UNIVERSITÀ DEGLI STUDI DELL'INSUBRIA - VARESE

DiSTA

Dipartimento di Scienze Teoriche e Applicate

P H D T H E S I S

to obtain the title of

PhD of Science

Specialty : Computer Science

Defended by

CÜNEYT GÜRCAN AKÇORA

PROFILING USER INTERACTIONS ON ONLINE SOCIAL NETWORKS

Advisor: Barbara Carminati

Advisor: Elena Ferrari

defended on January 7, 2013

Jury :

Reviewers :	James Joshi	-	University of Pittsburgh, USA
President :	Claudio Gentile	-	University of Insubria
Examinators :	... ...	-	...
	... ...	-	...
	... ...	-	...
Invited :	... ...	-	....

Benim oğlum büyük adam olacak diyen anneme...

Acknowledgment

I would like to express the deepest gratitude to my advisors Elena Ferrari and Barbara Carminati. In addition to their academic guidance, they have shown a great deal of patience towards me and supported my research in a way that goes beyond a professional relationship. Their caring attitude has always given me the strength to continue my research endeavor.

I would like to thank William Lucia for his work with me in hist last year in Master's degree.

My lab partners Lorenzo Bossi, Dr. Stefano Braghin, Dr. Paolo Brivio, Michele Guglielmi, Lui Geng and Dr. Pietro Colombo have contributed to my research in discussions, shown me the Italian hospitality and enriched my life greatly. Without them, my Ph.D would never be such a great experience.

I have spent many enjoyable moments with professors Alberto Trombetta, Claudio Gentile, Marco Tarini, Paolo Massazza and Mauro Ferrari. I would like to thank them for their kindness.

I would like to thank Mauro Santabarbara for his time in helping me with servers and computers that I used during my research. Not officially, but he has always been my third advisor.

Finally, I would like to thank the Italian people for their warmth and grace, which made me feel at home during these years. Your country is a heaven, not only because of your lakes, mountains and valleys, but also because of your cultured life, tolerance and kindness.

Abstract

Over the last couple of years, there has been significant research effort in mining user behavior on online social networks for applications ranging from sentiment analysis to marketing. In most of those applications, usually a snapshot of user attributes or user relationships are analyzed to build the data mining models, without considering how user attributes and user relationships can be utilized together.

In this thesis, we will describe how user relationships within a social network can be further augmented by information gathered from user generated texts to analyze large scale dynamics of social networks. Specifically, we aim at explaining social network interactions by using information gleaned from friendships, profiles, and status posts of users. Our approach profiles user interactions in terms of shared similarities among users, and applies the gained knowledge to help users in understanding the inherent reasons, consequences and benefits of interacting with other social network users.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Almost half a century has passed since the ARPANET network crashed when for the first time researchers attempted to transmit a word from a workstation at University of California to another one at the Stanford Research Institute. What started as a government funded defense project with the ARPANET has in years evolved into a huge network that carries and supports resources and services, such as personal blogs, news media sites, social networks, video sharing sites and online gaming platforms.

In the early days of the Internet, although work stations could communicate with each other, ordinary users were still out of the picture as the Internet was limited to communications among universities and research centres. Starting with the 90s, the Internet saw the creation of hyper-text documents and the World Wide Web, which connected these documents. Web browsers, such as the Netscape Navigator, were developed to surf the WWW, and scripting languages, such as JavaScript, were created to provide interactive hyper-text documents.

Despite interactive elements, the first hyper-text documents were static in the sense that they were loaded into the web browsers as "static screenfuls".[1] In 1999, the Web 2.0 concept was pushed forward to open web sites and documents to ordinary user input, and allow users to create, mesh and change documents. The Web 2.0 concept advanced the idea that user created content could replace or augment static web pages, and web sites could benefit from user collaborations. Although some services existed before the concept, with Web 2.0, online social networks, personal blogs, collaboration wikis, and video sharing services grew more popular.

While these fundamental changes were happening on how the Internet was being used, research works have been looking into network communications, and finding global patterns that explain dynamics within the Internet topology [44, 27, 132]. Once the WWW came into existence, research works turned their attention to human communications, and found that the Internet communication patterns, to some degree, also applied to human interactions. For example, the power law theory [44] that has originated from works on the Internet topology has also been observed on social networks [96].

[1] Excerpted from Fragmented Future by Darcy DiNucci.

Increased attention to human interactions has also benefited from a growing number of online human population. Web 2.0 influenced services, such as social networks, personal blogs and even video sharing services, have become very popular in recent years and attracted a huge registered user base. Among these, online social networks hold the most data about the Internet users; starting with the SixDegrees.com in 1997, online social networks, such as Friendster in 2002 and Facebook in 2004, became popular among hundreds of millions of users.

The growing population of online social network users has resulted in an abundance of user generated data, whether of the new friendships users make or profile information they enter. Social networks have also developed new functionalities such as "likes" and "shares" on Facebook, that increase the number of user generated data items (i.e., traces or texts of user involvement on the social network). From an information retrieval point of view, regardless of the studied social network, users' friendships, or connections to other social network users are the easiest data items to retrieve and analyze. A connection is also easy to represent on a global graph; the two users are denoted as two nodes, and the connection becomes the edge between them. By considering all friendships, a global graph can be constructed, and problems, such as influence propagation [66] or user clustering [78], can be studied with techniques that have been developed in graph theory works [103].

A specific problem that hindered efforts to utilize user generated text data, such as profile information or status posts, is that these data items were unstructured or noisy. For example, user locations on Twitter are still manually typed, therefore unstructured, because users do not have to choose their city, region, country values from a list. For this reason, aggregating user locations from many Twitter users are difficult and time consuming. Despite these problems, several research works have utilized user generated text and profile data. For example, the influence of gender and race/ethnicity of users have been studied to understand network behavior [79] on Facebook, and Twitter users have been modeled to provide personalized news services [1]. However, difficulties in aggregating text based user data has hindered efforts to combine network information (e.g., friendships) with text based information (e.g., users' posts and bios).

In recent years, the unstructured user data problem has been alleviated by the choice of online social network companies to provide an API access to their social network data. For their APIs, online social network companies have created structured user profiles. For example, Facebook has transitioned to a structured version of its user profiles by accepting most frequent manually entered data values as its predefined data values. A further data standardizing effort on social networks has resulted in a machine readable ontology in FOAF (Friend of a Friend) [51]. With FOAF, social network users and their relations to each other can be described in attributes within and across different social networks.

Our work shares a common goal with the FOAF approach; once the edges in a social network graph are labeled with shared attributes, or users are profiled with their attributes, a large scale analysis of shared attributes and created edges can be completed to have a better understanding of why and how interactions happen among social network users. Such an analysis can be used to understand and verify the sociological theories. Specifically we focus on the homophily [92] and heterophily [113] theories from sociology.

According to homophily, personal attributes and connections affect creation of new ties in social life and result in homogeneous groups of individuals. Political party supporters and religious communities are examples of homophilous groups. In contrast to homophily, heterophily is the tendency to interact with people that are not similar to you. In social networks, this tendency is mainly motivated by the fact that social interactions (e.g., creation of new relationships) might give a user some benefits in terms of acquaintance of new information.

In this thesis, we will describe how network connections within a social network can be further augmented by information gathered from user generated texts to analyze large scale dynamics of social networks. Specifically, we aim at explaining social network interactions by using information gleaned from friendships, profiles, and status posts of users. Our approach profiles user interactions in terms of shared similarities among users, and applies the gained knowledge to help users in understanding the inherent reasons, consequences and benefits of interacting with other social network users.

1.0.1 Outline of the Dissertation

The content of this dissertation is described in the following:

Related Work

In the related work chapter, we will explain previous work on finding similarities of users, and give the related work which deal with user interactions in privacy and risk domains. Afterwards, we will outline existing work in detecting which interactions among social network users can be labeled as anomalies. The chapter closes with the related work on micro analysis of user generated data to explain interactions. The content of this chapter is based on "Graphical User Interfaces for Privacy Settings" [10] and "Similarity Metrics on Social Networks" [11].

User Similarity in Interactions

In this chapter, we build on current research and propose measures for evaluating social network users' similarity according to both their connections and profile attributes. From the network connections perspective, we propose a novel measure that facilitates finding user similarity without observing an important fraction of the network. This approach reduces computational costs by limiting the number of queries to be performed on the network. We also propose a profile similarity measure to find semantic similarities among users. Moreover, since user profile data could be missing, we present a technique to infer them from profile items of a user's contacts. Metrics that are studied in this chapter constitute the building block of our studies in the following chapters. The results presented in this chapter has been published in [9].

Explaining Privacy with Interactions

In this chapter, we focus on the risk of new interactions from a privacy point of view. We investigate a risk measure to help users in judging another user with whom they might create a friendship relation. We assume that risks of interactions are impacted by several different dimensions, such as the other user's characteristics and shared friends, whose importance may greatly vary from user to user. Our risk measure captures the personal judgment of users, and helps them in forming their own opinions. This study has been tested with real life social network data through a human experiment, and the results validate our approach. The results presented in this chapter has been published in [7].

Risks of Interactions

Once we have learned privacy decisions of users in Chapter 4, we turn our attention to labeling users' interactions with their one and two degree neighbors on the social network. Based on neighbors' characteristics, in this chapter we categorize social network users, and find that all neighbors can be put into a relatively small number of groups, regardless of their differing characteristics. The results presented in this chapter has been published in [8].

Detecting Anomalies in Interactions

In this chapter, we propose novel anomaly detection measures that use both categories of users and the evolution of the social network graph. Our proposed anomaly measures model a user based on his/her social network structure and the characteristics of the other users she/he follows on the network. In this study we consider the follower relationship on Twitter as an interaction which is initiated to benefit from learning about another user's tweets. Through a large user experiment, we empirically show that anomalies in interactions can be found by analyzing data consumption patterns in a social network graph. The results presented in this chapter has been published in [64].

Micro Analysis of User Interactions

In this chapter, we study the problem of how interactions are governed by certain characteristics of posts across Facebook and Twitter. Rather than external features such as the network structure or user information, we focus on tweets/posts themselves to understand what aspects of tweets/posts help them to get retweeted and liked. Moreover, we analyze how location information influences retweet, comment and like interactions. The results presented in this chapter has been published in [85].

Chapter 2

Related Work

2.1 Introduction

With the increasing popularity of online social networks, many studies have been carried on to better understand how and why humans interact on social networks (see e.g., [42, 131]). As our research has focused on using both the network information and the user generated data, we will start discussing the related work that have focused on finding and explaining commonalities between users with similarity metrics. Afterwards, we will give the state of art in profiling user interactions from privacy, risk and anomaly perspectives. We conclude this chapter with the related work which explain user interactions with certain dimensions of user generated data, such as the sentiments expressed in user posts.

2.2 Similarity metrics

In the literature, the term *similarity* has been used in different meanings (e.g., the short distance between two users, shared features or shared actions) to quantize similarity for different application fields. Some work have attempted to define similarity rigorously [112, 54, 81]. Among these, the four principles by Lin [81] are widely used to implement similarity metrics on social networks. We will explain these four principles with commonality, differences, maximum and minimum similarity. In *commonality*, shared commonalities (e.g., race, gender, sex of users) increase the similarity of two users. On the other hand, the more *differences* lead to smaller similarity. Regardless of the number of features, two users are said to have the *maximum similarity* when they are identical in every feature. Similarly, regardless of the number of features, two users are the least similar when they are different in every feature. In current studies, the maximum and minimum similarity values are given as 0 and 1, respectively.

A more rigorous set of properties for similarity metrics can be adopted from distance metrics by considering $similarity = 1 - distance$. For any given distance metric, these properties are 1) symmetry, 2) identity, 3) non-negativity, and the 4) triangle equality. In the identity property, $distance(a, b) = 0$, when $a = b$. On social networks this property

can have different explanations; on the graph structure $distance(a, b) = 0$ is assumed to be true when the two nodes have the same set of friends, whereas if profile information are considered, two users must have the same values for every profile item. In setting a lower bound for distance, the non-negativity property defines $distance(x, y) \geq 0$ for any user pair. The symmetry condition assures that $distance(a, b) = distance(b, a)$, while in the triangle equality $distance(a, c) \leq distance(a, b) + distance(b, c)$.

The absence of some of these properties can be used to classify different formulas. For example, *quasi-metrics* do not provide the symmetry property, whereas *semi-metrics* do not have the triangle property. Traditionally, the symmetry property is not applicable in directed networks, because directions of edges can lead to different similarity values for a pair of users. In this case, two different values are computed; $sim(a, b)$ denotes the similarity value according to a user a and $sim(b, a)$ denotes a potentially different value from the perspective of user b.

Similarly, the triangle equality is difficult to achieve for similarity of user profiles because profiles can consist of more than one dimension (i.e., profile item). As a result, although the triangle property holds for one dimension, similarities for three user profiles with multiple dimensions might not adhere to the triangle property. Ideally, any formula that does not carry all these four properties should be called a measure, but researchers still prefer to use the word metric interchangeably with the term measure.

Founded upon these theoretical definitions, similarity metrics that have been proved efficient and practical on social networks are those that exploit a locality principle in similarity computations. The basic idea underlying such metrics is that, given two social network users, their similarity is computed by observing only a subset of vertices (e.g., friends of the two users) in the social network. This approach restricts required information about the social network to a minimum, and reduces the required time of calculations. Even though global measures (e.g., the shortest path between users, or the community membership of users) can be used in the same context, they are more costly in time and computational power because they require too much information about the social network. For example, shortest path calculation might require observing friendship links of many users. Moreover, even though the costs can be undertaken, researchers and companies cannot have access to the whole social network data because of privacy issues. Only owners of social networking services can have the complete data that is required to compute global similarity measures. Therefore, local similarity metrics provide a simple alternative in the face of these costly issues.

In 1970s, early attempts at defining similarity metrics involved finding similarities among text based documents [91] that were modeled as a collection of words. Similarity metrics were used to discover relations between documents or rank the documents according to their similarity to a given query. These efforts have resulted in several well known metrics, such as the Cosine and Jaccard similarities [55]. With the advent of the Internet, researchers have applied document based similarity metrics to user generated web items, such as friendships and status posts, to discover relationships between web users. Specifically, similarity research on user generated data has focused on predicting links (relationships) among users and mining past user behavior to predict future actions.

In the link prediction problem [80], similarity of social network users has been exploited to predict new friendships. In this context, high similarity between two social network users is assumed to increase the probability of them creating a new friendship [118]. With generalization, this idea has been explored in the homophily theory which states that people tend to be friends with other people who are similar to them along personal attributes, such as gender, race and religion [92].

In addition to user characteristics, actions of a user is exploited to predict actions of similar users. This idea has been studied in recommender systems to observe existing item ratings (for movies, books, songs etc.) of users and predict ratings of unseen items [93]. Similar users are assumed to give similar ratings to similar items. From this assumption, similarity of ratings are predicted by finding either similar users (i.e., user based) or similar items (i.e., item based).

On social networks, user generated data are classified into two types: profile data, which refers to user entered textual information, such as personal information, and network data, that is information on created relationships, such as friendships with other users on the social network. Depending on the type of user data, similarity metrics differ in how they model a social network user. Furthermore, some similarity metrics can be used only on one type of user generated data. By taking this into account, we will first explain how similarity metrics work on network data, and then continue to explain metrics for profile data.

2.2.1 Network Similarity

In network similarity metrics, existing user relationships are exploited to find similarity. For example, a big number of shared friends between two users can be assumed to imply their high similarity. Network data that represent relationships can be modeled as a graph $\mathcal{G} = (V, E)$, where each user a in the network is considered a vertex $v_a \in V$, and a relationship between users a and b is an edge $e_{ab} \in E$ on the graph $\mathcal{G}$. If relationships are established by mutual consent of two users, an edge between them is said to be undirected (the edge has no start and end points), and it can be called a *friendship*. Friendship relations on social networks, such as Facebook, Orkut, and LinkedIn, are modeled with undirected graphs. On an undirected graph, first level neighbors of a vertex v_a are a set of vertices $\Gamma(v_a)$ who share an edge with v_a (e.g., friends of a). Similarly, second level neighbors $\Gamma(\Gamma(v_a))$ (e.g., friends of friends of a) share an edge with first level neighbors of a. If relationships can be created without mutual consent, an edge is said to be directed; it starts from the user who initiated the relationship (i.e., source vertex) and ends at the user with whom the relationship was established (i.e., target vertex). For example, the popular social networks Twitter and Google+ are directed social networks; when user a starts following user b, an edge is created, starting from vertex v_a and ending at vertex v_b. On directed graphs, the *friendship* term from undirected graphs is replaced with *in-neighbors* and *out-neighbors*. In-neighbors $\Gamma^-(v_a)$ and out-neighbors $\Gamma^+(v_a)$ of a vertex v_a are defined as target and source vertices of all edges that have vertex v_a as their source and target vertex, respectively.

Measure	Formula	Description
Overlap	$\lvert\Gamma(v_a) \cap \Gamma(v_b)\rvert$	The number of common neighbors.
Preferential Attachment	$\lvert\Gamma(v_a)\rvert \times \lvert\Gamma(v_b)\rvert$	Multiplied neighbor counts of both users.
Jaccard	$(\lvert\Gamma(v_a) \cap \Gamma(v_b)\rvert)/(\lvert\Gamma(v_a) \cup \Gamma(v_b)\rvert)$	The percentage of shared neighbors over all neighbors.
Cosine	$(\lvert\Gamma(v_a) \cap \Gamma(v_b)\rvert)/(\sqrt{(\lvert\Gamma(v_a)\rvert \,.\, \lvert\Gamma(v_b)\rvert)})$	The number of common neighbors normalized by multiplied neighbor counts.
Adamic & Adar	$\sum_{v_c \in \{\Gamma(v_a) \cap \Gamma(v_b)\}} \frac{1}{Log(\lvert\Gamma(v_c)\rvert)}$	Common neighbors who have very few neighbors are given more importance.
Point-wise Mutual Information	$P(\Gamma(v_a), \Gamma(v_b)) \times log\left(\frac{P(\Gamma(v_a), \Gamma(v_b))}{P(\Gamma(v_a)).P(\Gamma(v_b))}\right)$	How much the probability of having the current set of common neighbors differs from the case where neighbors would be added by users on the graph randomly.
Katz's Measure	$\sum_{p=1}^{+\infty} \beta^p \times NumOfPath(v_a, v_b, p)$	Similarity is implied by the number and length of paths that connect two users on the graph. Each path $0 < p < +\infty$, whereas $0 < \beta < 1$.

Table 2.1: Network Similarity Metrics

Although the similarity metrics are designed to work with the *neighborhood* notion of undirected graphs, they can be applied to directed networks by small modifications. For example, direction of edges can be removed to make the graph undirected. Another approach is to consider only one type of edges (out-neighbors or in-neighbors) as neighbors while using the metrics. As these modifications can be used to define neighbors of a user on directed graphs, we will give metric definitions in terms of user neighbors. Assume that a similarity function $sim(v_a, v_b)$ computes the similarity of users v_a and v_b by considering their neighbors $\Gamma(v_a)$ and $\Gamma(v_b)$, respectively. We will denote one of their common neighbors with v_c, i.e., $v_c \in (\Gamma(v_a) \cap \Gamma(v_b))$. High similarity between two users who do not have a relationship will be assumed to increase the probability of them creating a relationship edge. With these definitions, metric formulas that we will explain are given in Table 2.1. Next we will discuss these network similarity metrics in more details.

Overlap: The overlap measure [124] counts the number of common friends of v_a and v_b to compute similarity.

Preferential Attachment: For relationship creation on social networks, the preferential attachment metric reflects the "rich gets richer" notion from sociology [14]. The metric assumes that highly connected vertices (i.e., users who have many neighbors) are more likely to create relationships with each other.

Jaccard ($\mathcal{L}1$ norm): The Jaccard metric [124] counts the number of common neighbors as in the overlap metric, but it normalizes this value by using the total number of neighbors of users v_a and v_b.

Cosine ($\mathcal{L}2$ norm): Cosine similarity was originally devised to find the similarity of two documents by computing the cosine of the angle between their feature vectors [124]. When the angle between the two vectors is 0, they are considered identical, and the cosine of the angle equals the maximum similarity value 1.

Adamic & Adar: Like the overlap metric, Adamic & Adar considers common neighbors, but each neighbor's impact on the similarity value depends on the number of its neighbors [3]. If a common neighbor has few neighbors, its impact on the similarity is assumed to be higher. For example, if users v_a and v_b are the only neighbors of v_c, $1/log(2)$ is added to the overall similarity. The similarity is computed by summing values from all common neighbors.

Point-wise Mutual Information (Positive correlations): Point-wise mutual information [22] is computed in probabilistic terms where joint probability distribution function $P(\Gamma(v_a), \Gamma(v_b))$ computes the probability of a graph vertex $v_x \in V | x \neq a \neq b$ sharing edges with both v_a and v_b, whereas marginal probability distribution functions $P(\Gamma(v_a))$ and $P(\Gamma(v_b))$ are probabilities of a graph vertex sharing an edge with v_a and v_b, respectively. If edges are assumed to represent friendships, $P(\Gamma(v_a), \Gamma(v_b))$ is equal to $\frac{\#mutual\ friends}{\#users\ in\ the\ social\ network}$. Similarly, $P(\Gamma(v_a))$ is equal to $\frac{\#friends\ of\ v_a}{\#users\ in\ the\ social\ network}$. In other words, point-wise mutual information shows whether two users share mutual friends due to randomness. Note that due to computing probability with the total number of users in the social network, point-wise mutual information produces very low average similarity values, because $\#users\ in\ the\ social\ network$ can be in millions.

Katz's Measure: Katz's measure [65] was designed in 1950's to find the status of a

vertex on a graph. The vertex which had the biggest number of shortest paths to the other vertices was considered a central vertex with high status. To compute $sim(v_a, v_b)$, Katz's measure finds the number of paths that connect v_a and v_b for path length p, $1 < p < +\infty$. The number of paths (i.e., the value of $NumOfPath(v_a, v_b, p)$) is dampened by a β^p value, where $0 < \beta < 1$. In practice, the β value is chosen as small as 0.005 [80]. Although p values can be increased to cover a big portion of the graph, usually $p = 2$ or $p = 3$ values are chosen to find similarity, because computations for $p > 3$ contribute very less to the overall value. Note that although the Katz's measure can be used as a global measure with big p values, small p values (e.g., 2 or 3) make it a practical measure for fast, local similarity computations.

In research work, performance of similarity metrics has been compared by making predictions based on similarity values, and validating the results [80, 118]. Typically, metrics are used to predict top-k relationships (e.g., k most probable future friendships that will be created between users) on graphs at a time t_1, and these predictions are validated at a time $t_2 > t_1$. The performance of a metric can be computed by counting the number of correct predictions.

In Liben et al. [80], Adamic & Adar has been shown to perform better than preferential attachment, Jaccard, overlap and Katz's measure on a scientific co-authorship network. On another social network, Orkut.com, Spertus et al. [118] have found that cosine similarity performed better than Jaccard and point-wise mutual information metrics.

Despite these comparisons, it is important to understand that each metric has its weaknesses in different application fields. Preferential attachment is widely used in social networks to predict friendships, but unlike Jaccard or cosine similarity its computed value does not reside within [0,1]. In fact, its max value is only bounded by the total number of users in a social network, because there are no theoretical limits to prevent a user from having every other social network user as a neighbor. However, some social networks may choose to limit the number of neighbors; for example, an undirected network, Facebook, allows up to 5000 neighbors, whereas a directed network, Twitter, does not have such a limit. Because of this, preferential attachment can not be used to quantify how much *percent* two users are similar. When the graph has many vertices (i.e., the probability of an edge between two users are very small), computed value of point-wise mutual information can be very small, and it can not be used to define how much *percent* two users are similar. Point-wise mutual information and preferential attachment can be best used in ranking a set of users according to their similarity to a specific user. There are some limitations in using Katz's measure too. In popular social networks, discovering edge counts for paths of length 2 or more can be restricted because social networking services do not allow access to social network data. Furthermore, Katz's measure can be costly to compute when the social network is large. Considering these limitations, research work [61, 134] have mostly used Jaccard, cosine or Adamic & Adar in their user similarity computations, because these measures are fast and easier to interpret.

2.2.2 Profile Similarity

Along with network data, profile data constitute the second type of user generated data on social networks. We will call similarity metrics which work with profile data as profile similarity metrics. On social networks, we will consider profile information as a set of unique items (e.g., hometown, location, education of a user), which can have one or multiple sub-fields for each value.

After modeling profile data with a set of items, similarity between two users is computed by first finding the similarity of individual item values on the two profiles. For example, similarity of two users according to the gender item compares gender values on the two profiles. If the considered item has many values, or each value has multiple sub-fields, an aggregation function is required to find item similarity. An overall profile similarity value is determined by aggregating (e.g., by weight averaging) all item similarity values. However in practice, most of the research work on profile similarity consider a user profile to be a set of unstructured keywords. For example, in [16], Facebook user profiles only consist of user values from the "hobbies" item. With this simplification, profile similarity of two users is found by computing item similarity of hobbies on two profiles.

In a more detailed study, similarity of items which have multiple values or subfields have been weight averaged to model the importance of similarity for some items or sub-fields [6]. For example, when user profiles consist of hometown and hobbies fields, hometown similarity can be weighted with a bigger coefficient to show that hometown similarity is more important than hobbies similarity.

When similarity computations are reduced to find item similarities, the type of item values determines the way similarity is computed. Although some items have numerical values (e.g., age, zip code), most of the item values on social networks are text based categorical data which can not be ordered on an axis to find similarity/distance of two data points. Using simple approaches, such as string matching, is not efficient because the text represents an identity (e.g., hometown:Barcelona), or partial (n-gram) similarities (e.g., b**AR**celona:p**AR**is) are trivial.

Two main approaches are used to find similarity of item values: ontology based [63, 94] and our social graph based [6]. In ontology based approaches, a graph of entities is created to define their relationships or distance [37]. For example, considering hometown similarity of three social network users with values Barcelona, Madrid and New York, an ontology can classify Barcelona and Madrid as Spanish cities, whereas New York is classified as an American city, and compute a higher similarity for users from Barcelona and Madrid. The main disadvantage of this approach is that it requires a reliable ontology which can be difficult to create. Furthermore, as social networks are dynamic, new item values are added in time and the ontology must be updated frequently.

In the social graph based approach (see Chapter 3) we assume that neighbors (e.g., friends/coauthors) of a user v_a are similar to v_a along profile attributes. For example, if hometown of v_a is Barcelona, we can expect many of its neighbors to be from Barcelona and other Spanish cities. When hometown values of neighbors are observed, another city (e.g., Madrid) can be found similar to Barcelona without explicitly creating an ontology.

With this intuition, a user v_b is said to be similar to v_a if its hometown is similar to the hometown values of v_a's neighbors. Furthermore, even when a user has a blank profile, using its neighbors in such a way allows one to compute its similarity with other social network users. The social graph approach has also been found effective for network similarity metrics [38]. The disadvantage of this approach is that neighbors are assumed to be similar to users. This assumption is more applicable in undirected social networks where mutual consent is required to create a relationship edge. However, if the network is undirected, neighbors can have very different characteristics from users, and performance of social graph based approaches can deteriorate.

2.3 Privacy in user interactions

Nissenbaum's work [105] on defining personal privacy in terms of interaction *contexts* has led to influential privacy work (e.g., [45, 39]) from machine learning and community detection communities. Considering context to be a role within which users interact, contextual privacy assumes that users publicize different parts of their social lives to specific people. For example, social network friends who are colleagues from a workplace are assumed to be interacting within office related roles, and their *private* pictures are not made visible to each other. In other words, "contextual privacy requires that particular information from users' roles in another context should not discredit a convincing performance in the current situation" [128]. To this end, Berg et al. finds two issues to be central to contextual privacy: user awareness about a specific context should be raised, and "the user should be provided with tools to help him/her make sure that certain users who interact with the user in a context will not be the individuals who witness them in another of their roles" [128].

The research work on understanding the context of user interactions has resulted in two main techniques: audience segregation and content segregation.

Audience Segregation

The audience segregation methods aim at grouping social network users automatically such that when privacy settings for some members of the group are learned, the same privacy settings can be applied to the remaining members in the group. Each group is then called an audience, and information sharing is targeted to different audiences. Audience segregation approaches use five methods to find distinct audiences: manual selection by the user, selections by attribute rules, contact interactions, attribute similarity and contact relationships [102]. Among these approaches, selection by contact interactions has been commercially deployed by Google Buzz, where the number of exchanged e-mails was chosen to create audiences. The drawbacks of this approach has been found to be very severe, because exchanged mails without considering the content of information has been found not to reflect audiences or social tie strengths.

From the remaining options, privacy research has mostly focused on contact relationships [39, 45] where the social graph of a user is analyzed by community detection algo-

rithms to find distinct audiences.

Current day social networks employ audience segregation to a degree, but research work offer richer representation of audiences. For example, on many social networks, users are divided into four audiences by default: friends, friends of friends, network/group members and everyone. Research work create finer granularity audiences such as university friends and work colleagues, and this approach has only been recently employed by Facebook.com. With this richness in audiences, research work aim at presenting the user with a simple and intuitive privacy setting interface to compensate the difficulty of choosing the correct privacy setting for each audience. To this end, research work represent audiences within graph layouts, such as in the Fruchtermann-Reingold layout [48].

Audiences can also consist of overlapping communities, and user features can be used to choose privacy settings with a finer granularity in the audience. For example, gender attribute can be used in a community to select a finer granularity setting for female users [45]. More than showing communities as nodes in a graph, a community can also be labeled with textual or numerical flags to distinguish its members from the rest of the nodes in the graph [90], and nodes in the community can be colored differently (e.g., in white or black) to show whether they can access a data type or not.

Regardless of presenting the user with or without a graph layout, audience segregation methods pick some users from each audience, and asks the user to choose *yes* or *no* on whether a data type should be accessible by the selected users. This approach has been shown to reduce required user effort in setting privacy [45] because members within an audience have been found to receive similar privacy settings from the user.

Further validation of audience segregation approaches have been done by measuring the number of time spent and the number of clicks needed by users to complete privacy setting tasks. Mazzia et al. have used 36 tasks which consisted of single and a group of privacy settings to show the efficiency of their results [90]. In addition to required time and clicks, success rates to asked questions about chosen privacy settings and correctly set privacy rules have been used to validate interface efficiency [109].

A further inquiry into the efficiency of audience segregation research can be made by asking whether the found audiences receive privacy settings that are distinct enough across audiences to justify asking a user the same question for multiple audiences. In a work by Onwuasoanya et al., 14 social network users out of 20 have chosen to manually create only one group from all of their friends "meaning that Facebook's default privacy settings for all friends accurately reflect their privacy settings" [106]. For the mentioned 14 users, multiple audiences would require choosing privacy settings for each audience, but the resulting privacy settings would not be different than those chosen by applying a privacy setting to all audiences at once. However, from the same study we note that for some users audience segregation would be desirable. To the best of our knowledge, no large scale survey work exists in understanding the percentage of users who would like to use audience segregation methods.

A critique of audience segregation work can be made regarding the scope of these studies. Although the idea of contextual privacy from Nissenbaum's perspective [105] was not defined to be applied to *friends only*, the current state of art in contextual privacy have

largely avoided creating audiences from friends of friends or other non-friend social network users. This shortcoming has mainly been the result of social network users' reluctance to deal with privacy issues beyond information disclosure to friends. Stutzman et al. suggest that "due to a normative lack of interaction with outsiders, Facebook users may have constructed a boundary of privacy with weak ties" [120].

Content Segregation

Content segregation is based on the idea that user generated instances (e.g., a single photo) of a data type (e.g., photo) can belong to different contexts [87]. Whereas audience segregation aims at finding the appropriate audience for a data type (i.e., which audiences should be allowed to access a specific data type), content segregation aims at finding the context of each data instance regardless of its type in order to specify audiences with respect to contexts. Social network users already employ such strategies by selectively allowing different users to access a data instance. For example, although party photos are considered very private and hidden from the public, photos of mountains/cities, or even the profile photo are not considered to be private enough to be hidden from everybody.

Among contextual privacy works, we have found Madejski et al. provide the first approach in content segregation [87]. In this work, information categories (i.e., contexts for content) for data instances "were selected based on themes that were most likely to elicit user intent to show or hide from certain profile groups". 12 such categories have been "defined by a careful consideration of Facebook data, Facebook terminology lists and tags as religious, political, alcohol, drugs, sexual, explicit, academic, work, negative, interests, personal and family".

By this approach, users' desire to hide data related to certain concepts can also be realized with ease. For example, "drunk photos, or check-ins to pubs can both be contextualized with alcohol and hidden from selected friends" [87]. We believe that when used with powerful machine learning/data mining techniques, such as face recognition and sentiment analysis, content segregation can facilitate privacy settings and prepare a coherent view across all data types for better privacy.

2.4 Risk in user interactions

In the literature, risk models [28, 130, 41] have been proposed to put privacy in a context so that social network users would be empowered to understand privacy risks. Privacy risks have been evaluated from access control [29, 52] and information leakage [71] perspectives. Lately, analyzing the social graph of a user [71, 20, 95] has been found successful in revealing personal information even when the user does not enter personal data to the social network.

Similar to our research focus, the risk of maintaining an online presence has been studied in [84], from which we borrowed our visibility measure for items in Chapter 4. In [84] a privacy score is computed for social network users, which measures the user's potential privacy risk due to his/her online information sharing behaviors. Such a measure depends on the sensitivity of the information being revealed and the visibility the revealed

information gets in the network. In contrast, in our work we are interested in finding risks of becoming friends with certain social network users.

Friends' role in user interactions has been studied in sociology [114], but observing it on a wide scale has not been possible until online social networks attracted millions of users and provided researchers with social network data. For online social networks, Ellison et al. [43] defined friends as social capital in terms of an individual's ability to stay connected with members of a previously inhabited community. Differing from this work, we study how friends can help users to interact with new people on social networks. Although these interactions can increase users' contributions to the network [120] and help the social network evolve by creation of new friendships [127], they can also impact the privacy of users by disclosing profile data. Squicciarini et al. [119] have addressed concerns of data disclosure by defining access rules that are tailored by 1) the users' privacy preferences, 2) sensitivity of profile data and 3) the objective risk of disclosing this data to other users. Similarly, Terzi et al. [84] has considered the sensitivity of data to compute a privacy score for users. Our work differs in scope from these in that we study the role of friends who connect users on the social network graph and facilitate interactions. Indeed, research works (see [15] for a review) have been limited to finding the best privacy settings by observing the interaction intensity of user-friend pairs [13] or by asking the user to choose privacy settings [45]. Without explicit user involvement, Leskovec et al. [77] have shown that the attitude of a user toward another can be estimated from evidence provided by their relationships with other members of the social network. Similar works try to find friendship levels of two social network users (see [5] for a survey). Although these work can explain relations between social network users, they cannot show how existence of mutual friends can change these relations.

Privacy risks that are associated by friends' actions in information disclosure has been studied in [125], but the authors work with direct actions (e.g., re-sharing user's photos) of friends, rather than their friendship patterns.

Recent privacy research focused on creating global models of risk or privacy rather than finding the best privacy settings, so that ideal privacy settings can be mined automatically and presented to the user more easily. Terzi et al. [84] has modeled privacy by considering how sensitive personal data is disclosed in interactions. Although users assign global privacy or risk scores to other social network users, friend roles in information disclosure are ignored in these work.

An advantage of global models is that once they are learned, privacy settings can be transfered and applied to other users. In such a shared privacy work, Bonneau et al. [19] use *suites* of privacy settings which are specified by friends or trusted experts. However, such works do not use a global risk/privacy model, and users should know which suites to use without knowing the risk of social network users surrounding him/her.

2.5 Detecting Anomalies in Interactions

Our work on detecting anomalies in interactions focuses on profiling Twitter users, and analyzing their friendships in time to detect anomalous interactions. As such, in addition to traditional approaches for anomaly detection [32], there are several research work on topic modeling for microblogging data, topical anomaly detection, graph based and classifier based anomaly detection on microblogs that are relevant to our work.

Early work on using topic models with microblogging data focused on choosing a strategy that would allow using traditional topic models with Twitter's short tweets. Similar to our tweet aggregation process, Hong et al. [58] experimented with different user representations with tweets, and checked the accuracy of their aggregation strategy by comparing a user's topics to an external ground truth (i.e., listings of web services that group users based on several criteria). However their work does not consider friends in defining a user's profile. Another early effort to look beyond traditional topic models and design a topic model that would work with Twitter data is from Ramage et al. [111]. They acknowledge the existence of some new dimensions in Twitter data, such as "substance, style, status, and social characteristics of posts" rather than a set of words to describe topics. Similar to using tweets and bios as two different datasets to find user topics, Zhao et al. [133] performs a study of comparing New York Times newspaper articles to Twitter news. Their approach aims at finding the best set of topics, without representing users with topic distributions.

With early applications in more traditional settings, graph based anomaly detection approaches [108] are established methods, but they are computationally expensive. A more local and cheap approach involves using classifier based anomaly detection. In "Warning-Bird", Lee et al. trains a classifier to detect suspicious URLs on Twitter [74]. In another classification work, Chu et al. [35] aims at classifying Twitter accounts into Human, Cyborg and Bot classes. Using our approach, anomalies can be detected as well to identify "Cyborg" and "Bot" accounts, because they are known to follow a large number of users regardless of user characteristics.

In user behavior, Anantharam et al. [12] analyze contents of URLs that are given in tweets to find topically anomalous tweets. By using this information, users can be labeled as having anomalous behavior. Similar to our friendship oriented approach, Takahashi et al. [122] propose a probability model to detect the emergence of a new topic between two users from the anomaly measured through the model. Their assumption is that users tweet about changes in their behavior.

2.6 Micro analysis of Interactions

Other than finding similar users or profiling users with certain topics, some related work focus on a certain dimension of the available profile or network data to study user interactions. These related work can be considered as the micro analysis of user interactions. For example, the use of hashtags (i.e., Twitter topics) and addressivity (i.e., @ sign) has been studied to understand interactions among Twitter users [57]. A comprehensive analysis has

found that online social networks, such as Twitter, can also act as an information source [72] to a high degree, thereby pointing to an information seeking behavior among Twitter users. This functionality of Twitter is provided by tweets and conversations that occur among users [59, 101, 121].

From a theoretical point of view, sentiment analysis of user generated texts is a widely studied problem in research work [70, 4], but its impact on conversational interactions has not been explored in detail. In these research works, sentiments are used to find aggregate public opinion about a given topic such as the approval rates of political actors.

Another dimension of user interactions is related to geolocation studies. Recent works by Takhteyevet al. [123] and Leetaru et al. [75] have analyzed the geography of Twitter and found that users communicate more often with those closest to them. However such works have been limited to studying Twitter users only, because accessing user location data on other social networks, such as Facebook.com, has been fraught with privacy and security problems, which makes a large scale analysis very difficult. In a limited study on Facebook, locations of friends have been studied from a privacy point of view on Facebook [73]. In this work, researchers come to the conclusion that friends/followers share common locations to a high degree, but an analysis of the impact of locations on conversational interactions has not been studied.

Chapter 3

User Similarity and Interactions

3.1 Introduction

Today, social network data are regarded as one of the most highly valuable source of information over the Web [25], as they can broaden our horizons in keeping track of the life, in learning about societies, or, simply, in making advertising more profitable. In this framework, social network studies [80, 116, 26] help us in understanding how user relationships will evolve, and which paths should be taken to spread specific news, ads and political views. The basis of the majority of such work [104, 92] is finding user similarity from different perspectives.

In our work, we build on current research and propose measures for evaluating social network users' similarity according to both their connections and profile attributes. From the network connections perspective, we propose a novel measure that facilitates finding user similarity without observing an important fraction of the network. This approach reduces computational costs by limiting the number of queries to be performed on the network. We also propose a profile similarity measure to find semantic similarities among users. Moreover, since user profile data could be missing, we present a technique to infer them from profile items of a user's contacts.

Besides proposing the measures, this chapter presents a comprehensive evaluation study aimed at comparing the proposed similarity measures with those previously proposed in the literature. The main goal of the experimental evaluation is to test how the considered measures are able to forecast creation of new relationships. We have conducted our experiments on different kinds of graph data (i.e., Facebook, Youtube, Epinions, and DBLP data sets) to see how different scenarios and graph sizes affect performance. Considered scenarios allow us to see the influence on the measure performance of different key dimensions of graph data, such as whether the relations are directed (e.g., Epinions), or undirected (e.g., Facebook), or they are explicitly (e.g., Facebook) or implicitly given by the semantics of the considered data (e.g., Youtube). Moreover, we show how the proposed technique to infer missing profile items performs on the undirected social network Facebook. Finally, to observe how profile and network similarities interact, we carried out experiments on a

DBLP data set enriched with profile data from the ACM digital library.

Specifically, we detail the relation between profile and network similarity measures, and explain how they are affected by the characteristics of studied networks. As such, we believe that the results reported in this chapter can be useful not only for evaluating the performance of our measures, but also to evaluate future developments in the area.

The rest of this chapter is organized as follows. Section 3.2 presents our similarity measures. Section 3.3 describes our experimentation methodology. Section 3.4 reports the experiments on undirected Facebook and DBLP datasets, whereas Section 3.5 illustrates the experiments on directed Youtube and Epinions datasets. Section 3.6 gives absolute performance results for our similarity measures. In Section 3.7 we discuss experimental results with a holistic view.

3.2 Similarity Measures

Our measures have been mainly designed to predict new relationships in the network. More precisely, we take into account the theory of homophily [92], from which we learn two issues related to the establishment of a new relationship in social networks. The first is that, as shown in [92], people tend to create new relationships with people that are closer to them on a social graph. As an example of this trend, in the Facebook data used for our experiments (cfr. Section 3.4), 83% of new edges are created between users and contacts of their friends. The second fact is that users tend to create relationships with people that are similar to them along certain profile attributes, such as gender, education and religion. Therefore, we introduce two distinct similarity measures, namely, *network* and *profile similarity*, and show how these two can be combined to find user similarity. Both of these measures are designed in such a way that we only consider the data in the social graph between the target user u and the new possible friend x, who is selected from u's friends of friends. Any user x who is connected to u within a two-hop distance is hereafter called a stranger. We model a social network $\mathcal{G} = (E, N)$ as a graph, such that the set of nodes N represents social network users, whereas the set of edges $E \subseteq N \times N$ denotes relationships. Moreover, we use dot notation to refer to a specific component within a graph.

3.2.1 Network Similarity

Similar to existing measures [40, 36], we compute network similarity between a target user u and a stranger x based on the use of network information of both u and x. A common way to estimate this similarity is to count the number of mutual friends between u and x. However, in existing approaches relationships among mutual friends of u and x are ignored, losing valuable information. Our intuition is that, if a stranger is connected to friends of u who themselves are well connected to each other, the stranger is more connected to the target user u. To understand this, assume that in one extreme, relationships among mutual friends of u and x provide a complete graph, and, in the other extreme, mutual friends have no friendship relations among them. In existing measures that just use the count of mutual friends, these cases are considered the same.

Our measure takes into account friendship edges among the mutual friends of u and x, in addition to the number of mutual friends. To find out how friendship edges should be used in the network similarity measure, we define the *mutual friends graph* of u and x as a subgraph of the target social network $\mathcal{G}$, where the node set consists of u, x and their mutual friends. The edge set includes all edges among the above considered nodes. Formally, we define mutual friends graph as follows.

Definition 1 (Mutual Friends Graph) *Given a social network $\mathcal{G}$ and two nodes $u, x \in \mathcal{G}.N$, the **mutual friends graph** of u and x, denoted as $MFG(u,x)$, is a sub-graph of $\mathcal{G}$ where: (1) $MFG(u,x).N = \{u,x\} \cup \{n \in \mathcal{G}.N | n \neq x, n \neq u, \exists e, e' \in \mathcal{G}.E, e = \langle u,n\rangle \wedge e' = \langle n,x\rangle\}$; (2) $MFG(u,x).E = \{\langle n,n'\rangle \in \mathcal{G}.E | n,\ n' \in MFG(u,x).N\}$.*

Edge count of the mutual friends graph shows strength of ties between two nodes in the graph. When computing how the target user u is similar to a stranger x with respect to the network, we compare edge count of mutual friends graph of u and x to edge count of the *friendship graph* of the target user u, that is, a graph consisting of all friends of u and all edges between the corresponding nodes. This is defined as follows.

Definition 2 (Friendship Graph) *Given a social network $\mathcal{G}$ and a node $u \in \mathcal{G}.N$, the **friendship graph** of u, denoted as $FG(u)$, is a sub-graph of $\mathcal{G}$ where: (1) $FG(u).N = \{u\} \cup \{n \in \mathcal{G}.N | n \neq u, \exists e \in \mathcal{G}.E, e = \langle u,n\rangle\}$; (2) $FG(u).E = \{\langle u,n\rangle \in \mathcal{G}.E | n \in FG(u).N\} \cup \{\langle n,n'\rangle \in \mathcal{G}.E | n,n' \in FG(u).N\}$.*

Network similarity can thus be measured through a comparison between the number of edges in the mutual friends graph of u and x and the number of edges in the friendship graph of u, as follows.

Definition 3 (Network Similarity) *Network similarity of node x to u is defined as:*

$$NS(u,x) = \frac{Log(|MFG(u,x).E|)}{Log(2\,|FG(u).E|)}$$

where $|MFG(u,x).E|$ and $|FG(u).E|$ are the number of edges in $MFG(u,x)$ and $FG(u)$, respectively.

The value of the proposed network similarity measure is damped by the log function, and a constant *2* is used to normalize the similarity value. This normalization is motivated by the fact that when a stranger x is connected to all friends of a target user u, the following holds: $|MFG(u,x).N| = |FG(u).N| + 1$, and $|MFG(u,x).E| \leq 2 \times |FG(u).E|$. Thus, when comparing the number of edges in the mutual friends graph to the number of edges in the friendship graph, we normalize the similarity function by taking this into account. If strangers do not have mutual friends with the target user, the mutual friendship graph consists only of two nodes, i.e., $MFG(u,x).N = \{u,x\}$ and no edges. Hence, network similarity is zero.

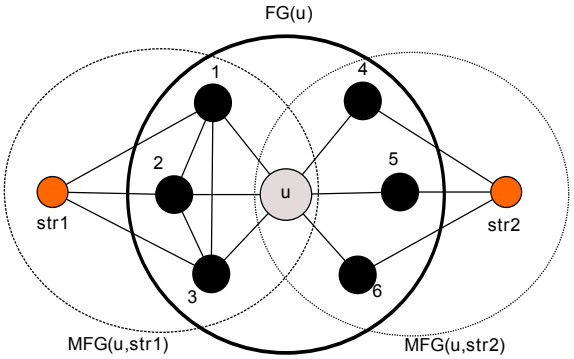


Figure 3.1: An exemplary social graph for a target user u. Six friends and two strangers are shown in the graph as nodes, and their friendship relations are denoted by edges.

Example 3.2.1 *Consider the social network graph in Figure 3.1. Friendship graph of user u has 7 nodes and 9 edges. Stranger str1 has 3 mutual friends with user u and there are 9 edges in the mutual friends graph MFG(u,str1); stranger str2 also has 3 mutual friends with u but the mutual friends graph MFG(u,str2) has only 6 edges. Network similarity of str1 to u is then $NS(u, str1) = Log(9)/Log(2 \times 9) = 0.76$, whereas $NS(u, str2) = Log(6)/Log(2 \times 9) = 0.61$. Our measure favors str1, as it is connected to a stronger community around u than str2.*

3.2.2 Profile Similarity

Profile similarity compares personal data stored in the profile items associated with two social network users, say u and x, to determine how much they are similar. The easiest way is to use a string matching approach to compute individual item similarity. Items on two profiles are said to be similar if their values are identical, and dissimilar otherwise. Therefore, similarity increases only when the target user u and the stranger x have the same values for an item. However, while this solution is quite simple to perform, it results in low average similarities in presence of profiles with high percentage of missing values. Unfortunately, this is indeed the case of user profiles in current social networks. In our experiments on Facebook data, we found that the percentage of missing profile values is 64% (see Section 3.4). To overcome this limitation that could affect also our similarity measure, we propose a method to infer a portion of the missing values of a stranger profile before computing profile similarity. This inferral mechanism reduces the number of blank profile items, and helps profile similarity to be computed with more profile information. In the following, we first present the inferring technique, then we show how the proposed profile similarity is computed. We assume that the target user u has all the information entered, so item values are inferred for strangers of u.

Inferring Missing Item Values. In general, each social network user is associated with a profile, consisting of a set of items I storing user personal data (e.g., date of birth, religion, school, hobby). Each item could be optionally structured into subfields (e.g., date

of birth=[day, month, year]) and could contain several values (e.g., hobby=[reading, chess, music]).

Hereafter, given a profile item $i \in I$, we use the function $values(i)$ to denote the set of values associated with it or one of its subfields (e.g., birthday.year denotes the year subfield of the birthday item, values(birthday.year) returns the value of the year subfield).

Typically, inferring a stranger's profile information is done by aggregating all profile information of his/her friends or looking at group memberships of the stranger. In the literature, this approach has been used to infer political views, sexual orientation or hidden parts of profiles (see [82] for an example). The implicit assumption in this approach is that we can easily obtain group memberships of the stranger or profile informations of stranger's friends from the social network. However, current social networks no longer allow this access, and we cannot even get the complete friends list of a stranger. For example, in the past Facebook had allowed researchers to access all information about users in a local network (e.g., users from a city, users from a university) but growing privacy threats compelled Facebook to discontinue this service. Moreover, even if the social networks allow this access, social network users are now able to hide their profile information by strict privacy settings. As a result, all the information we can get about a stranger is limited to the mutual friends he/she has with the target user, because by default social network APIs allow us to retrieve the list of mutual friends.

To overcome this dearth of information about a stranger, we propose an inferral method based on a localized approach. This is motivated again by the homophily theory, stating that users with similar profiles will more likely create a relationship. Therefore, given users u and x, the proposed technique to infer x's missing item values exploits profile information of mutual friends of u and x. By doing so, we avoid considering profiles of all x's friends. More precisely, we make use of the majority voting scheme [110, 97] to infer the value of a missing item i in x profile from the i's values in the mutual friends profiles. In majority voting, each mutual friend has a vote, and a count of votes V_v from all mutual friends is associated with each possible value v of item i (e.g., if there are two mutual friends who live in London, then London will have 2 votes). Upon collecting votes from n mutual friends, for each possible item value v, we compute the percentage of vote P_v as V_v/n. Then, the value with the highest percentage of vote is chosen as the inferred value for that given missing item.

In this process, two limitations can affect the inference: (1) small number of votes, (2) no clear majority for a value in voting. To prevent inferences from a very small number of votes, we use a required minimum number of votes from friends, denoted as f. Similarly, a minimum percentage (i.e., a threshold t) of votes is used to prevent inferences when there is no clear majority in the voting. Thus, a value v is inferred, if $V_v \geq f$ and $P_v \geq t$.

Example 3.2.2 *Assume that we are trying to infer current location of a stranger who has 5 mutual friends with the target user. Let us assume that mutual friends have current locations listed as A, B, A, C and A. There are* $m = 3$ *distinct item values (A, B, C),* $n = 5$ *votes, and* $V_A = 3$, $V_B = 1$, *and* $V_C = 1$. *Hence,* $P_A = 3/5$. *If we set* $f = 7$ *or* $t = 0.85$, *we cannot make an inference, because at least 7 friend votes or a value with* 0.85

percentage of all votes is required. If $f = 3$ and $t = 0.5$, then we can infer the current location of the stranger as A.

The system learns the required number of friend votes f and the threshold t for user u by inferring item values of each of u's friends and comparing the inferred values with friends' existing values. Note that f and t values are user specific. With every f and t value pair, count of inferred values and precision in inferrals are expected to be different. At this phase, we had two goals in mind; inference should be precise and it should allow us to fill in as much missing profile items as possible. Small f and t values help us to infer more values with smaller precision. Big f and t values, on the other hand, bring better precision but fewer inferred values.

To detect the optimum value of f and t, we use the following modularity function:

$$\mathcal{L}(f,t) = \mathcal{P}(f,t) \times Log(\mathcal{C}(f,t))$$

where $\mathcal{C}(f,t)$ is the number of correct inferrals and $\mathcal{P}(f,t)$ denotes precision in inferral. Thus, values of f and t are chosen to maximize the modularity function $\mathcal{L}(f,t)$. During the selection of this modularity function we heuristically experimented with different variations. A possibility was to multiply precision and the number of inferred items as in $\mathcal{L}(f,t) = \mathcal{P}(f,t) \times \mathcal{C}(f,t)$. This did not work because while $\mathcal{P}(f,t)$ showed a linear behavior (it increases or decreases linearly), the number of inferred items were exponentially increasing or decreasing. To smooth the exponential part (number of inferred items) we therefore used a log transformation and experiments with this specific formula provided better results. As the experimental results in Section 3.4.3 will show, this formula achieves precisions as high as 71.7%. Even with unstructured [1] multiple value items, we have found the performance of the modularity function satisfactory.

Profile Similarity Measure. In defining our similarity measure, we have taken into account that user profile mainly contains categorical data. Literature offers several similarity measures for categorical data (see [21] for a survey). Among these, the most basic measure is the overlap similarity which returns a similarity of 1 if two item values are identical and 0, otherwise. Other more complex measures return a non-zero similarity in case of non-identical item values, as well as a value different from 1, in case of two identical item values. In general, this is achieved by taking into account the item values distribution in a dataset. This dataset is used to discover similarity relations between item values, therefore its creation requires careful consideration. Adhering to the homophily theory, we assume that friends of a target user are similar to him/her, and profile items of friends can guide us in computing similarity of strangers. For this reason, we propose aggregating item values from friends of the target user u to create the dataset. With this approach, the measure learns the dataset from social network data automatically, therefore creating an ontology or defining a dictionary is not needed to compute similarity between value pairs.

Given a dataset, various measures differ in their approach to compute a similarity value for two item value pairs. We will give two examples to illustrate different behaviors. Both

[1] User inputs with Facebook auto-completion and aggregation was imposed just in recent years. Before this, users could enter unstructured texts.

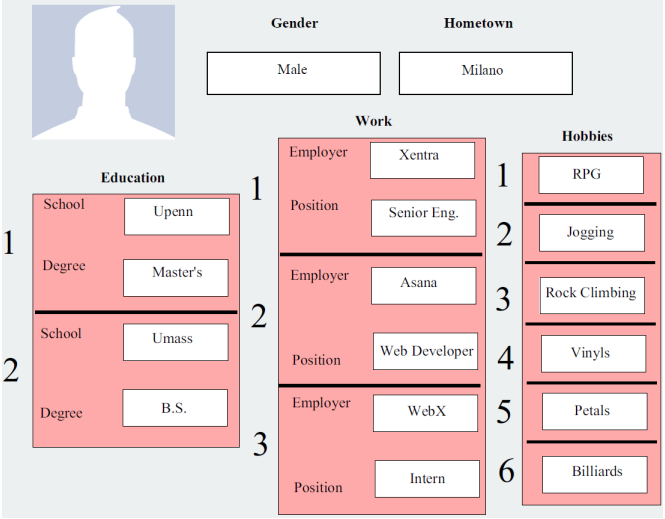


Figure 3.2: An exemplary profile for target user u. The profile consists of single valued gender and hometown items, as well as multiple valued education and work items.

in Occurrence and Inverse Occurrence Frequency two identical item values are assigned 1 for similarity, but similarity for non-identical item values is computed in different ways. In Occurrence Frequency (OF), "mismatches on more frequent values are assigned higher similarity" [21]; if two values are non-identical but highly frequent in the dataset, similarity is higher. In comparison, Inverse Occurrence Frequency (IOF) gives higher similarity to mismatches on less frequent values.

Among all measures discussed in [21], we chose to use the occurrence frequency for profile similarity, because our dataset consists of items from all friends of user u, and, according to homophily theory, we expect that friends are in some way similar to each other. Thus, we chose a measure that gives 1 for two identical profile item values, and, in case of non-identical values, gives a non-zero value which is computed by considering the frequency of the item values in the dataset (i.e., the friend profiles). We have validated this choice by considering other categorical similarity measures and observing their performance on social network data (see experimental results in Section 3.4.2).

Before giving our profile similarity definitions, profile items of a target user u are shown in Figure 3.2 to familiarize the reader with the structure of profile data on a social network. In general, items on a social network profile can be classified according to two orthogonal dimensions. A first classification is based on whether they are single value (e.g., gender) or multiple value (e.g., hobbies) items. The second dimension is related to whether they contain multiple sub-fields (e.g., education contains degree and school) or not. In Figure 3.2, gender and hometown are single value items and hobbies is a multiple value item. Furthermore, education and work are multiple value items with multiple subfields. Values of these items are shown with numbers, such as 1, 2 and 3. Similarity of single-value (sub) items is computed using Occurrence Frequency (OF) as follows:

Definition 4 (Single-value similarity) *Let i be a single value profile item consisting of a set of subfields Sb.*[2] *Let i_u and i_x be the value of item i in u and x profiles, respectively. The value of the n-th subfield of i_u is denoted as $i_u^{(n)}$. Furthermore, we create a dataset D that contains item values from profiles of u's friends. In this dataset, A and B are defined as $A = log(|D|/r(i_u^{(n)}))$ and $B = log(|D|/r(i_x^{(n)}))$, where $r(i_*)$ denotes the number of friends who have item values i_* in D. Single value similarity is given by the following formula:*

$$OF(i_u, i_x) = \frac{1}{|Sb|} \times \sum_{n \in Sb} \begin{cases} 1 & \text{if } i_u^{(n)} = i_x^{(n)} \\ (1 + A \times B)^{-1} & \text{if } i_u^{(n)} \neq i_x^{(n)} \end{cases}$$

Single-value similarity computes the similarity of two single-value items on different profiles. As shown in Figure 3.2, some items (e.g., education) can have more than one value (e.g., {(Upenn, Master's),(Umass, B.S.)} and therefore similarity of single value items must be aggregated to compute the item similarity. In this aggregation, we first compute single-value similarities for all possible item value pairs on the two profiles, and pick the highest valued ones. More precisely, Item Similarity is computed as follows.

Definition 5 (Item similarity) *Let i be a profile item consisting of a set of subfields Sb. Let values(i_u), values(i_x) be the set of values for item i in u and x profiles, respectively. $\forall h \in values(i_u)$, let SimItem($h$) $= \{OF(h,k) | k \in values(i_x)\}$ be the set of single-value item similarities computed between an element h in values(i_u) and all elements in values(i_x). Similarity for item i is defined as:*

$$S_i(u, x) = \frac{1}{|values(i_u)|} \times \sum_{h \in values(i_u)} max(SimItem(h))$$

Item similarity computes similarity of an item (e.g., education, hometown) on two profiles, however social network profiles do not consist of a single item. In fact, most social networks have tens of profile items, such as current location, education and hometown. To find the profile similarity, these item similarities must be once again aggregated. As some items can have different weights in computing the similarity, we employ a weighted average model in aggregation. This allows us to indicate importance of some items over others. Therefore, we define profile similarity of stranger x to target user u as follows.

Definition 6 (Profile similarity) *Profile similarity of users u and x is defined as:*

$$PS(u, x) = \frac{1}{|I|} \times \sum_{i \in I} \beta_i \times S_i(u, x)$$

where β_i and $S_i(u, x)$ are the importance and the similarity of the ith item, respectively, whereas I is the item set on user's profile. Importance coefficients are user-defined.

User	School	Dept.	Degree
u	Umass (75)	CSE (57)	Ph.D.(8)
u	SUNY (23)	CSE (57)	B.S.(37)
x	UPenn (21)	CSE (57)	Ph.D. (8)
x	UPenn (21)	EE (11)	B.S. (37)

Table 3.1: Education items of a stranger (x) and a target user (u). Number of occurrences of item values on profiles of u's friends are given in parentheses. These values in parantheses are what we termed as the dataset, and they are used to find the similarity between two non-identical item values.

Example 3.2.3 *Values of the education item of target user u and stranger x are those given in Table 3.1. For instance, target user u and stranger x each have two education values on their profiles. Target user u had her B.S. degree from SUNY in Computer Science. Next to item values, we give the number of u's friends who have the same item value. In total, we assume that there are 200 friends of u. For example, 23 of u's friends have studied in SUNY. Target user u also got her Ph.D. degree in computer science from Umass. 75 friends of u have also studied in Umass, but only 57 of them studied computer science, and of those only 8 had their Ph.D. in Computer Science. Stranger x studied EE in UPenn as an undergraduate student and had a Ph.D. in CSE from UPenn. In the table, we see that 21 friends of u have studied in Upenn. Similarly, we see that 11 studied EE in various degrees (B.S., M.S. or Ph.D.).*

Given the information in Table 3.1, we want to compute the similarity of x's education values to education values of u. There are three subfields (school, department, degree) in education, so $Sb = 3$.

For the two item values of user x, we need to compute single-value similarities of all item value pairs on u and x profiles. These are $OF(edu_{u.1}, edu_{x.1})$, $OF(edu_{u.1}, edu_{x.2})$, $OF(edu_{u.2}, edu_{x.1})$ *and* $OF(edu_{u.2}, edu_{x.2})$. *As we have two item values on the profile of u, we need to choose the two highest values among them. From Table 3.1, we can see that* $OF(edu_{u.1}, edu_{x.1})$ *returns the highest similarity value because out of three subfields, two of them (i.e., dept:CSE and degree:Ph.D.) are identical in both item values. When all three subfields are considered, the single-value similarity for these item values is computed as:*

$$\mathrm{OF}(\mathrm{edu_{u.1}}, \mathrm{edu_{x.1}}) = \frac{1}{3} \times \{ \frac{1}{1 + \log(\frac{200}{75}) \times \log(\frac{200}{21})} + 1 + 1 \}$$

The second and the third subfields (CSE and Ph.D.) are identical on both profiles ($edu_u^{(2)} = edu_x^{(2)}$ *and* $edu_u^{(3)} = edu_x^{(3)}$), *so from Definition 4 similarity is 1 for these two subfields. For the first education subfield,* $edu_u^{(1)} \neq edu_x^{(1)}$, *therefore occurrence frequencies*

[2] where $|Sb| = 1$ in case of non-structured items, like gender.

of $edu_u^{(1)}$ (University: Umass) and $edu_x^{(1)}$ (University: Upenn) are found from the data set to compute similarity. In this calculation, 75 of 200 friends of u have Umass educations, whereas 21 have Upenn educations.

Once the similarity between first education values from both profiles are found, we are left with $OF(edu_{u.2}, edu_{x.2})$ only, because $OF(edu_{u.1}, edu_{x.1})$, $OF(edu_{u.1}, edu_{x.2})$ and $OF(edu_{u.2}, edu_{x.1})$ involved $edu_{u.1}$ or $edu_{x.1}$ but these education values had already been used. Here only the degrees (B.S.) are identical on both profiles, whereas other two subfields are non-identical:

$$OF(edu_{u.2}, edu_{x.2}) = \frac{1}{3} \times \{ \frac{1}{1 + log(\frac{200}{57}) \times log(\frac{200}{11})} + \frac{1}{1 + log(\frac{200}{23}) \times log(\frac{200}{21})} + 1\}$$

Once those two similarity values are found, item similarity for education can be computed as follows:

$$\begin{aligned} S_{edu}(u, x) &= \frac{1}{2} \times \{OF(edu_{u.1}, edu_{x.1}) + OF(edu_{u.2}, edu_{x.2})\} \\ &= \frac{1}{2} \times \{0.901 + 0.703\} = 0.802. \end{aligned}$$

3.3 Experimental Methodology

We conducted several experiments to provide a comparison between the proposed similarity measures and those discussed in the literature. The comparison is done by showing how similarity measures can forecast the creation of a new relationship in a graph. With this aim, we considered several types of graphs, which can be grouped into "directed graphs" (i.e., Youtube and Epinions datasets) and "undirected graphs" (i.e., Facebook and DBLP datasets).

To verify how much similarity measures correctly forecast the creation of a new edge, we adopt the following methodology. We assume to have a social network graph $\mathcal{G}$, where each edge is complemented with information on the time of its creation. Then, given $\mathcal{G}$: (1) we randomly select a target user u and one of its strangers x; (2) we generate the graph $\mathcal{G}$ at a given time instant T, denoted as $\mathcal{G}_T$,[3] where T just precedes the time of creation of edge $e = \langle u, x \rangle$ in $\mathcal{G}$; (3) we compute the similarity between u and each of its strangers, x included, in $\mathcal{G}_T$ according to the considered measures; (4) we evaluate how the network and profile similarity values between u and x in $\mathcal{G}_T$ imply the creation of the edge $e = \langle u, x \rangle$. The strength of implication is given by how the similarity measure ranks x among all other

[3]This is done by creating the graph using only edges established before time T.

strangers in G_T. In the ideal case, the measures would rank x as the first stranger that the target user would add.

We conduct several experiments by focusing on edge prediction based on profile similarity, network similarity, as well as a combination of them. Before describing the results of experiments, we want to clarify the relationship between rank and similarity, since they can confuse the reader about prediction performances.

Rank and Similarity. Although similarity measures involve using different data (network and profile information), for each one of them, edge prediction aims at producing *for each target user* an ordered list of potential new friends from his/her strangers. In this ordered list, position of a stranger is called his/her rank, and the goal is to compute the list in such a way that strangers who are most likely to create new friendships with target users are ranked the highest.

Ranking requires ordering strangers on a scale; similarity measures use similarity as the scale and order strangers according to their similarity values. As a result, the most similar stranger has rank 1, whereas the least similar stranger is at the bottom of the list. The rank of this lowest stranger depends on the total number of strangers. Because of this, ranks cannot be compared in their absolute values. Therefore, we normalize the ranks such that percentage ranks 1 and 0 correspond to the top and bottom strangers, respectively. More precisely, for each target user, percentage rank for an m ranked stranger in a list of n total strangers is computed as $1 - m/n$. For example, if a stranger is ranked 10th in a list of 100 strangers, he/she is assigned the rank 0.9. In what follows, we will use *rank* and *percentage rank* interchangeably to mean percentage rank.

Although similarity and rank values of a stranger are directly proportional because similarity is used to compute rank, similarity and rank cannot be used interchangeably. Rank of a stranger is not equal to his/her similarity, because rank is found by normalizing an ordered list of all strangers. For example, a rank value of 0.9 does not mean that the similarity value is 0.9. Two strangers ranked 0.9 from the prediction lists of two target users can have drastically different similarity values such as 0.90 or 0.10. For the same reasons, a similarity value of 0.9 does not imply a rank value of 0.9.

A further issue is about comparing values from different similarity measures. The characteristics of the measures can be different, therefore similarity values from two different similarity measures cannot be directly compared. This is because one measure can assign very low similarities to all strangers, whereas another one can assign high similarities to all. However, percentage ranks of the same stranger from two similarity measures can be compared because ranks are normalized by taking similarities of all strangers into account.

Having explained similarity and rank notions, we will now review the network and profile similarity measures against which we compare our measures.

Network Similarity. We compare our network similarity measure (NS) with other well known similarity measures, namely L1 Norm, Cosine Similarity [40] and positive correlation Pointwise Mutual Information (PMI) [36]. We modify the considered measures to take into account friendship links as follows:

Measure	Assigns 0	Assigns 1	For multiple value pairs
OF	never	identical	simple average
IOF	never	identical	simple average
Eskin	never	identical	simple average
Overlap	non-identical	identical	simple average
Lin	possible	possible	function of log frequencies
Goodall	non-identical	possible	simple average

Table 3.2: Similarity values for identical and non-identical item values for different categorical similarity measures. Assigning 0 implies that there is no similarity between two item values. The highest similarity value is 1, and it is assigned to indicate maximum similarity. By *possible*, we mean that a specific frequency distribution of values in the dataset can result in a 0 or 1 similarity. In computation with multiple item value pairs, an overall similarity value is computed by simple averaging, or by using the similarity value of each pair in a function.

$$L1(u,s) = \frac{|F_u \cap F_s|}{|F_u| \,.\, |F_s|}, Cos(u,s) = \frac{|F_u \cap F_s|}{\sqrt{|F_u| \,.\, |F_s|}}$$

where with F_u and F_s we denote friends of user u and friends of stranger s, respectively. Pointwise mutual information is computed in probabilistic terms as follows:

$$PMI(u,s) = P(F_u, F_s) \,.\, log \frac{P(F_u, F_s)}{P(F_u).P(F_s)}$$

where joint probability distribution function $P(F_u, F_s)$ computes the probability of a social network user being friends of both u and s, whereas marginal probability distribution functions $P(F_u)$ and $P(F_s)$ are probabilities of a social network user being a friend of u and s, respectively.

Given a new edge e between a stranger x and user u, we say that a network similarity measure m_1 wins over another similarity measure m_2, if rank of x according to m_1, denoted as $R_{m_1}(x)$, is higher than the rank obtained according to m_2, i.e., $R_{m_1}(x) > R_{m_2}(x)$. Similarly, a loss for m_1 is recorded when $R_{m_1}(x) < R_{m_2}(x)$, finally a draw occurs when $R_{m_1}(x) = R_{m_2}(x)$.

For L1, Cosine and PMI measures, total number of wins, losses and ties are computed against our network similarity measure by considering all new edges in the dataset. We will detail this process by the following example.

Example 3.3.1 *Assume that we are given a timestamped edge e connecting a user u to a stranger x. Creation time of e is denoted by T_1. Assume that at time $T < T_1$ user u has only three strangers $S = \{x, x_2, x_3\}$. Furthermore x_2 and x_3 will not become friends of u until a time $T_2 > T_1$. Note that $x \in S$. So, at time T, u and x are not yet friends.*

Measure	x	x_2	x_3
NS	0,6	0,4	0,1
PMI	0,06	0,05	0,023
Cosine	0,2	0,3	0,1
L1	0,3	0,35	0,4

Table 3.3: Similarity values of three strangers for corresponding categorical similarity measures.

Assume that $NS(.)$, $PMI(.)$, $L1(.)$ and $Cos(.)$ similarity values for (u, x) at time T are given in Table 3.3.

Ordering similarities of x, x_2 and x_3 in each measure, we see the percentage ranks in the list as $R_{NS}(x) = 1$, $R_{Cos}(x) = 0.33$, $R_{L1}(x) = 0$ and $R_{PMI}(x) = 1$. Note that each measure orders the similarity values of x, x_1 and x_3 separately. As a result, x has rank 1 with 0.06 similarity in PMI, whereas it has the rank 0 with 0.3 similarity in L1. Hence, we say that for this edge, NS wins over L1 and Cosine, but it ties with PMI.

Comparison of our network similarity measure against L1, Cosine, and PMI will be done with total win numbers with respect to the considered measures. We denote these numbers as $W(NS, m)$, $L(NS, m)$, and $T(NS, m)$, respectively, where $m \in \{L1, PMI, Cosine\}$. Win ratio of our measure NS against other measures is defined as:

$$WinRatio(NS, m) = \frac{W(NS, m)}{W(NS, m) + T(NS, m) + L(NS, m)}$$

Profile Similarity.

In the experiments we will compute profile similarity with Overlap, IOF, OF, Eskin, Lin and Goodall measures which are defined for categorical data and discussed in [21]. All the considered measures assign similarity values to strangers based on their item value frequencies in a dataset. Table 3.2 shows the behavior of the considered similarity measures according to different dimensions. For Lin and Goodall measures, there are two and four versions, respectively, that are slight alterations of their original formula. These variations are not shown in Table 3.2 because their behaviors in considered dimensions are identical. Performance results of these variations will be given in Section 3.4.2.

The considered measures can be grouped according to the similarity value assigned to identical and non-identical item values. As reported in Table 3.2, Overlap and Goodall assign zero similarity for non-identical item value pairs, whereas OF, IOF, Eskin and Lin variations return a similarity value based on item frequencies of the dataset. Some measures also have opposite behaviors. For instance, for non-identical item values, IOF assigns lower similarity on more frequent item value pairs, whereas OF gives higher similarities to more frequent item values. When user and stranger have more than one value for an item, these measures employ different weighting schemes. In Lin variations, a weight based

	Win	Loss	Equal
NS, L1 Norm	213 241	179 052	7 708
NS, Cosine	246 956	145 979	7 066
NS, PMI	206 732	168 963	24 306

Table 3.4: Comparison of win/loss/draw counts

on logarithmic item frequencies is used, whereas in other measures simple averaging is employed.

Although we chose OF as our measure in profile similarity, we believe that the evaluation of all measures in experiments with real social network data will better highlight their usefulness and shortcomings so that in future studies researchers can be better informed about them.

In profile similarity experiments, we will follow a methodology similar to the one we used in network similarity. With timestamped new edges of user-stranger pairs, we will compute profile similarity values for strangers by using different measures to compute item similarity. Based on these results, we will rank strangers and find which similarity measure gives the best performance.

3.4 Experimental Results on Undirected Graphs

In this section, we will discuss experimental results for network and profile similarity on undirected graphs.

3.4.1 Comparison of Network Similarity Measures

In this experiment, we used the real Facebook data set from [129], denoted as FBdata in what follows, where it is possible to determine the time of creation of each edge. The dataset consists of 16K users and 1.5M edges. However, we have considered only a subset of this dataset, consisting of the oldest 480 274 edges, according to their creation dates. This is only to limit the computation cost. In 480 274 edges, 400 000 (83%) edges were created between users and their friends of friends (i.e., 2 hop strangers). Only 80 274 (17%) edges were created between 3 or more hop distance pairs. For each edge e in FBdata, we choose the first node as user u and the other one as user x. We then create the graph status in the time instant before creation of $e = \langle u, x \rangle$, denoted as G_{T_e}, and compute similarity measures among u and all of its strangers in G_{T_e}, including x.

The numbers of win/loss/draw of our network similarity measure (NS) against L1 Norm, Cosine and PMI are given in Table 3.4, where we show that overall, NS has the best performance.

In Figure 3.3, we further analyze the performance, by showing the ratio of wins to losses of NS against the other measures. Figure 3.3 clearly shows that the proposed measure

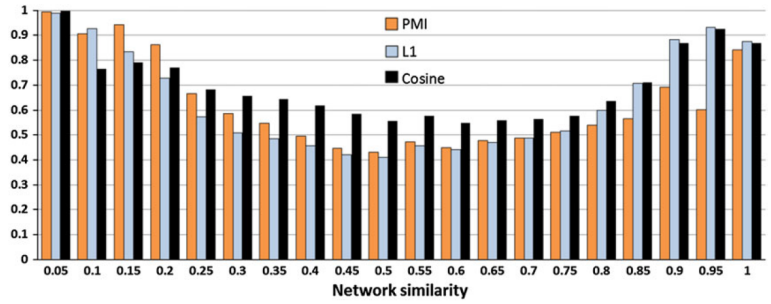


Figure 3.3: Win ratio against L1, Cosine, and PMI. The x-axis shows increasing network similarity values.

outperforms others when network similarity value is less than 0.3 and bigger than 0.7, whereas in the interval between 0.3 and 0.7 we see that the difference of number of wins for NS is small against PMI.

(a) Win/loss ratio over PMI (b) Win/loss ratio over L1

Figure 3.4: Win/loss ratios for [0.3,0.7] similarity range

A better view of this middle interval is given in Figure 3.4. The lowest win ratio against L1 occurs at 0.45 abscissa, with an ordinate value of 0.41. At this point the edges among mutual friends have an unexpected impact on the network similarity, and they increase similarity more than needed. In a sense, connections among mutual friends are given too much importance. This can happen in situations when the user has not many friends, and the mutual friends are well connected.

3.4.2 Comparison of Profile Similarity Measures

In this experiment, we wanted to find out which categorical similarity measure can better predict new edge formation between user and target stranger pairs. From FBdata, we created a dataset for each user-target stranger pair (u, x) by including item values of stranger x, user u and friends of user u. In Table 3.5, we give similarity and rank results for the different measures in average terms. In the similarity column, we report the average similarity of all strangers who have become friends with target users. Similarly, the rank column shows average rank of these new friends. A 0 value for the *rank column* means that the target stranger was ranked the last among all strangers, whereas a 1 value means that the target stranger was ranked the first. In the zero similarity column, we report the

Measure	Similarity	Rank	Zero similarity
OF	0.621	0.483	0.023
IOF	0.623	0.38	0.023
Eskin	0.852	0.322	0.023
Overlap	0.469	0.309	0.247
Lin	0.053	0.428	0.359
Lin1	0.033	0.345	0.431
Goodall1	0.343	0.317	0.247
Goodall2	0.345	0.307	0.247
Goodall3	0.146	0.568	0.278
Goodall4	0.214	0.306	0.249

Table 3.5: Comparison of profile similarity measures. Higher ranks show better performance, and the zero similarity column shows the percentage of strangers that are assigned zero similarity.

percentage of strangers whose similarity to target users were found to be zero.

In average, Eskin gives the highest similarity to target strangers, but it ranks them only higher than 32.2% of all strangers. Average similarity does not convey much information on performance, because a similarity measure can assign high similarity to all strangers. As a result, we cannot make a distinction among strangers. By the rank column, we can have a better view on how similarity measures can predict new edges. Higher average rank means that the measure gives higher probability to edge formation between user-target stranger pairs. Goodall3 gives the best average rank overall; it puts a target stranger higher than 56.8% of all strangers. However, this measure fails to find a similarity for 27.8% of all strangers. This is because all Goodall versions assign zero similarity to non-indentical value pairs. They rank fewer strangers the best but fail to find a similarity value for a big portion of strangers. Our chosen measure OF has the second best average ranking performance, and it returns a zero similarity value only when the stranger or the user has no profile items (i.e., their profiles are blank).

For these reasons, we believe that the experiments on undirected graphs confirm our choice of using OF.

3.4.3 Inferrals in Item Similarity

In doing this experiment we used our data extracted from Facebook consisting of 39 users and 12 567 friends of those users. The dataset consists of network and profile information of these users received by an application from Facebook during a one year period. All information is timestamped, and new friendships or friendship removals are updated in the dataset. Each user profile reflects the profile structure of Facebook, and changes in profile items and their values are also updated. In this dataset, the missing values are distributed as described in Figure 3.5, where for each item the percentage of missing value on 12 567

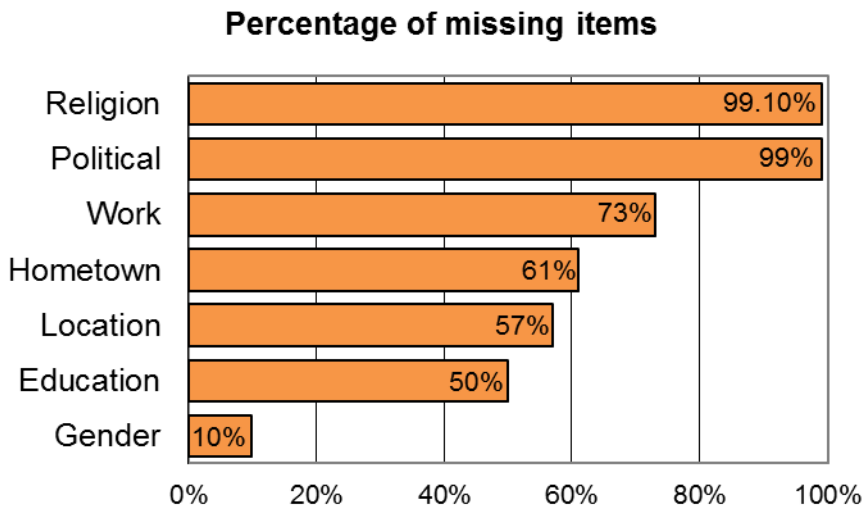


Figure 3.5: Percentage of missing item values for the considered dataset.

	friends (f)	thr. (t)	#inferred	prec.
Gender	3.34	0.59	594	0.728
Location	3.20	0.55	634	0.648
Hometown	3.55	0.61	982	0.717

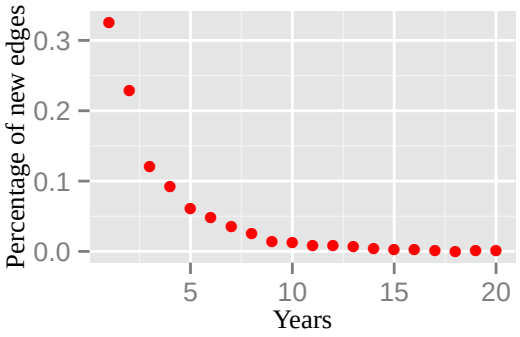
Table 3.6: Inferral results

friend profiles is shown.

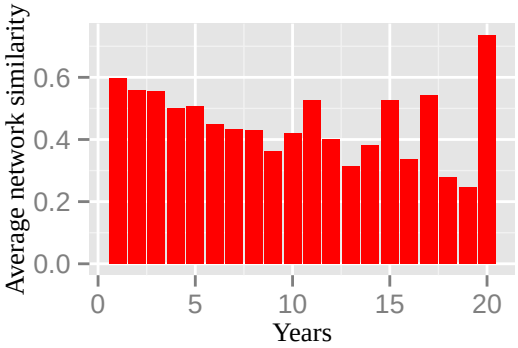
Table 3.6 shows the learned inferral parameters for gender, hometown and location. System learns the friend count f and the threshold t for a user u by inferring item values of each of u's friends and comparing the inferred values with friends' existing values.[4] From our modularity function, number of friends (f) that are required to make an inferral are found to be around 3, and threshold value (t) is close to 0.5. Among considered items, hometown has the biggest friend count for an inferral. In average, we need 3.55 friends to be able to infer a hometown value. Location, on the other hand, has the lowest precision with 64.8%.

Among single value items, political view and religion are the most difficult to infer because only 1% of all users enter this data. Inferring multi-value items proved to be very difficult; as Facebook has been accepting user entered information for these items until recently, abbreviations of schools and work places are prevalent, and natural language processing is required before starting an inferral.

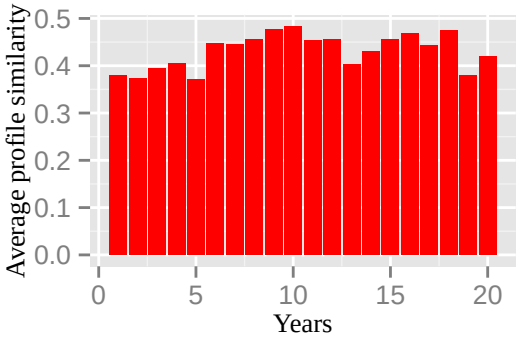
[4]In the table, precision is the correct inferrals over all inferrals, i.e., $precision = \#correct\ inferrals/\#all\ inferrals$.



(a) Percentage of newly created edges.



(b) Average network similarity for edge creation.



(c) Average profile similarity for edge creation.

Figure 3.6: The effect of network and profile similarity for new edge creation.

3.4.4 Combining Network and Profile similarity

To evaluate how our network similarity measure compares to the profile similarity measure on undirected graphs, we carried out experiments on data extracted from DBLP datasets[5] and enriched with author information from the ACM digital library. By exploiting these data, we generated a co-authorship graph, where each edge is complemented with the year of its creation, and each user profile contains research fields and subfields organized according to the ACM classification. More precisely, the obtained graph, which we refer to as PCA, has been generated by crawling DBLP authorship relationships starting from a given author X to other nodes who are at most 3 hops away from X. In its current form, PCA contains 40 115 edges and 3651 nodes.

From PCA, we tried to observe how network and profile similarity affect co-authorships. We defined friendship of two users as their co-authorship in a research article. The oldest co-authored research article in the DBLP dataset denoted a friendship edge between two users, and the edge is timestamped with the publication date of the article. Given a node u and one of its strangers x (i.e., for u and x, there exists a node f which is a co-author of both u and x), we analyze the time interval, denoted as $\hat{t}(u, x)$, since x became a friend of f until x becomes a coauthor of u. We analyze the time interval by comparing how it changes against network and profile similarity values.

Figure 3.6 shows three graphs for years of edge creation, where the axes show the $\hat{t}(u, x)$ values. Figure 3.6a shows edge creation as $\hat{t}(u, x)$ increases. Interestingly, 33% of all the new co-authorship edges are created within one year after having a stranger. Extreme $\hat{t}(u, x)$ values are also possible, but very unlikely: only 1, 3 and 2 pairs of users coauthor a publication 18, 19 and 20 years after they were linked through a mutual coauthor. Despite these exceptional values, in average it takes $\hat{t}(u, x) = 3.29$ years to create a link between two users once they have a common coauthor, and the median time is 2 years.

Figure 3.6b shows how network similarity affects the edge creation rate. Despite some exceptions, as $\hat{t}(u, x)$ increases, average network similarities decrease, which highlights that users that are more connected (in terms of network), coauthor a publication sooner than those that are less connected. Exceptions are in 11, 15, 17 and 20 years as the average similarity in these years increase unexpectedly. However, these years only represent 1.5% of the all edges. In general, this graph pointed out how human connections between authors, modeled as network similarity, affect creation of new co-authorships. It also shows that this kind of relationships is very much affected by the time. Figure 3.6c shows the profile similarity values between u and x when $\hat{t}(u, x)$ increases. Despite some exceptions, profile similarity distribution along years is more uniform than it is in the network similarity graph. This highlights that authors can create coauthorship relationships even after a long term if they have common research fields (i.e., they have higher profile similarity values), regardless of how much they are connected.

[5] http://dblp.uni-trier.de/xml/

3.5 Experimental Results on Directed Graphs

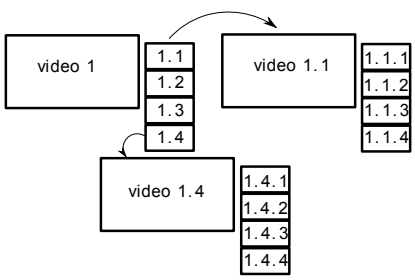
For directed graph experiments, we evaluated similarity measures on two datasets. The first dataset consists of Youtube videos, from which we created a directed graph based on "relations" between a video and those videos suggested by Youtube as "related videos". The second dataset is the Epinions.com dataset [88], where trust / distrust relations between users are mapped onto a directed graph. In the following subsections, we will discuss our findings.

3.5.1 Comparison on Network Similarity Measures

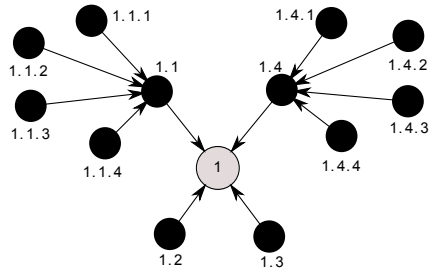
In this section we will explain experiments for network similarity on both Youtube and Epinions graphs.

Experiments on Youtube

The Youtube dataset that we have considered has been crawled by Cheng et al. [33] between February 22nd, 2007 and July 27th, 2008. In their 59 crawls, Youtube video pages have been visited, and for each video, 20 related videos have been added to the dataset. From this dataset we created a directed graph by adding each crawled video and the corresponding related videos as nodes and by inserting a new edge to the graph from a node representing a related video to the node denoting the crawled video. More precisely, in our graph we represent a video as a target user and its related videos (20 related videos) as friends. Hence, related videos of related videos are the strangers of a video, i.e., strangers for a target user. Figure 3.7a shows a Youtube interface of videos and related videos, whereas Figure 3.7b shows how these videos are represented on a graph. For example, video node 1 is a friend of video 1.1. Similarly, video 1.1 is a friend of video 1.1.1, whereas video 1 is a stranger of video 1.1.1.



(a) Related videos of crawled ones are added to the future crawl queue, and crawled afterwards. Arrows show the crawling sequence starting from video 1.



(b) Related Youtube videos are modeled having directed edges to the crawled video in the graph. From a page, first four of the 20 related videos are shown.

Figure 3.7: Graph creation process in Youtube dataset.

First, we will explain the usage of Youtube dataset for evaluating our measures.

When a video is first updated to Youtube, it does not have user views and hence the first data that can be used to compute its related videos are user-generated tags, title and description of the video ([34]). These information pieces are mostly required to upload a video (e.g., entering a title for the video or choosing a category).

A similar process happens in a social network (e.g., Twitter.com) during registration, since users enter few profile information. At this step, the user does not have any preference about possible new friends, so these can be suggested by only looking at user profile data. For new friends / followings, existing social networks, like Twitter.com, offer lists of people to follow based on user profile information (users with the same interests can be offered as potential friends).[6] Similarly, at the beginning, Youtube offers related videos that have similar tags or titles. After the video is uploaded, a percentage of social network users watch the video and Youtube creates a more robust related videos list depending on what other videos these users watched.

As related videos are aggregated from these user watch lists by employing the recommendation algorithms of Youtube, we argue that choosing the most related videos is similar to user decisions on which other social network users to follow and be friends with. Validity of our assumption is then directly linked to how efficient these algorithms are in finding related videos. In the past, the quality of Youtube algorithms has been observed by a work on a Youtube dataset. In ([135]), Zhou et al. note that "YouTube recommendation helps viewers discover videos of their interest rather than popular videos only". This result implies that related videos are indeed reflective of the video's content because same users have shown interest in the same videos. [34] also find that Youtube provides a well clustered network of videos. If the Youtube recommendation algorithms cannot effectively find the most related videos, clustering would be poor (for a traditional social network clustering would also be poor if social network users followed other users randomly). This is also what we expect to see in a social network because of homophily (finding similar nodes close to each other on the graph, and dissimilar nodes distant from each other).
A further important consideration is due to the fact that the network evolves in time. Indeed, although a video itself is not a living entity and does not choose its related videos list[7], users' video watching patterns change gradually in time. For instance, due to sociocultural issues some videos are more frequently viewed together. An example of this can be Mitt Romney-Barack Obama videos which have become very relevant to each other starting from the beginning of the US election campaign, and now appear together on Youtube very often. This is the time dimension that we focused on. Therefore, videos, although not related in the past, can be relevant to each other in the future. Our experiments are about observing the network in time, and making predictions about new relationships before videos become relevant, that is, before they are recommended by Youtube.

For computational complexity issues, we limited the graph size to a predefined number of seed nodes and their relations. Starting from the oldest crawled Youtube videos and

[6] If there is no profile information, a generic list of celebrity accounts are offered.

[7] Related videos list of a video can not be determined by the user who uploaded the video: http://support.google.com/youtube/bin/answer.py?hl=en&answer=92651

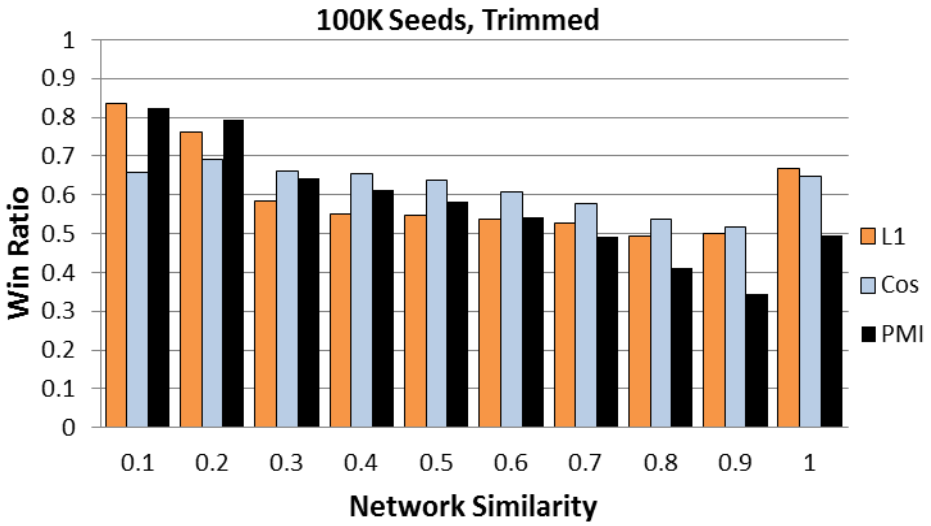


Figure 3.8: Win ratios of NS against the other similarity measures for 100K seed nodes in a trimmed network.

their related 20 videos, we construct a directed graph such that the number of nodes in the graph is equal to a number s. As such, we stopped the graph creation when the number of nodes reaches s.

To evaluate the impact of graph density in similarity measures, we further elaborate the obtained graphs by removing nodes with less than 10 degrees. We will refer to this node removal as "trimming". As such, experiments have been conducted on graphs computed on: (1) 100K seeds ($s = 100k$); (2) 100K seeds and trimmed; (3) 200K seeds; (4) 200K seeds and trimmed.

Note that adopting the methodology introduced in Section 3.3 for the Youtube dataset implies forecasting the creation of a relation between videos. As such, given an edge between videos u and x in the Youtube dataset, we create the graph $\mathcal{G}_T$ at the time instant T just before the edge creation and compute the similarity measure between u and all its strangers, x included.

Performance evaluation involves performance comparison of our network similarity measure with the network similarity measures discussed before. Following the evaluation methodology reported in Section 3.4.1, similarity values for all strangers of a node u are computed according to PMI, L1, Cosine and our network similarity measure (NS).

Figure 3.8 shows the win ratio of our network similarity measure on a 100K seeds, trimmed Youtube graph. The x-axis shows increasing network similarity values in [0.0,1.0]. Our measure performs better than all other measures, and just like in undirected Facebook graph (see Figure 3.3), PMI performs better than L1 and Cosine similarity against our measure. Win ratios against L1 and Cosine similarity measures stay above 0.5 most of the time, whereas L1 is correlated with Cosine in performance.

Seed Number	Trimmed	Number of predictions
100K	No	29 130
100K	Yes	19 379
200K	No	43 816
200K	Yes	29 098

Table 3.7: Number of future friends found in depth 2. Trimmed graphs contain fewer future friends than untrimmed ones

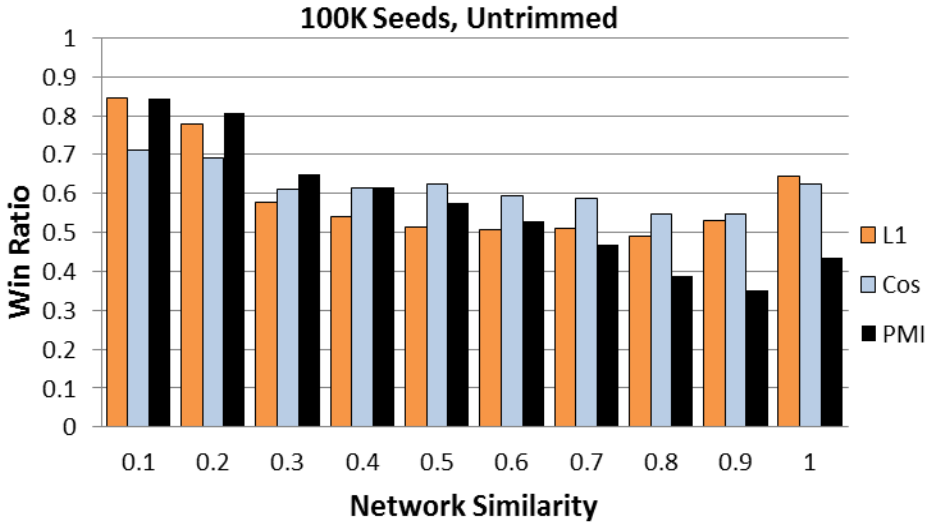


Figure 3.9: Win ratios of NS against the other similarity measures for 100K seed nodes in an untrimmed network

In Figure 3.9 we show the impact of a more sparse graph (i.e., 100K seeds, untrimmed graph) on performance. Win ratios have been lowered around the middle section, but the difference is not too big. This implies that edge density has not made a big difference in performance. As a result, trimmed graphs can be favored when computational complexity issues arise in such big datasets. However, when trimmed graphs lose many strangers (friends of friends), the number of future friends which are found at depth two decreases. As a result, measures produce fewer predictions, as presented in Table 3.7; 200K trimmed Youtube graph produces only as many predictions as 100K untrimmed Youtube graph.

In Figures 3.10 and 3.11, we use a 200K seeded graph to see the impact of graph size on performance evaluation. In the trimmed graph, Cosine similarity performance is worse than it was in the 100K seed graphs, but PMI and L1 do not show a performance fluctuation with PMI having the most stable performance in both 100K and 200K graphs.

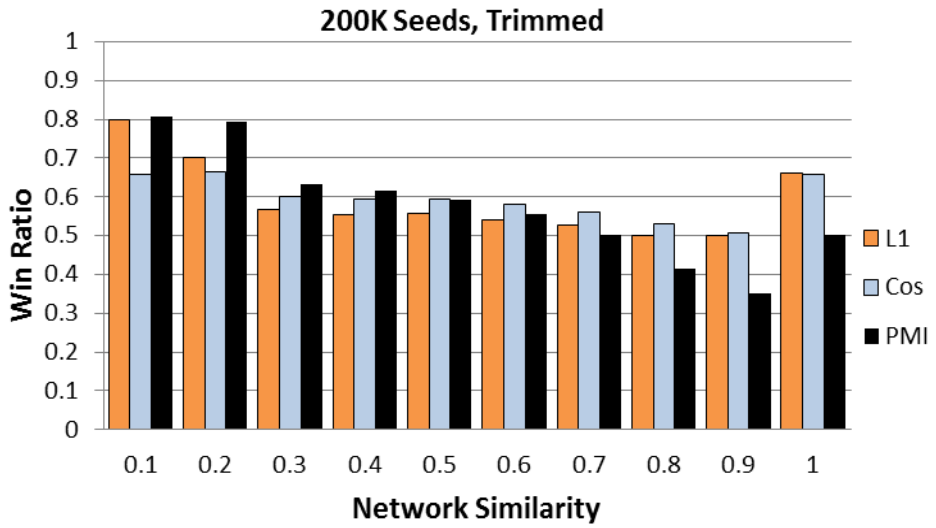


Figure 3.10: Win ratios of NS against the other similarity measures for 200K seed nodes in a trimmed network

In Figure 3.11, untrimmed graph shows better performance for Cosine similarity. Indeed, Cosine similarity is shown to perform poorly when the graph is sparse, whereas L1 performance increases for sparse graphs.

Experiments on Epinions

The Epinions data set ([88]) includes both network and profile information. User profile information consists of subject ids associated with authored articles, where timestamped ratings (+1, -1) represent graph edges. A user gives a +1 rating if he/she likes the content or the behavior of a stranger (i.e., a two hop user) and would like to see more of what the stranger does in the site. Social networks where edges can be weighted to show friendliness or antagonism are called signed networks. If user u rates user x with 1, a trust relation is said to be formed. If the rating is -1, a distrust relation is observed. We divide network information into these two (trust/distrust) sets, and create graphs from each set. A mixed set is also used to combine both trust and distrust graphs.

Due to computational costs, we restricted our network dataset to 498K edges, where 400K edges are timestamped with the first date in the dataset (i.e., 2001 / 01/10). In a 2 year period (2001/01/17 - 2003/08/07), 98K new edges have been added to the graph. Each edge carries a trust (+1) or distrust (-1) sign. In creating the directed Epinions graph, we consider a directed edge originating from user u to the rated user x with its corresponding sign. In experiments, we try to predict these future edges based on the network information of 400K initial edges among nodes. Experiments consisted of predicting 80K edges in the trust network, and 18K edges in the distrust network. In Figure 3.12 predictions[8] are

[8]In Graph theory, triadic closure is used to refer to predictions for such graphs where two pairs of nodes have strong ties, and a weak tie among them is expected, i.e., the dashed line already exists or it is expected

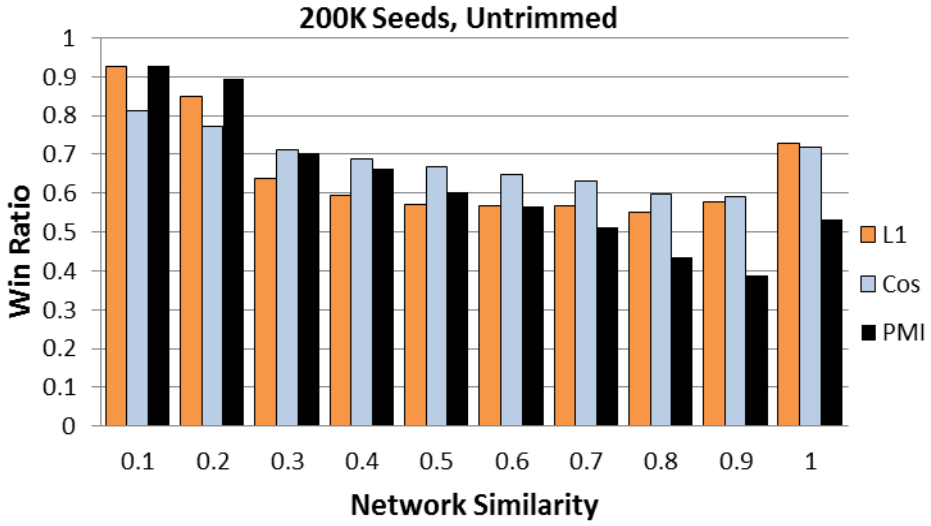


Figure 3.11: Win ratios of NS against the other similarity measures for 200K seed nodes in an untrimmed network

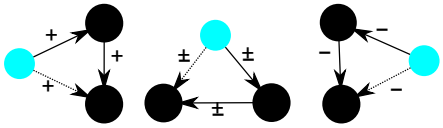


Figure 3.12: Trust, distrust and mixed relations on the graph. Dashed edges are the predictions.

shown with dashed edges for all three graphs. Note that in the mixed graph, edges can have trust or distrust (1,-1) weights.

In Figure 3.13, performance of our network similarity measure is shown for the distrust graph. Overall we see that PMI, L1 and Cosine similarity measures show a poor performance against our similarity measure. However, PMI performs better than L1 and Cosine similarities. In Figure 3.14, we see a distinctly different performance shape for the trust graph. Performance greatly varies in trust and distrust graphs, and our measure performs much better in the distrust graph of Figure 3.13. We will defer the discussion on why this behavior is observed to Section 3.5.2.

We also experimented with a mixed graph of trust and distrust edges. This helps us build a denser graph where edges in a triadic closure can now have alternating trust and distrust signs. Note that the number of predictions in the mixed prediction will be higher than the total sum of closures in trust and distrust graphs. Performance for this graph is shown in Figure 3.15, and it is very similar to what we had found for the trust graph.

to be formed in future. Our experiments try to predict this edge.

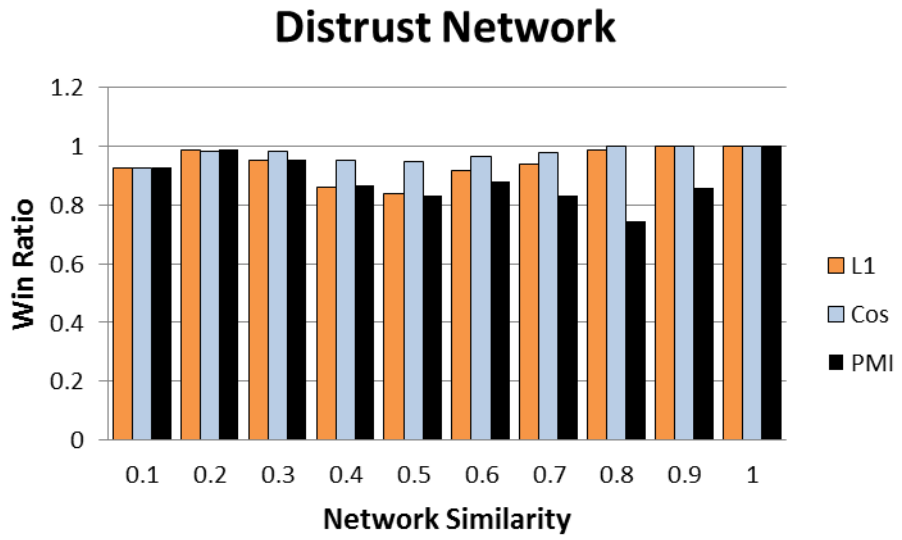


Figure 3.13: Win ratios of NS against other similarity measures for Epinions distrust network. X-axis values are increasing NS values

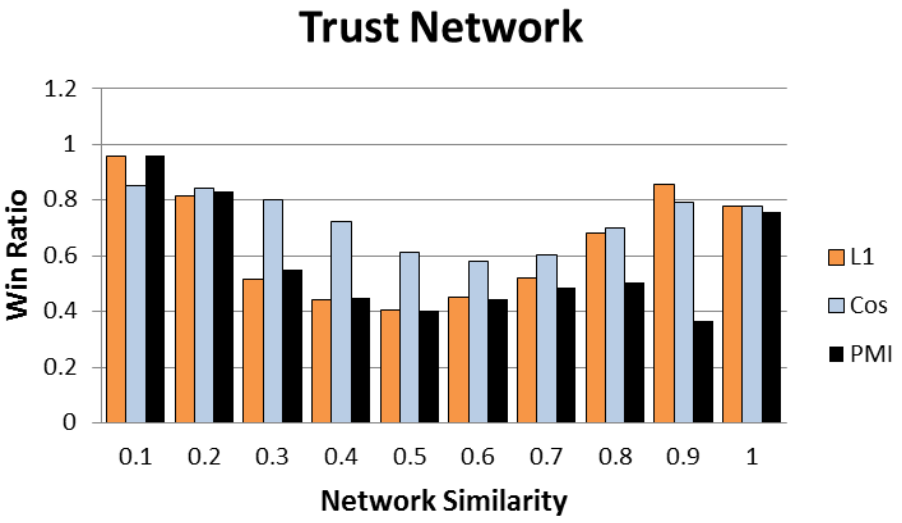


Figure 3.14: Win ratios of NS against other similarity measures for Epinions trust network

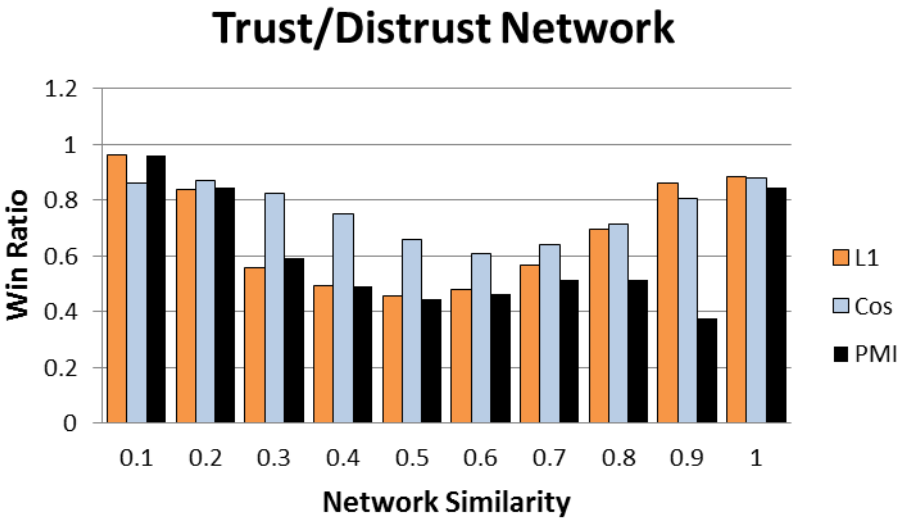


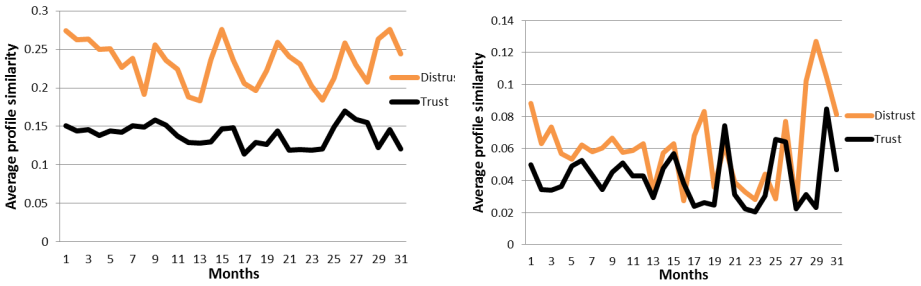
Figure 3.15: Win ratios of NS against other similarity measures for combined Epinions network

3.5.2 Comparison of Profile Similarity Measures on Directed Graphs

Similar to the undirected DBLP graph experiments, we start with the initial graph of 400K Epinions edges that are timestamped with the oldest data (i.e., 2001/01/10). For the future edges of a 2.5 year period (2001 - 2003), we evaluated profile similarities of users and strangers. In doing so, we used Occurrence Frequency (OF) and Goodall3 categorical similarity measures in our profile similarity computations. We chose these two because in Section 3.4.2 we found them to be the top two best performing measures in average rankings of strangers. Figure 3.16 shows the results we had by employing these two measures.

Overall, +1 ratings (Trust) occur between less similar users, whereas -1 ratings (Distrust) are given to higher similarity strangers. This user behavior differs from what we have seen on undirected networks where similar users tend to create friendship edges between them.

To understand this, semantics of this signed Epinions graph should be taken into account. Trust/distrust ratings are the result of stranger articles and content. We believe that high similarity between two users indicate that they can judge each other's articles with more authority, and in less similarity users do not feel confident to rate articles with -1 (Distrust). As a result, people tend to use -1 ratings for articles whose subjects are well known to them.



(a) Average profile similarity with Occurrence Frequency.

(b) Average profile similarity of strangers with Goodall3 measure.

Figure 3.16: Figure shows average profile similarity as time progresses. The x-axis values are number of months since a user first rated another Epinions user.

3.6 Absolute Performance Values

Until now, we have compared our similarity measures to other ones in the literature. In this section, we will present the absolute performance of our similarity measures.

3.6.1 Profile Similarity

The absolute performance results for our profile similarity measure can give a better understanding of how much profile similarity can explain the probability of edge formation.

In Table 3.8 we give the results of our profile similarity measure in predicting next edges on FBdata and DBLP datasets. In the first column, we give the percentage ranks, where rank 0 shows the least likely strangers to become friends with a target user, whereas rank 1 shows new edges with the most likely strangers.[9] The second and third columns show the percentage of strangers that formed new edges with target users in the considered datasets. For example, in FBdata, 9% of new edges were formed with the least likely strangers (rank 0), while this ratio is 31% for the DBLP data set. From this table we see that our profile similarity measure works better for the Facebook dataset, with 11% of new friendships have been established with the rank 1 strangers. The results for both the Facebook and DBLP datasets show that a big percentage of new friends come from lower ranks. We see a similar performance on the directed Epinions dataset for Trust and Distrust rankings. Table 3.9 shows that more than 30% of new friendships on both networks have been created with strangers whose similarity values to target users are in [0, 0.05).

[9] Values are rounded to two decimal points.

Rank	**Percentage of new edges**	
	Facebook	DBLP
0.0	0.09	0.31
0.1	0.08	0.08
0.2	0.11	0.10
0.3	0.09	0.09
0.4	0.08	0.07
0.5	0.19	0.11
0.6	0.05	0.06
0.7	0.10	0.07
0.8	0.02	0.06
0.9	0.08	0.04
1.0	0.11	0.003

Table 3.8: Profile similarity rankings for strangers of newly formed friendship edges on Facebook and DBLP datasets.

Rank	**Percentage of new edges**	
	Distrust	Trust
0	0.312	0.362
0.1	0.084	0.09
0.2	0.106	0.11
0.3	0.087	0.09
0.4	0.072	0.067
0.5	0.11	0.117
0.6	0.06	0.047
0.7	0.067	0.049
0.8	0.063	0.042
0.9	0.036	0.024
1.0	0.003	0.002

Table 3.9: Profile similarity rankings for strangers of newly formed friendship edges on the Epinions dataset.

These results imply that profile similarity is not very powerful to predict new edges when used alone, but this low performance is due to a combination of both missing profile items and semantics of edge creation.

Missing profile items on datasets lower the performance of our measure because we have no data to compute similarity. As a result, many strangers are given zero similarity, and correspondingly zero ranks. In missing item values, Facebook dataset is the best dataset with only 20% of users having blank profiles, whereas this number is 28% for the DBLP data and as much as 35% for the Epinion dataset.

Semantics of edge creation on social networks also plays a role in how much of the profile items are available before an edge is created. On Facebook, profiles are populated at the beginning of memberships, and predicting new edges with these already entered profile data is easier. On the other hand, DBLP profile information is populated by coauthoring new papers; research field is learned after the edges/co-authorships are already formed. If the researcher has not yet authored enough papers, profile similarity performs poorly because it does not have enough profile information to predict new edges. [10]

However, even with these issues profile similarity allows us to understand reasons behind edge formation by discovering common profile values between users who formed the new edge. Indeed, by using profile similarity, edge formation can be attributed to these common profile items such as nationality, education or hometown. For this reason we believe that even if profile similarity results in poor predictions, it provides a good way to understand edge formation.

3.6.2 Network Similarity

In this section we will show the absolute performance of our network similarity measure on all the datasets we considered. These performance values will be given in terms of ranks; we will show the percentage of newly added friends with their corresponding rank values among all strangers. Our similarity measures can then be judged by how much percent of new friends were rank 1 strangers.

In Figure 3.17 we show the percentage of new friends and their corresponding ranks for the DBLP dataset. Like all the other figures in this section, in Figure 3.17 the x-axis is shown for the rank interval [0.8-1.0] for a better view. The y-axis gives the percentage of strangers for their corresponding ranks on the x-axis. The figure shows that the prediction of next friends is accurate 8% of the time (i.e., rank 1 strangers are added as friends 8% of the time). These results also show that 20% of new friends are ranked within the interval 0.97-1.0.

Similarly, rankings for the Facebook dataset are given in Figure 3.18. Here we see that Facebook results are better than DBLP results, because 33% of the time rank 1 strangers are added as friends. This rate drops to 12% for rank 0.99 strangers. Despite this, on Facebook 50% of new friends are ranked within the interval 0.97-1.0.

These results show that our network similarity measure is more efficient to predict new

[10] This problem is known as the cold start problem in recommender systems.

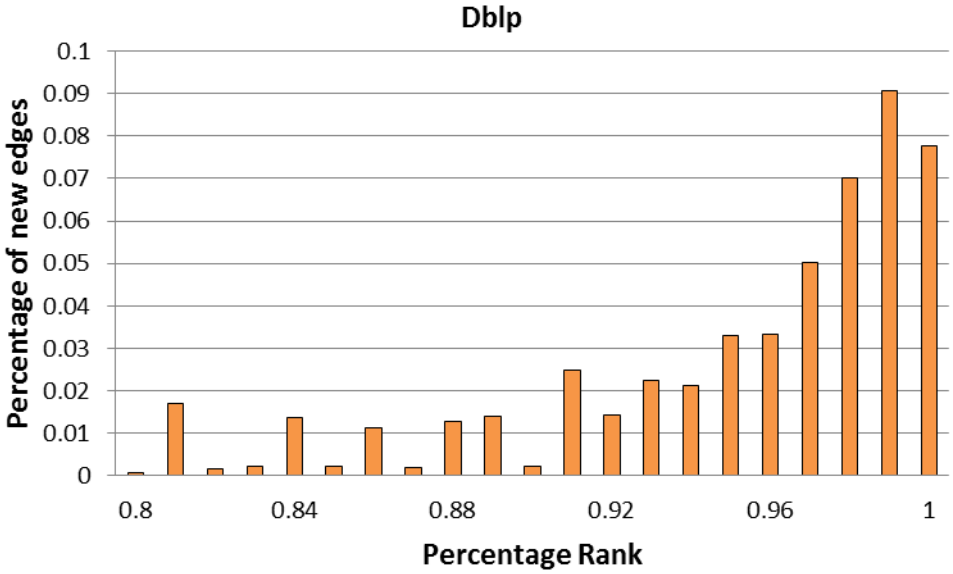


Figure 3.17: Percentage of strangers of newly formed edges on the DBLP dataset by their ranks according to NS.

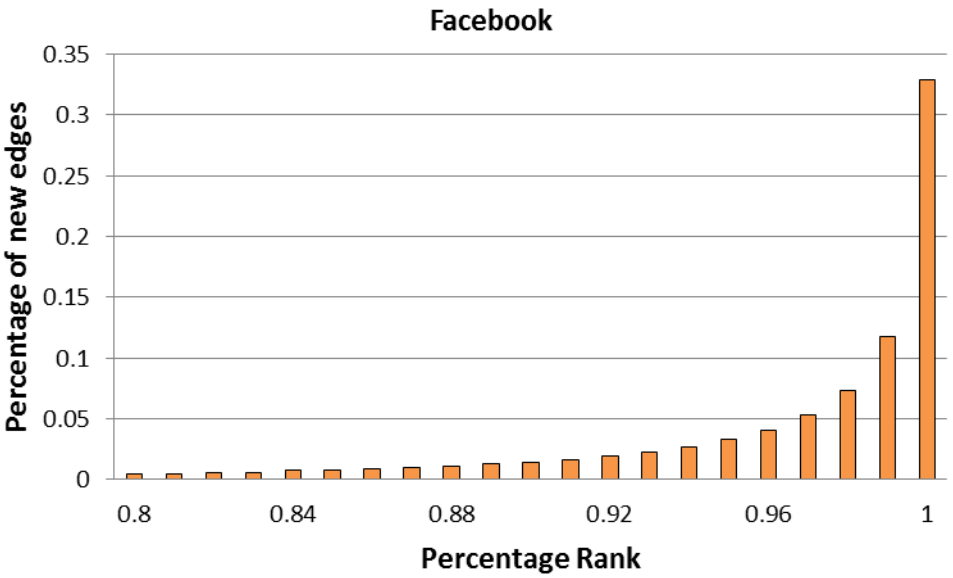


Figure 3.18: Percentage of strangers of newly formed edges on the Facebook dataset by their ranks according to NS.

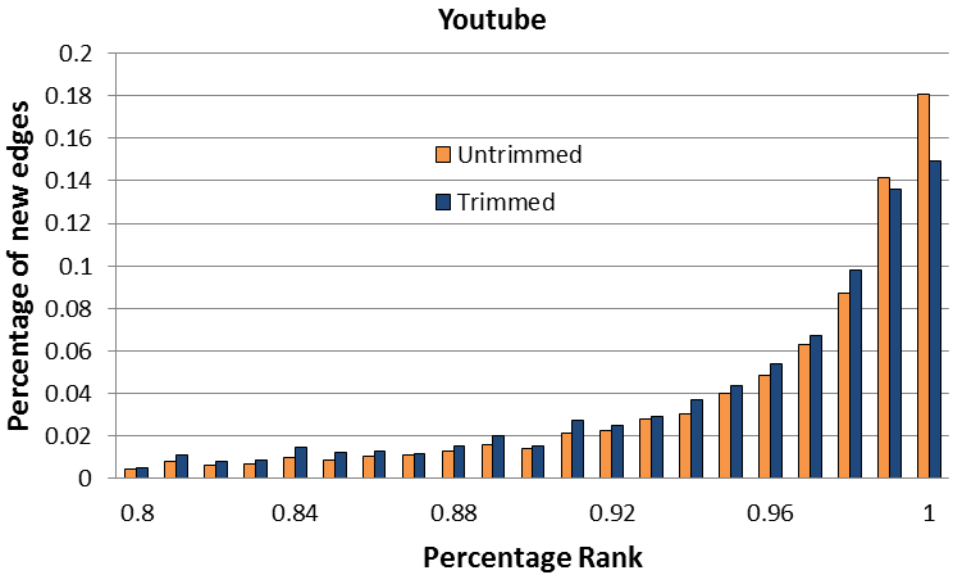


Figure 3.19: Percentage of strangers of newly formed edges on the Youtube dataset by their ranks according to NS.

relations on the Facebook dataset than on the DBLP dataset. Although both networks are undirected social networks, Facebook exhibits stronger homophily effects, i.e., friends of friends tend to create more friendships.

Homophily effects are not limited to undirected graphs. In Figure 3.19, we see that the Youtube dataset performance of our network similarity measure is comparable to Facebook results (in Figure 3.18). For untrimmed and trimmed Youtube networks, 18% and 14% of new friends can be predicted to be rank 1 strangers. In accordance with our comparative results in Section 3.5.1, trimmed network performance versus untrimmed network performance values do not seem to differ very much. This result shows that when computational costs are considered, the Youtube network can be trimmed while still retaining a good performance.

Strong homophily effects can also be seen on the directed Trust and Distrust networks of Epinions. Figure 3.20 shows these rankings for newly formed edges between the target users and their strangers. An interesting observation is that Distrust performance is better than the trust performance.

In Section 3.5, we have shown the comparative performance of our network similarity measure on the Epinions dataset. From Figures 3.13 and 3.14 we observed that distrust prediction of our similarity measure performed better than it did for the trust prediction. 25% of distrust strangers who have rank 1 become friends with target users (i.e., in 25% of the cases we correctly predict who will get a distrust rating from the user first). In the trust network, this figure drops down to 22%. The distrust performance is also better than the trust performance for all ranges. This result is interesting because, although

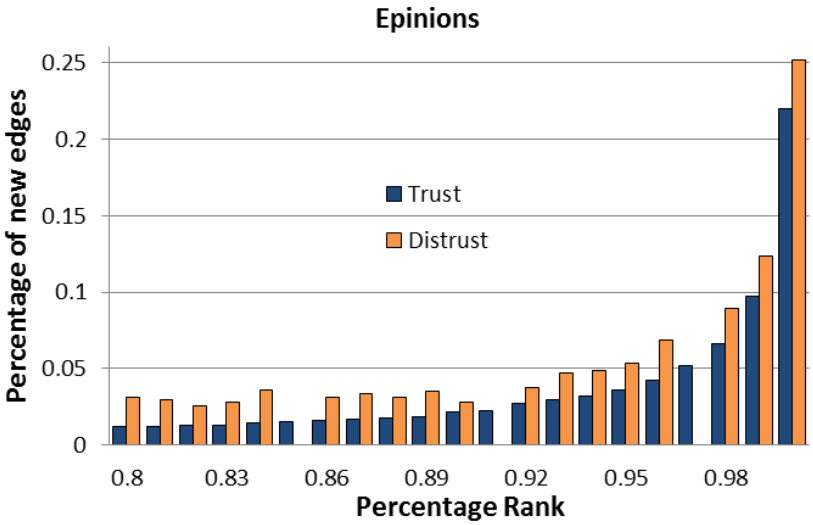


Figure 3.20: Percentage of strangers of newly formed edges on the Epinions dataset by their ranks according to NS.

trust has been used in transitive models with success, distrust transitivity can have mixed performance results. For instance, in some settings on the same dataset [53] found that one step propagation of distrust can help in predicting trust or distrust. Our experiments also assume distrust propagation, but it is propagated for one step only (i.e., A distrusts B, B distrusts C, hence A distrusts C).

From our experiments on profile similarity for the distrust network (see Figure 3.16), we were able to observe that distrust propagates to the users with similar interests (i.e., from the target user to other users who write about the target users's article categories). As such, similar users create edges between themselves by rating each other's articles. This results in a denser network and increases the performance of our network similarity measure because our measure takes into account these rating actions (ratings given by friends of the target user) in the target user's social graph.

As a result of our network and profile similarity experiments, we believe that there is transitivity in the Epinions distrust network, but it is not the distrust notion per se that propagates. Rather, the transitive notion is the expertise that users have in article categories; user ratings of distrust reflect their capability to judge others' work in categories that are similar to theirs.

3.7 Discussion on Experimental Results

In this chapter we mentioned that edge formation on directed and undirected graphs follows different social network dynamics. Whereas consent of both users is required for a friendship in undirected graphs, directed graphs do not have this requirement. As a result,

friend nodes on undirected graphs are more similar in profile than they are in directed graphs. This can be seen in Figure 3.6c, where we evaluated profile similarity of coauthors in an undirected DBLP graph. There are consistent high similarity values in the graph throughout the years. This is because researchers need to work on related topics to coauthor a paper. Similarly, the profile similarity experiments (Table 3.5) on the Facebook dataset show that the Occurrence Frequency that emphasizes similarity of strangers to users' already existing friends has the best performance; new friends should either be similar to the user, or they should be similar to user's friends. Here the underlying notion is that friends are a good indication of user characteristics on undirected networks.

This argument does not hold on directed graphs, because edges are not formed by mutual consent. In the Epinions dataset, we found that similarity is very low compared to undirected graphs (see Figure 3.16). Furthermore, profile similarity patterns are very dependent on the sign of the edge; distrust ratings are given to more similar users, whereas trust ratings are given to less similar users.

The correlation of profile and network similarities is also different on different types of graphs. In undirected graphs we see that profile similarity is a good indicator of network similarity, i.e., users with similar profile characteristics tend to be closer on the social graph and they share mutual friends with the same characteristics. This fact helped the performance of our network similarity measure, because the underlying notion in our network similarity measure is that users create edges with strangers who are connected to a denser community in their social graphs.

On directed graphs, the reason behind edge formation has a big impact on the correlation between network and profile similarity. In the Epinions dataset, there are two motivations; users rate strangers with trust to see more of what they do on the website, whereas the distrust ratings are given to indicate that a stranger's content and activity are of no interest to the user. In Figure 3.13 we showed that our network similarity measure performs better for the distrust graph. However, profile similarity for these new edges is shown to be very low in Figure 3.16. Clearly, here network similarity provides more information than profile similarity for the distrust graph.

However, in real life situations, the motivation for edge formation might be difficult to know a priori. It is important that new techniques are devised to mine the graph type and edge motivation from the dataset. This knowledge is vital to understand which characteristics should be observed in similarity studies.

3.8 Conclusion

In this chapter, we presented a method for evaluating social networks according to network connections and profile attributes. We comparatively evaluated our measures on real life datasets. Our work highlights how network types can influence interactions among social network users, and what type of social network information can be used for different network types.

This study can be extended along two main dimensions. First, studies can focus on the

impact of friends on user interactions. As our experimental results show, friends of friends constitute a big percentage of new friends, so we can assume that they increase interactions of a target user. Friends can be modeled in such a way that their contribution to a target user's social interactions can be aggregated, and the impact of friends can be deduced.

Moreover, how network similarity implies profile similarity (or vice versa) can be further studied. Specifically, by using network similarity, changes in profiles of users can be learned even when users do not update their profile information. In Chapter 6 we study this aspect.

Chapter 4

Explaining Privacy with Interactions

4.1 Introduction

Having an account in an Online Social Network (OSN) opens a path to opportunities but it also brings about certain risks; social network users can be bullied, their pictures can be stolen or their status posts can reach unwanted audiences. Even when profiles do not list any information, social graphs can be analyzed to infer personal information. Despite all these, social networks have hundreds of millions of users, because users think that positives outweigh negatives and they maintain their online presence. However, this user attitude is not totally care-free; individual cases of privacy breaches and their consequences have been widely discussed in social media and privacy risks have only grown with time as social networks have grown in size exponentially [24].

Several research efforts have been carried out to alleviate these problems, with the results of some tools helping users to be more privacy-aware. Notable examples are relationship-based access control mechanisms [30], tools on support of privacy preference specification [45, 83], as well as more expressive privacy settings recently adopted by commercial OSNs, like Facebook. However, despite the relevance of these proposals, we believe there is still the lack of a conceptual model on top of which privacy tools have to be designed. Central to this model, that should be the basis for any conscious user decision about profile information disclosure, should be the concept of *risk* one might be exposed when interacting with other OSN users.

The introduction of this concept, new in OSNs, is mainly motivated by the consideration that a large amount of friendships in social networks are just virtual. Indeed, people connected in an OSN very often have never met before establishing a relationship (or a long period has passed before they keep in touch again in an OSN). However, several events reported in the media testify that users establish virtual relationships with never-met people without considering the consequences these new relationships might have for their privacy and, some time, personal safety. Indeed, a new relationship in an OSN always

implies the release of some personal information. In general, this release is controlled by user privacy settings. Unfortunately, empirical studies show that social network users are not used to specifying privacy settings, and very often they do not change the default privacy settings that are very permissive [62, 23]. As a consequence, the creation of new relationships without specifying the appropriate privacy settings might expose a user to the risk of unconscious release of personal data. This is due to the fact that in almost all OSNs users can reference resources of other users in their social graph; and it is generally not possible or very difficult for a user to control the resources published by another user. This uncontrolled information flow highlights the fact that creating a new relationship might expose one to some privacy risk. In addition, as said before, new relationships with never-met users might have most serious consequences, since social networks have drawn the attention of sexual predators and child molesters, just to mention the worst.

Despite the possible serious consequences of the establishment of new relationships, current situation is that social network users are poorly informed about their friends, and with whom their friends keep company. Hugely popular social networks still assume that a friend is someone with whom a user can share everything, and furthermore, that the user can even share most of his/her profile information with friends of friends. Recommended privacy settings are in accordance with this friend notion. For instance, on Facebook, friends of friends are by default allowed to see most parts of a user profile.

To cope with these issues, we investigate a risk measure to help users in judging a stranger (i.e., a never-met user) with whom they might establish a relationship so as to be informed of how much it might potentially be risky, in terms of disclosure of private information, to have interactions with him/her. In defining the proposed measure we made several design choices, inspired by real cases. The first is that risk evaluation is impacted by several different dimensions, whose importance may greatly vary from user to user. Indeed, different from other scenarios where user risk evaluation can be made by a unique trusted entity (e.g., a bank, an insurance company) by processing profile information and users' past activities, we believe that in OSNs user risk attitude should also be taken into consideration. As such, we adopt a solution closer to the social computing paradigm, where rather than a trust entity judging social network users, we prefer the judgments given by community members. This paradigm has resulted to be a winner choice in other communities, where knowledge deriving from the community is used to take decisions, such as recommendation systems, collaborative filtering, and so on. To this end, we try to assess the risk of a stranger through the eyes of the user, hence the risk score is user-dependent and subjective. Thus, our risk measure captures the personal judgment that a user, hereafter called owner, expresses on a stranger Y with respect to his/her risk. However, to help owners in forming their own opinion about risk, we provide them with objective information about strangers. More precisely, inspired by homophily and heterophily theories, we provide information on *similarity* (see Chapter 3) between owner and the stranger as well as the *benefits* owner might have in terms of new information from strangers profile he/she might be authorized to access. We believe that, by using this information, users are able to evaluate the trade-off between similarities and benefits so as to estimate the risk of a stranger.

Other design choices for our risk measure are motivated by social network statistics,

which show that new relationships are mainly established among contacts of direct friends (i.e., second level friends). As such, we focus on second hop connections which act as strangers in the rest of this chapter. Statistics also show that this set of users can be very large (as an example, according to Facebook statistics each user has 16,900 strangers on the average). This necessitates a prediction technique, since asking an owner to insert risk judgments for each stranger is unfeasible. To this purpose, we adopt an active learning process for risk estimation, where owner risk attitude is learned from few required interactions. According to the proposed learning algorithm, once the classifier is built with the training data (i.e., the owner risk labels), the system can predict risk labels of all those strangers that were not included in the training data without any user intervention.

The risk estimation process discussed here has been developed into a Facebook application and tested on real Facebook data. Our experiments show that with limited user involvement, we can get accurate risk estimation. In particular, as experiments in Section 4.4 show, we are able to correctly predict risk labels with 83.38% of accuracy. Furthermore, in Section 4.4 we discuss how user decisions in assigning risk levels are affected by the characteristics of other social network users.

To the best of our knowledge, we are the first to provide a framework for computing OSN users' privacy risks which takes into account a rich set of dimensions, ranging from structural properties of the social graph and profile similarity, to benefits arising from user interactions and subjective risk perception.

The remainder of this chapter is organized as follows. We discuss our risk measure in Section 4.2, whereas Section 4.3 details the three phases of the risk learning process. In Section 4.4 we discuss the performance of risk computation.

4.2 How to measure risk

In designing the proposed risk measure we take into account several design principles that have been inspired from social network theory. In the following, we discuss these principles and our assumptions, whereas in Section 4.3 we present our approach to estimate risk levels according to them.

Strangers. As mentioned in the introduction, social network users tend to interact and create new relationships with those that are close to them in the social network graph [92, 68]. More precisely, given a social network user, hereafter *owner*, we compute risk levels for those users that are connected to a friend of owner's friends. In the following, we will refer to these second-level contacts as *strangers*.

Subjective risk perception. Risk attitude has been found to be very subjective [2, 47]. As such, we do not believe that it is possible to automatically estimate the risk just on the basis of the characteristics of a user social graph and/or strangers' profiles. In contrast, we believe that we need to take into account also users' risk attitude. To cope with this issue, during the first phase of our risk estimation process we ask for owner feedback. More precisely, we ask users to provide risk judgments for few selected strangers from the social graph. Then, from the collected risk judgments, the process learns how to

estimate risk levels of the remaining strangers.

Risk judgment. Since we want to estimate how much it might be risky to interact with a stranger, the risk notion we want to model is related to social interactions. We believe that to receive an educated risk estimation for a given stranger, we should provide the owner with meaningful information that can help him/her to make a correct decision. To determine what kind of information should be provided to owners, we take into account two key tendencies that, according to social network theory, regulate social interactions.

Homophily. This is the tendency of people to interact with those that are similar to them [92]. In social networks, users who are similar along certain attributes, such as gender, education and hometown, tend to create new friendship links. Furthermore, as previously discussed, users create new friendship links with others who are closer to them on the social graph.

To take homophily into account, we provide owner with information on how much the stranger, for which the risk judgment is required, is similar to him/her. This information is provided by means of a 'similarity measure'. It is worth noting that literature offers several similarity measures [118], we adopt a similarity function that considers both network similarity (e.g., information on owner and strangers mutual friends), as well as profile similarity (e.g., how many common/similar profile attributes they have).

Benefits. In contrast to homophily, heterophily is the tendency to interact with people that are not similar to you [113]. In social networks, this tendency is mainly motivated by the fact that social interactions (e.g., creation of new relationships with a stranger) might give owner some benefits in terms of acquaintance of new information (e.g., strangers profile and social graph).

An illustrious example of heterophily can be observed on the social network Twitter. Twitter users do not need approval from others before they can 'follow' them and view their status posts. This allows users to follow people whenever they can get benefits, and, in many cases, followed users are distinctly different from the followers in their profile and social graph. We wish to keep also this tendency into account, and therefore we provide owner with information on which benefits he/she might get in starting social interactions with the stranger.

To have a measure of benefits, the owner assigns an importance coefficient θ_i for each benefit item i, whereas the benefit an owner o gets from a stranger s is defined as:

$$B(o,s) \;=\; \frac{1}{|M|} \times \sum_{i \in M} (\theta_i \times V_s(i,o))$$

where M is the set of benefit items on the profile of s (e.g., photos, friends, wall), θ_i is the importance of being able to see item i, and $V_s(i,o) = 1$, if item i of the profile of s is visible to the owner, $V_s(i,o) = 0$, otherwise.

4.3 Risk Learning Process

Our risk learning process relies on owner feedback to take into account the subjective nature of risk. Feedback is gathered as owner feeling/estimation about how much it could be risky to start social interactions with a given stranger. Since the owner might not know the stranger, risk estimation is based both on the similarity the stranger has with the owner and the benefits he/she might provide to the owner (cfr. the principles discussed in the previous section). The benefits are considered as a pay-off for showing personal data to strangers. This data exposure brings a sense of privacy loss, and the benefits must weigh more than the perceived privacy loss. Collected risk judgments, hereafter *risk labels*, are then used to learn owner risk attitude and predict risk estimation for those strangers for which the owner does not provide a direct judgment.

The proposed risk learning process is based on supervised learning techniques [69], where, by analyzing training instances (i.e., few labeled instances) it is possible to generate a classifier to predict labels for unlabeled instances. In designing the supervised learning process to be adopted for risk estimation we have considered two main issues. The first is the potentially big number of strangers an owner might have. As an example, in our experiments the average count of strangers per owner is 3,661.

Another important factor is the dynamic nature of the owner's social graph, in that stranger connections might change very fast. These changes are due to acquisition of new strangers (i.e., some friends create new contacts) as well as new connections between strangers themselves, which might impact their similarity measures with the owner. This means that in our scenario it is not efficient to adopt a pre-defined and fixed training set. Rather, it is preferable to select the training set on the fly so that changes in the social graph are immediately reflected, and the number of required data to be labeled (i.e., strangers) can be reduced as the accuracy of predictions improves. Due to these requirements, from the many existing supervised learning models, we choose the active learning paradigm [117], where the learning phase is computed in several rounds. During each round an annotator (i.e., the owner) is interactively queried for labels (i.e., risk judgments) of selected unlabeled data samples (i.e., unlabeled strangers).

Using active learning in our context implies that initially all strangers belong to the set of unlabeled strangers. In each round, the risk learning process asks the owner to give risk labels to a selection of the most informative strangers[1] among those not labeled, which are then moved into the labeled strangers set. Then, collected labels are used by a classifier to predict labels for unlabeled strangers. Rounds of labeling and prediction continue until a good level of prediction is met, where the required level is given as input by the owner to take into account different risk attitude. The learning process is summarized in Figure 4.1.

To adopt active learning in our scenario, we need to cope with several issues. (1) In order to really get the owner risk attitude, the learning process has to pose an appropriate question to the owner, so as to get a meaningful feedback with respect to his/her risk

[1]Most informative strangers are selected according to a clustering-based approach explained in Section 4.3.2.

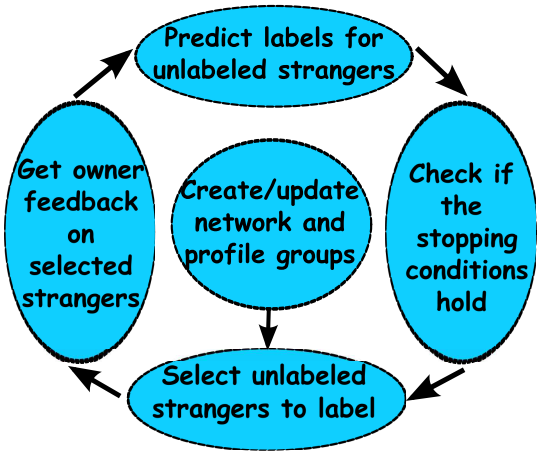


Figure 4.1: Risk learning process

perception. Possible risk label values have to be carefully designed too. (2) Particular attention has to be paid to the strategy for the most informative strangers selection, which should be tailored to the social network scenario. (3) A good classifier should be chosen to predict risk labels for unlabeled strangers. (4) Finally, we need to design stopping criteria, which ensure the required accuracy.

All the above issues are discussed in the rest of the section.

4.3.1 Risk Labels

In order to explain owners that our risk estimation is based on similarity and benefits, we query the owner for the risk level of a stranger with the following question:

"You and *stranger_name* are $x/100$ similar and he/she provides you $y/100$ benefits in terms of information you are allowed to see now on his/her Facebook profile. Do you think it might be risky to establish a relationship with *stranger_name*? Please respond by considering how much you are similar to *stranger_name* and that, after you become friends of him/her, benefits might increase as you might be allowed to see more resources in addition to his/her profile, e.g., his/her posts, photos, if privacy settings allow you."

Rather than allowing owners to select any value in [0,1], we give them only three options for risk labels, namely *very risky*=3 , *risky*=2, and *not risky*=1. We believe that these three values are more easily understandable by average OSN users.

4.3.2 Stranger sampling

The literature offers several methods for sampling selection (see e.g., [117] for a survey). Among these, we are interested in pool-based criteria according to which data to be labeled in each round are greedily selected from the pool of unlabeled strangers.

Our idea is to cluster strangers based on their network connections and profiles, so as to

generate a set of pools. Thus, the set of strangers is divided into a set of disjoint pools $\mathcal{P}_{st}$ from which a number of strangers are randomly selected at each round i to be labeled by the owner. Therefore, at each round i each pool $P \in \mathcal{P}_{st}$ has associated the corresponding set of labeled strangers L^i_{st} as well as the gathered labels L^i, whereas U^i_{st} denotes the set of unlabeled strangers. Then, based on labels in L^i the classifier predicts labels for unlabeled strangers in U^i_{st}. The process is iterated until the required accuracy in labeling prediction is met in the corresponding pool P.

Pools are defined by clustering strangers according to two dimensions. The first takes into account how much a stranger is connected to the owner in the network. Therefore, we create a first-level grouping based on strangers' network similarity with the owner as follows:

Definition 7 (Network similarity groups) *Let S_o be the set of strangers for a given owner o. Let NS be a function returning the network similarity between o and $s \in S_o$, $NS(o,s) \in [0,1]$. The Network Similarity Groups for o (NSG) consists of α disjoint sets of strangers $nsg_1, \ldots, nsg_\alpha$, such that for each $x \in \{1, \ldots, \alpha\}$, $nsg_x = \{s_i \in S_o \mid \frac{x-1}{\alpha} \leq NS(o,s_i) < \frac{x}{\alpha}$ and $0 < x \leq \alpha\}$.*[2]

To estimate the network similarity between owner o and a stranger s, we use our measure NS() defined in 3. Unlike existing similarity measures [118] which only consider mutual friends of the owner and a stranger, the measure works by also considering the connections among mutual friends. If the stranger is connected to a dense community around the owner, the measure returns a higher similarity value.

Additional knowledge that can help in defining stranger pools is the one contained in OSN user profiles. Indeed, these can be used to further refine network similarity groups, so as to cluster similar strangers together, which we expect to improve label prediction accuracy. Therefore, the first-level groups of strangers are based on network similarity with the owner, whereas the second-level groups of strangers within the same network similarity group are determined according to their mutual profile similarity.

In selecting the algorithm for profile-based clustering [49] that better fits our scenario, we have to take into account that (1) social network profiles mainly contain categorical data (such as hometown, gender and education); (2) it might be required to perform profile-based clustering several times, since we need to generate separate profile clusters for each network similarity group. Due to these requirements, we selected the Squeezer algorithm [56]. This is a clustering algorithm for categorical data that makes only one pass on the data set, and this helps us to deal with computational complexity issues when there are thousands of strangers in a network similarity group. A further benefit of Squeezer is that it allows us to customize clustering by associating different weights with each profile item (e.g., age, hometown, educations). As it will be discussed in the section reporting experiments, these weights help us in catching the relevance of some profile items over the others while grouping strangers. In order to apply Squeezer to our scenario, we have adapted it as follows. Given a network similarity group nsg, the algorithm starts with the

[2]We defer the discussion on α value to the experimental results section.

first stranger as a profile similarity cluster *psg* and iterates over each stranger $s \in nsg$. An overall similarity of stranger s with profile similarity cluster *psg* is computed by weighted averaging all profile attribute similarities. More precisely, to find the similarity of a stranger to a *psg*, the algorithm checks how many strangers from *psg* have the same value as s for each profile attribute. Stranger s is put into the most similar cluster. If no similar cluster can be found, that is, the similarity is below a given threshold, the stranger s itself creates a new cluster. We have adapted the similarity measure defined in [56] for relational tuples, to profile attributes as follows.

Definition 8 (Profile Similarity) *Let S_o be the set of strangers for a given owner o. Let nsg be a network similarity group generated on S_o. Let c be a subset of strangers in nsg (i.e., a cluster) and s be a stranger such that $s \notin c$. Let PA be the set of profile attributes, we denote with $s.pa_i$ the value of the i-th attribute in s profile, where $i \in |PA|$. The profile similarity of c and s is given by:*

$$Sim(s,c) = \sum_{i \in |PA|} w_i \left(\frac{Sup(s.pa_i)}{\sum_{x \in VAL_{pa_i}(c)} Sup(x)} \right)$$

where w_i is the weight associated with the i-th profile attribute, $VAL_{pa_i}(c)$ is the set of values for attribute pa_i in the profiles of strangers in c, and $Sup(x) = |\{s_i \in c | s_i.pa_i = x\}|$ returns the support of attribute x.

By using the above profile similarity definition in Squeezer, we define the second-level grouping obtaining thus the set of network and profile based stranger pools $\mathcal{P}_{st}$, defined as follows.

Definition 9 (Network and profile based pools) *Let S_o be the set of strangers for a given owner o. Let α and β be two parameters. Let NSG be the set of α network similarity groups generated according to Definition 7. For each $nsg \in NSG$, let $Squeezer(nsg, \beta)$ be the set of profile-based clusters formed on nsg by algorithm Squeezer using the similarity measure in Definition 8 and β as similarity threshold to create a new cluster. The set of disjoined pools $\mathcal{P}_{st}$ on S_o is given by $\bigcup_{nsg \in NSG} Squeezer(nsg, \beta)$.*

Figure 4.2 graphically depicts the resulting network and profile based pools.

Up to this point, we explained how strangers can be grouped so that sampling can be done. Next we will detail the prediction process which is performed over each distinct pool, and the related termination condition. Therefore, in what follows we explain the process for a given network and profile based pool $P \in \mathcal{P}_{st}$.

4.3.3 Classifier

In searching a classifier for our scenario, we focused on those tailored for network data. In this field, literature offers several classifiers, among these we selected the graph based

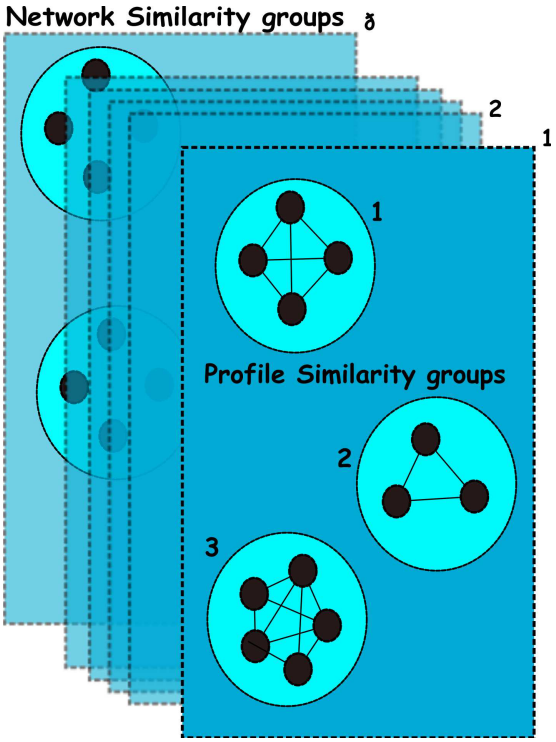


Figure 4.2: Network and profile based pools

approach by Zhu et al. [137]. This is a graph based classifier that works well with few labeled samples, and this complies with our need to reduce labeling efforts.

According to Zhu's approach, both labeled and unlabeled strangers are represented as nodes in a graph, where each pair of nodes is connected by a weighted edge. Weights represent the similarity according to the Euclidean similarity metric. The authors note that different similarity metrics can be used in different settings. In our scenario data are categorical, so we use edge weights based on strangers' profile similarity. To compute this similarity, we adopt the profile similarity function PS() defined in 3 which takes as input two profiles. For each attribute, if values are identical on both profiles the attribute similarity is set to 1. If they are non-identical, a non-zero value is computed by considering the frequency of the item values in the data set (i.e., the profiles in the considered pool). We refer the reader to 3 for more details.

After weights have been computed, the classifier predicts similar labels for similar neighbors on the graph, by exploiting the random walk strategy presented in [137].

In our privacy context, erroneous predictions can have two types of consequences; a stranger's label can be predicted lower or higher than what the owner would give. Higher label prediction poses no immediate threat to privacy; it only calls for more vigilance. On the other hand, lower prediction can have the system assume that the owner is safe when there is a real privacy threat.

4.3.4 Termination

In general, the risk learning process aims at soliciting owner labels till a good accuracy is reached. Accuracy can be estimated by comparing the predicted labels against the labels given by the owner. If the accuracy is low, we can ask owner to continue labeling. In the case of a good accuracy value, we can stop and spare owner effort in labeling. Since accuracy validation requires owner effort (i.e., owner labeling), we need to stop the process even if not all predicted labels have been validated against owner labels.

Several measures have been proposed to understand when to stop labeling when all predictions cannot be validated [136]. Among these, we chose the classification change principle. Classification change looks into predicted labels of strangers in consecutive rounds. If predicted labels do not change in these rounds, we conclude that further owner labeling will not change them as well and stop owner labeling. In what follows, we give more details on the adopted accuracy and stabilization measures.

Accuracy. The basic idea to estimate accuracy is to compute the root mean square error between labels predicted by the classifier and labels given by the owner for the same set of strangers. This error can be iteratively computed. More precisely, in each round r_i, the learning process asks the owner to give risk labels for some strangers s for which labels have been already predicted in previous round r_{i-1}. The error is computed as the root mean square between the predicted and owner-defined labels.

Definition 10 (Root Mean Square Error) *Let $P \in \mathcal{P}_{st}$ be a network and profile based pool, let U^i_{st} and L^i_{st} be the subsets of P denoting, respectively, the set of unlabeled strangers*

and the set of strangers for which the owner has given a risk label at round r_i. Moreover, let L^i be the set of collected owner labels for pool P till round r_i, and $\hat{L}^i_{st}$ be the set of predicted labels for strangers in U^i_{st}. Let S be a set of strangers selected from those that have been labeled at round r_{i+1} (i.e., from $L^{i+1}_{st} \setminus L^i_{st}$), where, for each $s \in S$, $\hat{L}(s)^i$ and $L(s)^{i+1}$ denote the label for s predicted at round r_i and the label given by owner at round r_{i+1}, respectively. Root mean square error of predictions for the set S is then:

$$RMSE_S = \sqrt{\frac{\sum_{s \in S}(L(s)^{i+1} - \hat{L}(s)^i)^2}{|S|}}$$

Since risk labels have values in the range [1,3] (i.e., very risky=3, risky=2, not risky=1), the root mean square error can have a value in [0,2], where 0 implies that all the predictions were correct.

Stabilization. For employing classification change in our unvalidated predictions, we solicit owner to give a confidence value $c \in [0,100]$, which is then used as a tolerance value for the classification change, as the following definition clarifies.

Definition 11 (Classification Change) *Let $P \in \mathcal{P}_{st}$ be a network and profile based pool, and let $\hat{L}(s)^i$ and $\hat{L}(s)^{i+1}$ be labels for stranger $s \in P$ predicted at round r_i and r_{i+1}, respectively. Let c be the confidence value selected by owner o. We say that P is stabilized according to confidence c if, for all $s \in P$, the following condition does not hold:*

$$\hat{L}(s)^{i+1} - \hat{L}(s)^i \geq \frac{(L_{max} - L_{min}) \times (100 - c)}{100}$$

where L_{max} and L_{min} are the lower and upper bound of the labels range.

Note that, if the owner wants to manually label all strangers, the confidence value can be set as $c = 100$.

Stopping Condition. On the one hand, accuracy helps in validating label predictions, but it requires owner effort. On the other hand, stabilization in predicted labels does not guarantee accuracy in labeling; it merely informs us that further labeling by the owner will not change predicted labels. To keep into account both these aspects, we adopt a stopping condition that combines accuracy and stabilization. That is, we stop the risk learning process when risk labels are predicted with a good accuracy (i.e., RMSE between owner given and predicted labels has to be less than 0.5) and for at least n rounds there should be no classification changes with a confidence c selected by the owner, where n is a system-defined parameter.

4.4 Experiments on Facebook data

In this section, we discuss the effectiveness of the proposed risk measure and related learning process. At this aim, we conducted several experiments to validate that our proposed risk

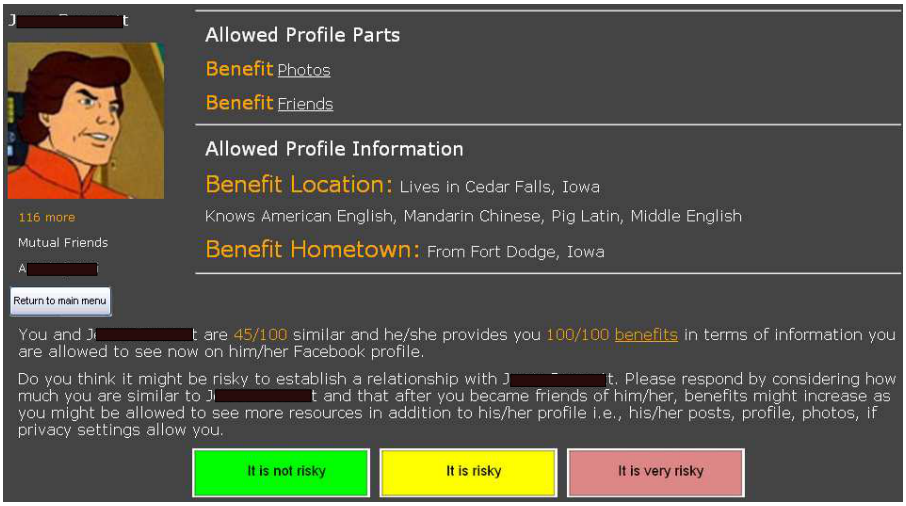


Figure 4.3: Snapshot from the extension

measure indeed captures the risk perception of social network users and that the learning process correctly predicts strangers' risk labels. Before giving the details of experiments, we introduce the dataset that has been used.

4.4.1 Facebook Dataset

To perform the experiments on Facebook real data, we developed a Facebook application (called Sight[3]). The aim of this application is to: 1) gather strangers' information, so as to generate the network and profile based pools, 2) ask owner (i.e., the social network user that launched the application) risk labels only for selected strangers, on the basis of similarities and benefits. To perform this last task we developed a Google Chrome extension. Due to Facebook API limitations we cannot retrieve the complete social graph at once. Rather, we listen owner profile to see friends' interactions (e.g., tagging, posting) and, once a friend of friend is found, we query Facebook for its mutual friends/profile information. According to our experiments, the time period to learn a big portion of the social graph (4,000 strangers) can take up to 1 week. However, the user can start label and learn about the risk since the first day. In our experiments, in 2 months we were able to discover around 30,000 strangers.

Through Sight, owner can specify the benefit coefficients for each single item (e.g., profile attributes, wall, photos), as well as the confidence he/she wishes to reach. We list received benefits on the extension interface, along with the similarity value of the selected strangers, and ask the owner our query from Section 4.3.1. A snapshot from the extension interface is shown in Figure 4.3.

During the test, Sight application has been used by 47 Facebook users, where 32 were

[3]http://www.facebook.com/developers/apps.php?app_id= 103656703030138

males and 15 were females, all aged in the range [18-35]. 17 of the users were from Turkey, 5 from Italy, 9 from USA, 1 from India, and 7 from Poland.[4] We collected thus a total of 172,091 stranger profiles, and 4,013 owner-defined risk labels. On the average, owners have 3,661 strangers and gave 86 risk labels.

4.4.2 Parameters setting

The first couple of parameters are used to customize the creation of network and profile based pools. These are α and β. We recall that the first determines the number of network similarity groups computed using network similarity function NS(). As this function returns value in [0,1], for this first set of experiments, for simplicity, we have set α=10, that should represent 10% of strangers for each network similarity group under the assumption that strangers follow a uniform distribution w.r.t. their network similarity with owner.

However, Figure 4.4 reports that stranger distribution in network similarity groups is not uniform. More precisely, in the x-axis network similarity groups are ordered based on increasing network similarity values. Thus, for example strangers in the 3rd group have higher network similarity with owner than strangers in the 2nd network similarity group. Figure 4.4 highlights that most of the strangers are weakly connected with owners. Moreover, we found that no stranger has network similarity values greater than 0.6, so in the following we do not report experiments for these network similarity groups.

In contrast, β represents the threshold for the creation of a new profile based cluster, so it constraints the number of profile based clusters generated by Squeezer over each network similarity group. Thus, increasing β could result in too many profile based clusters each of which with few strangers, which implies too many distinct learning processes to be executed. To avoid this situation, we select β=0.4.

A further parameter is related to the number of strangers that are required to be labeled by the owner in each round, which in these experiments have been set to 3, to keep minimum the owner effort. Finally, a cluster is considered stabilized when classification change does not happen in 2 rounds of labeling (i.e., parameter n in Section 4.3.4).

4.4.3 Risk learning process

In order to show the effectiveness of the proposed risk learning process we run several experiments.

Risk Label Prediction. As explained in Section 4.3.4, the risk learning process stops when both the accuracy and stabilization conditions hold. We recall that if both these hold it means that risk labels are predicted with a good accuracy (i.e., RMSE between owner and predicted labels is less than 0.5) and for at least 2 rounds there is no classification changes with a confidence c selected by the owner.

At the average, the confidence selected by our 47 users is about 80% (i.e., 78,39). With these confidence values, experiments highlight that labels prediction stabilize in about 3 rounds (i.e., 3,29 is the average).

[4]The statistics have been computed on those available user profiles.

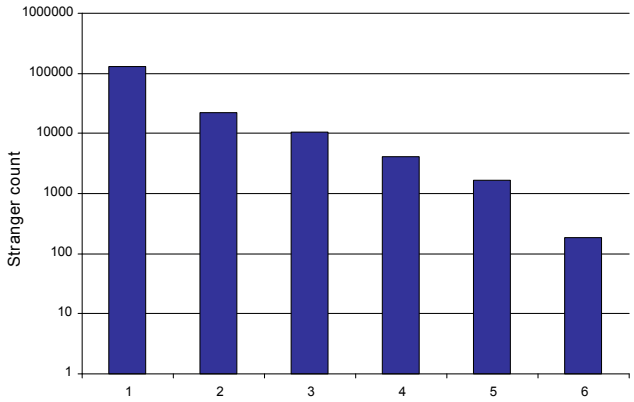


Figure 4.4: Stranger count for network similarity groups

Whilst these results confirm that we have good accuracy (an RMSE error less than 0.5), we are also interested in those predicted labels that exactly match what the owner would have given (i.e., $\hat{L}(s)^{i+1} = L(s)^i$). To have this estimation, during the accuracy evaluation, we count the number of predicted labels that match the owner labels. The promising result is that 83,36% of predicted labels exactly match the owner labels.

Stranger sampling. To validate the strategy adopted for pool generation, we perform experiments to compare network and profile based pools (NPP) against pools generated by only considering network similarity (NSP).

The impact of profile similarity grouping can be seen in Figures 4.5 and 4.6, showing better results for NPP in terms of root mean square error and stabilization per round.

4.4.4 Risk Measure

As discussed in Section 4.2, risk labels are based on similarity and benefits. In the following experimental setting, we will evaluate how these two factors affect the owner's risk judgment.

Network and Profile Similarity. We first consider network similarity. In creating network similarity groups, we wanted to capture how owner judgment is affected by network connections. Our data shows that some strangers can have more than 40 mutual friends with an owner. Hence, with increasing network similarity we assume that the possibility of an acquaintance between owner and stranger increases. We expect that this increasing possibility is reflected in lower assigned risky labels. Indeed, Figure 4.7 shows that with increasing network similarity, the percentage of very risky labels in network similarity groups consistently decreases.

Along with network similarity, we evaluated the importance of stranger profile similarity in owner labeling. To this purpose, we look for patterns in profile attributes of strangers that are assigned by owner to the same risk labels, so to determine how much each stranger profile attribute affects owner decision. As an example, if risk labels given by owner show

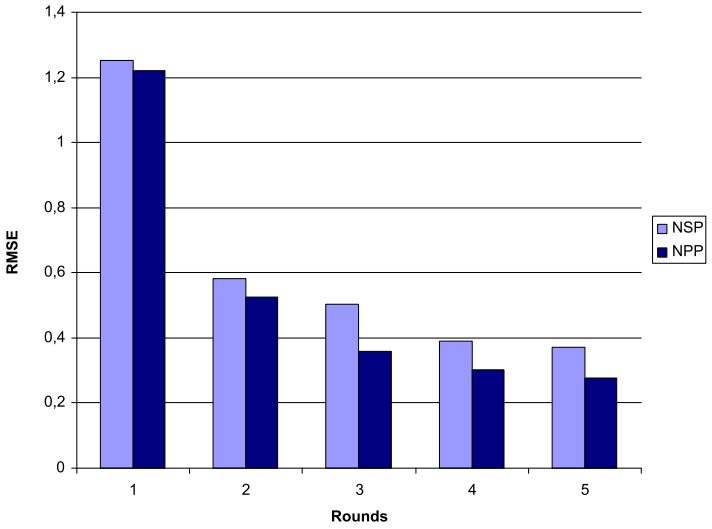


Figure 4.5: Error rate by rounds for NPP and NSP pools.

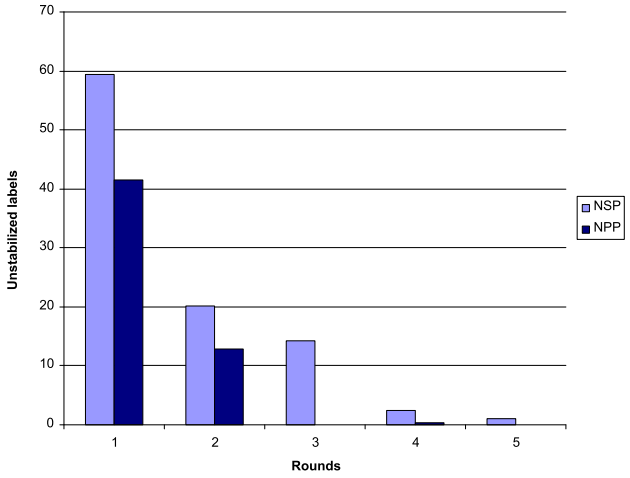


Figure 4.6: Average number of unstabilized labels for NPP and NSP pools.

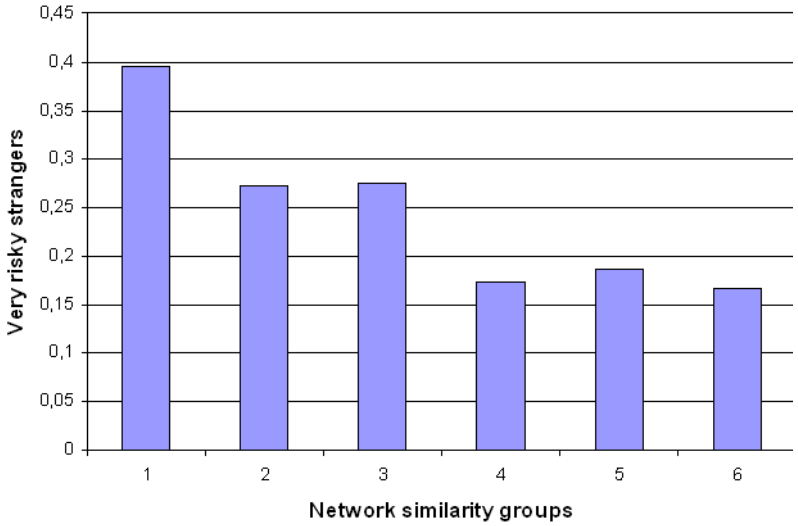


Figure 4.7: Percentage of very risky strangers in network similarity groups

a pattern with gender (e.g., all males are labeled as very risky), we can infer that gender is an important attribute in owner decision. In information theory, information gain ratio [86] is used to capture the importance of a variable in decision trees. More precisely, a high information gain ratio for a profile attribute implies a reduction in entropy[5] caused by attribute value. A higher gain ratio item carries more information in understanding the rationale behind owner labeling (e.g., all males are risky).

In our setting, we use three profile attributes for clustering with Squeezer algorithm. These are gender, last_name and locale, where this latter denotes Facebook web interface language of the owner. For example, English speaking owners from USA have EN_US locale, while owners from Great Britain have EN_GB locale.

Definition 12 (Attribute importance) *Given a profile attribute $pa_i \in PA$, we estimate its importance $\mathcal{I}_{pa_i}$ by normalizing the information gain ratio as follows:*

$$\mathcal{I}_{pa_i} = \frac{IGR(pa_i)}{\sum_{pa_j \in PA} IGR(pa_j)}$$

where $IGR(pa_i)$ denotes information gain ratio of attribute pa_i.

In Table 4.1, we show profile attribute importance for our owners in an ordered list, where $\mathcal{I}_1$ refers to the most important profile attribute (highest information gain ratio), and $\mathcal{I}_3$ refers to the least important attribute. For example, in the table, values in $\mathcal{I}_i$-th column and gender row correspond to the number of owners for which gender is the i-th most important item. As highlighted in the table, gender has the biggest average weight

[5]Here we refer to the entropy of label distribution in three risk classes (i.e., very risky, risky, not risky).

Table 4.1: Profile attributes importance

	$\mathcal{I}_1$	$\mathcal{I}_2$	$\mathcal{I}_3$	Avg Imp.
gender	34	12	1	0.6231
locale	13	33	1	0.3226
last_name	0	2	45	0.0542

Table 4.2: Mined importance of benefits

	$\mathcal{I}_1$	$\mathcal{I}_2$	$\mathcal{I}_3$	$\mathcal{I}_4$	$\mathcal{I}_5$	$\mathcal{I}_6$	$\mathcal{I}_7$	Avg Imp.
photo	21	8	6	4	3	0	5	0.27
education	11	9	4	3	10	4	6	0.143
work	8	7	9	7	5	7	4	0.140
friends	2	10	7	6	6	8	8	0.13
hometown	0	7	9	11	6	9	5	0.11
location	1	4	8	9	11	8	6	0.092
wall	4	2	4	7	6	11	13	0.091

for owners, and for 34 owners it is the most important item ($\mathcal{I}_1$). Gender is followed by locale, and last_name. Although last_name has low average weight, it is more important than locale for two owners.

Benefits. The discussion on similarity aimed at finding similarity patterns in owners' risk judgments. Now we will look at what benefit patterns can be found in owners' risk judgment. More precisely, we adopt for benefit items the same importance definition from information gain ratio (Definition 12).

Whereas in similarity we have categorical item values such as gender:male, in benefits we work with visibility values such as photos:1. Here item values 0 and 1 correspond to visibility:false and visibility:true, respectively on stranger profiles.

In line with importance of items in similarity, we found importance of benefits in owners' risk judgment. In Table 4.2 we show the average importance as well as the order of items' importance. As reported, photos are considered the most important benefit item for 21 users, followed by education and work. Home wall has the least average importance, but for 4 owners it is the most important benefit.

According to our benefit measure, we asked owners to give a θ weight to each benefit item. In Table 4.3 those θ weights are shown. When comparing these weights to the ones we found in Table 4.2, we see that home wall benefit is given a low importance in both tables and some other weights are correlated. The relevant issue this table highlights is that for some benefit items it is better to use system suggested weights.

Another aspect of benefits is related to stranger profile attributes, as some patterns can be found with respect to the benefits they provide to owners. On social networks, gender of a stranger has been known to affect the visibility of profile items. In [47], Fogel et al. show that females have stricter privacy settings than males. As a result, the benefits they provide as strangers are lower than those of male users. To understand how provided benefits change with gender, we computed visibility of profile items based on gender. In

Table 4.3: Owner given θ weights

	Average θ weight
hometown	0.155
friend	0.149
photo	0.147
location	0.143
education	0.1393
wall	0.1328
work	0.1321

Table 4.4: Item visibility for different genders

	wall	photo	friend	loc.	edu.	work	hometown
male	25 %	88 %	56 %	42 %	35 %	20 %	41%
female	16 %	87 %	47 %	32 %	28 %	12 %	30%

Table 4.4, female strangers are shown to have lower visibility values for profile items. However, in photos male and female strangers do not show a big difference and visibility is almost the same for both genders. In Fogel's study, social network users were asked this question: *Do you include a picture of yourself on your profile?*. This *picture of yourself* can be the profile picture, or a picture in photo albums. In our experiment, the photos are photo albums on Facebook, excluding a profile picture. Fogel reports that the answer to the asked question was yes for 90.4% of males, and 81.6% of females. In our experiment, we found the visibility of photos to 88% and 87% for males and females, respectively. However, in our experiments we could not check if the visible photos on Facebook were self-pictures. Visibility percentages can be lower when non personal photos are excluded.

Similar to male/female difference, benefits show a pattern with locale values. In Table 4.5 we show visibility of profile items for strangers from 7 locales (TR:Turkey, DE:Germany, US:USA, IT:Italy, GB:Britain,ES:Spain, PL:Poland). Work has the lowest visibility among items.

It is interesting to note that although home walls are considered very private and hidden, home was one of the lowest importance items in both Tables 4.3 and 4.2. It seems that social network users do not share their walls with strangers, but they are also not interested in other strangers' walls. Photos have very high visibility among all locales while friends list visibility ranges from 41% to 72%. In Table 4.5 we can also see some correlation between locales. Visibility of profile items do not change greatly between IT and ES locales and the difference between percentages is around 5%.

4.5 Conclusions

We have proposed a measure for estimating the risk, in terms of disclosure of personal information, of interacting with OSN users. Risk levels are computed for 2 hop connections by taking into account both similarity and benefit metrics as well as user risk attitude. They

Table 4.5: Visibility of profile items for different locale strangers

	wall	photo	friend	loc.	edu.	work	hometown
TR	20%	84%	41%	36%	31%	15%	32%
DE	20%	77%	46%	34%	17%	17%	34%
US	17%	89%	52%	42%	34%	18%	37%
IT	27%	92%	68%	32%	38%	14%	41%
GB	12%	91%	46%	38%	25%	17%	32%
ES	22%	87%	63%	37%	28%	13%	37%
PL	31%	95%	72%	33%	23%	13%	31%

are computed through an active learning process, where owner risk attitude is learned from few required interactions. The initial experiments we have performed show the effectiveness of our proposal. We plan to extend this work along several directions. First, we plan to extend our tests to a greater data set and to data sets coming from different social networks. Moreover, we plan to develop techniques to mine from the data most of the values for the parameters on which our learning process relies. Finally, we envision a variety of applications for our risk labels that we would like to explore in the future, such as privacy settings/friendships suggestion or label-based access control.

Chapter 5

Risks of Interactions

5.1 Introduction

Users register on social networks to keep in touch with friends, as well as to meet with new people. Research works have shown that a big majority of people that we meet online and add as friends are not random social network users; these people are introduced into our social graph by friends. Although friends can enrich the social graph of users, they can also be a source of privacy risk, because a new relationship always implies the release of some personal information to the new friend as well as to friends of the new friend, which are *strangers* for the user. This problem is aggravated by the fact that users can reference resources of other users in their social graph; and make it very difficult to control the resources published by a user. This uncontrolled information flow highlights the fact that creating a new relationship might expose users to some privacy risks.

We cannot assume that friends will make the right choices about friendships, because friends may have a different view on people they want to be friends with. Considering this, privacy of a social network user should be protected by building a model that observes friendship choices of friends, and assigns a risk label to friends accordingly. Such a model requires knowing a user's perception on the risks of friends of friends. We made a first effort in this direction in Chapter 4 by proposing a risk model to learn risk labels of strangers by considering several dimensions. To validate the model, we developed a browser extension showing for each stranger (i) his/her profile features, (ii) his/her privacy settings, and (iii) mutual friends. Based on this information, the user is asked to give a risk label $l \in \{1, 2, 3\}$ to the stranger. These risk labels correspond to *not risky, risky* and *very risky* classification of a stranger. Through the extension, 47 users (32 male, 15 female users) have labeled 4013 strangers. However, *we did not consider risk of friends.*

This new work starts with considering two factors in assigned risk labels. First, strangers can be risky only because of their profile features. Second, a friend himself can increase or decrease the risk of a stranger. Increases and decreases will be termed as *negative and positive friend impacts*, respectively. In any case, if a risky stranger is introduced into the user's social graph it is because of his/her friendship with a friend.

However, determining the friend impacts can help us to determine which privacy actions should be taken to avoid data disclosure. We aim at learning how risk labels are assigned to strangers depending only on their profile features, and how much a friend can impact (i.e., increase or decrease) these labels. If strangers are risky just because of their profile features, privacy settings can be restricted to avoid only these strangers. On the other hand, if a friend increases the risk labels of strangers, all of his/her strangers should be avoided.

In Section 5.2 we explain the building blocks of our model and Section 5.3 shows how we use our dataset efficiently. In Section 5.4 we discuss the role profile features in risk labels, and in Section 5.5 we show how impacts of friends are modeled. Section 5.6 explains finding risk labels of friends from friend impacts, and in Section 5.7 we give the experimental results.

5.2 Overall Approach

We will start this section by explaining the terminology that will be used. In what follows, on a social graph $\mathcal{G}_u$, 1 hop distance nodes from u are called friends of u, and 2 hop distance nodes are called strangers of u, i.e., strangers of user u are friends of friends of u. We will denote all strangers of user u with S_u, and risk label of each stranger $s \in S_u$ that was labeled by u will be denoted as $l_{us} \in \{1, 2, 3\}$.

A social network $\mathcal{G} = (N, E, Profiles)$ is a collection of N nodes and $E \subseteq N \times N$ undirected edges. $Profiles$ is a set of profiles, one for each node $n \in \{1, ..., |N|\}$. A social graph $\mathcal{G}_u = (V, R, F)$ is constructed from the social network $\mathcal{G}$ for each user $u \in N$, such that, the node set $V = \{\forall n \in N | distance(n, u) \leq 2\}$. Nodes in $\mathcal{G}_u$ consist of friends and strangers of u. Similarly, edge set R consists of all edges in $\mathcal{G}$ among nodes in V. Each node $v \in V$ in a social graph will be associated with a feature vector $f_v \in F$. Cells of f_v correspond to profile feature values from the associated user profile in $Profiles$.

The goal of our model is to assign risk labels to friends according to the risk labels of their friends (i.e., strangers). As we stated before, risk labels of strangers depend on stranger features as well as mutual friends. We do not assume that all friends can change users' risk perception in the same way. Some friends can make strangers look less risky and facilitate interactions with them (i.e., friends decrease the risk of strangers). On the other hand, some friends can make strangers more risky (i.e., friends increase the risk of strangers). For example, if users do not want to interact with some friends, they might avoid friends of these friends as well. We will use positive and negative impacts to refer to decreases and increases in stranger risk labels, respectively. To understand whether friends have negative or positive impacts, our model must be able to know what risk label the stranger would receive from the user if there were no mutual friends. This corresponds to the case where the user given label depends only on stranger features. We will term this projected label as the baseline label, and show it with b_{us}. For instance, assume that if there are no mutual friends, a user u considers all male users as very risky, and avoids interacting with them. In this case, the baseline label for a male stranger s is very risky,

i.e., $b_{us} = very\ risky$. However, if the same male stranger s has a mutual friend with user u, we assume that the user given label l_{us} might not be equal to the baseline label b_{us} (i.e.,$l_{us} \neq b_{us}$), because the mutual friend might increase or decrease the risk perception of the user. This difference between the baseline and user given labels will be used to find out friend impacts.

Finding baseline labels and friend impacts requires different approaches. In baseline estimates, we use logistic regression on stranger features, and for the friend impacts we use multiple linear regression [98]. Both of these regression techniques require many user given labels to compute baseline labels and friend impacts with high confidence. However, users are reluctant to label many strangers, therefore we have to exploit few labels to achieve better results. To this end, we transform our risk dataset, and use the resulting dataset in regression analyses. In the next sections, this transformation and regression steps will be described in detail. Overall, we divide our work into four phases as follows:

1. **Transformation:** Exploit the risk label dataset in such a way that regression analyses for baseline labels and friend impacts can find results with high confidence. With this step, we increase the number of labels that can be used to estimate baseline labels and friend impacts.

2. **Baseline Estimation:** Find baseline labels of strangers by logistic regression analysis of their features.

3. **Learning Friend Impacts:** Create a multiple linear regression model to find friends that can change users' opinion about strangers and result in a different stranger label than the one found by baseline estimation.

4. **Assigning Risk Labels to Friends:** Analyze the sign of friend impacts, and assign higher risk labels to friends who have negative impacts.

5.3 Transforming Data

By transforming the data, we aim at using the available data efficiently to find friend impacts with higher confidence. To this end, we first transform profile features of friends and strangers to use *k-means* and hierarchical clustering algorithms [49] on the resulting profile data. This section will discuss the transformation, and briefly explain the clustering algorithms.

Our model has to work with few stranger labels, because users are reluctant to label many strangers. This limitation is also shared in Recommender Systems (RS) [93] where the goal is to predict ratings for items with minimum number of past ratings. In neighborhood based RS [67], ratings of other similar users are exploited to predict ratings for a specific user. Traditionally, the definition of similarity depend on the characteristics of data (e.g., ordinal or categorical data), and it has to be chosen carefully. We use profile data of friends and strangers in defining similar friends and strangers, respectively. Friend impacts of a user u is learned from impacts of similar friends from all other users. To this end, we

transform profile data of friends and strangers in such a way that friends and strangers of different users are clustered into global friend and stranger clusters. Next sections will describe the aims and methods of friend and stranger clustering in detail.

5.3.1 Clustering Friends

Clustering friends aim at learning friend impacts for a cluster of friends. This is because we might not have enough stranger labels to learn impacts of individual friends with high confidence. To overcome this data disadvantage, impact of a friend f can be used to find the impacts of other friends who belong to the same cluster. For example, a user from Milano can have a friend from Milano, whereas a user from Berlin can have a friend from Berlin. Although these two friends have different hometown values (Milano and Berlin), we can assume that both friends can be clustered together because their hometown feature values are similar to user values. This hometown example demonstrates a clustering based on a single friend profile feature and it results in only two clusters: friends who are from Milano/Berlin and friends who are from somewhere else. However, in real life social networks, friends have many values for a feature, some of which can be more similar to the user's value than others. For example, Italian friends of a user from Milano can be from Italian cities other than Milano, and these friends should not be considered as dissimilar as friends from Berlin. By considering these, we transform categorical friend values to numerical values in such a way that similarities between friend and user values become more accurate.

Our transformation uses the homophily [92] assumption which states that people create friendships with other people who are similar to them along profile features such as gender, education etc. In other words, we assume that all friends of a user u can be used to judge the similarity of a social network user to u. For example, considering the case where the user u is from Milano, a social network user from Rome is similar to the user if the user has many friends from Rome. Moreover, we assume that different users will have similar clusters of friends, e.g., friends from user's hometown, alma mater etc. and friend impact values will be correlated with their corresponding clusters, e.g., friends from hometowns will have similar impact values. More precisely, the transformation of friends' data maps a categorical feature value of a friend, such as hometown:Milano, to a numerical value which is equal to the frequency of the feature value among profiles of all friends of a user. For example, if a friend f has profile feature value hometown:Milano, and there are 15 out of 100 friends with similar hometown:Milano values, hometown feature of f will be represented with $15/100 = 0.15$. After applying this numerical transformation to all friends of all users, we compute a Social Frequency Matrix for Friends (SFMF) where each row represents numerical transformation of feature vector of a user's friend.

Definition 13 (Social Freq. Matrix for friends) *The Social Frequency Matrix associated with a social network $\mathcal{G}$ is defined as $|N| \times |F| \times n$, where N is the set of users in $\mathcal{G}$, $F \subset N$ is the set of user in $\mathcal{G}$ that are friends of at least one user $u \in N$, and n is the number of features of user profiles. Each element value of the matrix is given by:*

$$SFMF[u, f, v] = \frac{Sup(\vec{f_v})}{|F_u|}$$

where $F_u \subset F$ *is the set of friends of* u, $Sup(\vec{f_v}) = \left|\{g \in F_u | \vec{g}_v = \vec{f_v}\}\right|$ *and* $f \in F_u$, *whereas* $\vec{g}_v$ *and* $\vec{f_v}$ *show the value of profile feature* v *for users* g *and* f, *respectively.*

Having transformed friend data into numerical form, we can now use a clustering algorithm to create clusters of friends. After applying a clustering algorithm to the Social Frequency Matrix for friends, output friend clusters will be denoted by FC.

5.3.2 Clustering Strangers

By clustering friends, we can learn impacts of friends from different clusters, but this raises another question: do friends have impact on all strangers of users? Our assumption is that correlation between stranger and friend profile features can reduce or increase friend impact. For example, if a student user u labels friends of a classmate friend f, we might expect friends of f who are professors to have higher risk labels than student friends of f, because u might not want his/her professors to see his/her activities and photos. Here the work feature of strangers changes friend impact of f by increasing the risk label of professor friends of f. To see how friend and stranger features change friend impacts, we transform strangers' profile data to numerical data and cluster the resulting matrix just like we clustered friends. This clustered stranger representation helps us detect clusters of strangers for whom certain clusters of friends can change risk perception of users the most. Formally, we prepare a social frequency matrix as follows:

Definition 14 (Social Freq. Matrix for strangers) *The Social Frequency Matrix for Strangers associated with a social network* $\mathcal{G}$ *is defined as* $|N| \times |S| \times n$, *where* N *is the set of users in* $\mathcal{G}$, $S \subset N$ *is the set of user in* $\mathcal{G}$ *that are strangers of at least one user* $u \in N$, *and* n *is the number of features of user profiles. Each element value of the matrix is given by:*

$$SFMS[u, s, v] = \frac{Sup(\vec{s}_v)}{|F_u|}$$

where $Sup(\vec{s}_v) = |\{g \in F_u | \vec{g}_v = \vec{s}_v\}|$ *and* $S \in N$, *whereas* $\vec{s}_v$ *shows the value of feature* v *for stranger* s.

Note that we still use friend profiles in the denominator to transform stranger data. This is because we cannot see all strangers of a friend due to API limitations of popular social networks. To overcome this problem, we use friend profiles because we expect them to be similar to profiles of their own friends (strangers). We again use the Social Frequency Matrix for strangers to create clusters of strangers. We will denote stranger these stranger clusters by SC.

5.3.3 Clustering Algorithms

In our experiments, we used the *k-means* and hierarchical algorithms [49] to produce clusters of friends and strangers. This section will briefly explain these algorithms. In what follows, we will use data points and strangers/friends interchangeably to mean elements in a cluster.

The *k-means* clustering algorithm takes the number of final clusters as input and clusters the data by successively choosing cluster seeds and refining the distance within cluster data points. The required input for the number of final clusters is usually unknown beforehand and this makes *k-means* unfeasible in some scenarios. However, in our model it gives us the flexibility to experiment with different sizes of clusters. *k-means* is also a fast clustering algorithm which suits our model for the cases where all friends of all users can reach a few thousands. In our experiments, we used different k values to find optimal performance. In hierarchical clustering[1], a tree structure is formed by joining clusters and the tree is cut horizontally at some level to produce a number of clusters.

In friend and stranger clustering, choosing the number of final clusters or the horizontal level requires some trade-offs. The advantage of using many clusters is that data points in each cluster are more similar to each other (i.e., friends or strangers in a cluster are more similar in profile feature values). On the other hand, too many clusters decreases the average number of data points in a cluster, and our model may not be trained on these clusters with high confidence, i.e., there may not be enough data points in a cluster to prove anything. Using too few clusters also has a disadvantage. Final clusters may contain too many data points that are not very similar to each other. This decreases the quality of inferences because what we infer from some data points might not be valid for others in the same cluster. Despite this, if data points are naturally homogeneous, the similarity among data points in a big cluster can be high. As a result, a big cluster may offer more data to prove our inferences with more confidence.

After transforming our data and creating friend and stranger clusters, we will now explain baseline label estimation for stranger clusters.

5.4 Baseline Estimation

Baseline estimation analyzes how feature values on stranger profiles bring users to assign specific risk labels to strangers. The baseline estimation process results in baseline labels for each stranger $s \in S$. These labels are found by using statistical regression methods on already given user labels and stranger profile features. In this section we will discuss this process.

Baseline estimation corresponds to the case where a user would assign a risk label to a stranger without knowing which one of his/her friends are also friends with the stranger. Figure 5.1 shows an example of baseline estimation. In the figure, each stranger $s \in S_u$ is a

[1]We used the agglomerative form where a new stranger is added to clusters by considering the complete distance. Height of the tree was 3.

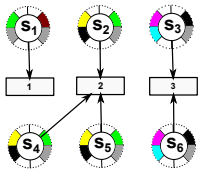


Figure 5.1: Features and risk labels

node surrounded by a ring representing his/her feature vector f_s. Each cell in the feature vector corresponds to a feature value of the stranger (e.g., hometown:Milano). Different colors for the same cell position represent different values for the same feature on different stranger profiles. In the example shown in Figure 5.1 strangers S_2, S_4 and S_5 are labeled with 2 (i.e., the risky label). These three strangers share the same feature vector as shown with the same colored cells. Based on these observations, if any stranger has the same feature vector with S_2, S_4 and S_5, the stranger will be given label 2. The evidence to support this statement comes from the three strangers (S_2, S_4 and S_5), and the number of such strangers determine the confidence of the system in assigning baseline labels.

Although in Figure 5.1 stranger features are shown to be the only parameter in defining stranger labels, in our dataset labels of strangers have been collected from users by explicitly showing at least one mutual friend in addition to the stranger feature values. Because of this, stranger labels that are learned from users can be different from baseline labels; they can be higher (more risky) or lower (less risky) depending on the friend impact. Considering this, in baseline estimation we use the labels of strangers who have the least number of mutual friends with users. These are the subset of labels which were given to strangers who have only one friend in common with users, i.e., for user u and stranger s, $|F_u \cap F_s| = 1$. In what follows, we will use *first group dataset* to refer to these strangers.

In our approach, we use logistic regression to learn the baseline labels from available data. This allows us to work with categorical response variables (i.e., one of the tree risk labels). Stranger features are used as explanatory (independent) variables and risk labels as the response (dependent) variable which is determined by values of explanatory variables (i.e., feature values). Although the response variable has categorical values, it can be considered ordinal because risk labels can be ordered as not risky (label 1), risky (label 2) and very risky (label 3).

Ordinary Logistic Regression is used to model cases with binary response values, such as 1 (a specific event happens) or 0 (that specific event does not happen), whereas multinomial logistic regression is used when there are more than two response values. As multinomial logistic regression a basic variant of logistic regression, we will first start with the definition of logistic regression. For this purpose, assume that our three risk labels are reduced to two (risky, not risky).

Suppose that π represents the probability of a particular outcome, such as a stranger being labeled with risky, given his/her profile features as a set of explanatory variables

$x_1, ..., x_n$:

$$P(l = risky) = \pi = \frac{e^{(\alpha + \sum \beta_k X_k)}}{1 + e^{(\alpha + \sum \beta_k X_k)}}$$

where $0 \leq \pi \leq 1$, X_k is a feature value, α is an intercept and βs are feature coefficients, i.e., weights for feature values. The logit transformation $log[\frac{\pi}{1-\pi}]$ is used to linearize the regression model:

$$log[\frac{\pi}{1-\pi}] = \alpha + \sum \beta_k X_k$$

By transforming the probability (π) of the response variable to an odd-ratio ($log[\frac{\pi}{1-\pi}]$), we can now use a linear model. Given the already known stranger features and labels, we use Maximum Likelihood Estimation [99] to learn the intercept value and the coefficients of all features.

Although standard binary logistic regression and multinomial logistic regression use the same definition, they differ in one aspect: multinomial regression chooses a reference category and works with not one but $N - 1$ log odds where N is the number of response categories. In our model, $N = 3$, because the response has three labels (1, 2, 3). In both binary and multinomial logistic regression, intercept and coefficient values are found by using numerical methods to solve the linearized equation(s). With the found values, we can write the odd ratio as an equation. For example, in equation $log[\frac{\pi}{1-\pi}] = 0.7 + 1.2 \times X_1 + 0.3 \times X_2$, the intercept value is (0.7) and feature coefficients ($\beta_1 = 1.2$ and $\beta_2 = 0.3$). We can then plug in a new set of values (e.g., $X_1 = 0.5$) for features, and get the probabilities of response value being one of three labels. For example, for a specific stranger, the model can tell us that risk label probabilities of the stranger is distributed as %0.9 very risky, %0.09 risky and %0.01 not risky. As we can compute baseline label in real values, a stranger $s \in S$ is assigned a baseline label by weight averaging the probabilities of risk labels.

5.5 Friend Impact

So far, we have discussed clustering and baseline label estimation. In this section we will first discuss how these two aspects of our model are combined to compute friend impacts. After finding friend impacts, we will discuss how risk labels can be assigned to friends by considering the sign of impact values. In computing friend impacts, we use multiple linear regression [98], which learns friend impacts by comparing baseline and user given labels to strangers. To this end, we define an estimated label parameter to use in linear regression as follows:

Definition 15 (Estimated label) *For a stranger s and a user u, an estimated label is defined as:*

$$\hat{l}_{us} = b_{us} + \sum_{FC_i \in FC} FI(FC_i, SC_j) \times Past(u, s)$$

where $\hat{l}_{us}$ and b_{us} are estimated and baseline labels for a stranger s, and s belongs to the stranger cluster $SC_j \in SC$. Friend clusters FC are found by applying a algorithm to the mutual friends of user u and stranger s. $Past(u, s)$ denotes an intermediary value based on stranger labels given by user u, whereas $FI(FC_i, SC_j)$ represents impact of a friend f from a friend cluster FC_i on the label of stranger s from a stranger cluster SC_j.

In the rest of this section, we will define the $Past(.,.)$ and $FI(.,.)$ parameters, and explain how they are used to compute friend impacts.

5.5.1 The Past Labeling Parameter

We start by discussing the past parameter $Past(.,.)$ which returns a value from past labellings of strangers by user u.

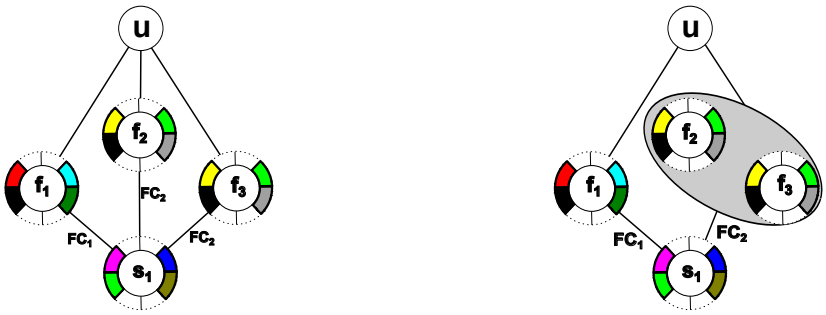
The past parameter is traditionally used in recommender systems to adjust baseline estimate [93]. The need for this parameter arises from the fact that baseline estimation is computed from labels of all strangers who have only one mutual friend with user u (i.e., first group dataset), and it tends to be a rough average. To overcome this, a subset of strangers, who are very *similar* to s and who have been labeled in the past by u, are observed and the baseline label is increased or decreased to make it more similar to the user given labels of these strangers.

In defining the past parameter, we consider two factors: how many similar strangers should be considered in this adjustment and what is an accurate metric for finding similarity of two strangers? For the first question, we use the computed stranger clusters. For a stranger s, similar strangers from the first group dataset are those (i) that are labeled by the same user u, and (ii) that belong to the same stranger cluster with s. Although we use stranger clusters to choose similar strangers, the similarity of strangers in a cluster can be low or high depending on the clustering process. With too few clusters and too many clusters, similarity of strangers in a cluster can be low and high respectively. We adjust the baseline labels by considering labels given to most similar users. To this end, we use the profile similarity measure that we proposed in Chapter 3. This measure assigns a similarity value of 1 to strangers with identical profiles, and for non-identical profiles the similarity value is higher for strangers whose profile feature values are more common in profile features of u's friends. Formally, we define the past labeling as follows:

Definition 16 (Past Labeling Parameter)
For a given user u and stranger s, the past labeling parameter is defined as:

$$Past(u, s) = \frac{1}{|SC_i|} \sum_{x \in C_i} PS(s, x) \times (l_{ux} - b_{ux})$$

where $PS()$ denotes the profile similarity between two strangers, l_{ux} is the user given label of stranger x, and b_{ux} is the baseline label of x. Strangers s and x belong to the same stranger cluster C_i.



(a) Multiple impacts for a friend cluster. (b) Single impact for a friend cluster.

Figure 5.2: Friend impact definitions by considering the number of friends from the same cluster. In the single impact definition, two friends do not increase the friend impact.

5.5.2 The Friend Impact Parameter

The second parameter from Definition 15, $FI(f, s)$, is used to show impacts of mutual friends on the risk label given to s by u. In modeling friend impacts, we wanted to see how friends from different clusters changed the baseline label. By using this approach, we explain impacts of friend clusters in terms of friend features that shape friend clusters. If there is at least one mutual friend from a friend cluster FC_i, we say that friend cluster FC_i may have impacted the label given to the stranger s. For the cases where a stranger s has two or more mutual friends from a friend cluster FC_i, we experimented with both options for $FI(f, s)$. Next, we will explain these options.

Multiple Impact for the Friend Cluster

In our first approach, we assume that a bigger number of mutual friends from friend cluster $FC_i \in FC$ will impact user labeling. Assume that from a friend cluster $FC_i \in FC$, we are given a set of mutual friends $MF_i = \{\forall f | f \in FC_i, f \in \{F_u \cap F_s\}\}$ of user u and stranger s. We define the impact of friend cluster FC_i on the label of stranger $s \in SC_j$ as follows:

$$FI_2(FC_i, SC_j) = |MF_i| \times I_{FC_i,SC_j}$$

where I_{FC_i,SC_j} is the impact of a cluster $FC_i | f \in \{FC_i \cap MF_i\}$ on the label of stranger $s \in SC_j$. Note that this impact (I_{FC_i,SC_j}) is the unknown value that our system will learn.

Single Impact for the Friend Cluster

In the second approach, we assume that a bigger number of friends from the same cluster does not make a difference in user labeling; at least one friend from the cluster is required, but more friends do not bring additional impact. This approach is shown in Figure 5.2b, where friends are shown with their cluster ids, and two friends from friend cluster FC_2 bring a single impact. Assume that from a friend cluster $FC_i \in FC$, we are given a set of

mutual friends of user u and stranger s. We give the impact of friend cluster FC_i on the label of stranger s as follows:

$$FI_1(FC_i, SC_j) = I_{FC_i,SC_j}$$

where I_{FC_i,SC_j} is the impact of a friend cluster FC_i on label of stranger $s \in SC_j$.

These different friend impact approaches change the model by including different numbers of friend impacts. The unknown impact variable I_{FC_*,SC_*} is learned by the least squares method [60]. The least squares method provides an approximate solution when there are more equations than unknown variables. In our model, each stranger's label provides an equation to compute impacts of k_1 friend clusters on k_2 stranger clusters (k_1 and k_2 are the final numbers of friend and stranger clusters in the k-means algorithm). In Example 5.5.1, we will explain these points and give equations of one stranger for single and multiple impact definitions.

Example 5.5.1 *Given a stranger $s_1 \in SC_1$ who is labeled by u, assume that the user given label $l_{us_1} = 2.3$, while the baseline label is $b_{us_1} = 2.7$. Again assume that $Past(u, s) = -0.2$. Equations for the stranger s with single and multiple friend impact definitions are respectively given as follows:*

$$2.3 = 2.7 + (I_{FC_2,SC_1} + I_{FC_1,SC_1}) \times -0.2$$
$$2.3 = 2.7 + (2 \times I_{FC_2,SC_1} + I_{FC_1,SC_1}) \times -0.2$$

After choosing one of these definitions of friend impact, we input one equation for each stranger s to the least squares method to compute impact values of friend clusters on stranger clusters. In the experimental results, we will discuss the definition that yielded the best results.

5.6 Friend Risk Labels

Learning impact values allows us to see the percentage of positive and negative impacts for each friend cluster. Negative impact values for a friend cluster shows that the friend cluster increases the risk label of strangers. Depending on a user's choice, friend clusters which have negative impacts less than $x\%$ of the time can be considered not risky. Similarly, a threshold $y\%$ can be chosen to determine very risky friend clusters. In our experiments, we heuristically chose $x = 20$ and $y = 50$. With these threshold values for risk labels, we formally define the risk label of a friend f as follows:

Definition 17 (Friend Risk Label) *Assume that the percentage of positive and negative impact values for a cluster $FC_i \in FC$ are denoted with Im_i^+ and Im_i^- respectively, where $Im_i^+ + Im_i^- = 1$. We assign a risk label to a friend f who is a member of the friend cluster*

FC_i (i.e., $f \in FC_i$) according to the negative impact percentage of the friend cluster FC_i as follows:

$$l(u,f) = \left\{ \begin{array}{lll} \textit{not risky} & \textit{if} & Im_i^- < 0.2 \\ \textit{risky} & \textit{if} & 0.2 \leq Im_i^- < 0.5 \\ \textit{very risky} & \textit{if} & Im_i^- \geq 0.5 \end{array} \right\}$$

Next we will give the experimental results of our model performance.

5.7 Experimental Results

In this section we will validate our model assumptions, and then continue to give detailed analysis of performance under different parameter/setting scenarios.

5.7.1 Validating Model Assumptions

Before finding friend impacts, we validated our model assumption (i.e., mutual friends have an impact on the risk label of a stranger) by using logistic regression on the whole dataset (4013 stranger labels and profiles). For this, we included *the number of mutual friends* as a parameter, and computed the *significance*[2] of model parameters. In overall regression, photo visibility, wall visibility, education and work parameters were excluded from the model because they were found to be non-significant. For significant parameters, $Pr(>|t|)$ values are shown in Table 5.1.

In the regression, there are two friend related parameters: the number of mutual friends and the friendlist visibility. Differing from the number of mutual friends, friendlist visibility is a categorical variable which takes 0 when the stranger hides his/her friendlist from the user and 1 otherwise. From Table 5.1[3], we see that seeing a stranger's friendlist increases the probability of the stranger getting label 1, whereas it is not an important parameter for label 3. Our main focus in regression analysis was to verify that the number of mutual friends parameter is significant. We found that an increasing number of mutual friends indeed helps a stranger get label 1, and decreases the probability of getting label 3. This result tells us that friends have an impact on user decisions and our assumption about the existence of friend impacts holds true. After validating our model assumption, we continue to the baseline label estimations.

5.7.2 Training for Baseline

Baseline calculation predicts labels for strangers without friend impacts. For this purpose we take strangers who have one mutual friend with users ($|MF| = 1$) into a new dataset (first group dataset), and train a logistic regression model. Logistic regression on the first

[2]Significance is measured by p-values. The p-value is the probability of having a result at least as extreme as the one that was actually observed in the sample. Traditionally, a $p - value$ of less than 0.05 is considered significant.

[3] Notes: Reference category for the equation is label 2. Standard errors in parentheses. Significance codes: '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Table 5.1: Regression results for all data points. p-value=2.22e-16. Total N=4013.

	Label 1	Label 3
Intercept	-1.2668850***	-0.7626810***
	(0.146)	(0.138)
Mutual friends	0.0379547***	-0.0467834***
	(0.008)	(0.012)
Gender	-0.3696749**	0.3480055**
	(0.118)	(0.113)
Friendlist visibility	0.6203365***	-0.0642952
	(0.125)	(0.118)
Locale	0.6167273***	0.7070663***
	(0.180)	(0.172)
Location	0.1347104	0.2708697*
	(0.128)	(0.125)
N	1116	1161

Table 5.2: Regression results for the first group data points. p-value = 5.6701e-11. Total N=1520.

	Label 1	Label 3
Intercept	-2.5400***	-0.8661***
	(0.6305)	(0.2791)
Gender	-1.1026**	0.6985*
	(0.4108)	(0.3350)
Friendlist visibility	0.4705*	0.5214
	(0.2075)	(0.1706)
Wall	0.4173.	-0.1595
	(0.2463)	(0.2262)
Photo	1.9425**	0.1361
	(0.6093)	(0.2339)
Locale	0.1446	0.5846*
	(0.2881)	(0.2277)
N	278	588

group dataset finds how stranger features bring users to label strangers. Table 5.2 shows model parameters and their corresponding p-values.

In Table 5.2, we see that when users label the first group strangers, photo and wall visibility are significant parameters. If these items are visible on stranger profiles, the probability of strangers getting label 1 increases. In the whole dataset (see Table 5.1), these two parameters were found to be insignificant. Another interesting result is that locale[4] is significant for label 3 whereas it is non/significant for label 1. A high locale value means that the stranger is similar to existing friends of users, but this high similarity is shown to increase the probability of strangers being labeled as very risky, i.e., receiving

[4]Locale is the web interface language of the user on the social networking site (e.g., IT for Italian and RU for Russian).

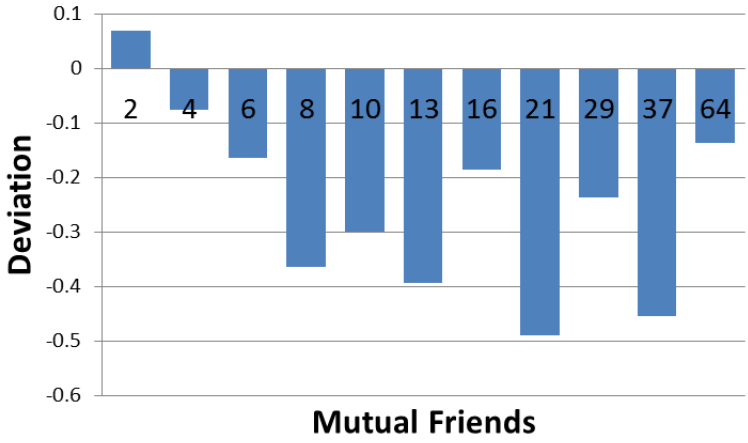


Figure 5.3: Deviation of user given labels from baseline labels. Values in the x-axis are the number of mutual friends between a stranger and user.

label 3.

After computing a baseline label for all strangers, we use the difference between user given and baseline labels ($l_{us} - b_{us}$) to model the friend impact. These differences (deviations from the baseline label) are shown in Figure 5.3. In the figure, we see that user given labels are lower than the computed baseline label, which shows that in overall friends have positive impacts (i.e., thanks to mutual friends, users assign lower risk labels to strangers.). Overall, we found that there is not a linear relation between the number of mutual friends and the deviation values. This non-linearity changes how we define the impacts of friend clusters. In Section 5.5 we gave two definitions for friend impacts (see Figure 5.2) to account for deviations from the baseline label.

In multiple friend impacts we assumed that more mutual friends from a friend cluster bring additional impacts. On the other hand, in single friend impact one friend was enough to have the impact of a friend cluster. This finding implies that more friends of the same cluster do not provide any benefits to strangers on Facebook and mutual friends from different clusters are more suitable to change the user's risk perception about a stranger. We believe that this can be generalized to other undirected social networks.

In the rest of the experiments, we will give the results computed by using the single friend impact definition. We will now explain the model performance under different clustering settings.

5.7.3 Clustering

For clustering 12659 friends, and 4013 strangers we experimented with k-means and hierarchical clustering algorithms. In our experiments with different numbers of final clusters, the k-means algorithm yielded the best results for friend clustering, whereas hierarchical clustering was better for stranger clustering. Due to space limitations, we will omit

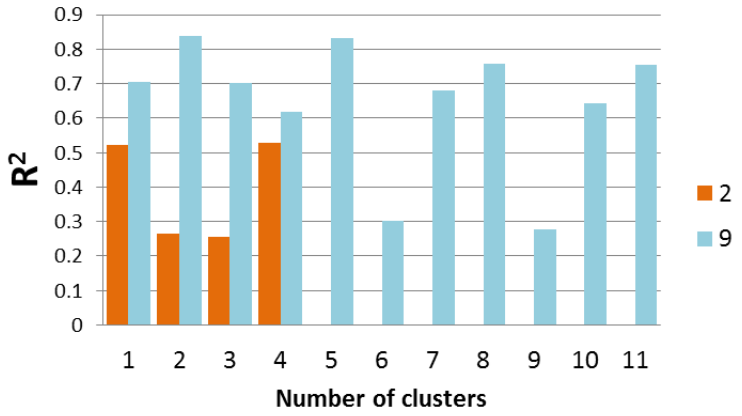


Figure 5.4: Coefficient of determination (R^2) values for 2 and 9 friend clusters

hierarchical clustering results for friends and k-means results for strangers.

Friend Clustering: In Figures 5.4 and 5.5, we show the adjusted coefficient of determination[5] (R^2) of our multiple regression model with different k values for friend clustering. The x-axis gives the number of stranger clusters for which at least one friend cluster has an impact. In Figure 5.4 we see the performance for maximum and minimum number of friend clusters. For $k = 2$, friend clusters are very roughly clustered, and each cluster is not homogeneous enough (i.e., contains different types of friends) to mine friend impacts[6]. As a result, we can observe friend impacts on very few clusters. For $k = 9$, friend clusters are more homogeneous, but in this case our multiple regression model does not have many data points (strangers) to learn the impacts of friend clusters.

Figure 5.5 shows the results for $k = 5, 6, 7$ values. For two k values, 5 and 6, we have the best results. Our model hence suggests that friends of social network users can be put into 5 or 6 clusters when considering how much they can affect user decisions on stranger labeling.

Stranger Clustering: In Figure 5.6 we show how the R^2 values change for the biggest and smallest numbers of stranger clusters. With 8 stranger clusters, our model can detect friend cluster impacts on 5 out of 8 stranger clusters only, whereas for 158 clusters the number is 15 out of 158. For 158 stranger clusters, R^2 values are generally low because strangers are distributed into too many clusters, and each stranger cluster does not have many data points (strangers) to learn from. Although finding impacts on 5 out of 8 stranger clusters seems like a good performance, low R^2 values (lower than 0.5) show that the model can explain less than 50% of the variation in data. In Figure 5.7 we see that more stranger

[5]The adjusted coefficient of determination is the proportion of variability in a data set that can be explained by the statistical model. This value shows how well future outcomes can be predicted by the model. R^2 can take 0 as minimum, and 1 as maximum.

[6]We use F-ratio probability to test the significance of parameters, i.e., a low probability (we use .05 as cutoff) for the F-ratio suggests that at least some of the friend cluster impacts are significant.

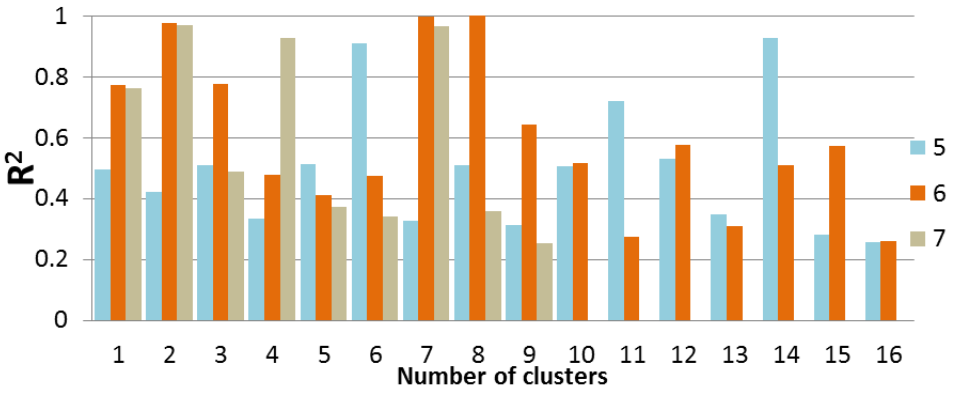


Figure 5.5: Coefficient of determination (R^2) values for 5, 6 and 7 friend clusters

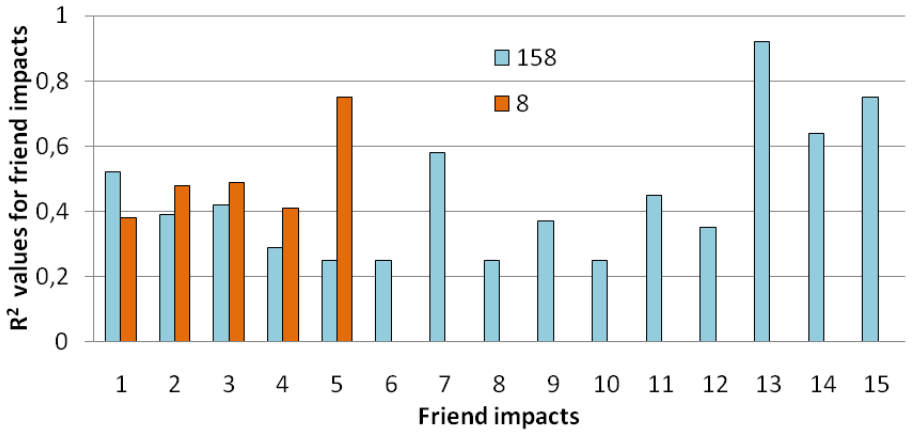


Figure 5.6: Coefficient of determination (R^2) values of friend impacts for 158 and 8 stranger clusters.

clusters can improve the model performance and this leads to R^2 values close to 1. For 26 stranger clusters, R^2 values are better, and we can find friend impacts in 16 out of 26 stranger clusters.

Cross Validation: A major point in statistical modeling is the response to out of sample validation; a statistical model can be over-fitted to the training data, and it can perform poorly when applied to new testing data. After clustering and prior to learning friend cluster impacts, we prepare a test set for validating our model. We remove 10% of strangers from stranger clusters and set those aside as the test strangers (T). Once friend impacts are found for stranger clusters, we plug in the set of test strangers, and calculate the root mean square value (RMSE) of their labels. RMSE for a stranger s and user u is defined by using the predicted label $\hat{L}_{us}$ and user given label L_{us} as $RMSE = \sqrt{\frac{\sum_{s \in T} (L_{us} - \hat{L}_{us})}{|T|}}$.

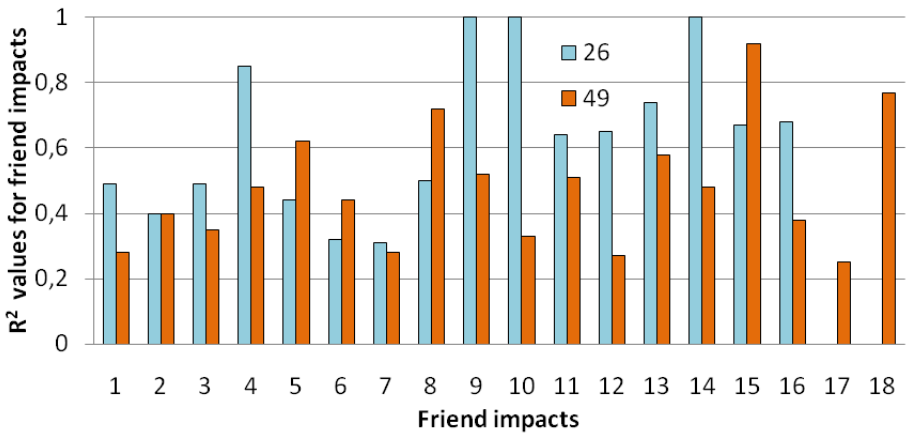


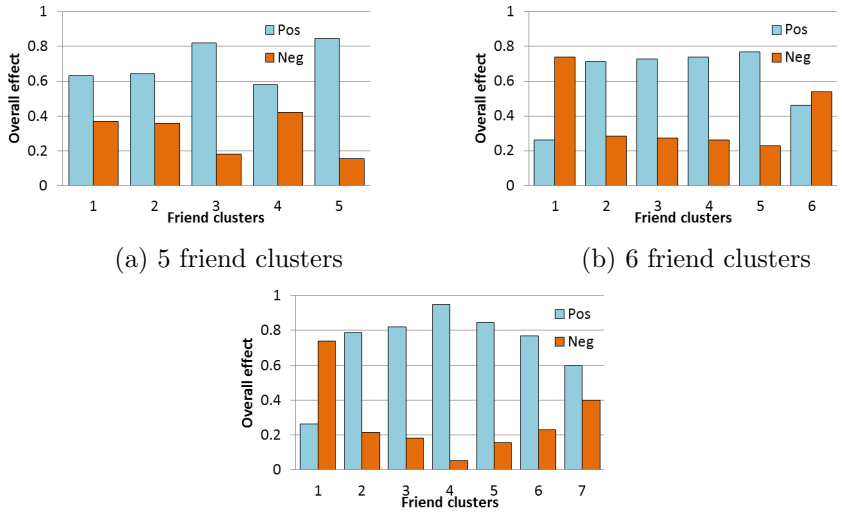
Figure 5.7: Coefficient of determination (R^2) values of friend impacts for 26 and 49 stranger clusters

Table 5.3: Performance values for different numbers of stranger clusters.

Cluster count	8	26	49	82	158
R^2	0.51	0.64	0.48	0.54	0.45
Median Size	62	25	16	12	7
Validation points	179	99	69	48	27
RMSE	0.35	0.45	0.62	0.97	0.94

Cross validation results for different numbers of stranger clusters is detailed in Table 5.3 by using 6 friend clusters. The first row of the table shows the number of stranger clusters, whereas the second row shows the average R^2 values in these clusters. In the third row, we show the median size of stranger clusters; with increasing numbers of clusters, the number of strangers in each cluster decreases. In the case of 158, the average number of strangers in a cluster is reduced to 7, and this results in a poor performance because the model cannot have enough data to learn friend impacts on stranger clusters. The average number of validation points are shown in the fourth row. An increasing number of stranger clusters results in fewer validation points because some clusters will have less than 10 strangers themselves. In the fifth row, the root mean square values (RMSE) are shown for these validation points. In 26 stranger cluster our model yields the best R^2 and $RMSE$ pair results.

These experimental results suggest that the optimal number of stranger clusters (26) is bigger than the optimal number of friend clusters ($k = 5, 6$). We explain this by the fact that although users can choose friends of specific characteristics, they cannot do so with strangers. As a result, strangers are more diverse than friends, and they need to be clustered differently from friends.



(a) 5 friend clusters

(b) 6 friend clusters

(c) 7 friend clusters

Figure 5.8: Percentage of positive and negative impact values for friend clusters.

5.7.4 Friend Impacts and Risk Labels

In this section we will give computed friend cluster impacts, and show how friends are assigned risk labels.

The rationale behind clustering was to observe different friend cluster impacts on different stranger clusters. Although a friend cluster can have an overall positive impact (i.e., reduces the risk label of most strangers), friend clusters might have different signs and multitudes of impact values on stranger clusters. In Figure 5.8 we show how different friend clusters can have positive and negative impact values for different k values (number of friend clusters). Note that clusters are not identical across these figures, i.e., cluster 1 can have different members in each figure. This is because with different number of final clusters, the clustering algorithms produce potentially different clusters of data points. As seen in Figure 5.8a, when we increase the number of friend clusters from $k = 5$ to $k = 6$, positive and negative impact frequencies change for each cluster because either friend clusters became more homogeneous or some clusters did not have enough data points to learn from. Figure 5.8b shows two friend clusters with overall negative impacts (friend clusters 1 and 6). Figure 5.8c shows the positive and negative impact frequencies for $k = 7$, where frequencies are more emphasized for negative and positive impacts of a cluster. Note that the number of overall negative clusters is reduced from 2 to 1 here. Similar to a transition from 5 to 6 clusters, friends of two negative clusters might be put into the same cluster (cluster 1) or there were no longer enough strangers for some friend clusters to learn a negative impact.

The existence of both positive and negative impact values for each friend cluster confirms our intuition that impacts of friend clusters vary depending on a stranger cluster. A

friend is assigned a higher risk label when a friend cluster has a big percentage of negative impact values. In Section 5.6, we gave definitions of friend risk labels according to two threshold values (x=20, y=50) of negative impact percentages. By using k=6 friend clusters, from Figure 5.8b we see that friends from friend clusters 1 and 6 are labeled as very risky because the negative impact percentages for the clusters are > 0.6. In the figure, we also see that none of the clusters have < 0.2 negative impacts, hence no friends cluster is said to be not risky (label 1).

We tested the accuracy of our risk definition for friends by observing 261 deleted friendships of users. As a performance measure, we assumed that the deleted friends should come from friends who are labeled as very risky, i.e., friends who belong to the 1st and 6th clusters. We have found 117 of the 261 deleted friends were found to belong to the 1st and 6th friend clusters.

Although we chose to use specific values for very risky and not risky label thresholds (x=20, y=50) in assigning risk labels to strangers, our model can ask social network users to define these threshold values on their own. With this approach, our risk model for friends can be personalized by users and applied to privacy settings on social networks.

5.8 Conclusion

In this chapter, we looked into risks of friendships and analyzed how the risk labels of friends of friends can be used to compute risk labels of friends. We found that the number of mutual friends is not very important to change the risk perception of a user towards a friend of friend. On the other hand, having different types of mutual friends (i.e., friends from different friend clusters) with a friend of friend plays a bigger role in users' risk perception. Our results showed that in terms of risk, friends can be grouped into 6-7 clusters, whereas the number of groups for strangers can reach 26 or more. These results show that even though user numbers reach millions, friends for each user have similar roles. We have validated risk labels of friends on deleted Facebook friendships, and showed that risks of friendships can indeed be learned by considering users' risk perception towards friends of friends. In the future, we want to create sets of global privacy settings by using our risk model, so that privacy settings can be automatically applied to different social network users.

Chapter 6

Detecting Anomalies in Interactions

6.1 Introduction

Over the last couple of years, there has been significant research effort in mining user behavior on social media for applications ranging from sentiment analysis [107] to marketing [126]. In most of those applications, usually a snapshot of the user behavior and user relationships are analyzed to build the data mining models [18, 115]. Although such analysis are really useful, in many cases, they do not find significant changes happening to a user's behavior over time.

For example, for marketing purposes, a user's tweets may be analyzed to infer his/her current interests. Although, understanding the current interests of a user is helpful for showing display ads, detecting significant changes in interests over time could be as important. For example, after a significant life event such as having a baby, a user may start tweeting more about baby food. Furthermore, such a user may start to follow a leading pediatric expert on Twitter who tweets about best practices related to infant feeding. Detecting such significant changes or anomalies[1] may have important applications. For instance, if we detect such an "anomaly" in the context of the above scenario, we may predict that in the future such a user may be interested in child seats even though his/her current tweets may not show any interest. Detecting such anomalies in user behavior may have other important applications. For example, anomalies may indicate the hijacking of an account. Of course, in order to make such kind of predictions, we should start with understanding and detecting such significant changes (i.e., anomalies) in user behaviors.

Although recent work in emerging topic detection [31] or trend analysis [89] have studied changes in mentioned or tweeted topics, they are prone to miss some significant changes in individual user behavior because they require users to tweet about these changes. In other words, detecting behavior changes by monitoring tweets depends on users' willingness to tweet about these changes. However, recent work show that Twitter's content stream

[1] We use the term "anomaly" to represent such significant changes in user behavior.

is dominated by a small number of users, and the top 15 percent of users account for 85 percent of all tweets [75]. Furthermore, a quarter of users active during a one month period are reported to tweet just once, while half tweet between one and four times. With this dearth of user tweets, it is clear that detecting behavior changes by considering only tweets may lead to undetected anomalies for a big portion of users.

In contrast, despite a shortage of tweets, users are more active in following new people. For example, in our experiments we have found that 78 thousand users followed 16 million new friends in a four year period. We show that, by observing the topics of new friends, behavior changes can be found by a larger portion of Twitter users.

6.2 Overview of the Problem

We are interested in analyzing two definitions of anomaly of user behavior in Twitter. In our first definition, we aim at capturing the divergence of individual behavior from user's past behavior. As such, a user is said to exhibit an anomalous behavior if newly followed accounts have different topics from the ones followed in the past. In the second definition, we take a wider view on Twitter, as we aim at finding a divergence of individual behavior from collective behavior. As an example, a teenager's new interest in following users from academia can constitute an individual anomaly (i.e., divergence from his/her past behavior). However, when we look at friendships of other teenager users we might deduce that in collective behavior, teenagers start following academia users after their college enrollments. Looking at collective behavior helps us put newly emerging patterns in individual behaviors into perspective. Thus, we define anomaly as an individual's interest in following topics that are not followed by other similar Twitter users.

To highlight potential anomalies in user behavior we need to compare two different snapshots. For this purpose, we have used the Twitter dataset crawled in 2009 by Kwak et al. [72] as the first snapshot. For the second snapshot, we have crawled Twitter in 2013, by querying the Twitter API [2] for friends and tweets/bios of users. Although an API call can return the last 200 tweets of a user, we stored only the latest 50 tweets due to storage limitations. Each tweet t_x of a user u_x is a short text of maximum 140 characters. In what follows, for any user account u_x, b_x denotes its bio, whereas its tweets are denoted as $T_x = \{\cup t_x^{(m)} | m = 1, ..., 50\}$.

During data collection, we have discovered 11M Twitter accounts, and stored bios/tweets of 3 million users. From these stored users, we chose 100K seed users $\mathcal{S}$ that existed in the Twitter crawl of 2009 by Kwak et al. [72]. From 2009 to 2013, users have added new friends, and deleted some existing ones. We call friends of users in 2009 and 2013 as old and new friends, respectively. More precisely, we will denote old and new friends of a seed user u_s with Γ'_s and Γ''_s, respectively, where $\Gamma_s = \Gamma'_s \cup \Gamma''_s$. In Figure 6.1, the edges which existed in dataset [72] are labeled with 2009, whereas new edges that we discovered are labeled with 2013. For instance, for the seed node B, its old friends are E,

[2] https://dev.twitter.com/

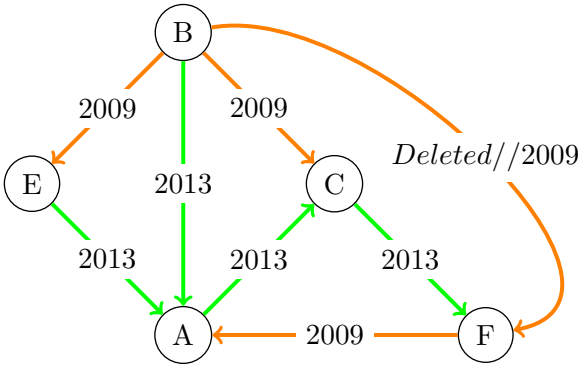


Figure 6.1: Old and new friends of users

C, and F, whereas the new friends are E, C and A. As depicted by the broken edge, F is no longer a friend of B in 2013.

Other than seed users and their old/new friends, we have also queried 1.2 million twitter users to avoid learning from a small sample of all Twitter users. These additional users have been added to the dataset regardless of whether they existed in 2009 or not. In what follows, the additional Twitter users will be denoted with Λ.

Based on 3 million user accounts, we have found a median number of 352 (1896 average) friends per each user. For some service accounts, such as CNN news, this number reaches up to several millions. As we are mainly interested in personal user accounts in our anomaly metrics, we decided to use a cut off point to separate personal accounts from these service accounts. To this end, we have only queried accounts that have less than 500 friends. This pre-screening also reduces the number of queries sent to the Twitter API for bio information and recent tweets. After pre-screening, an average number of 208 friends have been queried for each seed user. These seed users represent the dataset on which we analyzed the possible anomalies.

To discover anomalies in *individual behavior*, we compared 2009 and 2013 friends lists of each seed user, so as to determine if these two lists are similar. As described in Section 6.3.1, similarity is computed as cosine similarity on labels associated with friends in the lists, that is, descriptive labels associated with tweets and bio of friends. In Section 6.3.1, we explain how these friend labels are computed.

In contrast to individual behavior, anomaly detection with *collective behavior* tries to capture how other users that are similar to a seed user change their friends lists from 2009 to 2013. An important question is therefore how to find users that are similar to the considered seed user, so that collective behavior can be studied. For this problem we employ clustering techniques, and put seed users into k clusters (see Section 6.3.2).

Given a new friendship, we look into its compliance with both individual and collective behavior of a user, so as to determine if this new friendship has to be considered anomaly or not. We detail this process in Section 6.3.3.

6.3 Methodology

In this section, we first discuss how we build topic models to capture user interests. Later on, we discuss how we cluster users to find group of users with similar interests. Finally, we discuss the anomaly measures used in our study.

6.3.1 Topic Models

To create topic models, we have used two kinds of information related to Twitter users: bio texts and tweets. In both cases, stop words and punctuation marks are removed from the documents before building the models.

In the first case, a bio text is used as a single document to label its owner. In the second scenario, each tweet is considered as a separate document and multiple tweets are used to label a user. Indeed, although each user has a single bio text, there can be thousands of tweets from a user. As it is too costly to include all tweets of a user in our computations, we had to limit the number of tweets used in the labeling process. Furthermore, using tweets for labeling requires selecting the most informative tweets of a user, since unlike bios, tweets can be about daily conversations and have fewer words after stop word removal.

Given 2013 bios/tweets of all users (i.e., seed users, their friends and additional twitter users, Λ) represented as a set of documents, we use the latent Dirichlet allocation (LDA) [17] topic modeling to find a predefined number of latent topics in the documents. LDA is a generative model that explains similarity of some documents by latent topics. During initialization, LDA requires a total number of topics K and gives two sets of results that we use to label users.

The first result is a set of latent topics as a mixture of words, where ϕ_k is the word distribution for topic k. In other words, topic k is given as a set of words where each word w has a probability $\phi_{k,w}$ of occurring in topic k (e.g., topic 1, social:0.85, media:0.15, $\phi_{1,media} = 0.15$).

The second result of LDA is the topic distribution for a document t, denoted as θ_t. Each document t is represented by a K-dimensional probability distribution of topics, where $\sum_{k=1}^{K} \theta_{t,k} = 1$. For example, document 2 can have the topic distribution 1:0.8, 7:0.2, which indicates that probability of topic 1 occurring in document 2 is $\theta_{2,1} = 0.8$.

By itself, LDA does not name the topics that are discovered. Instead, the most probable words related to a topic can be used to represent a topic. For example, a top 5 word list of "university, student, school, college, professor" can reveal that the topic is about academia.

We use topic distribution (θ_t) of documents to assign labels to users. More specifically, each user x is assigned a profile vector P_x of K dimensions, where K are the number of latent topics found by LDA. The k^{th} value in a profile vector P_x shows the probability of user u_x being labeled with the k^{th} topic.

Each user is said to have two profile vectors: a tweetLDA profile vector when tweets are used as an input for LDA, and a bioLDA profile vector when bios are used as input for LDA.

In the case of bios, a user's profile vector is the topic distribution of his/her bio. However, when using tweets as our LDA input, we need to aggregate topic distributions of multiple tweets from a user to create a user's profile vector. To this end, we employ a basic weight aggregation, and define profile vector P_x of user x as follows:

Definition 18 (Profile vector for tweetLDA) *Given a user x, let TW be the set of tweets of user x processed by LDA with K topics. The $k-th$ element of the tweetLDA profile vector for x is defined as $P_{x,k} = \frac{1}{|TW|} \cdot \sum_{t \in TW} \theta_{t,k}$, where $\theta_{t,k}$ is the probability of topic k in tweet t, and $0 < k \leq K$.*

6.3.2 Clusters

By clustering, we aim at grouping users with similar profiles into the same clusters, so that we can compare users' new friendships with those of other similar users. To this end, we created a dataset $\mathcal{PC}$ from profile vectors of seed users, where profile vectors are computed by using tweetLDA or bioLDA. We have experimented with k-means and hierarchical clustering [49]. This clustering process results in $|C|$ clusters, where each cluster $C_c \in C$ holds a set of similar seed users.

In addition to clustering, we have considered using a similarity threshold or choosing the $top-n$ most similar users. The main difficulty of these approaches was to choose an optimum threshold similarity or the n value so that only the most similar Twitter users are analyzed for collective behavior. Furthermore, computational complexity issues require choosing another threshold for how many similar users are chosen for analyzing collective behavior. If $top-n$ similar users are considered in the collective behavior analysis, n has to be reasonably large to ensure that collective behavior is adequately observed. On the other hand, if we choose a very large n value, we face the risk of including too many Twitter users and making computations unfeasible.

Faced with these issues, clustering provides an efficient choice because it considers all similarities among seed users, and does not require a similarity threshold value. Although choosing a total number of clusters $|C|$ is similar to choosing the n value, clustering does not create fixed sized clusters and it ensures that regardless of cluster size, users in a cluster are more similar to each other than to users from other clusters.

6.3.3 Anomaly Definitions

We propose two definitions for anomaly detection: individual and collective anomaly.

In individual anomaly detection, we are interested in detecting changing interests of individual users over time. To this end, we define divergence as the dissimilarity of friends' profile vectors taken at time t from profile vectors of friends taken at time t'. In other words, a user is said to exhibit anomalous behavior if newly followed accounts are labeled differently from the ones followed in the past. We have experimented with several definitions for the dissimilarity. In two extreme cases, (1) new friends can be highly similar to few old friends, or (2) new friends can have low similarity with many old friends. As our old data comes from the early years of the Twitter social network when users had just started

to use it intensively, we have decided to use definition 1, since it allows users to expand their friends list in time by following users who are very similar to few old friends. If a new friend is not very similar to at least some old friends, we define this new friendship as an individual anomaly. Formally, we define anomalous friendship of a user as follows:

Definition 19 (Individual anomaly) *Let u_s be a user and let Γ' and Γ'' be his/her friend sets at time t and t' ($t' > t$), respectively. A new friend $u_f \in \Gamma'' \setminus \Gamma'$ is individually anomalous, if*

$$\xi > \arg\max_{u_o \in \Gamma'} sim(P_f, P_o)$$

where P_o is the profile of u_o, ξ is a global similarity threshold, and $sim()$ is a similarity function. A user profile P_* is computed by using either bioLDA or tweetLDA.

In collective anomaly, we take a wider view on Twitter, and define anomaly as a seed user's interest in following topics that are not followed by other similar Twitter users. We use clusters for forming the similarity definition; similar users of u_s are those that belong to the same cluster. Given a seed user u_s, a user's collective divergence is computed as the dissimilarity of its new friend from the old friends set of users belonging to its cluster.

Definition 20 (Collective anomaly) *Let u_s be a user and let C_c be the cluster of users such that $u_s \in C_c$. Let Γ'' and Γ' be u_s friend sets at time t' and t ($t' > t$), respectively. A new friend $u_f \in \Gamma'' \setminus \Gamma'$ is collectively anomalous, if:*

$$\xi > \arg\max_{u_o \in \Gamma'^*} sim(P_f, P_o)$$

where $sim()$ is a similarity function, and $\Gamma'^* = \bigcup_{u_c \in C_c} \Gamma'_c$ is the union of old friends of users from cluster C_c.

For both anomaly definitions, we use cosine similarity to compute the similarity between profile vectors.

By definition, a new friendship of a seed user u_s is individually anomalous if the profile vector of the new friend is not very similar to at least one old friend of u_s. This new friend is also collectively anomalous if his/her profile vector is dissimilar to the old friends of users who are similar to u_s.

We term a new friend u_f as a verified anomaly ($v_f = 1$) if both individual and collective anomaly conditions hold, and unverified ($v_f = 0$) if both conditions do not hold. A user is said to show more anomaly when the number of anomalous friends increases. However, importance of one friend's anomaly for a user also depends on the size of both Γ', and Γ'^*. When these old friend sets are larger in size, the probability of a new friend being an anomaly decreases because there will be a bigger number of old friends that can be similar. Considering this, we give more importance to anomalous seed users who have a bigger number of old friends, or have a bigger number of similar users. In other words, if seed users have already well established Twitter accounts, we do not expect their interests to diverge greatly in the future. To quantify the anomalous behavior of a user, we take into account old friends of users and cluster members, and define an anomaly score as follows.

Definition 21 (Anomaly Score) *Let u_s be a user with an old friends set Γ', and union set Γ'^*, the anomaly score is defined as:*

$$\lambda_s = \frac{log10(|\Gamma'^*|) + \sum_{u_f \in \Gamma''_s \setminus \Gamma'_s} v_f}{|\Gamma'| + log10(|\Gamma'^*|)}$$

where $v_f = 1$, if friend u_f is a verified anomaly, Γ'^* is the union set of old friends of cluster members from cluster $C_c | u_s \in C_c$, and a log function is used to dampen the impact of cluster members' friends.

In Definition 21, a bigger value for $|\Gamma'^*|$ ensures a higher anomaly score. In other words, if a user has many anomalous friendships despite a bigger union set, the user will be assigned a higher anomaly score.

6.4 Experimental Analysis

In this section, we first describe how we use text based Twitter data in our experiments, and explain how the topic model has been constructed from tweets and bio texts of users. Then, in Section 6.4.2 we look into topic model, and analyze its performance. In Section 6.4.3, we discuss some characteristics of Twitter network data, and show how new friendships change the way data is consumed on Twitter. Section 6.4.4 describes anomaly detection results. Finally, Section 6.4.5 describes the validation of found anomalies.

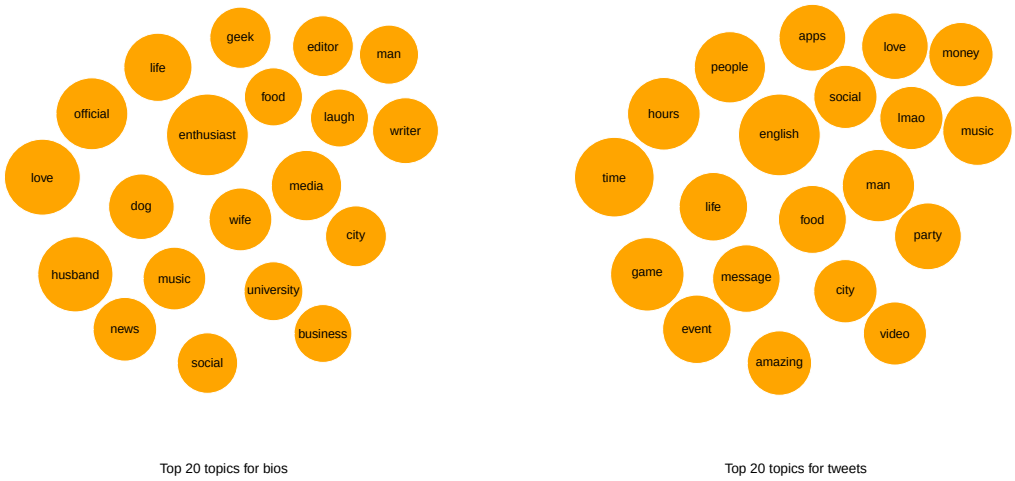
6.4.1 Text Data Characteristics

For our topic model input, we have used an expanded stopwords list (554 words) and also removed from tweets and bios internet related technical terms, such as http, tinyurl and bit.ly. After this trimming, tweets and bio texts of more than 30 characters have been used as input documents. With this criteria, %4 of all tweets, and %16 of all bios have been removed from the dataset. The number of removed bios is higher, because bios are poorly populated by many users.

Although we have used all available bios, we had to limit the number of tweets due to computational costs. To this end, we have picked at most 10 longest tweets from each user, and used a total of 21686103 tweets in our topic model. In the next section, we describe the topics extracted from our text data.

6.4.2 LDA Results for Topics

In topic modeling, we experimented with $K = 100$, $K = 150$ and $K = 200$ numbers of topics to describe our dataset. Overall, building a LDA model using 100 topics has given better sets of words for documents, therefore in the rest of this chapter, the results are shown for LDA models with 100 topics.



(a) Top 20 topics found from bio texts of users.

(b) Top 20 topics from tweets.

Figure 6.2: Topics and their top words.

In LDA topic modeling, we have used 2013 bios and tweets of users as separate datasets, and found two different (potentially) topic distributions for each user (bioLDA and tweetLDA, respectively). As bio texts define a user's profile, whereas tweets are mostly messages about daily lives, we considered bioLDA to explain who a user is, and tweetLDA to show what a user writes about. Top 20 topics from tweetLDA and bioLDA are shown in Figure 6.2.

In order to understand the performance of bioLDA and tweetLDA, we created a list from official Twitter accounts of 41 U.S. senators. Among these, 27 and 14 accounts belong to senators from the Democratic Party and the Republican Party, respectively. Nine of these senators are excluded in our experiments, because they are not followed by any user in our LDA datasets. Out of 41 senators, 22 democrats and 10 republicans have been found to be in our tweetLDA dataset, whereas bioLDA dataset includes 20 democrats and 10 republicans. [3]

By summing the number of times a topic appears in senators' profile vectors, we have computed five most frequent topics for democratic and republican senators. Table 6.1 shows top-5 topics and party affiliations for tweetLDA and bioLDA. To represent each topic, we have used its top two most frequent words. For example in tweetLDA, senators from both parties have topic 42 as the most common topic in their profile vectors, and this topic is represented by two words: *obama* and *president*. The 42nd topic also includes many political terms, such as vote, election, gop, state, debate, house, and bill. In the table, tweetLDA results show that senators from both parties share one topic, but other

[3]Two senators are excluded in bioLDA because of short or blank bios.

Table 6.1: Top words from each one of the top-5 topics of U.S. senators found by tweetLDA and bioLDA

tweetLDA		bioLDA	
Democrat	Republican	Democrat	Republican
Obama, president	Obama, president	city, area	city, area
city, park	tax, year	official, account	official, account
business, innovation	art, photo	news, latest	conservative, christian
energy, solar	war, Israel	conservative, christian	official, love
great, love	live, show	public, views	Facebook, Youtube

four topics are different. The fourth most common topic for democrats is about *solar* and *energy*, whereas this topic appears 23rd in the list of topics for republicans. [4] The second most common topic (i.e., tax, year) for republicans is the 12th most common topic in democrats' accounts.

Overall, bioLDA results show that technical words, such as official and Facebook, lead to found topics that can describe the function of an account (e.g., Official Twitter feed for Senator Frank R. Lautenberg), but these topics fail to give us a sense of what the Twitter profiles are about. On the other hand, tweetLDA is more useful to understand characteristics of users, because tweetLDA uses more information (i.e., multiple tweets instead of a short bio) and diverse texts (i.e., multiple tweets about different topics).

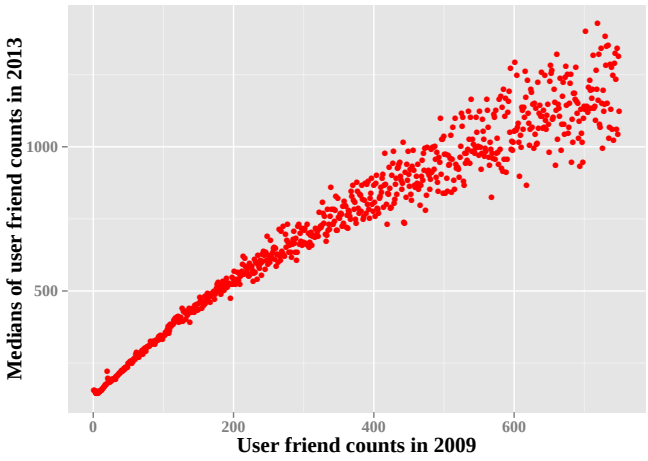
6.4.3 Network Data and Changing Data Interests

Our anomaly definitions are based on a dynamic network where both network connections and user generated data can be observed over time. In line with this assumption, we have tracked changes in friendship counts of twitter users from 2009 to 2013. In Figure 6.3, we show this change by plotting friendships counts in 2009 for friendship counts in 2013. With this increase in friend counts, we also assumed that Twitter users change their interest in who to follow for their data.

Similar to the "who and what" argument we discussed in topic modeling, new friends of a user can tell us "how" a user consumes data. In order to find a global pattern on data consumption, we tracked the relationship between profile vectors of users, and profile vectors of their new friends. Given a seed user's profile vector P_s and the profile vector of his/her new friend P_f, we found prominent topics (i.e., the highest probability valued topic) in each profile vector, and counted the number of times users from these two topics become

[4] Other words from the topic include words such as green, water, power, wind, oil and gas

Figure 6.3: Changing friend counts on the Twitter network. The x-axis values are numbers of friends in 2009. In the y-axis, median values of 2013 friend counts are shown. As shown in the left bottom corner of the chart, Twitter users who had almost zero friends in 2009 have a median number of 150 friends in 2013. X-axis values are limited to a maximum of 750 for a better viewing experience.



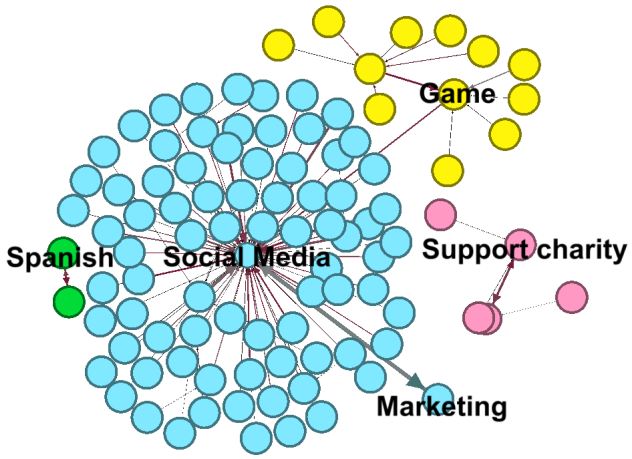
friends. Figure 6.4 shows the results in a graph, where each one of 100 tweetLDA topics is represented as a node, and a directed edge shows the number of friendships between users that are identified with those topics. In the graph, the users identified with *social media* topic (given with the word social in both parts of Figure 6.2) has the biggest number of friendships from users identified with other topics. Although *social media* is the most popular topic in the graph, its users become friends with users identified with the *marketing* topic, which is not popular among users identified with other topics.

Unlike undirected networks, such as Facebook.com, Twitter friendships are not built on mutual consent. As such, friend counts of users can be vastly bigger than those on undirected networks. Figure 6.1 shows this phenomenon, where many users have more than 1000 friends in 2013. However, Figure 6.4 shows that friendships are not uniformly distributed among users identified with different topics. In other words, many new friendships are created in time, but new friends are chosen from users identified with small number of topics.

6.4.4 Anomaly Results

In anomaly detection experiments, we used both tweetLDA and bioLDA topics to understand the friendship behavior of Twitter users. Our experiments have worked with 100K seed users (i.e., users who had a Twitter account in 2009), and after removal of inactive accounts (i.e., users with no tweets, hidden tweets, blank bios or no new friends), we have examined 78K seed users and their 16016152 new friendships and 2905004 deleted

Figure 6.4: Number of friendships among users of different tweetLDA topics.



friendships from 2009 to 2013.[5]

Our anomaly experiments start with finding individual anomalies of new friendships, and continues with checking discovered individual anomalies for collective anomalies. In the next sections we will first discuss similarity of new friends to old friends and give performance results of tweetLDA and bioLDA. Afterwards, we will detail our findings in collective anomaly.

Individual Anomaly

In individual anomaly, we are interested in seeing how cosine similarity values between 2009 and 2013 friends are distributed. According to our definition (see Section 6.3.3), a threshold similarity value of ξ is used to define an individually anomalous friend. As such, if similarities of new friends are low, more new friends will be signaled as individual anomalies.

Our experiments with bioLDA resulted in lower similarities and higher variance values while computing profile vector similarities. A total of 1200392 new friends were found to have 0 similarity to old friends, whereas by using profiles vectors from tweetLDA we only had 87890 new friends with zero similarity. Furthermore, other than the zero similarity region in the distribution, bioLDA also results in high variance values for similarity values above 0.75. We give the bioLDA similarity distribution in Figure 6.5, where percentage of friends with zero similarity (i.e., x=0) is not shown to increase clarity of the figure.

Unlike bioLDA, as shown in Figure 6.6, tweetLDA results in low variance values in the similarity distribution.

[5]The number of new friendships is greater than the total number of queried Twitter users because we have queried Twitter breadth first, and many new friendships are shared by seed users.

Figure 6.5: Similarity of newly added friends to old friends with bioLDA used as the topic model to define profile vectors.

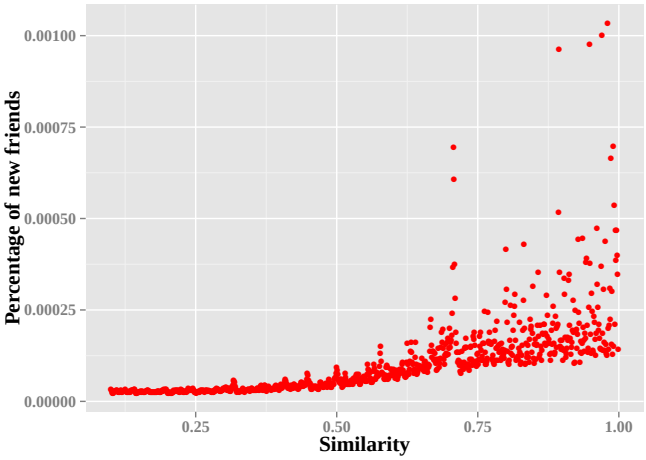
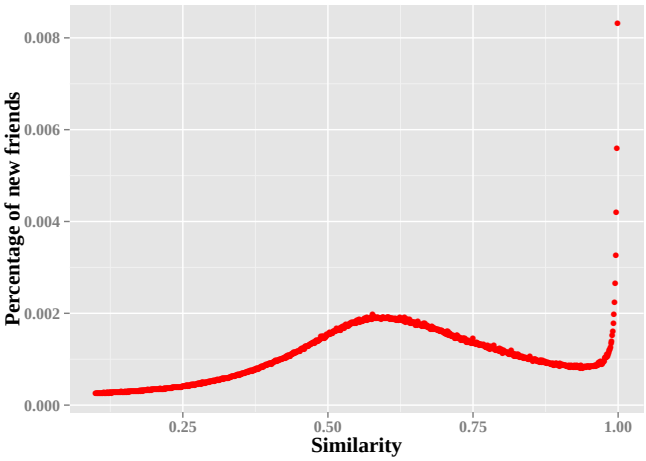
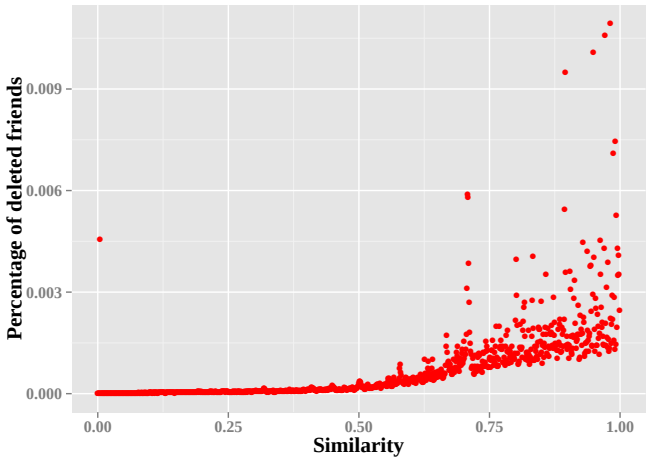


Figure 6.6: Similarity of newly added friends to old friends with tweetLDA used as the topic model to define profile vectors.



In order to better understand users' behavior on data consumption, we have also looked into the similarity distribution of deleted old friends. In Figures 6.7 and 6.8 we show these values.

Figure 6.7: Similarity of deleted friends to old friends with bioLDA used as the topic model to define profile vectors.



By comparing deleted old friends and added new friends, we see that new friends are more similar to old friends than deleted friends are, in both bioLDA and tweetLDA. This is an expected outcome, because as time progresses, users refine their data interests and conserve friendships that appeal to them (i.e., consuming more data from same friends), while deleting friends whose data are no longer interesting.

In finding individual anomalies, we experimented with different values of the similarity threshold parameter ξ. In Table 6.2 we show percentages of users and friendships that were labeled as anomalous for different ξ values. From the table, we see that for $\xi = 0.1$ only 1% of new friends are found to be anomalous. However, the percentage of users who have an anomalous friendship reaches up to 20% of all seed users. The percentage of users with an anomalous friendship rises quickly, and reaches 80% by $\xi = 0.5$. In contrast, the percentage of anomalous friendships rises slowly, and barely reaches 22% by a ξ value as high as 0.7.

From these results, it can be deduced that most of the new friends of seed users are similar to their old friends. However, most of the seed users become friends with at least one user who is not similar to old friends. In the next section, we will check these new anomalous friends for collective anomalies.

Collective Anomaly

In collective anomaly detection we are given a list of users and their new friendships which are individually anomalous. Our goal is to compare found anomalies with collective be-

Figure 6.8: Similarity of deleted friends to old friends with tweetLDA used as the topic model to define profile vectors.

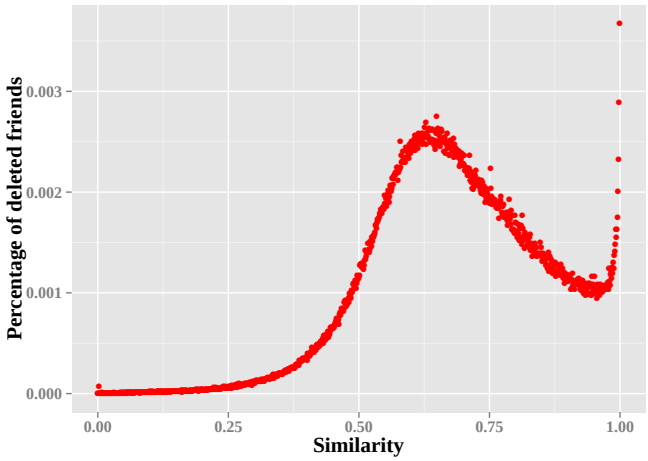


Table 6.2: Percentages of users and friendships that were labeled as anomalous for different ξ values.

ξ	% Users	% new friends judged as anomaly
0.1	0,20	0,01
0.2	0,40	0,01
0.3	0,60	0,02
0.4	0,76	0,04
0.5	0,82	0,08
0.6	0,83	0,15
0.7	0,83	0,22

havior and assign an anomaly score to seed users. To this end, we first cluster seed users in order to find their similar users. As we described in Section 6.3.2, finding similar users requires setting a total number of clusters for the k-means algorithm. For this number we experimented with k=100, 200, 500 and 1000. For k=500 we had the best trade off between cluster size and within-cluster similarity. In the rest of this chapter we will present our results for k=500.

After finding the cluster C_s of a seed user u_s, we compute the similarity of a seed user's anomalous friends to the old friends of users from the cluster C_s. For simplicity, in the rest of this chapter we will call these old friends of users from a seed user's cluster C_s as the seed user's extended friends.

Similar to individual anomaly results, value of the similarity threshold ξ determines the number of individual anomalies that are verified by collective anomaly. For small ξ values, individual anomalies are rarely labeled as collective anomalies because an extended friend's similarity to the anomalous friend can exceed or equal ξ. For example, from our experiments with tweetLDA results, none of the individual anomalies is labeled as a collective anomaly for $\xi = 0.1$. With increasing ξ values the number of collective anomalies also increases, because even from the extended friends set there will not be a friend whose similarity to the individual anomalous friend exceeds or equals ξ. In Figure 6.9, for different ξ values we give the number of individual anomaly friends who were also found to be collective anomalies.

Increasing the ξ value from 0.6 to 0.7, in Figures 6.9b and 6.9a, we see that even users with a big number of old friends are labeled as anomalous.

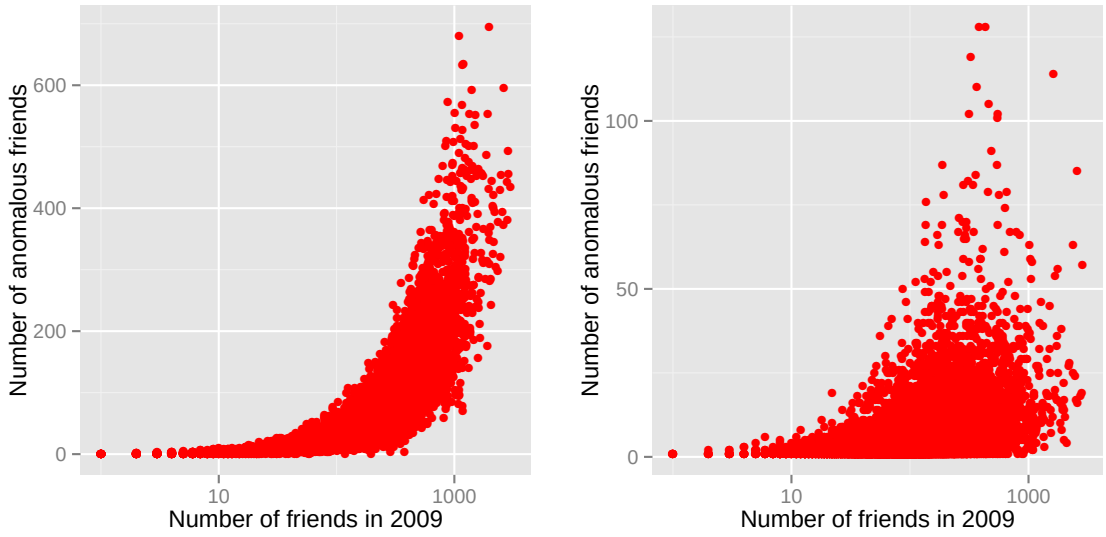
As we described in Section 6.3.3, we assume that established users with an already big number of friends are less likely to have anomalous friendships, because they should already be following users of topics whose data is interesting to them. In order to choose a value for ξ, we used this assumption and computed the median count of friends of anomalous users for different ξ values. These median values are 3, 6, 23, 62, 121 and 150 for ξ values of $0.2, 0.3, ..., 0.7$. From these values, we chose $\xi = 0.6$ which provides a median friend count of 121, because it provides the best trade off between median friend count, and number of found anomalies.

By using $\xi = 0.6$, we have detected 2334627 individual anomalous friendships and 79657 of these were also labeled as collective anomalies. In the next section we will give validation results for a subset of these verified anomalies.

6.4.5 Validation of Anomalies

We have employed a manual validation scheme for a subset of verified anomalies. The validation process entailed a web interface[6] with two panels. In the left panel, we listed the bio and last ten tweets of a user, as well as a summary of the accounts he/she currently follows. The summary consisted of 20 most frequent words, and their frequencies (e.g., actor[25]) from the followed accounts. In the right panel, a new friend was shown with

[6] `http://sight.dicom.uninsubria.it/anomaly/`



(a) Number of collective anomaly friends for a threshold similarity of 0.7

(b) Number of collective anomaly friends for a threshold similarity of 0.6

Figure 6.9: The relationship between number of collective anomaly friends and number of old friends for $\xi = 0.6$ and $\xi = 0.7$. An increasing ξ value results in a bigger number of anomalous friends.

his/her bio text and last ten tweets.

The validation process has been completed on the Amazon Mechanical Turk[7] by 245 users. Works of 183 users were accepted and used in the following experimental discussion. 62 validators' works were rejected because they either did not give proper explanations when labeling, or finished their tasks in very short times without properly analyzing the friendships.

Validators were given three labels to indicate whether a friendship between the user and the friend were anomalous, or not: *yes*, *no* and *possible* (i.e., not enough data to tell). In a separate text field, validators were instructed to give an explanation for their answers. Although the *possible* label does not require an elaborate explanation, we still solicited input so that the validators would not use this label to finish the experiment in a shorter time. From the accepted works, in average each explanation was 50.93 characters long.

We randomly chose 40 user clusters, and extracted a total of 1960 anomalies from them to be labeled by validators. In addition to anomalous friendships, we added 840 non-anomalous friendships to the list of user pairs to be labeled by users. Each friendship was given to five validators, so that aggregated labels would be based on decisions from multiple validators. Our validation process was divided into training and test phases. Each validator was trained by showing him/her five new friendships from a given cluster C_c (see Section 6.3.2). In the training phase, we aimed at familiarizing validators with the user interface, and help them learn expected user behavior (i.e., new friendship patterns). Afterwards, each validator was taken to the test phase where he/she was asked to label at most 45 new friendships from the same cluster C_c. The expected answer for the anomalous and non-anomalous friendships were *yes* and *no*, respectively.

We measured the consistency of labels given to friendships by validators and found a Fleiss' Kappa [46] value of $\kappa = 0.328$[8]. Fleiss' Kappa is a statistical measure that computes the reliability of agreement between validators when assigning the three labels to a number of items (i.e., friendships).

87% anomalous friendships have been given at least one *yes* label, whereas for 94% of them the number of possible and yes labels are bigger than the number of no labels. In 50% anomalous friendships, number of yes labels exceeds the number of no labels. Considering that friendships which are labeled with *possible* can be anomalous or non-anomalous, for 1960 asked anomalous friendships, the precision of our metrics lies in the interval $(\#yes/1960) \leq p \leq (\#yes + \#possible)/1960$, that is, $0.50 \leq p \leq 0.94$.

In terms of recall, 59 of the 840 non-anomalous friendships have been labeled as anomalies by a majority of the validators. Overall, the precision of our metrics can reach 0.87, which provides a good accuracy.

[7]Approved by the Office of Research Compliance - University of Texas at Dallas, human experiment IRB MR 13-231.

[8]For Fleiss' Kappa, > 0.2 Fair agreement, > 0.40 Moderate agreement, > 0.6 Substantial agreement

(a) Yes, anomaly (b) Possible, not enough data (c) No, not an anomaly

Figure 6.10: Wordclouds of explanations for validator given decisions to the friendships which were detected as anomalies by our methods.

6.5 Experimental Discussion

In this section we will analyze explanations of validators and evaluate the cases where our method mistakenly or correctly labels new friendships as anomalies. Furthermore, we will categorize user explanations, so that we can understand which types of new friendships are considered anomalies by human subjects. In Figure 6.10, we show wordclouds of explanations for each type of user decision.

In categorization, we parsed given explanations, so that explanations can be assigned to categories, and the accuracy of our methods can be analyzed for each category. Tables 6.3 and 6.4 show our categories, their frequencies for true and false positives and an examplary explanation for the considered category. In total we have empirically detected 47 categories, and here we show the most frequent ones.

Table 6.3 shows the results where human validators did not agree with the *anomaly* label given by our methods. The most common explanation is related to shared interests among user-friend pairs.

Table 6.3: Most frequent explanations for false positive anomalies.

Explanation Category	**Freq.**	**Representative explanation from the category**
Same interests	29.4%	Language difference, but both are interested in public transport and trains.
Same professions	16.6%	Both accounts are speakers and deal with the public.
Follows his/her interest	10.8%	Person likes geek and digital things, what he followed is a popular internet meme.
Same location	10.6%	Makes sense that a lawyer from Philade(l)phia would follow a local chef and restaurant owner.
Same backgrounds	6.0%	I know these people mentioned and they are real leftists. They are at least in solidarity.
Similar interests	4.6%	J***** is talking about windows 8 tech, and j***** is talking about developers.
Provides same service	3.9%	Both are Twitter accounts for companies on social media.
Similar professions	2.8%	A marketing strategist and creative director in media can be friends.
Same works	2.3%	Both are at Harvard Kennedy School. So chances of mixing up is high.
Follows celebrity	1.6%	I think this seems normal as L** could just be a fan.
Promotes himself	1.3%	L** is interested in social/media, from london, h***** seems well connected.

Table 6.4: Most frequent explanations for true positive anomalies.

Explanation Category	**Freq.**	**Representative explanation from the category**
Different interests	25.2%	E*** seems more interested in writing, nomad is more into marketing.
Different backgrounds	16.0%	One is stripper, and the other one is a working busy person
Different locations	11.1%	H*** is from Netherlands, J** visited and probably met H***. He is not a typical friend.
Different professions	10.3%	One is a poet and one is a coder.
Unlikely interest	9.0%	M***** is a businesswoman and is unlikely to be interested in cats and products for cats
Unrelated tweets	6.8%	Bio and tweets of both have nothing in common.
Follows bot	6.8%	O******* seems spammy. Not sure why D would be interested in them.
Different languages	5.1%	I**** A***'s tweets are in japanese, language barrier to friendship.
Unlike previous friends	3.0%	Doesn't seem (like) other followed accounts.
Follows comm. account	2.6%	Bright Eyes is an eye place. Like, not literally an eye, but a place specializing in eye stuff.

For these false positive anomalies, validators have given reasons ranging from "users are from the same location (i.e., India)" and "data engineer is interested in hacker" to "both users work at Apple Inc.". A closer look at these given reasons shows that users' backgrounds, such as being Indian or working at Apple Inc., can lead to friendships that cannot be explained by a data interest of user in the the friend's account.

These type of friendships are mostly due to shared location, hometown or work. In some cases, intrinsic characteristics of users can also lead to anomalous friendships. For example, we have noticed that a female user had many anomalous friendships, but there was an underlying theme in the newly followed friend accounts. These new friends were all influential women from different fields who were tweeting about a wide variety of topics.

Another type of anomaly that results from a lack of data interest is related to friendships which are initiated by anomalous friends. In these cases, an ad(vertisement) account starts following a personal user account, and expects the user to follow back. After a time period, users who have not reciprocated the friendship are unfollowed by the ad account. If the user reciprocates the friendship, our metrics identify these ad accounts as anomalous friends, because they tweet spam ads. Similarly hacked friend accounts are easily identified as anomalies once they start tweeting spams.

Our validation process shows that some user characteristics may lead to the creation of diverse friendships, however the main reason behind user friendships on Twitter is data interests. Thus, a model that is trained on limited information from users (e.g., last 10 tweets) can provide an accurate global view on Twitter. By analyzing this information, we can accurately predict which friendships are unexpected, and should be considered anomalies.

6.6 Conclusions

In this chapter, we outlined an approach to consider both user generated text data and network connections in detecting anomalous user behavior on Twitter. By this approach, a user's data interests on the social network is expected to comply with his/her past behavior and significant changes with respect to past behavior are treated as anomalies. Our results indicate that proposed anomaly detection models could be used to effectively analyze data consumption patterns in a Twitter graph.

By extending our model from detecting anomalies to understanding why friendships are formed in terms of shared topics, in the future we will study the evolution of the Twitter network according to topical interests. This approach can be useful in not only detecting, but also in explaining the long term user behavior on Twitter.

Chapter 7

Micro Analysis of User Interactions

7.1 Introduction

Recent years have seen an increasing number of successful online social networks catering to different social needs on the Internet. From professional social network LinkedIn to personal social network Facebook, online social networks are diversified to meet needs of a growing online population. As a result, an Internet user has several accounts on online social networks, where his/her activities are governed by different motivations.

Two of the most successful online social networks, Facebook and Twitter, have grown in recent years to accommodate millions of social network users, and hold millions of personal profiles for the same sets of users. Dynamics of these online social networks are affected by different factors; the directed nature of Twitter enables fast and efficient information propagation, whereas undirected Facebook is still meshed by close familial or regional networks, and users interact in more conservative ways.

Despite these differences, both Facebook and Twitter provide means (i.e., like and retweet, respectively) to make another user's post visible to a larger audience. Analyzing how retweets and likes are employed provides clues in understanding how to spread ideas, disseminate news and propagate influence on a social network.

In this chapter, we study the problem of how a post is liked/tweeted by social network users across Facebook and Twitter. Rather than external features such as the network structure or user information, we focus on tweets/posts themselves to understand what aspects of tweets/posts help them to get retweeted and liked. Moreover, we analyze how location information influences retweet, comment and like interactions.

To this end, we extract the following features from text based social network posts: named entities, lexical errors and sentiments. In named entity recognition, we locate atomic elements of seven categories: people, organizations, locations, date, time, percentage and money. By observing found entities, we show how the functionality (i.e., usage purpose) of online social networks can be mined.

In lexical analysis, we locate lexical errors of texts, and classify these errors into ten most frequently committed errors. An analysis of these lexical errors show that social network users make similar mistakes on both sites.

Sentiment analysis aims at studying user interactions for different sentiments that are found in social network posts. We investigate the impact of positive/negative/neutral sentiments on conversation patterns.

Furthermore, using location information from Twitter bios and Facebook profiles, we show that users' interaction patterns are heavily dependent on their current locations. Although online social networks have connected millions of users from all over the world, interaction patterns are still location limited, and governed by densely connected networks.

The chapter is organized as follows. Section 7.2 explains our data collection process on Facebook and Twitter. Section 7.3 explains our methodology, and discusses the software tools we used. In Section 7.4, we analyze the four dimensions we considered, whereas Section 7.5 shows how considered dimensions affect each other. Finally, Section 7.6 combines our insights with geolocation data, and presents our results in a visual way.

7.2 Data collection

Data sets of this chapter have been created by querying Twitter.com and Facebook.com APIs.

A Facebook application [1] was used to query Facebook for user data in an offline manner and store posts from the 2008-2013 period. Through the application, 75 users have given us permission to track data items (i.e., status posts, photos, videos, etc.) along with comments and likes these items receive from other Facebook users. This data crawling allowed us to track conversations of 670K Facebook users. Each considered data item has been posted on Facebook by users, their friends or friends of friends. Similarly, likes/comments on these data items belong to the users, their friends or friends of friends. Facebook status posts can contain pictures and videos. As we cannot analyze contents of these additional data items, we limit our analysis to Facebook comments that consist of textual data only. In what follows, we will refer to Facebook comments as Facebook texts.

Our Twitter dataset comes from a crawl between December 2012 and April 2013, and the considered tweets have been posted between 2008 and 2013. By using a Twitter application [2], we have stored bio information and last 10 tweets of 11M Twitter users.

In what follows, we will use the terms Twitter texts and tweets interchangeably. For geolocation analysis we used bio information on user profiles. Bio information for each user contains a current location field, where unstructured texts can be entered for city/country values, e.g., Boston, MA. We have used Google Geocode API [3] to convert these location texts into longitude-latitude values.

[1] `developers.facebook.com`

[2] `dev.twitter.com`

[3] `developers.google.com/maps/documentation/geocoding/`

7.3 Methodology and tools

In order to understand what factors improve the chances of a text getting liked/retweeted by other users, we have considered the following dimensions: named entity recognition, lexical and sentiment analyses. In what follows, we will discuss our methodology for mining each of these dimensions.

In given sentences, named entity recognition (NER) [100] locates atomic elements of predefined categories such as names of people, cities, locations, time mentions, and money values. For entity recognition, we used Apache OpenNLP [4] that allows training its classifier with seven different categories. We have used the categories "people, organizations, locations, date, time, percentage and money". This tool has a precision of 0.8 and a recall of 0.74.

Lexical errors are grammar and spelling mistakes/typos found in sentences. These are noun-verb agreement errors, missing words, extra words, wrong words, confusion of similar words, wrong word order, comma errors, and whitespace errors. For Facebook and Twitter texts, we used LanguageTool[5]
to find lexical errors. These errors are predefined text patterns defined in an XML file. The tool reaches a precision of 93% on the English texts.

Sentiment analysis aims at extracting the general sentiment from a given sentence, identifying whether it expresses a positive, negative or neutral emotion. For this purpose, we used the *Sentiment140 API*, a Sentiment Analysis tool developed by Go et al. [50], which is a recent and widely used tool. The tool is based on a machine learning algorithm for classifying the sentiment of Twitter messages using distant supervision. The training data consists of Twitter messages with emoticons (i.e., pictorial representation of a facial expression), which are used as noisy labels. This type of training data is abundantly available and can be obtained through automated means. The underlying idea is to use emoticons to learn which words co-appear with emoticons, and use this information in machine learning algorithms, such as naive bayes and support vector machines. They show that the tool has an accuracy above 80% when trained with emoticon data [50].

7.4 Mining Dimensions

In mining the considered dimensions, we have focused on English language texts. To this end, we have used the Ldig library in Python [6], which has a precision of 99.1% in detecting the English language.

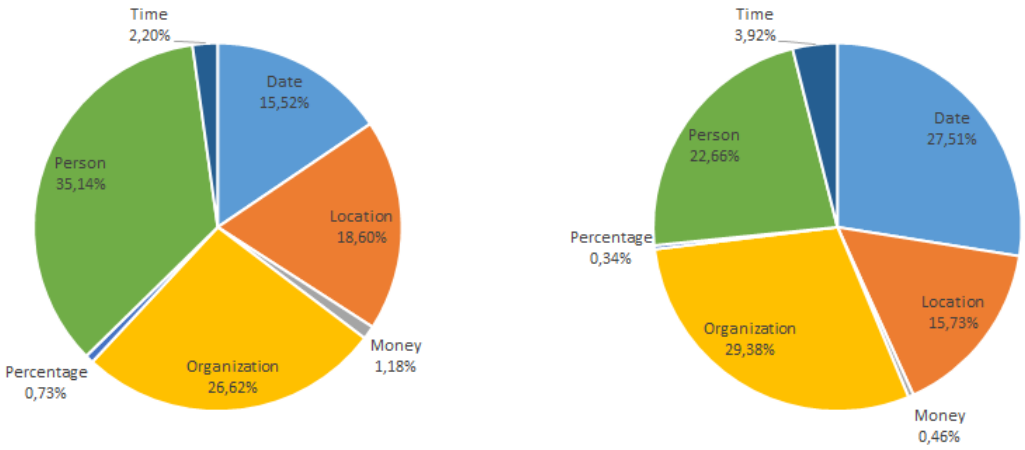
For Facebook and Twitter datasets, counts of English language texts and dimensional statistics are given in Table 7.1. In columns named entity, error and sentiment, we give the percentages of English texts which contain at least one named entity, lexical error and sentiment tag, respectively. In the following sections, these values will be explained in

[4] `http://opennlp.apache.org/`
[5] `http://www.languagetool.org`
[6] `https://github.com/shuyo/ldig`

Table 7.1: Dimensional statistics and counts of English texts. Entitity, error and sentiment values are given in percentages.

	Count	Entity	Error	Sentiment
Twitter	10.6M	29.8%	81%	32%
Facebook	1M	13%	69%	22%

detail.



(a) Facebook entities.

(b) Twitter entities.

Figure 7.1: Entities and their percentage in Facebook and Twitter posts.

7.4.1 Entity Recognition

In Table 7.1, percentage of English texts which contain at least one named entity are 29.8% and 13% for Twitter and Facebook texts, respectively. Figure 7.1 further details the composition of entities in these texts. Along seven different categories, we see that on Facebook, Persons are mentioned in texts 35.14% of the time, whereas this value is 22.66% for tweets. Date and time entities are mentioned twice as much on Twitter, but organizations have similar percentage values. We attribute the difference in date/time values to Twitter users' high mobility (i.e., mobile phone usage) compared to other social network users [76]. Because of this mobility, tweets are more related to events happening in real time. For example, 1.5% of tweets contain the word today, whereas this value is only 0.02% for Facebook texts. In time entities, we found a similar pattern with tonight appearing in 0.06% of tweets and 0.0012% of Facebook texts, respectively.

As seen on Figures 7.2 and 7.4, some organizations are frequently mentioned on both Twitter and Facebook, but users on Twitter are more likely to mention a broader variety of organizations, ranging from commercial brands to news agencies. In contrast, the two

most frequently mentioned organizations on Facebook are related to basketball, and other organizations are not very frequently mentioned.

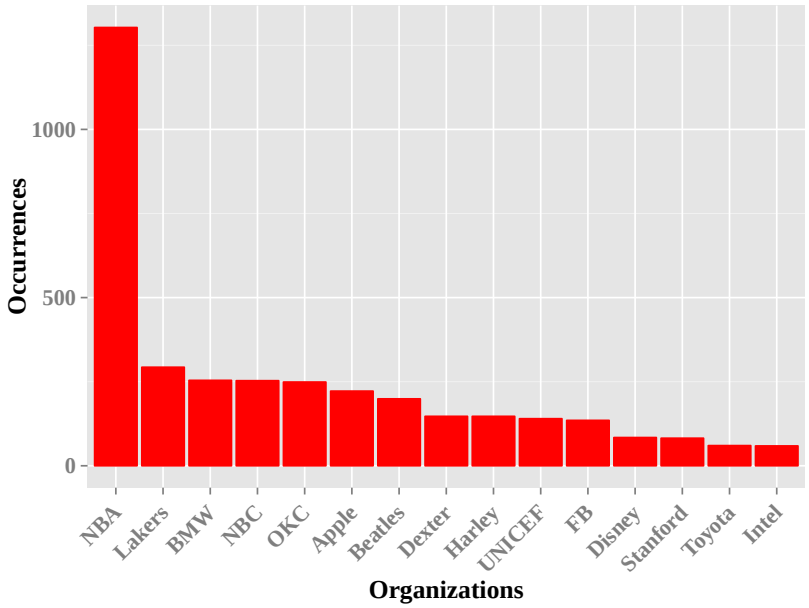


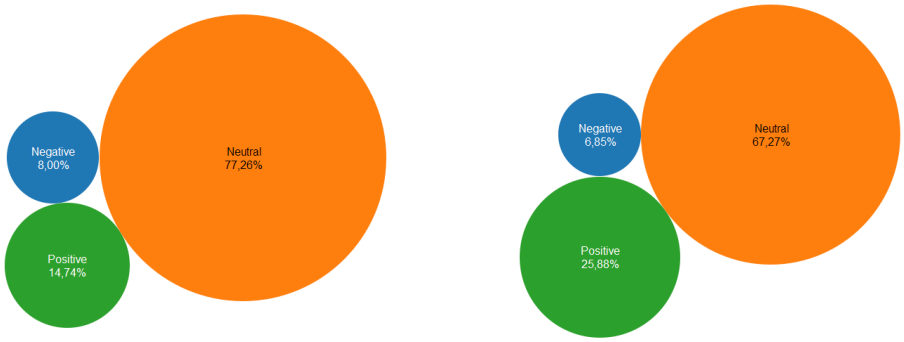
Figure 7.2: Most frequently mentioned organizations on Facebook.

7.4.2 Sentiment Analysis

We have found positive sentiment to be more common than negative sentiment on both Facebook and Twitter. Percentages of posts with negative, positive and neutral sentiment tags are shown in Figure 7.3. Although Facebook friends are more likely to be real life acquaintances than Twitter followers, and Facebook posts are more likely to be directed to a private audience, only 22% of Facebook posts contain a sentiment. On Twitter, tweets are public, but 32% of tweets contain a sentiment, and 14% are positive sentiment tweets. Negative sentiments are expressed less often on both sites, with 8% and 6.85% on Facebook and Twitter, respectively. In Tables 7.2 and 7.3 we show some post examples with neutral, positive and negative sentiments.

As posts on both sites are short texts, sentiments that are expressed in posts are mostly dependent on a limited number of words. For example, in Table 7.3, the first negative tweet has its sentiment expressed by the word 'lonely'. However, the sentiment tool performs well in labeling sentiments when texts include profanity or other swear words. The following tweets are labeled as:

- **Negative**: i dont care what others think any more so if you dont like me suck it up and keep your mouth shut or go f*** off. yes im in a bad mood nite



(a) Facebook sentiment (b) Twitter sentiment

Figure 7.3: Percentage of sentiments in Facebook and Twitter posts.

- **Positive**: here's to me actually making some f****** money!!!!!!!!! whoo hoo!!!!!!!!!!!!!!
- **Neutral**: F*** I, we all do.

Table 7.2: Examples of Facebook posts with sentiments.

Positive	I love this kid!!
	I like Cinebistro too! Cant wait to see the Artist
	Thanks AJ and SW crew for bringing both you and Thursday back to our shores!!
Negative	No and no!!!
	I dont think it worth it, it doesn't give back as I look at it, its ugly and... why do they call it celtic?
	Of course it is. Their negative attacks on each other alone are bringing some nasty skeletons out of the closet that will hurt whichever Republican becomes the nominee in November.
Neutral	I never heard Shaolin monks went to Chinese university or teach some class in school, so they teach Chinese culture more than kungfu
	Im waiting on a special phone calll...cake up time
	They have a show in new Orleans on march 29

We give the precision and recall values for sentiment detection in Table 7.4. Overall, precision values for both online social networks are high, whereas the lowest values are obtained for positive recall for Facebook posts (54%) and negative recall for tweets and posts (63% and 64%, respectively). These values have been obtained by asking to a group of three validators to assign a sentiment to messages, given a sample composed by 200

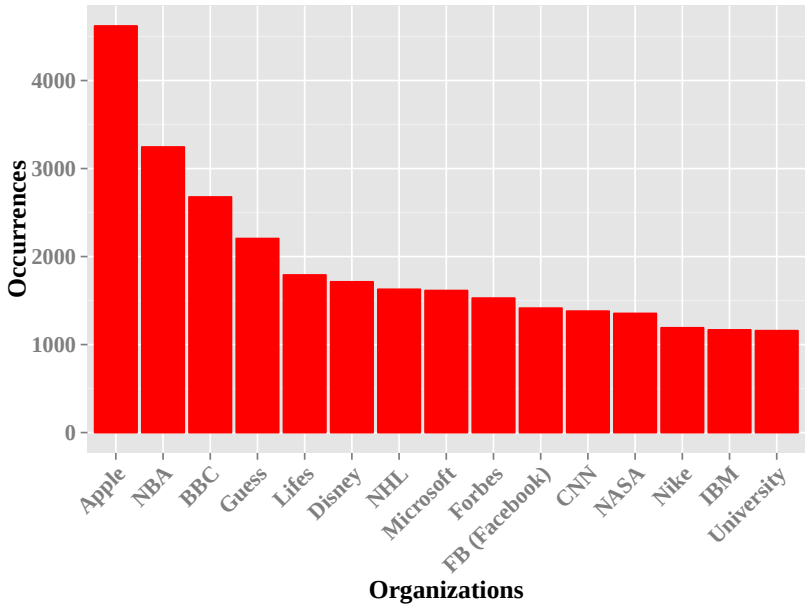
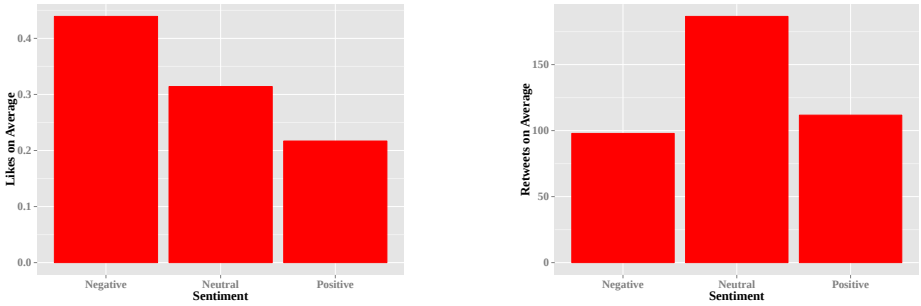


Figure 7.4: Most frequently mentioned organizations on Twitter.

posts and 200 tweets. Finally, we have computed the average values of precision and recall obtained from each validator.



(a) Likes for negative and positive comments.(b) Retweets for negative and positive tweets.

Figure 7.5: The impact of sentiments on like and retweet counts.

7.4.3 Lexical Analysis

In lexical analysis, we locate spelling or grammar mistakes within individual sentences of a text. To this end, we have replaced @usernames and #hashtags on Twitter with generic words before running the lexical analysis tool, so that site specific features (e.g., hashtag usage) are stripped and the tool can parse sentences without errors. For example, 'Ask

Table 7.3: Examples of tweets with sentiments.

Positive	watching a snuff movie, so funny, I love these things
	hey Laura thanks for the invite. I am eating grapes.
	Good Morning and good Ester to everyone from Turin (Italy)!!!
Neutral	Signing up for twitter
	Catching up with online things...
	Eating an apple
Negative	so lonely =(
	lonely days.....when will these lonely days leave me?
	is bored beyond belief

Table 7.4: Precision and recall values for Twitter and Facebook sentiments. + and - signs refer to positive and negative sentiments, whereas P. and R. refer to precision and recall, respectively.

	+P.	+R.	-P.	-R.	Neut. P.	Neut. R.
Twitter	88%	83%	79%	63%	87%	79%
Facebook	81%	54%	82%	64%	75%	82%

@user about #ny' becomes 'Ask William about New York'. With this transformation, Table 7.1 shows that 81% and 69% of Twitter and Facebook texts contain at least one lexical error.

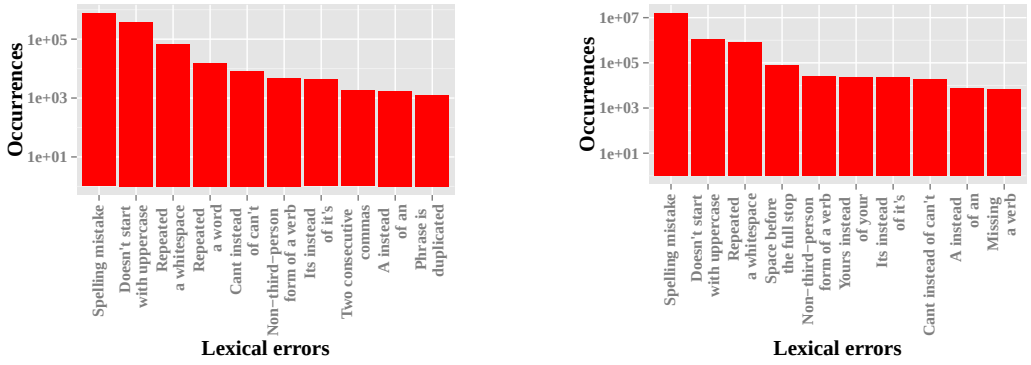
These errors are better analyzed in Figures 7.6a and 7.6b. An interesting error that appears on Twitter but not on Facebook is the absence of a proper verb in a sentence. Overall, errors on both online social networks are very similar; word spelling mistakes and absence of uppercase letters in the beginning of sentences are major errors.

7.5 Dimension interplay

An interesting part of mining conversational user interactions is finding how different dimensions affect each other or retweet/like counts individually. In this section, we will look at dimension correlations and analyze how these interplays affect likes/retweets of user texts on Facebook and Twitter. In particular, we will analyze the interplay between: i) retweet/like counts and sentiment, ii) sentiment and lexical errors, and iii) lexical errors and likes/retweets.

Figure 7.5 shows the impact of sentiments on retweet and like counts. In the figure, we see that negative texts receive more likes on average on Facebook, whereas on Twitter neutral tweets are retweeted more. Despite similar retweet counts, positive tweets are retweeted more often than the negative tweets.

We explain the high like count of negative Facebook posts by expressions of user com-



(a) Most common lexical errors on Facebook. (b) Most common lexical errors on Twitter.

Figure 7.6: Lexical errors on social networks.

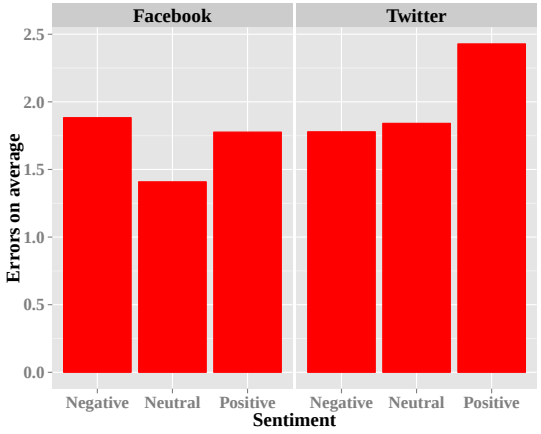


Figure 7.7: Average lexical errors by sentiments in Facebook and Twitter posts.

passion. Facebook posts, such as *"Lexi better not die tonight. :("* (96 likes), receive higher like counts. On Twitter neutral tweets, such as news, are retweeted by many users.

Mining both sentiments and lexical errors allows us to see how user emotions lead to lexical errors due to stress, anger or sadness. Figure 7.7 shows average lexical errors for positive and negative sentiment texts, for both Facebook and Twitter. Positive tweets have been found to contain more than 2 errors, whereas negative Facebook comments contain 1.8 errors in average.

In Tables 7.5 and 7.6 we show some text examples with neutral, positive and negative sentiments containing errors.

Lexical errors can also affect the number of times a text is retweeted/liked by other users. In Figures 7.8a and 7.8b we show the average value of likes/retweets a text receives for different values of lexical errors. Although there are as many as 15 lexical errors in

Table 7.5: Examples of Facebook posts with sentiments containing errors.

Positive	strawberries would be the best.. oh my how good... and make a chesecake from this... OH yes...
Negative	Cant believe I pay tuition money for this stuff...;-(
Neutral	Dear NBC: Please make your videos viewable in other countries outside America. Sincerly, - The rest of the world

Table 7.6: Examples of tweets with sentiments containing errors.

Positive	Oh twitter! Me loves me loves! It'll lead me to my preciousssssss!
Negative	write the wrong words for the wrong thing and make it worse and worse. wahahha
Neutral	looking at a little tiger cat who says hes in my server doing maintenance

both figures, number of posts with more than 5 errors are very low on both online social networks. An example with 15 errors is the tweet *omg omg omg omg omg omg omg 7 days too goo and i shall be in greece :) also my heads f***** stupid boy y does he always do this 2 me :(x*

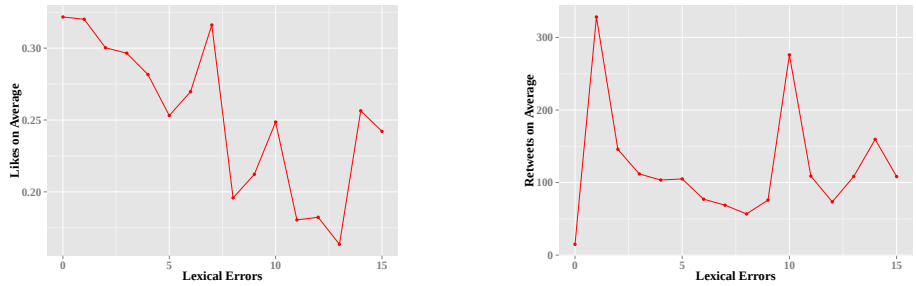
Especially on Figure 7.8a we see that an increasing number of lexical errors leads to lower like counts on Facebook. Figure 7.8b shows a similars pattern for Twitter posts, but in this case retweet counts decrease with increasing numbers of lexical errors after the first error.

7.6 Influence of Geolocation on Conversations

Although some research work [59, 101, 121] have worked on topical conversations on Twitter, a comparative analysis of locations of social network users across multiple online social networks has not been studied yet.

In order to analyze the impact of geolocations, we converted textual current locations of Twitter users into geographical longitude-latitude values. From these values, Figure 7.9 shows the current location of Twitter users on a world map. On this map, red points and green points correspond to users who post a tweet and retweet a posted tweet, respectively. Edges between these two types of users connect them on the map. From the map, we see that a large percentage of retweeted tweets come from the east coast of USA and north Europe. Similarly, most edges are created between these two parts of the world. The absence of China and most of the Russian territories are prominent features on the map. The high concentration of Canadian cities around the northern border of USA is also visible from the map.

Another presentation of this location data allows us to measure the distance between two users in miles. A zero distance shows that a user u who retweeted a tweet from user x



(a) Average like counts by number of lexical errors.

(b) Average retweet counts by number of lexical errors.

Figure 7.8: Lexical errors and interactions.

lives in the city where user x currently lives. By plotting the number of such user pairs for each distinct distance value [7] yields Figure 7.10. In the figure, we see that a big percentage of user pairs have zero distance between them. Numbers of user pairs are shown to decrease with increasing distances. An early anomaly in this trend is the low number of user pairs around 2500 miles. This distance corresponds to the separation between USA and Europe.

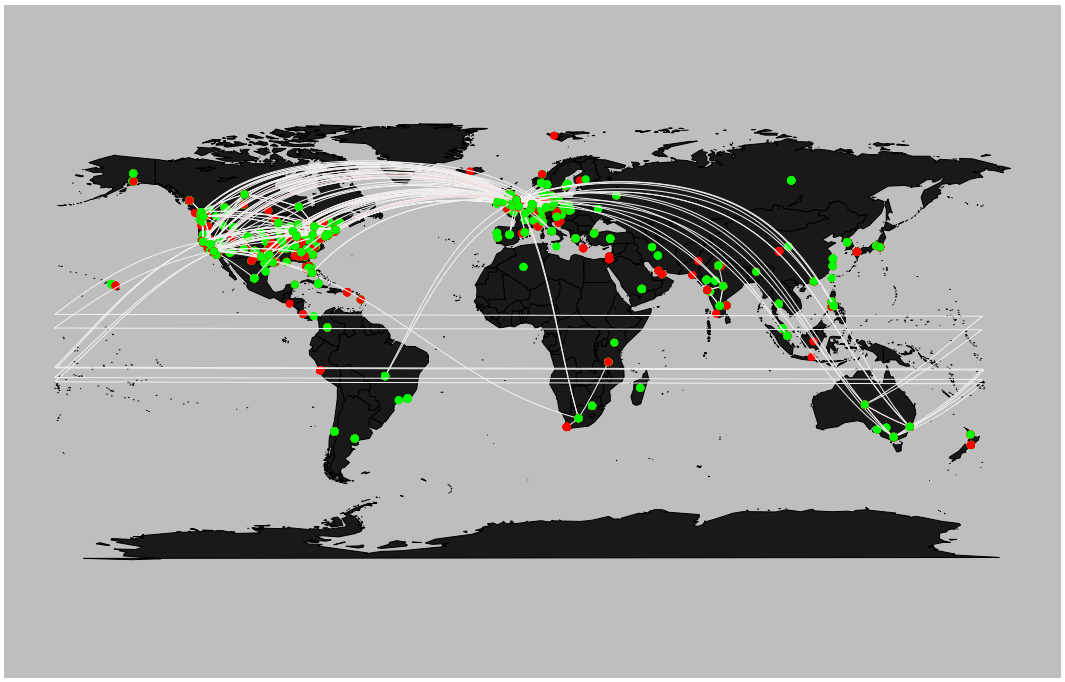


Figure 7.9: Locations of Twitter users. Edges show retweet behavior among Twitter users.

Unlike Twitter, Facebook privacy settings are very strict, and users are less willing to

[7] We have put distances into 100 mile buckets.

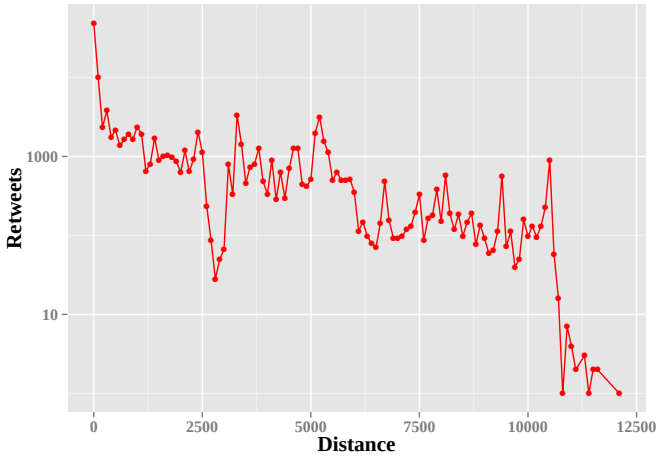


Figure 7.10: In miles, distances between user pairs who retweet each other's tweets.

share their location information. Due to this shortcoming, we could not repeat the distance experiment on the Facebook dataset. For a similar distance notion on Facebook dataset, we have used the locale [8] field to analyze user proximity. We explain this choice with the empirical observation that users from a country mainly use the official language of the country in the Facebook interface. The main exception to this observation is that English is also widely used by other nationalities.

Figure 7.11 shows locale pairs for Facebook users. In the figure, a locale value l is connected by an edge to another locale m if users from l constitute 30% or more of users who have liked Facebook comments of users from the locale m. Although EN_US is connected to a big portion of other locales, most locales have self loops, or they are connected to a small number of other locales. Another relevant feature of the figure is the community of Latin languages connected together, such as it_IT, es_ES, fr_FR, etc.; users of Latin languages interact more often.

In the previous sections, we have shown how different dimensions of user generated texts affect how many times a particular tweet/comment will be liked or commented. Regardless of these numbers, our geolocation experiments show that users who like/comment a text are more likely to live closer to the owner of the text.

[8] Locale is the user chosen interface language for Facebook.com, such as EN_US (English in USA), EN_GB (English in Great Britain)

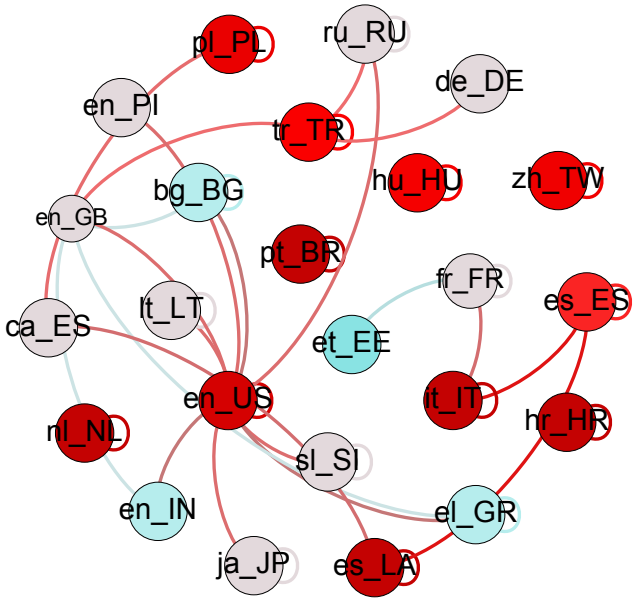


Figure 7.11: Like interactions among Facebook users of different locales.

Bibliography

[1] Fabian Abel, Qi Gao, Geert-Jan Houben, and Ke Tao. Analyzing user modeling on twitter for personalized news recommendations. In *User Modeling, Adaption and Personalization*, pages 1–12. Springer, 2011.

[2] A. Acquisti and R. Gross. Imagined communities: Awareness, information sharing, and privacy on the Facebook. In *Privacy Enhancing Technologies*, pages 36–58. Springer, 2006.

[3] L.A. Adamic and E. Adar. Friends and neighbors on the web. *Social networks*, 25(3):211–230, 2003.

[4] Apoorv Agarwal, Boyi Xie, Ilia Vovsha, Owen Rambow, and Rebecca Passonneau. Sentiment analysis of twitter data. In *Proceedings of the Workshop on Languages in Social Media*, pages 30–38. Association for Computational Linguistics, 2011.

[5] Waqar Ahmad and Asim Riaz. Predicting friendship levels in online social networks. Master's thesis, Blekinge Institute of Technology, 2010.

[6] C.G. Akcora, B. Carminati, and E. Ferrari. Network and profile based measures for user similarities on social networks. In *Information Reuse and Integration (IRI), 2011 IEEE International Conference on*, pages 292–298. IEEE, 2011.

[7] C.G. Akcora, B. Carminati, and E. Ferrari. Privacy in social networks: How risky is your social graph? In *Data Engineering (ICDE), 2012 IEEE 28th International Conference on*, pages 9–19. IEEE, 2012.

[8] Cuneyt Gurcan Akcora, Barbara Carminati, and Elena Ferrari. Risks of friendships on social networks. In *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, pages 810–815. IEEE, 2012.

[9] Cuneyt Gurcan Akcora, Barbara Carminati, and Elena Ferrari. User similarities on social networks. *Social Network Analysis and Mining*, 3:1–21, 2013.

[10] Cuneyt Gurcan Akcora and Elena Ferrari. Graphical user interfaces for privacy settings. In *Encyclopedia of Social Network Analysis and Mining*. Springer, 2014.

[11] Cuneyt Gurcan Akcora and Elena Ferrari. Similarity metrics on social networks. In *Encyclopedia of Social Network Analysis and Mining*. Springer, 2014.

[12] Pramod Anantharam and Amit Sheth. Topical anomaly detection from twitter stream. *In the Proceedings of ACM Web Science 2012*, pages 11–14, 2012.

[13] L. Banks and S.F. Wu. All friends are not created equal: An interaction intensity based approach to privacy in online social networks. In *Computational Science and Engineering, 2009. CSE'09. International Conference on*, volume 4, pages 970–974. IEEE, 2009.

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

[15] Michael Beye, Arjan Jeckmans, Zekeriya Erkin, Pieter Hartel, Reginald Lagendijk, and Qiang Tang. Literature overview - privacy in online social networks, October 2010.

[16] P. Bhattacharyya, A. Garg, and S.F. Wu. Analysis of user keyword similarity in online social networks. *Social network analysis and mining*, 1:1–16, 2010.

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

[18] Johan Bollen, Huina Mao, and Xiaojun Zeng. Twitter mood predicts the stock market. *Journal of Computational Science*, 2(1):1–8, 2011.

[19] J. Bonneau, J. Anderson, and L. Church. Privacy suites: Shared privacy for social networks. In *Symposium on Usable Privacy and Security (SOUPS)*, 2009.

[20] J. Bonneau, J. Anderson, and G. Danezis. Prying data out of a social network. In *First International Conference on Advances in Social Networks Analysis and Mining*. Citeseer, 2009.

[21] S. Boriah, V. Chandola, and V. Kumar. Similarity measures for categorical data: A comparative evaluation. *SIAM*, 30(2):243–254, 2008.

[22] G. Bouma. Normalized (pointwise) mutual information in collocation extraction. *Proceedings of GSCL Conference*, pages 31–40, 2009.

[23] Danah M Boyd and Eszter Hargittai. Facebook privacy settings: Who cares? *First Monday*, 15(8):23, 2010.

[24] D.M. Boyd and N.B. Ellison. Social network sites: Definition, history, and scholarship. *Journal of Computer-Mediated Communication*, 13(1):210–230, 2008.

[25] B. Bringmann, M. Berlingerio, F. Bonchi, and A. Gionis. Learning and Predicting the Evolution of Social Networks. *Intelligent Systems, IEEE*, 25(4):26–35, 2010.

[26] Piotr Brodka, Stanislaw Saganowski, and Przemyslaw Kazienko. Ged: the method for group evolution discovery in social networks. *Social Network Analysis and Mining*, 1:1–14, 2012. 10.1007/s13278-012-0058-8.

[27] Kenneth L Calvert, Matthew B Doar, and Ellen W Zegura. Modeling internet topology. *Communications Magazine, IEEE*, 35(6):160–163, 1997.

[28] B. Carminati, E. Ferrari, S. Morasca, and D. Taibi. A probability-based approach to modeling the risk of unauthorized propagation of information in on-line social networks. In *Proceedings of the first ACM conference on Data and application security and privacy*, pages 51–62. ACM, 2011.

[29] B. Carminati, E. Ferrari, and A. Perego. Enforcing access control in web-based social networks. *ACM Transactions on Information and System Security*, 13(1):1–38, 2009.

[30] Barbara Carminati and Elena Ferrari. Privacy-aware access control in social networks: Issues and solutions. In Xindong Wu, Jordi Nin, and Javier Herranz, editors, *Privacy and Anonymity in Information Management Systems*, Advanced Information and Knowledge Processing, pages 181–195. Springer London, 2010.

[31] Mario Cataldi, Luigi Di Caro, and Claudio Schifanella. Emerging topic detection on twitter based on temporal and social terms evaluation. In *Proceedings of the Tenth International Workshop on Multimedia Data Mining*, page 4. ACM, 2010.

[32] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Computing Surveys (CSUR)*, 41(3):15, 2009.

[33] X. Cheng, C. Dale, and J. Liu. Statistics and social network of youtube videos. In *Quality of Service, 2008. IWQoS 2008. 16th International Workshop on*, pages 229–238. IEEE, 2008.

[34] X. Cheng and J. Liu. Nettube: Exploring social networks for peer-to-peer short video sharing. In *INFOCOM 2009, IEEE*, pages 1152–1160. IEEE, 2009.

[35] Zi Chu, Steven Gianvecchio, Haining Wang, and Sushil Jajodia. Who is tweeting on twitter: human, bot, or cyborg? In *Proceedings of the 26th Annual Computer Security Applications Conference*, pages 21–30. ACM, 2010.

[36] Thomas M. Cover and Joy A. Thomas. *Elements of information theory*. Wiley-Interscience, New York, NY, USA, 1991.

[37] M. Cristani and R. Cuel. A survey on ontology creation methodologies. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 1(2):49–69, 2005.

[38] W. Cukierski, B. Hamner, and B. Yang. Graph-based features for supervised link prediction. In *Neural Networks (IJCNN), The 2011 International Joint Conference on*, pages 1237–1244. IEEE, 2011.

[39] G. Danezis. Inferring privacy policies for social networking services. In *Proceedings of the 2nd ACM workshop on Security and artificial intelligence*, pages 5–10. ACM, 2009.

[40] Mukund Deshpande and George Karypis. Item-based top-n recommendation algorithms. *ACM Trans. Inf. Syst.*, 22:143–177, January 2004.

[41] A. Dubrawski, P. Sarkar, and L. Chen. Trade-offs between agility and reliability of predictions in dynamic social networks used to model risk of microbial contamination of food. In *Social Network Analysis and Mining, 2009. ASONAM'09. International Conference on Advances in*, pages 125–130. IEEE, 2009.

[42] N.B. Ellison et al. Social network sites: Definition, history, and scholarship. *Journal of Computer-Mediated Communication*, 13(1):210–230, 2007.

[43] N.B. Ellison, C. Steinfield, and C. Lampe. The benefits of facebook "friends:" social capital and college students' use of online social network sites. *Journal of Computer-Mediated Communication*, 12(4):1143–1168, 2007.

[44] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On power-law relationships of the internet topology. In *ACM SIGCOMM Computer Communication Review*, volume 29, pages 251–262. ACM, 1999.

[45] L. Fang and K. LeFevre. Privacy wizards for social networking sites. In *Proceedings of the 19th international conference on World wide web*, pages 351–360. ACM, 2010.

[46] Joseph L Fleiss, Bruce Levin, and Myunghee Cho Paik. The measurement of interrater agreement. *Statistical methods for rates and proportions*, 2:212–236, 1981.

[47] J. Fogel and E. Nehmad. Internet social network communities: Risk taking, trust, and privacy concerns. *Computers in Human Behavior*, 25(1):153–160, 2009.

[48] T.M.J. Fruchterman and E.M. Reingold. Graph drawing by force-directed placement. *Software: Practice and experience*, 21(11):1129–1164, 1991.

[49] G. Gan, C. Ma, and J. Wu. Data clustering: theory, algorithms, and applications. *ASASIAM Series on Statistics and Applied Probability*, 20:219–230, 2007.

[50] Alec Go, Richa Bhayani, and Lei Huang. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, pages 1–12, 2009.

[51] Jennifer Golbeck and Matthew Rothstein. Linking social networks on the web with foaf: A semantic web case study. In *AAAI*, volume 8, pages 1138–1143, 2008.

[52] K.K. Gollu, S. Saroiu, and A. Wolman. A social networking-based access control scheme for personal content. In *Proceedings of the 21st ACM Symposium on Operating Systems Principles (SOSP2007)-Work-in-Progress Session*, 2007.

[53] R. Guha, Ravi Kumar, Prabhakar Raghavan, and Andrew Tomkins. Propagation of trust and distrust. In *Proceedings of the 13th international conference on World Wide Web*, WWW '04, pages 403–412, New York, NY, USA, 2004. ACM.

[54] V. Ha and P. Haddawy. Similarity of personal preferences: theoretical foundations and empirical analysis. *Artificial Intelligence*, 146(2):149–173, 2003.

[55] Jiawei Han, Micheline Kamber, and Jian Pei. *Data mining: concepts and techniques*. Morgan kaufmann, 2006.

[56] Z. He, X. Xu, and S. Deng. Squeezer: an efficient algorithm for clustering categorical data. *Journal of Computer Science and Technology*, 17(5):611–624, 2002.

[57] C Honey and Susan C Herring. Beyond microblogging: Conversation and collaboration via twitter. In *System Sciences, 2009. HICSS'09. 42nd Hawaii International Conference on*, pages 1–10. IEEE, 2009.

[58] L. Hong and B.D. Davison. Empirical study of topic modeling in twitter. In *Proceedings of the First Workshop on Social Media Analytics*, pages 80–88. ACM, 2010.

[59] Akshay Java, Xiaodan Song, Tim Finin, and Belle Tseng. Why we twitter: understanding microblogging usage and communities. In *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis*, pages 56–65. ACM, 2007.

[60] B. Jiang. On the least-squares method. *Computer methods in applied mechanics and engineering*, 152(1-2):239–257, 1998.

[61] L. Jin, H. Takabi, and J.B.D. Joshi. Towards active detection of identity clone attacks on online social networks. In *Proceedings of the first ACM conference on Data and application security and privacy*, pages 27–38. ACM, 2011.

[62] A.N. Joinson. Looking at, looking up or keeping up with people?: motives and use of facebook. In *Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 1027–1036. ACM, 2008.

[63] J. Jung and J. Euzenat. Towards semantic social networks. *The Semantic Web: Research and Applications*, 1:267–280, 2007.

[64] Cuneyt Gurcan Akcora & Barbara Carminati & Elena Ferrari & Murat Kantarcioglu. Detecting anomalies in social network data consumption. Under submission.

[65] L. Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, 1953.

[66] David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146. ACM, 2003.

[67] Y. Koren. Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 4(1):1, 2010.

[68] G. Kossinets and D.J. Watts. Empirical analysis of an evolving social network. *Science*, 311(5757):88, 2006.

[69] SB Kotsiantis, ID Zaharakis, and PE Pintelas. Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering: real word AI systems with applications in eHealth, HCI, information retrieval and pervasive technologies*, 160:3, 2007.

[70] Efthymios Kouloumpis, Theresa Wilson, and Johanna Moore. Twitter sentiment analysis: The good the bad and the omg. In *Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media*, pages 538–541, 2011.

[71] B. Krishnamurthy and C.E. Wills. On the leakage of personally identifiable information via online social networks. *ACM, Computer Communication Review*, 40(1):112–117, 2010.

[72] Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th international conference on World wide web*, pages 591–600. ACM, 2010.

[73] Sebastian Labitzke, Florian Werling, Jens Mittag, and Hannes Hartenstein. Do online social network friends still threaten my privacy? In *Proceedings of the third ACM conference on Data and application security and privacy*, pages 13–24. ACM, 2013.

[74] Sangho Lee and Jong Kim. Warningbird: Detecting suspicious urls in twitter stream. In *Symposium on Network and Distributed System Security (NDSS)*, 2012.

[75] Kalev Leetaru, Shaowen Wang, Guofeng Cao, Anand Padmanabhan, and Eric Shook. Mapping the global twitter heartbeat: The geography of twitter. *First Monday*, 18(5):1–15, 2013.

[76] Amanda Lenhart and Susannah Fox. *Twitter and status updating*. Pew Internet & American Life Project Washington DC, 2009.

[77] Jure Leskovec, Daniel Huttenlocher, and Jon Kleinberg. Predicting positive and negative links in online social networks. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 641–650, New York, NY, USA, 2010. ACM.

[78] Sheen S Levine and Robert Kurzban. Explaining clustering in social networks: Towards an evolutionary theory of cascading benefits. *Managerial and Decision Economics*, 27(2-3):173–187, 2006.

[79] Kevin Lewis, Jason Kaufman, Marco Gonzalez, Andreas Wimmer, and Nicholas Christakis. Tastes, ties, and time: A new social network dataset using facebook. com. *Social networks*, 30(4):330–342, 2008.

[80] D. Liben-Nowell and J. Kleinberg. The link-prediction problem for social networks. *Journal of the American Society for Information Science and Technology*, 58(7):1019–1031, 2007.

[81] D. Lin. An information-theoretic definition of similarity. In *Proceedings of the 15th international conference on Machine Learning*, volume 1, pages 296–304. San Francisco, 1998.

[82] J. Lindamood, R. Heatherly, M. Kantarcioglu, and B. Thuraisingham. Inferring private information using social network data. In *Proceedings of the 18th WWW*, pages 1145–1146. ACM, 2009.

[83] H.R. Lipford, A. Besmer, and J. Watson. Understanding privacy settings in facebook with an audience view. In *Proceedings of the 1st Conference on Usability, Psychology, and Security*, pages 1–8. USENIX Association Berkeley, CA, USA, 2008.

[84] K. Liu and E. Terzi. A framework for computing the privacy scores of users in online social networks. In *Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on*, pages 288–297. IEEE, 2009.

[85] William Lucia, Cuneyt Gurcan Akcora, and Elena Ferrari. Multi-dimensional conversation analysis across online social networks. In *Social Computing and its applications (SCA), 2013 Third International Conference on*, pages 331–336. IEEE, 2013.

[86] D.J.C. MacKay. *Information theory, inference, and learning algorithms*. Cambridge Univ Pr, 2003.

[87] M. Madejski, M. Johnson, and S. Bellovin. The failure of online social network privacy settings. *Dept. of Computer Science, Columbia University, Tech. Rep. CUCS-010-11*, 2011.

[88] P. Massa and P. Avesani. Trust-aware bootstrapping of recommender systems. In *Proceedings of ECAI 2006–Workshop on Recommender Systems*, pages 29–33. Citeseer, 2004.

[89] Michael Mathioudakis and Nick Koudas. Twittermonitor: trend detection over the twitter stream. In *Proceedings of the 2010 international conference on Management of data*, pages 1155–1158. ACM, 2010.

[90] A. Mazzia, K. LeFevre, and E. Adar. The pviz comprehension tool for social network privacy settings. *University of Michigan CSE Technical Report CSE-TR-570-11*, 2011.

[91] M. McGill. An evaluation of factors affecting document ranking by information retrieval systems., 1979.

[92] M. McPherson, L. Smith-Lovin, and J.M. Cook. Birds of a feather: Homophily in social networks. *Annual review of sociology*, 27:415–444, 2001.

[93] P. Melville and V. Sindhwani. Recommender systems. *Encyclopedia of Machine Learning*, 1:829–837, 2010.

[94] P. Mika. Ontologies are us: A unified model of social networks and semantics. *The Semantic Web–ISWC 2005*, 4:522–536, 2005.

[95] A. Mislove, B. Viswanath, K.P. Gummadi, and P. Druschel. You are who you know: inferring user profiles in online social networks. In *Proceedings of the third WSDM conference*, pages 251–260. ACM, 2010.

[96] Alan Mislove, Massimiliano Marcon, Krishna P Gummadi, Peter Druschel, and Bobby Bhattacharjee. Measurement and analysis of online social networks. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 29–42. ACM, 2007.

[97] D.C. Mueller. Public choice: A survey. *Journal of Economic Literature*, 14(2):395–433, 1976.

[98] R.H. Myers. *Classical and modern regression with applications*, volume 488. Duxbury Press Belmont, California, 1990.

[99] I.J. Myung. Tutorial on maximum likelihood estimation. *Journal of Mathematical Psychology*, 47(1):90–100, 2003.

[100] David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26, 2007.

[101] Meenakshi Nagarajan, Hemant Purohit, and Amit Sheth. A qualitative examination of topical tweet and retweet practices. In *International AAAI Conference on Weblogs and Social Media. AAAI*, 2010.

[102] M. Netter, M. Riesner, and G. Pernul. Assisted social identity management-enhancing privacy in the social web. In *10th International Conference on Wirtschaftsinformatik*, 2011.

[103] Mark EJ Newman, Duncan J Watts, and Steven H Strogatz. Random graph models of social networks. *Proceedings of the National Academy of Sciences of the United States of America*, 99(Suppl 1):2566–2572, 2002.

[104] M.E.J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.

[105] H. Nissenbaum. *Privacy in context: Technology, policy, and the integrity of social life*. Stanford Law Books, 2009.

[106] A. Onwuasoanya, M. Skornyakov, and J. Post. Enhancing privacy on social networks by segregating different social spheres. *Rutgers Governor's School of Engineering and Technology Research journal*, 2008.

[107] Alexander Pak and Patrick Paroubek. Twitter as a corpus for sentiment analysis and opinion mining. In *Proceedings of LREC*, volume 2010, 2010.

[108] Panagiotis Papadimitriou, Ali Dasdan, and Hector Garcia-Molina. Web graph similarity for anomaly detection. *Journal of Internet Services and Applications*, 1(1):19–30, 2010.

[109] T. Paul, D. Puscher, and T. Strufe. Improving the usability of privacy settings in facebook. *Arxiv preprint arXiv:1109.6046*, 1:1, 2011.

[110] L.S. Penrose. The elementary statistics of majority voting. *Journal of the Royal Statistical Society*, 109(1):53–57, 1946.

[111] D. Ramage, S. Dumais, and D. Liebling. Characterizing microblogs with topic models. 2010.

[112] M.M. Richter. Foundations of similarity and utility. In *The 20th International FLAIRS Conference, Key West, Florida*, 2007.

[113] E.M. Rogers. *Diffusion of innovations*. Free Pr, 1995.

[114] L.B. Rubin. *Just friends: The role of friendship in our lives*. Harper & Row New York, 1985.

[115] Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. In *Proceedings of the 19th international conference on World wide web*, pages 851–860. ACM, 2010.

[116] John Scott. Social network analysis: developments, advances, and prospects. *Social Network Analysis and Mining*, 1:21–26, 2011. 10.1007/s13278-010-0012-6.

[117] B. Settles. Active learning literature survey. *Machine Learning*, 15(2):201–221, 1994.

[118] E. Spertus, M. Sahami, and O. Buyukkokten. Evaluating similarity measures: a large-scale study in the orkut social network. In *Proceedings of the 11th SIGKDD*, pages 678–684. ACM, 2005.

[119] A. Squicciarini, F. Paci, and S. Sundareswaran. Prima: An effective privacy protection mechanism for social networks. In *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*, pages 320–323. ACM, 2010.

[120] F. Stutzman and J. Kramer-Duffield. Friends only: examining a privacy-enhancing behavior in facebook. In *Proceedings of the 28th international conference on Human factors in computing systems*, pages 1553–1562. ACM, 2010.

[121] Bongwon Suh, Lichan Hong, Peter Pirolli, and Ed H Chi. Want to be retweeted? large scale analytics on factors impacting retweet in twitter network. In *Social Computing (SocialCom), 2010 IEEE Second International Conference on*, pages 177–184. IEEE, 2010.

[122] Toshimitsu Takahashi, Ryota Tomioka, and Kenji Yamanishi. Discovering emerging topics in social streams via link anomaly detection. In *Data Mining (ICDM), 2011 IEEE 11th International Conference on*, pages 1230–1235. IEEE, 2011.

[123] Yuri Takhteyev, Anatoliy Gruzd, and Barry Wellman. Geography of twitter networks. *Social Networks*, 34(1):73–81, 2012.

[124] P.N. Tan, M. Steinbach, V. Kumar, et al. *Introduction to data mining*. Pearson Addison Wesley Boston, 2006.

[125] K. Thomas, C. Grier, and D. Nicol. unfriendly: Multi-party privacy risks in social networks. In *Privacy Enhancing Technologies*, pages 236–252. Springer, 2010.

[126] Hollis Thomases. *Twitter marketing: An hour a day*. Sybex, 2010.

[127] M. Valafar, R. Rejaie, and W. Willinger. Beyond friendship graphs: a study of user interactions in flickr. In *Proceedings of the 2nd ACM workshop on Online social networks*, pages 25–30. ACM, 2009.

[128] Bibi van den Berg and Ronald Leenes. Audience segregation in social network sites. In *Proceedings of the 2010 IEEE Second International Conference on Social Computing*, SOCIALCOM '10, pages 1111–1116, Washington, DC, USA, 2010. IEEE Computer Society.

[129] Bimal Viswanath, Alan Mislove, Meeyoung Cha, and Krishna P. Gummadi. On the evolution of user interaction in facebook. In *Proceedings of the 2nd ACM SIGCOMM Workshop on Social Networks (WOSN'09)*, August 2009.

[130] T. Wang, M. Srivatsa, D. Agrawal, and L. Liu. Modeling data flow in socio-information networks: A risk estimation approach. *SACMAT 2011, 16th ACM Symposium on Access Control Models and Technologies, Innsbruck, Austria, June 15-17, Proceedings*, 2011.

[131] Shaozhi Ye and S Felix Wu. Measuring message propagation and social influence on twitter. com. In *Social informatics*, pages 216–231. Springer, 2010.

[132] Soon-Hyung Yook, Hawoong Jeong, and Albert-László Barabási. Modeling the internet's large-scale topology. *Proceedings of the National Academy of Sciences*, 99(21):13382–13386, 2002.

[133] Wayne Zhao, Jing Jiang, Jianshu Weng, Jing He, Ee-Peng Lim, Hongfei Yan, and Xiaoming Li. Comparing twitter and traditional media using topic models. *Advances in Information Retrieval*, pages 338–349, 2011.

[134] E. Zheleva, L. Getoor, J. Golbeck, and U. Kuter. Using friendship ties and family circles for link prediction. *Advances in Social Network Mining and Analysis*, 2:97–113, 2010.

[135] R. Zhou, S. Khemmarat, and L. Gao. The impact of youtube recommendation system on video views. In *Proceedings of the 10th annual conference on Internet measurement*, pages 404–410. ACM, 2010.

[136] J. Zhu, H. Wang, and E. Hovy. Multi-criteria-based strategy to stop active learning for data annotation. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 1129–1136. Association for Computational Linguistics, 2008.

[137] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Machine learning-international workshop then conference-*, volume 20, page 912, 2003.

Chapter 8

Appendix: Experiment Interfaces

In this chapter, we will give the screenshots of our experiment website mentioned in Chapter 6. Figures 8.1 and 8.2 show the initial welcome page where we teach the human validator the basics of our testing website.

Our training screen for an anomalous user-friend pair is shown in Figure 8.3. In Figure 8.4, we show an exemplary test screen. In the figure, the user chooses a label for the friendship pair, and enters an explanation for his/her decision to the text area.

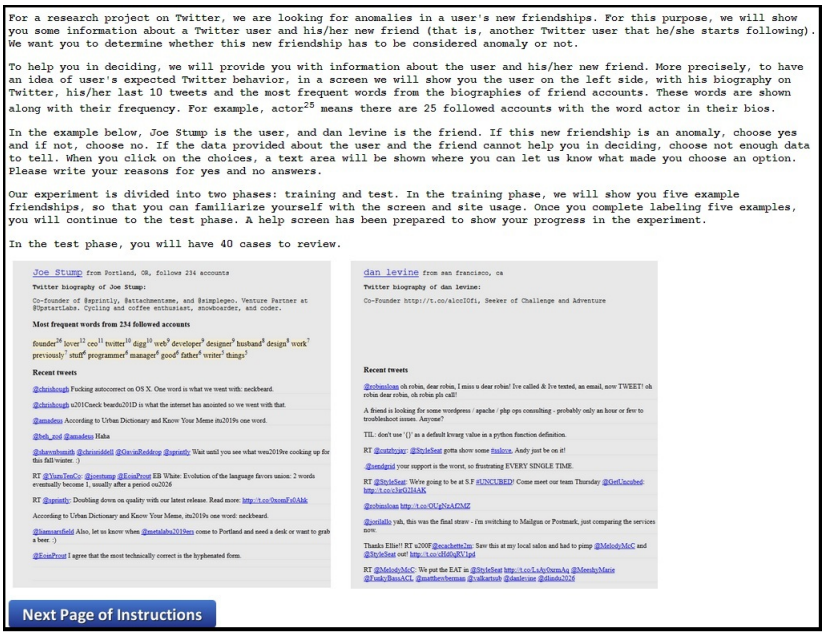


Figure 8.1: Help Interface 1

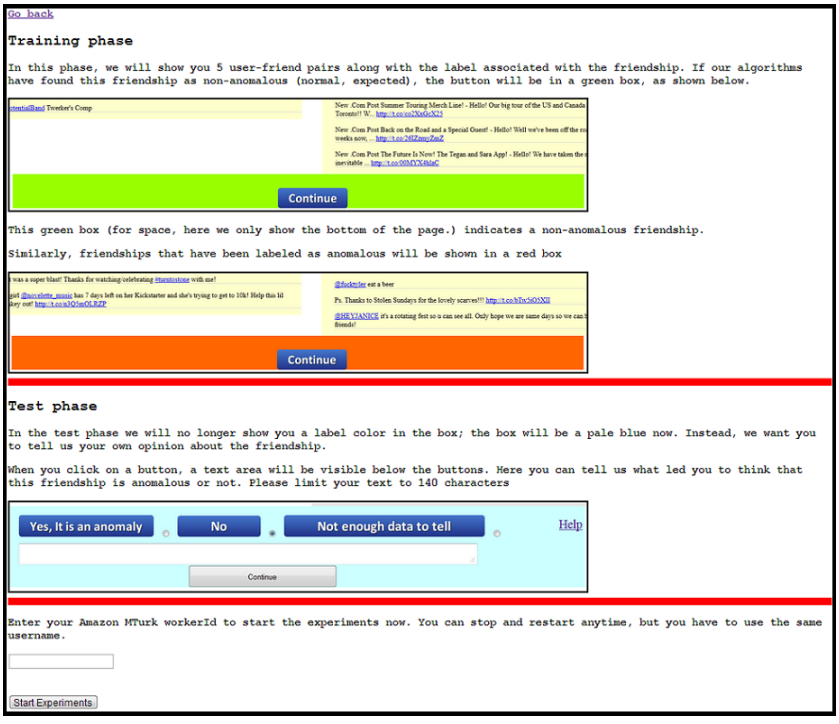


Figure 8.2: Help Interface 2

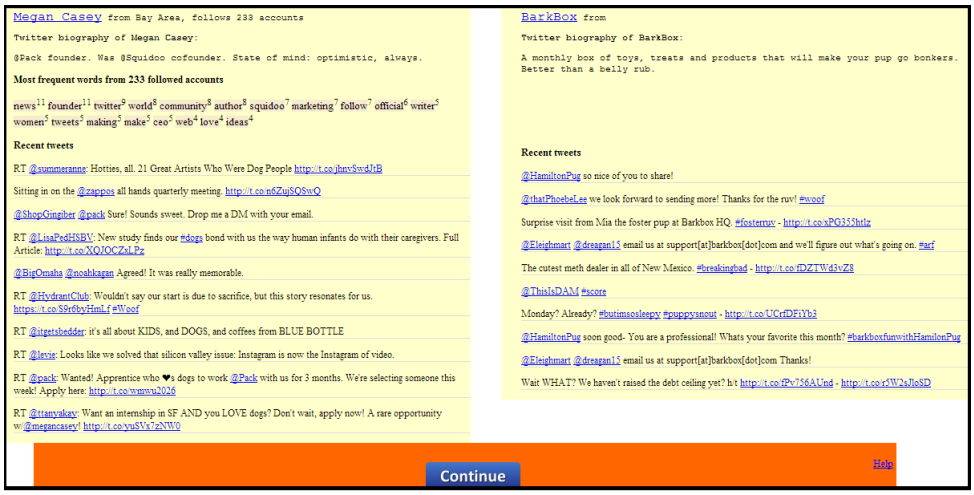


Figure 8.3: Training Interface

Figure 8.4: Test Interface