*.

[112] Řehůřek, R. & Sojka, P. (2010). Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks* (pp. 45–50). Valletta, Malta: ELRA. `http://is.muni.cz/publication/884893/en`.

[113] Ren, Y. & Ji, D. (2017). Neural networks for deceptive opinion spam detection: An empirical study. *Information Sciences*, 385, 213–224.

[114] Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994). Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*, CSCW '94 (pp. 175–186).

[115] Saif, H., He, Y., Fernandez, M., & Alani, H. (2016). Contextual semantics for sentiment analysis of twitter. *Information Processing & Management*, 52(1), 5–19.

[116] Salvetti, F., Lewis, S., & Reichenbach, C. (2004). Automatic opinion polarity classification of movie. *Colorado research in linguistics*, 17(1), 2.

[117] Seo, H., L. F. P. R. X. H. & Vekapu, S. (2015). Perceived usefulness of online reviews: Effects of review characteristics and reviewer attributes. In *Proceedings of the 65th International Communication Association Annual Conference*.

[118] Severyn, A. & Moschitti, A. (2015). Twitter sentiment analysis with deep convolutional neural networks. In *Proc. of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 959–962).

[119] Singh, A. (2012). Social networking for botnet command and control.

[120] Stackla (2019). Bridging the gap: Consumer & marketing perspectives on content in the digital age. `https://stackla.com/resources/reports/bridging-the-gap-consumer-marketing-perspectives-on-content-in-the-digital-age/`.

[121] Statusbrew (2018). 100 social media statistics you must know [2018] + infographic. `https://blog.statusbrew.com/social-media-statistics-2018-for-business/`.

[122] Stein, T., Chen, E., & Mangla, K. (2011). Facebook immune system. In *Proceedings of the 4th workshop on social network systems* (pp.$\tilde{8}$).: ACM.

[123] Stringhini, G., Kruegel, C., & Vigna, G. (2010). Detecting spammers on social networks. In *Proceedings of the 26th annual computer security applications conference* (pp. 1–9).: ACM.

[124] Tang, J., Hu, X., & Liu, H. (2014). Is distrust the negation of trust?: the value of distrust in social media. In *Proceedings of the 25th ACM conference on Hypertext and social media* (pp. 148–157).: ACM.

[125] Than, C. & Han, S. (2014). Improving recommender systems by incorporating similarity, trust and reputation. *Journal of Internet Services and Information Security (JISIS)*, 4(1), 64–76.

[126] theverge (2018). Twitter now lets you report accounts that you suspect are bots. `https://www.theverge.com/2018/10/31/18048838/twitter-report-fake-accounts-spam-bot-crackdown`.

[127] Titov, I. & McDonald, R. (2008). Modeling online reviews with multi-grain topic models. In *Proceedings of the 17th international conference on World Wide Web* (pp. 111–120).: ACM.

[128] Ungar, L. H. & Foster, D. P. (1998). Clustering methods for collaborative filtering. In *AAAI workshop on recommendation systems*, volume 1.

[129] Varol, O., Ferrara, E., Davis, C. A., Menczer, F., & Flammini, A. (2017). Online human-bot interactions: Detection, estimation, and characterization. In *Eleventh international AAAI conference on web and social media*.

[130] Vydiswaran, V., Zhai, C., & Roth, D. (2011). Content-driven trust propagation framework. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 974–982).: ACM.

[131] Wang, G., Mohanlal, M., Wilson, C., Wang, X., Metzger, M., Zheng, H., & Zhao, B. Y. (2012a). Social turing tests: Crowdsourcing sybil detection. *arXiv preprint arXiv:1205.3856*.

[132] Wang, G., Xie, S., Liu, B., & Yu, P. S. (2011a). Review graph based online store review spammer detection. In *Proceedings of the 2011 IEEE 11th International Conference on Data Mining*, ICDM '11 (pp. 1242–1247). Washington, DC, USA: IEEE Computer Society.

[133] Wang, G., Xie, S., Liu, B., & Yu, P. S. (2012b). Identify online store review spammers via social review graph. *ACM Trans. Intell. Syst. Technol.*, 3(4), 61:1–61:21.

[134] Wang, T., Cai, Y., Leung, H.-f., Lau, R. Y., Li, Q., & Min, H. (2014). Product aspect extraction supervised with online domain knowledge. *Knowledge-Based Systems*, 71, 86–100.

[135] Wang, W., Pan, S. J., Dahlmeier, D., & Xiao, X. (2017). Coupled multi-layer attentions for co-extraction of aspect and opinion terms. In *AAAI* (pp. 3316–3322).

[136] Wang, X., Liu, H., & Fan, W. (2011b). Connecting users with similar interests via tag network inference. In *Proceedings of the 20th ACM international conference on Information and knowledge management* (pp. 1019–1024).: ACM.

[137] Weng, J., Lim, E.-P., Jiang, J., & He, Q. (2010). Twitterrank: finding topic-sensitive influential twitterers. In *Proceedings of the third ACM international conference on Web search and data mining* (pp. 261–270).: ACM.

[138] Wikipedia contributors (2019a). Clustering coefficient — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/w/index.php?title=Clustering_coefficient&oldid=887685015`. [Online; accessed 10-June-2019].

[139] Wikipedia contributors (2019b). Hellinger distance — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/w/index.php?title=Hellinger_distance&oldid=898120854`. [Online; accessed 8-June-2019].

[140] Wikipedia contributors (2019c). Rand index — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/w/index.php?title=Rand_index&oldid=897752689`. [Online; accessed 23-June-2019].

[141] Wikipedia contributors (2019d). Scale-free network — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/w/index.php?title=Scale-free_network&oldid=900556552`. [Online; accessed 22-June-2019].

[142] Wilson, T., Wiebe, J., & Hoffmann, P. (2009). Recognizing contextual polarity: An exploration of features for phrase-level sentiment analysis. *Computational linguistics*, 35(3), 399–433.

[143] Wu, Y., Zhang, Q., Huang, X., & Wu, L. (2009). Phrase dependency parsing for opinion mining. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3 - Volume 3*, EMNLP '09 (pp. 1533–1541). Stroudsburg, PA, USA: Association for Computational Linguistics.

[144] Xiao, C., Freeman, D. M., & Hwa, T. (2015). Detecting clusters of fake accounts in online social networks. In *Proceedings of the 8th ACM Workshop on Artificial Intelligence and Security* (pp. 91–101).: ACM.

[145] Xie, S., Wang, G., Lin, S., & Yu, P. S. (2012a). Review spam detection via temporal pattern discovery. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12 (pp. 823–831). New York, NY, USA: ACM.

[146] Xie, Y., Yu, F., Ke, Q., Abadi, M., Gillum, E., Vitaldevaria, K., Walter, J., Huang, J., & Mao, Z. M. (2012b). Innocent by association: early recognition of legitimate users. In *Proceedings of the 2012 ACM conference on Computer and communications security* (pp. 353–364).: ACM.

[147] Xu, C., Tao, D., & Xu, C. (2013). A survey on multi-view learning. *arXiv preprint arXiv:1304.5634*.

[148] Xu, X., Yuruk, N., Feng, Z., & Schweiger, T. A. (2007). Scan: a structural clustering algorithm for networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 824–833).: ACM.

[149] Yan, Z., Xing, M., Zhang, D., & Ma, B. (2015). Exprs: An extended pagerank method for product feature extraction from online consumer reviews. *Information & Management*, 52(7), 850–858.

[150] Yang, Y., Lan, C., Li, X., Luo, B., & Huan, J. (2014a). Automatic social circle detection using multi-view clustering. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management* (pp. 1019–1028).: ACM.

[151] Yang, Z., Wilson, C., Wang, X., Gao, T., Zhao, B. Y., & Dai, Y. (2014b). Uncovering social network sybils in the wild. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 8(1), 2.

[152] Yao, Y., Viswanath, B., Cryan, J., Zheng, H., & Zhao, B. Y. (2017). Automated crowdturfing attacks and defenses in online review systems. *arXiv preprint arXiv:1708.08151*.

[153] Yildirim, H. & Krishnamoorthy, M. S. (2008). A random walk method for alleviating the sparsity problem in collaborative filtering. In *Proceedings of the 2008 ACM Conference on Recommender Systems*, RecSys '08 (pp. 131–138). New York, NY, USA: ACM.

[154] Yin, X., Han, J., & Philip, S. Y. (2008). Truth discovery with multiple conflicting information providers on the web. *IEEE Transactions on Knowledge and Data Engineering*, 20(6), 796–808.

[155] You, Z., Qian, T., & Liu, B. (2018). An attribute enhanced domain adaptive model for cold-start spam review detection. In *Proceedings of the 27th International Conference on Computational Linguistics* (pp. 1884–1895).

[156] Zhao, L., Hua, T., Lu, C.-T., & Chen, R. (2016). A topic-focused trust model for twitter. *Computer Communications*, 76, 1–11.

[157] Zhou, D., Manavoglu, E., Li, J., Giles, C. L., & Zha, H. (2006). Probabilistic models for discovering e-communities. In *Proceedings of the 15th international conference on World Wide Web* (pp. 173–182).: ACM.

[158] Zhou, W., Jin, H., & Liu, Y. (2012). Community discovery and profiling with social messages. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 388–396).: ACM.

[159] Ziegler, C.-N. & Lausen, G. (2005). Propagation models for trust and distrust in social networks. *Information Systems Frontiers*, 7(4-5), 337–358.