Towards Secure and Privacy-Preserving Online Social Networking Services

Zhenqiang Gong

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2015-76
http://www.eecs.berkeley.edu/Pubs/TechRpts/2015/EECS-2015-76.html

May 13, 2015

Towards Secure and Privacy-Preserving Online Social Networking Services

by

Zhenqiang Gong

A dissertation submitted in partial satisfaction
of the requirements for the degree of

Doctor of Philosophy

in

Computer Science

in the

GRADUATE DIVISION

of the

UNIVERSITY OF CALIFORNIA, BERKELEY

Committee in charge:

Professor Dawn Song, Chair
John Canny
Deirdre Mulligan
Doug Tygar

Spring 2015

Towards Secure and Privacy-Preserving Online Social Networking Services

by

Zhenqiang Gong

Abstract

Towards Secure and Privacy-Preserving Online Social Networking Services

by

Zhenqiang Gong

Doctor of Philosophy in Computer Science

University of California, Berkeley

Professor Dawn Song, Chair

Online social networking services (e.g., Facebook, Twitter, and Blogger) bring new benefits to almost all aspects of our lives. They have completely transformed how we communicate with each other, how we process information, and how we diffuse social influence. However, these social networking services are also plagued by both conventional and emerging threats to security and privacy. For instance, two fundamental security risks are 1) users' accounts are compromised by attackers or get lost and 2) attackers create massive fake (or Sybil) accounts to launch various malicious activities. In this thesis, we first design secure and usable account recovery methods based on users' trusted friends to recover compromised or lost user accounts. Second, we construct a scalable semi-supervised learning framework, which is based on probabilistic graphical model techniques, to detect Sybil accounts. Third, we demonstrate that diverse private information (e.g., private user demographics and hidden social connections) can be inferred with high accuracies from data that is publicly available on social networking sites, which has implications for the design of privacy-preserving online social networking services.

To my wife and parents

Contents

List of Figures

List of Tables

Acknowledgements

I want to thank my advisor, professor Dawn Song, for being an excellent academic advisor and guiding me towards exciting and challenging research directions. In addition, I would like to thank my committee members, including professor John Canny, professor Deirdre Mulligan, and professor Doug Tygar, for their guidance, support, and encouragement.

I also want to thank my collaborators and colleagues. During my graduate career, I have collaborated with Mario Frank, Ling Huang, Shouling Ji, Bin Liu, Lester Mackey, Prateek Mittal, Arvind Narayanan, Mathias Payer, Dawn Song, Elaine Shi, Emil Stefanov, Richard Shin, Vyas Sekar, Ameet Talwalkar, Di Wang, and others. Furthermore, I would like to thank other graduate students in the security group and teachers of the courses I have taken. I was lucky to study together with such fantastic graduate students and learn from excellent teachers. I have benefited a lot from discussing with my collaborators and colleagues.

Being able to pursue graduate degree at Berkeley was in a large part due to excellent teachers and mentors I encountered in my life. A special thanks goes to Guang-Zhong Sun, Xing Xie, and Jing Yuan, without whose mentorship and support, it would be unlikely for me to be accepted by Berkeley as a graduate student.

Finally, I want to thank my family. This thesis would be impossible without their love and support.

Chapter 1

Introduction

An online social networking service is a web service that allows users to create a profile, construct a list of other users with whom they share connections, and view and traverse through the connections [20]. For instance, Facebook, Twitter, and Google+ are popular online social networking services. Online social networking services have become increasingly important as they have completely transformed how we communicate with each other, how we process information, and how we diffuse social influence. Indeed, Facebook reported to own 1.4 billion active users as of December 2014 [6], and Twitter had 284 million active users as of December 2014 [5]. Furthermore, Facebook has become the second most visited website worldwide, just below the search engine Google, according to Alexa [2]. However, these online social networking services are also vulnerable to both classical and emerging threats to security and privacy, which are the focus of this thesis.

1.1 Trusted Friends based Secure Account Recovery

The first fundamental security risk is that users' accounts could get lost or be compromised by attackers. In particular, social networking services today most commonly rely on passwords to authenticate users. Two well-known issues in this paradigm are that users will inevitably forget their passwords, and that passwords could be compromised and reset by attackers. As a result, users cannot access their own accounts and lose all the data associated with the accounts. Therefore, service providers often provide a backup authentication mechanism (i.e., account recovery mechanism) for users to recover accounts. Since existing backup authentication mechanisms such as security questions were shown to be insecure or unreliable or both [17, 24, 106, 144], social networking services aim to leverage trusted social friends to design backup authentication mechanisms, i.e., *social authentication*. For instance, Facebook launched a social authentication system called *Trusted Contacts* in May 2013. In this mechanism, a user selects a few social friends as his/her trustees who will later help the user to recover his/her lost or compromised account.

We provide a systematic study about the security of such trustee-based social authentication mechanisms [50]. In particular, we identify that, unlike other authentication mechanisms, users' security in trustee-based social authentication is correlated, i.e., if a user's trustees are compro-

mised, then the user will also be compromised. Based on this key observation, we propose a new probabilistic security model to quantify the expected number of users that can be compromised by an attacker with a given resource. We also introduce various attacks and defenses, and we evaluate them quantitatively using our security model on a few real-world social network datasets. Our results have strong implications for the design of more secure trustee-based social authentication mechanisms including Facebook's Trusted Contacts.

1.2 Social Structure based Sybil Account Detection

The second threat is Sybil attacks, where attackers register a large number of fake (or Sybil) accounts to subvert the security and privacy of social networking services. For instance, in 2013 it was reported that 10% of Twitter accounts were Sybils [1]. These Sybil accounts are used for various malicious activities such as spreading spam or malware [117], stealing other users' private information [15, 43], and manipulating web search results via "+1" or "like" clicks [53].

We design SybilBelief [47] to detect Sybil accounts at scale using the social connections between users. The intuition is that it is hard for attackers to establish trust relationships between Sybil accounts and benign users, even though they can manipulate arbitrarily the Sybil accounts they created. From a machine learning perspective, detecting Sybil accounts is a binary classification problem with benign and Sybil as the two classes. Previous works [8, 26, 34, 120, 121, 125, 133, 139, 140] were one-class classification approaches since they leveraged either known benign accounts or known Sybil accounts, but not both, to learn their classification models. SybilBelief leverages information about both known benign accounts and known Sybil accounts, as well as the social connections amongst them and other unlabeled accounts, through a semi-supervised learning approach. SybilBelief is based on pairwise Markov Random Fields and Loopy Belief Propagation. In particular, SybilBelief models a social network as a pairwise Markov Random Fields, which is a joint probability distribution over the states (i.e., benign or Sybil) of all accounts; given some known benign and Sybil accounts, SybilBelief leverages Loopy Belief Propagation to infer the posterior probabilities of all other accounts being fake; and SybilBelief further uses the posterior probabilities to classify them. We demonstrate that SybilBelief substantially outperforms previous approaches with both synthetic and real-world social network datasets.

1.3 Attribute Inference and Link Prediction

Users could hide their sensitive attributes (e.g., sexual orientation, location, and major) and social connections in social networking services for various reasons. Therefore, two fundamental privacy attacks are to infer hidden attributes and hidden connections, which are called *attribute inference* and *link prediction*, respectively. While there is a large body of literature on link prediction [7, 16, 58, 77, 79, 87, 88, 108, 116, 126, 141], attribute inference and how user attributes can help link prediction are less explored.

Unlike prior works that studied attribute inference and link prediction separately, we propose to solve the two problems jointly [48, 49], which is based on our observations that user attributes and social structures influence each other [52]. In particular, we propose a framework called *Social-Attribute-Network (SAN)* to gracefully integrate social structure, user attributes, and their interac-

tions. Specifically, SAN augments a social network with additional nodes, each of which represents a binary attribute; and a link between a user and an attribute node indicates that the user has the corresponding attribute. Moreover, we propose a family of new link prediction and attribute inference algorithms under the SAN framework. We demonstrate that these algorithms outperform previous link prediction and attribute inference approaches via evaluations on a real-world Google+ dataset. Furthermore, we find that attribute inference can help link prediction, i.e., link prediction accuracy can be further improved by first performing attribute inference. Our results have implications for the design of privacy-preserving social networking services. In particular, our inference attacks should be considered in the next generation of social networking services.

Chapter 2

Trusted Friends based Secure Account Recovery

In this chapter, we present how to design secure account recovery methods based on trusted friends available on users' social networking accounts. This work was published in *IEEE Transactions on Information Forensics and Security, Vol.9, No.8*. This is a joint work with Di Wang.

2.1 Introduction

Web services (e.g., Gmail, Facebook, and online Bankings) today most commonly rely on passwords to authenticate users. Unfortunately, two serious issues in this paradigm are: users will inevitably forget their passwords, and their passwords could be compromised and changed by attackers, which result in the failures to access their own accounts.

Therefore, web services often provide users with backup authentication mechanisms to help users regain access to their accounts. Unfortunately, current widely used backup authentication mechanisms such as security questions and alternate email addresses are insecure or unreliable or both. Previous works [24, 106, 144] have shown that security questions are easily guessable and phished, and that users might forget their answers to the security questions. A previously registered alternate email address might expire upon the user's change of school or job. For the above reasons, it is important to design a secure and reliable backup authentication mechanism.

Recently, *trustee-based social authentication* has attracted increasing attentions and has been shown to be a promising backup authentication mechanism [21, 40, 41, 107]. Brainard et al. [21] first proposed trustee-based social authentication and combined it with other authenticators (e.g., password, security token) as a two-factor authentication mechanism. Later, trustee-based social authentication was adapted to be a backup authenticator [40, 41, 107]. In particular, Schechter et al. [107] designed and built a prototype of trusted-based social authentication system which was integrated into Microsoft's Windows Live ID. Schechter et al. found that trustee-based social

authentication is highly *reliable*. Moreover, Facebook announced its trustee-based social authentication system called *Trusted Friends* in October, 2011 [41], and it was redesigned and improved to be *Trusted Contacts* [40] in May, 2013.

However, these previous work either focus on security at individual levels [21, 107] or totally ignore security [40, 41]. In fact, security of users are correlated in trustee-based social authentications, in contrast to traditional authenticators (e.g., passwords, security questions, and fingerprint) where security of users are independent. Specifically, a user's security in trustee-based social authentications relies on the security of his or her trustees; if all trustees of a user are already compromised, then the attacker can also compromise him or her because the attacker can easily obtain the verification codes from the compromised trustees. The impact of this key difference has not been touched. Moreover, none of the existing work has studied the fundamental design problems such as *how to select trustees for users so that the system is more secure and how to set the system parameters (e.g., recovery threshold) to balance between security and usability.*

Our work: We provide the first systematic study about the security of trustee-based social authentications. To this end, we first propose a novel framework of attacks that are based on the observation that users' security are correlated in trustee-based social authentications. In these attacks, an attacker initially obtains a small number of compromised users which we call *seed users*. The attacker then iteratively attacks other users according to some *priority ordering* of them. In an *attack trial* to a user Alice, if at least k trustees of Alice are already compromised, then the attacker can easily compromise Alice; otherwise the attacker can (optionally) send spoofing messages to Alice's uncompromised trustees to request verification codes, and such spoofing attacks can succeed with some probability [107]. Our attacks are similar to forest fires which start from a few points and spread among the forests. Thus, we call them *forest fire attacks*.

Second, we construct a probabilistic model to formalize the threats of forest fire attacks and their costs for attackers. For each user, our model computes the *compromise probability* that the user is compromised after a given number of attack iterations. With those compromise probabilities, our model calculates the *expected number of compromised users* and treats it as the threat. Moreover, our model quantifies the costs of sending spoofing messages for attackers.

Third, we explore various scenarios where seed users have different properties and introduce strategies to construct priority orderings. For instance, one scenario could be that seed users happen to be appointed as trustees of a large number of users. Furthermore, we discuss a few defense strategies. For example, one strategy is to guarantee that no user is appointed as a trustee of a large number of users.

Results and impact of our work: We apply our framework to extensively evaluate various concrete attack scenarios, defense strategies, and the impact of system parameters using three real-world social networks. First, we find that forest fire attack is a potential big threat. In particular, when all the users with at least 10 friends in these social networks adopt trustee-based social authentications, an attacker can compromise tens of thousands of users in some cases even if the number of seed users is 0; using a small number (e.g., 1,000) of seed users, the attacker can further compromise two to three orders of magnitude more users with low (or even no) costs of sending spoofing messages. Second, our defense strategy, which guarantees that no users are selected as trustees by too many other users, can decrease the expected number of compromised users by one to two orders of magnitude and increase the costs for attackers by a few times in some cases. Third, we find that, in contrast to existing work [40, 41, 107] where the recover threshold is set to be three, it could be set to be four to better balance between security and usability.

In summary, our key contributions are as follows:

- We propose a novel framework of attacks, which we call *forest fire attacks*.
- We construct a model to formalize the threats of forest fire attacks and their costs for attackers. Moreover, we explore various attack scenarios and defense strategies.
- We apply our framework to extensively evaluate these attack scenarios, defense strategies, and the impact of system parameters using three real-world social networks. Our results have strong implications for designing more secure trustee-based social authentications.

2.2 Background

First, we overview how a trustee-based social authentication system works. Then, we introduce two basic concepts, i.e., *social networks* and *trustee networks*.

2.2.1 Trustee-based Social Authentications

A trustee-based social authentication includes two phases:

- **Registration Phase.** The system prepares trustees for a user Alice in this phase. Specifically, Alice is first authenticated with her main authenticator (i.e., password), and then a few (e.g., 5) friends, who also have accounts in the system, are selected by either Alice herself or the service provider from Alice's friend list and are appointed as Alice's trustees.
- **Recovery Phase.** When Alice forgets her password or her password was compromised and changed by an attacker, she recovers her account with the help of her trustees in this phase. Specifically, Alice first sends an account recovery request with her *username* to the service provider which then shows Alice an URL. Alice is required to share this URL with her trustees. Then, her trustees authenticate themselves into the system and retrieve verification codes using the given URL. Alice then obtains the verification codes from her trustees via emailing them, calling them, or meeting them in person. If Alice obtains a sufficient number (e.g., 3) of verification codes and presents them to the service provider, then Alice is authenticated and is directed to reset her password. We call the number of verification codes required to be authenticated the *recovery threshold*.

Note that it is important for Alice to know who her trustees are in the Recovery Phase. Schechter et al. [107] showed that users cannot remember their trustees via performing user studies. Thus, a usable trustee-based social authentication system should remind Alice of her trustees.

Next, we provide details about two representative trustee-based social authentication systems which were implemented by Microsoft [107] and Facebook [40, 41], respectively.

Microsoft's trustee-based social authentication: Schechter et al. [107] designed and built a trustee-based social authentication system and integrated it into Microsoft's Windows Live ID

service. In the Registration Phase, users provide *four* trustees. The recovery threshold is *three*. Moreover, users will be reminded of their trustees.

Facebook's trustee-based social authentication: Facebook's trustee-based social authentication system is called *Trusted Friends* [41], whose improved version is *Trusted Contacts* [40]. In the Registration Phase of Trusted Contacts, a user selects *three to five* friends from his or her friend list as trustees. The recovery threshold is also set to be *three*. Facebook does not remind a user of his or her trustees, but it asks the user to type in the names of his or her trustees instead. However, once the user gets one trustee correctly, Facebook will remind him or her of the remaining trustees.

Both trustee-based social authentication systems ask users to select their own trustees without any constraint. In our experiments (i.e., Section 2.7), we show that the service provider can constrain trustee selections via imposing that no users are selected as trustees by too many other users, which can achieve better security guarantees. Moreover, none of these work performed rigorous studies to support the choice of three as the recovery threshold. In fact, our experimental results show that setting the recovery threshold to be four could better balance between security and usability.

2.2.2 Social Networks and Trustee Networks

We denote a social network as $G = (V, E)$, where each node in V corresponds to a user in the service and an undirected edge (u, v) represents that users u and v are friends. Moreover, in a trustee-based social authentication system, users and their trustees form a directed network. We call this directed network a *trustee network* and denote it as $G_T = (V_T, E_T)$, where a node in V_T is a user in the service and a directed edge (v, u) in E_T means v is a trustee of u.

One fundamental challenge in trustee-based social authentication is how to construct the trustee network from a social network so that the system is more secure.

2.3 Threat Model

We first introduce attackers' background knowledge and then a novel family of attacks which we call *forest fire attacks*.

2.3.1 Background knowledge

We assume that attackers know the trustee network in the target service. The reasonableness of this threat model is supported by two evidences. First, attackers can obtain users' usernames. A username is usually a string of letters, digits, and special characters. Moreover, Bonneau et al. [18] showed that a majority (e.g., 96% in their studies) of websites enable attackers to probe if a string is a legitimate username. Thus, strong attackers, who have enough resources (e.g., a botnet) to perform username probings, can obtain all usernames in the target service. Second, Schechter et al. [107] found, via performing user studies, that users cannot remember their own trustees. Therefore, a usable trustee-based social authentication system must remind users of their trustees. Recall that an account recovery request only requires a username. As a result, an attacker could

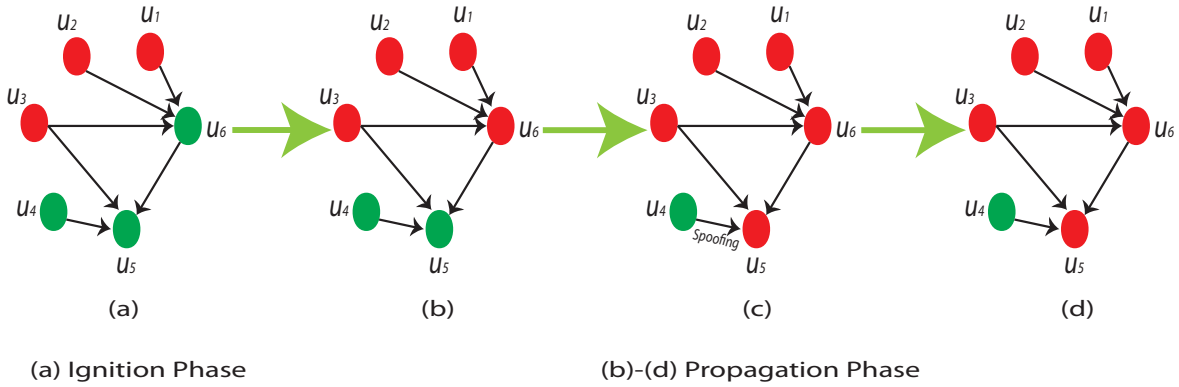


Figure 2.1: Illustration of a forest fire attack to a service with 6 users. The shown graph is the trustee network. Recovery threshold is three. Users u_5 and u_6 have adopted the trustee-based social authentication. The attack ordering is u_6, u_5, u_4. (a) u_1, u_2, and u_3 are compromised seed users. (b) u_6 is compromised because three of his or her trustees are already compromised. (c) u_5 is compromised because the attacker already compromises his or her trustees u_3 and u_6 and obtains a verification code from u_4 via spoofing attacks. (d) u_4 is not compromised because he or she hasn't adopted the service.

send account recovery requests with the collected usernames to the service provider which reminds the attacker of the trustees of each user.

Next, we take Facebook as an example to show how an attacker obtains the trustee network. First, Facebook provides an interface[1] to test if a user (represented by a username, real name, or email address) is in Facebook. Thus, the attacker can perform username probings to collect Facebook users. Second, the attacker sends account recovery requests to Facebook using the collected names. Recall that Facebook shows all trustees to a user once the user correctly types in one trustee. Moreover, Bilge et al. [4] showed that an attacker can obtain friend lists of around 90% of Facebook users. Thus, the attacker can repeatedly guess the trustees of a user until success. We note that Facebook only allows a user to try around 10 times for typing in the trustees within a short period of time. However, such rate limit cannot prevent a strong attacker from obtaining the trustee network eventually, though it can increase the attacker's cost.

2.3.2 Forest Fire Attacks

Our forest fire attacks consist of *Ignition Phase* and *Propagation Phase*.

Ignition Phase: In this phase, an attacker obtains a small number of compromised users which we call *seed users*. They could be obtained from phishing attacks, statistical guessings, and password

[1] `https://www.facebook.com/login/identify?ctx=recover`

database leaks, or they could be a coalition of users who collude each other. Indeed, a large number of social network accounts were reported to be compromised,[2] showing the feasibility of obtaining compromised seed users.

Propagation Phase: Given the seed users, the attacker iteratively attacks other users. In each *attack iteration*, the attacker performs one *attack trial* to each of the uncompromised users according to some *attack ordering* of them. In an attack trial to a user u, the attacker sends an account recovery request with u's username to the service provider, which issues different verification codes to u's trustees. The goal of the attacker is to obtain verification codes from at least k trustees. If at least k trustees of u are already compromised, the attacker can easily compromise u; otherwise, the attacker can impersonate u and send a spoofing message to each uncompromised trustee of u to request the verification code. Schechter et al. [107] found that such spoofing attacks can successfully retrieve a verification code with an average probability around 0.05.

Although the spoofing attacks can help attackers compromise more users, we want to stress that they are *optional*. We will show in our experiments that an attacker can still compromise a large number of users even if he does not use spoofing attacks to retrieve verification codes in some cases.

Example: Figure 2.1 shows a forest fire attack to a service with 6 users. Note that a good attack ordering can increase the probability that users are compromised and decrease the number of required spoofing messages (see our experimental results in Section 2.7). In our example, if the attacker performs attack trials with an attack ordering of u_5, u_6, u_4, the attacker needs to spoof both u_4 and u_6 to compromise u_5, which requires two spoofing messages. However, with the attack ordering of u_6, u_5, u_4, the attacker only needs to spoof u_4 to compromise u_5, which only requires one spoofing message and could succeed with a higher probability.

Compromised users could be recovered: Users could recover their compromised accounts to be uncompromised after they or the service provider detect suspicious activities of the accounts. For instance, a trustee of u receiving a spoofing message might report to u, who then changes his or her password; the phenomenon that a trustee requests lots of verification codes for different users within a short period of time is a possible indicator of forest fire attacks, and the service provider could then notify the users, whose trustees have requested verification codes, to change passwords. Moreover, a recovered account could be compromised again in future attack iterations, e.g., when the trustees of the recovered user are still compromised. The process of being compromised and being recovered could repeat for many attack iterations.

2.4 Security Model

In this section, we introduce our security model to formalize the threats of forest fire attacks and their costs for attackers.

[2]For instance, Gao et al. [44] showed that around 57,000 out of 3,500,000 (1.6%) Facebook accounts were compromised.

2.4.1 Formalizing Threats

We use $\Gamma_T(u)$ and $m_u = |\Gamma_T(u)|$ to denote the set of trustees and the number of trustees of u, respectively. We denote by $\Gamma_{T,o}(u)$ the set of users who select u as a trustee. We model a set of seed users (denoted as S) is obtained by a *seed users selection strategy*, and we denote it as $\mathcal{S}$. In the tth attack iteration, the attacker performs attack trials to uncompromised users according to an attack ordering $O^{(t)}$. The attack orderings are constructed by an *ordering construction strategy* which is denoted as $\mathcal{O}$.

We call the probability that u is compromised in the tth attack iteration *compromise probability*, and we denote it as $p_c^{(t)}(u)$. u is eventually compromised if it is compromised in at least one attack iteration. Thus, we denote by $p_a^{(t)}(u)$ the probability that u is compromised after t attack iterations, and $p_a^{(t)}(u)$ is called *aggregate compromise probability*. The compromise probabilities in the tth attack iteration depend on the aggregate compromise probabilities after $(t-1)$ attack iterations. Moreover, we use $p_c^{(t)}(V_T)$ and $p_a^{(t)}(V_T)$ to represent the vectors of compromise probabilities and aggregate compromise probabilities of all users in V_T, respectively.

Next, we elaborate the iterative computations of compromise probabilities and aggregate compromise probabilities.

Ignition Phase

If u is a seed user, then u's initial compromise probability is 1, otherwise we model u's initial compromise probability as 0. Formally, we have the initial compromise probability of u as follows:

$$p_a^{(0)}(u) = p_c^{(0)}(u) = \begin{cases} 1 \text{ if } u \in S \\ 0 \text{ otherwise} \end{cases} \tag{2.1}$$

Propagation Phase

The key component is to update the aggregate compromise probability of u when the aggregate compromise probabilities of u's trustees are given.

Obtaining one verification code: We denote by A the event that the attacker obtains a verification code from a trustee v of u and by $p^{(t)}(v,u)$ the probability that A happens in the tth attack iteration. Moreover, we denote the event that v is already compromised when the attacker attacks u in the tth attack iteration as B. Then we can represent $p^{(t)}(v,u)$ as:

$$\begin{aligned} p^{(t)}(v,u) &= Pr(A) \\ &= Pr(A|B)Pr(B) + Pr(A|^{\neg}B)Pr(^{\neg}B) \end{aligned} \tag{2.2}$$

, where $^{\neg}B$ represents that B does not happen. Next, we model $Pr(A|B)$, $Pr(B)$, $Pr(A|^{\neg}B)$, and $Pr(^{\neg}B)$, respectively.

When B happens, the attacker can obtain a verification code from v with a probability 1, i.e., $Pr(A|B) = 1$. $Pr(B)$ depends on whether the attacker attacks v before u or not. If v is attacked before u, then the probability that B happens is $p_a^{(t)}(v)$, otherwise it is $p_a^{(t-1)}(v)$. Formally, we

have:

$$Pr(B) = \begin{cases} p_a^{(t)}(v) & \text{if } v \text{ is ordered before } u \\ p_a^{(t-1)}(v) & \text{otherwise} \end{cases} \tag{2.3}$$

When B does not happen, the attacker can impersonate u and send a spoofing message to v to request a verification code. We call the probability that such spoofing attacks succeed *spoofing probability*. Spoofing probability might be different for different trustees. A trustee might behave differently to spoofing messages impersonating different users because he or she might have different levels of trusts with the users that are impersonated. Moreover, spoofing probability might be different in different attack iterations because trustees might gradually become aware of the spoofing attacks. Thus, we model the spoofing probability that the attacker obtains a verification code from v in an attack trial to u in the tth attack iteration as $p_s^{(t)}(v,u)$, i.e., $Pr(A|\neg B) = p_s^{(t)}(v,u)$.

In summary, we have:

$$p^{(t)}(v,u) = \begin{cases} p_a^{(t)}(v) + p_s^{(t)}(v,u)(1 - p_a^{(t)}(v)) \text{ if } v \text{ is ordered before } u \\ p_a^{(t-1)}(v) + p_s^{(t)}(v,u)(1 - p_a^{(t-1)}(v)) \text{ otherwise} \end{cases}$$

Computing compromise probabilities: Recall that u is compromised if the attacker can obtain verification codes from at least k trustees of u. Thus, given the natural assumption that u's trustees are independent, the compromise probability of u in the tth attack iteration is calculated using the following local update rule:

$$p_c^{(t)}(u) = \sum_{\phi} \prod_{v \in \phi} p^{(t)}(v,u) \prod_{v \in \Gamma_T(u) - \phi} (1 - p^{(t)}(v,u)) \tag{2.4}$$

, where $\phi \subseteq \Gamma_T(u)$ and $|\phi| \geq k$.

Aggregating compromise probabilities: Assuming that whether u is compromised in one attack iteration is independent with whether u is compromised in another attack iteration, we can iteratively compute the aggregate compromise probability of u as follows:

$$p_a^{(t)}(u) = 1 - (1 - p_a^{(t-1)}(u))(1 - p_c^{(t)}(u)) \tag{2.5}$$

Compromised users can recover to be uncompromised: As we have discussed in Section 2.3.2, compromised users can recover to be uncompromised and be compromised again due to various factors. We call the probability that a compromised user u recovers to be uncompromised in the tth attack iteration *recovery probability*, and we denote it as $p_r^{(t)}(u)$. Considering the recovery probability, we reformulate the aggregate compromise probability of u (i.e., Equation 2.5) as follows:

$$p_a^{(t)}(u) = (1 - p_r^{(t)}(u))(1 - (1 - p_a^{(t-1)}(u))(1 - p_c^{(t)}(u)))$$

Expected number of compromised users: Given a forest fire attack specified by the seed users selection strategy $\mathcal{S}$, the number of seed users n_s, the ordering construction strategy $\mathcal{O}$, and the number of attack iteratons n, we define the threat of the attack as the expected number of users it compromises. Moreover, we denote the expected number of compromised users as $n_c(G_T, k, n_s, n, \mathcal{S}, \mathcal{O})$, and it is formalized as follows:

$$n_c(G_T, k, n_s, n, \mathcal{S}, \mathcal{O}) = \sum_{u \in V_T} p_a^{(n)}(u) \tag{2.6}$$

2.4.2 Formalizing Costs

We consider the attacker's costs of obtaining seed users and sending spoofing messages. Suppose the cost to obtain seed users is c_I. In the following, we quantify the cost of sending spoofing messages.

In an attack trial to u, the attacker sends a spoofing message to u's trustee v when the following three independent events happen: u is uncompromised, v is uncompromised, and among the rest of u's trustees, the number of compromised ones is less than k.

In the tth attack iteration, the first event happens with a probability $(1-p_a^{(t-1)}(u))$. We denote by $q^{(t)}(v,u)$ the probability that the second event happens, and $q^{(t)}(v,u)$ depends on whether v is attacked before u or not. Formally, we have:

$$q^{(t)}(v,u) = \begin{cases} 1-p_a^{(t)}(v) \text{ if } v \text{ is ordered before } u \\ 1-p_a^{(t-1)}(v) \text{ otherwise} \end{cases} \tag{2.7}$$

, for any $v \in \Gamma_T(u)$. Moreover, the third event happens with the following probability:

$$r^{(t)}(v,u) = \sum_{\phi} \prod_{w \in \phi} (1-q^{(t)}(w,u)) \prod_{w \in \Gamma_T(u)-\phi-\{v\}} q^{(t)}(w,u) \tag{2.8}$$

, where $\phi \subseteq \Gamma_T(u) - \{v\}$ and $|\phi| < k$.

Therefore, the *expected* number of spoofing messages (denoted as $c^{(t)}(v,u)$) that the attacker sends to v in the attack trial to u in the tth attack iteration is formalized as follows:

$$c^{(t)}(v,u) = (1-p_a^{(t-1)}(u))q^{(t)}(v,u)r^{(t)}(v,u) \tag{2.9}$$

So the total expected number of spoofing messages (denoted as $c^{(t)}(u)$) that the attacker sends in the attack trial to u in the tth attack iteration is:

$$c^{(t)}(u) = (1-p_a^{(t-1)}(u)) \sum_{v \in \Gamma_T(u)} q^{(t)}(v,u)r^{(t)}(v,u) \tag{2.10}$$

Thus, we obtain the total expected cost of performing a forest fire attack as follows:

$$c(G_T, k, n_s, n, \mathcal{S}, \mathcal{O}) = c_I + c_e \sum_{t=1}^{n} \sum_{u \in V_T} c^{(t)}(u) \tag{2.11}$$

, where c_e is the average cost for the attacker to send one spoofing message.

2.5 Attack Strategies

The attacker could design the seed users selection strategy and the attack ordering construction strategy to maximize the expected number of compromised users. First, we show that finding the optimal set of seed users and the optimal ordering construction strategy is NP-Complete. Then, we explore various scenarios where seed users have different properties and introduce two ordering construction strategies.

2.5.1 NP-Completeness

Given G_T, k, n_s, and n, the attacker essentially aims to solve the following *attack maximization problem*:

Attack Maximization Problem:

$$n_c(G_T, k, n_s, n) = \max_{\mathcal{S},\mathcal{O}} n_c(G_T, k, n_s, n, \mathcal{S}, \mathcal{O})$$

We prove that this problem is NP-Complete. Formally, we have the following theorem.

Theorem 1. *The Attack Maximization Problem is NP-Complete*

Proof. To prove the hardness of the attack maximization problem, we only need to show the hardness of a corresponding decision problem, i.e., given G_T, k, n_s, n, and an integer l, it is NP-Complete to decide whether $n_c(G_T, k, n_s, n) \geq l$. This follows from a reduction from the NP-Complete Set Cover problem: given a ground set $X = \{x_1, \ldots, x_a\}$, and a collection of its subsets $S_1, S_2, \cdots, S_m$, we want to determine if there exists t subsets $S_{i_1}, S_{i_2}, \cdots, S_{i_t}$ such that $S_{i_1} \cup S_{i_2} \cup \cdots \cup S_{i_t} = X$. We show that this can be viewed as a special case of the attack maximization problem, where the spoofing probabilities and recovery probabilities are all 0.

Given any instance of a set cover problem, we construct a corresponding trustee network as follows: there are k nodes for each S_j, and one node for each x_i; each x_i has all the k copies of S_j as trustees iff $x_i \in S_j$; in addition, for each S_j we have a^2k^2 dummy nodes each of which designates the k copies of S_j as trustees. Then the Set Cover problem is satisfiable iff the attacker can compromise at least $l = ta^2k^2 + tk + a$ nodes with a seed set whose size is tk, i.e., $n_c(G_T, k, tk, n) \geq l$. □

2.5.2 Strategies for Selecting Seed Users

A seed users selection strategy $\mathcal{S}$ is essentially to assign a score which represents some metric of importance to each user and to select n_s users with the highest scores as seed users. This is closely related to the *node centrality problem* [95] in the network science community. In the following, we modify a few node centrality heuristics as seed users selection strategies. These strategies work on a trustee network, and we name them with a prefix 'S-' to indicate they are used to select seed users.

S-Random: As a baseline, this strategy assigns a random score ranging from 0 to 1 to each user in the trustee network.

S-Degree: Intuitively, if a user u is selected as a trustee by more users, then compromising u will increase the compromise probabilities of more users and thus the attacker has an opportunity to

compromise more users. Therefore, S-Degree treats the number of users that select u as a trustee (i.e., outdegree of u in the trustee network) as u's score.

S-BadRank: S-BadRank, adapted from BadRank [10], performs a random walk on the trustee network. Specifically, S-BadRank starts a random walk from u that is picked uniformly at random. Then S-BadRank iteratively performs one of the two operations: choosing a trustee v of u uniformly at random and walks to v with a probability $1 - \alpha$, and selecting a user w uniformly at random from the entire trustee network and walks to w with a probability α. Traditionally, α is called the *restart probability*.

The random walk will converge to a stationary probability distribution over all users in the trustee network. The stationary probability of u is roughly the frequency with which the random walk visits u and is used as u's score. Intuitively, S-BadRank tends to assign a high score to a user that is selected by many users as a trustee because he or she has a large chance to be visited by the random walk.

S-Closeness: The attacker could also select users that are close to all other users in the trustee network as seed users because compromising them could help the attacker compromise other users more quickly. This intuition is formally captured by the *closeness* metric [105]. Specifically, we define the distance between v and u as the length of the shortest directed path whose tail is v and whose head is u. Then the closeness of u is defined as the inverse of the sum of its distances to all other users, and it is treated as u's score. In our experiments, we adopt the approximate algorithm developed by Okamoto et al. [96] to find the top-n_s users with the highest closeness scores since it is scalable to large networks.

2.5.3 Strategies for Constructing Attack Orderings

We name these strategies with a prefix 'O-' to indicate that they are used to construct attack orderings. Essentially, these strategies assign a score to each user in the trustee network and rank them in a decreasing order according to their scores.

O-Random: As a baseline, this strategy assigns a random score ranging from 0 to 1 to each user in the trustee network.

O-Gradient: Algorithm 1 shows O-Gradient. Given the current aggregate compromise probabilities $p_a^{(t)}(V_T)$ of all users after t attack iterations, the attacker calculates a *predicted* aggregate compromise probability $q_a^{(t)}(u)$ for each user u by simulating an attack trial to u. Note that $q_a^{(t)}(u)$ might be different from $p_a^{(t+1)}(u)$ because $q_a^{(t)}(u)$ is calculated with the fixed aggregate compromise probabilities of u's trustees while $p_a^{(t+1)}(u)$ is updated with the newest aggregate compromise probabilities of u's trustees. Intuitively, a user u with a larger difference of $q_a^{(t)}(u) - p_a^{(t)}(u)$ could be attacked with a higher priority because such an attack trial could bring more increase to the aggregate compromise probabilities of users who select u as a trustee. Thus, O-Gradient calculates the differences of $q_a^{(t)}(u) - p_a^{(t)}(u)$ for all users and ranks them decreasingly according to these differences.

Algorithm 1: O-Gradient

Input: $G_T = (V_T, E_T)$ and $p_a^{(t)}(V_T)$.

Output: An ordering O of all users in V_T.

begin
  for $u \in V_T$ **do**
    for $v \in \Gamma_T(u)$ **do**
      //Probability of getting one verification code
      $p_v \longleftarrow p_a^{(t)}(v) + p_s^{(t+1)}(v,u)(1 - p_a^{(t)}(v))$
    end
    $p_u \longleftarrow \sum_{\phi} \prod_{v \in \phi} p_v \prod_{v \in \Gamma_T(u) - \phi} (1 - p_v)$, where $\phi \subseteq \Gamma_T(u)$ and $|\phi| \geq k$.
    $q_a^{(t)}(u) \longleftarrow 1 - (1 - p_a^{(t)}(u))(1 - p_u)$
  end
  for $u \in V_T$ **do**
    $d_a^{(t)}(u) \longleftarrow q_a^{(t)}(u) - p_a^{(t)}(u)$
  end
  $O \longleftarrow$ Sort users decreasingly using $d_a^{(t)}(u)$
  return O
end

2.6 Defense Strategies

We discuss potential defense strategies from three aspects, i.e., hiding trustee networks from attackers, mitigating spoofing attacks, and constraining the selection of trustees.

2.6.1 Hiding Trustee Networks

Preventing attackers from obtaining a trustee network is an essential step towards the defense of forest fire attacks. In the currently deployed social authentication systems, attackers can obtain a trustee network because *users need to know their trustees to retrieve verification codes from them* in the Recovery Phase. An alternative implementation of the Recovery Phase is that the service provider directly sends verification codes to the trustees of the user when receiving an account recovery request, and the trustees are required to actively share the verification codes to the user. This implementation does not require users to know their trustees, and thus it is hard for attackers to obtain the trustee network.

However, this implementation is *unreliable* and could annoy users and their trustees. Specifically, u's trustees might already forget they are trustees of u, and thus they might simply treat those verification codes as spams and not share them with u, which results in low reliability. Moreover, users do forget who their trustees are [107], and thus it is highly impossible for u to actively request verification codes from its trustees. If the trustees do actively share the codes with u, then attackers can frequently send account recovery requests to the service provider which immediately sends verification codes to the trustees, and the trustees will (possibly) frequently share the codes with u, which could be annoying to both u and its trustees. More seriously, after u's trustees realize that the verification codes are just spams, they might not actively share the verification codes with u even if she is really trying to recover her account, which again results in low reliability.

Therefore, hiding trustee networks from attackers sacrifices reliability, which possibly explains why existing trustee-based social authentication systems didn't adopt this alternative implementation of the Recovery Phase.

2.6.2 Mitigating Spoofing Attacks

Another way to defend against forest fire attacks is to remind trustees of not sharing verification codes via messages. This strategy is not novel, and we include it for completeness. Indeed, existing social authentication systems [40, 107] already try to mitigate spoofing attacks. For instance, Microsoft's system [107] asks a trustee why she is requesting the verification code and encourages her to share the code with the user via phone or meeting in person.

However, it is hard to control how trustees share the verification codes with others in practice. Indeed, an attacker can still obtain a verification code from a trustee with an average probability around 0.05 via message-based spoofing attacks in the Microsoft's system [107]. It is an interesting future work to design better user interfaces to further reduce this spoofing probability.

2.6.3 Constraining Trustee Selections

Finally, we introduce strategies to contrain trustee selections, which are easy to implement and effective at defending against forest fire attacks. We consider both *local trustee selection strategies* and *global trustee selection strategies*. A local trustee selection strategy is based on a user's local social network structure while a global one is based on the entire social network structure. We name these strategies with a prefix 'T-' to indicate that they are used to select trustees.

We note that how users select their trustees in a real trustee-based social authentication system such as Facebook's Trusteed Contacts is not clear and thus might not be one of our strategies. However, our work focuses on a comparative study about different trustee selection strategies and can shed light on which strategy is more secure.

Local trustee selection strategies

For a user u, a local trustee selection strategy essentially computes a score $s(v, u)$ for each friend v of u and then selects m_u friends with the highest scores as u's trustees.

T-Random: As a baseline, T-Random assigns a random number ranging from 0 to 1 as the score $s(v, u)$.

T-CF: As was shown by Gilbert and Karahalios [46], the number of common friends of two users is an informative indicator about the level of trust between them. Thus, one speculation is that a user might select friends with which he or she shares many common friends as trustees. To quantify the security of this speculation, we design the strategy T-CF (i.e.,T-CommonFriends), which uses the number of common friends shared by u and his or her friend v as the score $s(v, u)$, i.e., $s(v, u) = |\Gamma(v) \cap \Gamma(u)|$.

However, there are two drawbacks of T-CF. First, the fact that u shares many friends with a popular user v doesn't necessarily mean that u and v have a high level of trust because it is normal for many friends of u to be in v's friend list. Second, if a common friend of u and v is a popular user, then sharing him or her doesn't necessarily indicate a high level of trust between u and v. Next, we introduce two strategies to overcome the two drawbacks, respectively.

T-JC: To overcome the first drawback of T-CF, we design the trustee selection strategy T-JC (i.e., T-JaccardCoefficient), which uses the Jaccard Coefficient [61] of the two sets $\Gamma(u)$ and $\Gamma(v)$ as the score $s(v, u)$, i.e., $s(v, u) = \frac{|\Gamma(v) \cap \Gamma(u)|}{|\Gamma(v) \cup \Gamma(u)|}$.

T-AA: To overcome the second drawback, we design the T-AA (i.e., T-AdamicAda) strategy, which uses Adamic-Ada [7] similarity between u and v as the score $s(v, u)$. Adamic-Ada similarity penalizes each common friend by its popularity (i.e., the number of friends). Formally, we have $s(v, u) = \sum_{w \in \Gamma(v) \cap \Gamma(u)} \frac{1}{\log |\Gamma(w)|}$.

Global trustee selection strategies

Global strategies leverage the entire social network structure and thus are potentially better than local strategies.

T-Degree: As we discussed in Section 2.5.2, seed users could be those having large outdegrees in the trustee network, and they could enable an attacker to compromise many other users. Thus, we propose the T-Degree strategy to minimize the maximum outdegree in the trustee network. Intuitively, T-Degree constrains that no users are selected as trustees by too many other users.

Algorithm 2 shows our T-Degree strategy. T-Degree selects trustees for users one by one. For each user u that has adopted the trustee-based social authentication service, T-Degree selects his or her m_u friends whose current outdegrees in the trustee network are the smallest as u's trustees; ties are broken uniform at random.

Algorithm 2: T-Degree

Input: $G = (V, E)$, V_a, and m_u of all $u \in V$.

Output: A trustee network $G_T = (V_T, E_T)$.

begin
  $V_T \longleftarrow V$
  for $u \in V_T$ **do**
    $d_o(u) \longleftarrow 0$
  end
  for $u \in V_a$ **do**
    $\phi \longleftarrow \emptyset$
    while $|\phi| < m_u$ **do**
      $v \longleftarrow \text{argmin}_{v \in \Gamma(u) - \phi} d_o(v)$
      $\phi \longleftarrow \phi \cup \{v\}$
      $d_o(v) \longleftarrow d_o(v) + 1$
      $E_T \longleftarrow E_T \cup (v, u)$
    end
  end
  return $G_T = (V_T, E_T)$
end

2.7 Experiments

2.7.1 Experimental Setups

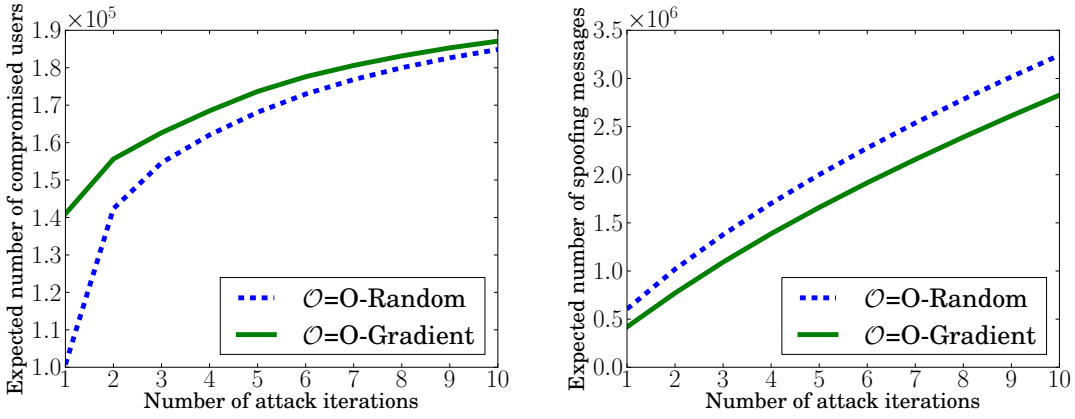
Parameter settings: Unless otherwise mentioned, we set $n = 10$, $m_u = m = 5$, $k = 3$,[3] $n_s = 1,000$, $p_r^{(t)}(u) = p_r = 0$ for every u, and $\alpha = 0.9$;[4] we use the O-Gradient strategy to construct attack orderings, i.e., $\mathcal{O}$=O-Gradient; according to Schechter et al. [107], the average message-based spoofing probability is around 0.05, thus we set $p_s^{(t)}(v, u) = p_s = 0.05$ for every edge $(v, u) \in E_T$.

[3]This is the recovery threshold chosen by the Microsoft's and Facebook's trustee-based social authentication systems.

[4]We tried α from 0 to 0.9 with a step size 0.1, and we found that S-BadRank with $\alpha = 0.9$ achieves the largest expected number of compromised users.

Table 2.1: The number of users, the number of users who have at least 10 friends, the fraction of such users, and the average number of friends of such users in the three social networks.

	Flickr	Google+	Twitter
# Nodes	1,551,824	10,230,332	21,297,771
# Nodes (degree $\geq$ 10)	233,067	3,144,370	4,540,483
Fractions	15%	31%	21%
Average degrees	75	27	107



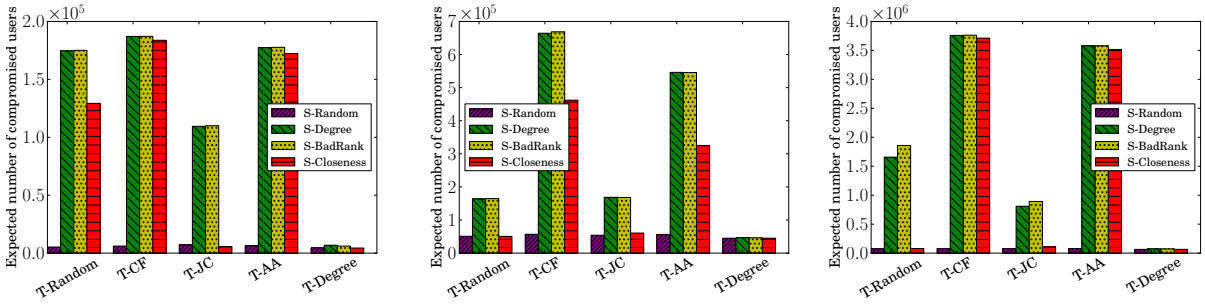
(a) Number of compromised users

(b) Number of spoofing messages

Figure 2.2: Impact of attackers' resources and attack orderings on Flickr. First, we find that an attacker can perform forest fire attacks with low costs. Second, we find that O-Gradient compromises more users and requires fewer spoofing messages than O-Random when the attacker performs a given number of attack iterations.

Datasets: We use three social network datasets. They are Flickr, Google+, and Twitter. We obtained the Flickr dataset from Mislove et al. [90]. In this social network, we take each user as a node and connect two users if they are in each other's friend lists. We obtained a Google+ snapshot from Gong et al. [52]. In this social network, each node is a Google+ user and two users are connected if they are in each other's circles. The Twitter dataset was obtained from Kwak et al. [73]. In this social network, each node is a Twitter user and two users are connected if they follow each other.

Since a trustee-based social authentication requires a user to have enough number of friends from which trustees can be selected, we assume that users who have at least 10 friends are appropriate to adopt the trustee-based social authentication service. In our experiments, we consider the *worst*



(a) Flickr (b) Google+ (c) Twitter

Figure 2.3: The expected number of compromised users for different seed users selection strategies and trustee selection strategies. We find that, with 1,000 seed users, an attacker can compromise two to three orders of magnitude more users with low costs of sending spoofing messages in some cases. However, our strategy T-Degree can decrease the expected number of compromised users by one to two orders of magnitude.

scenario in which every such user has adopted the trustee-based social authentication service.[5] Table 3.1 shows the statistics of our interest in the three social networks.

For simplicity, we only show results on the Flickr dataset in some of our experiments, but similar conclusions are applicable to the other two datasets.

2.7.2 Experimental Results

Impact of attackers' resources and attack orderings: The number of attack iterations is closely related to the costs of sending spoofing messages. Figure 2.2 illustrates the expected number of compromised users (Figure 2.2a) and the expected number of required spoofing messages (Figure 2.2b) as a function of the number of attack iterations on Flickr for the ordering construction strategies O-Random and O-Gradient. The seed selection strategy is S-Degree and the trustee selection strategy is T-CF.

First, we find that a strong attacker (e.g., an attacker controlling a botnet) can perform firest fire attacks with low costs. This is because such an attacker can send out billions of messages *per day* with a low cost [112], which is far more than that needed to spoof trustees in our experiments.

Second, we find that the ordering construction strategy O-Gradient compromises more users and requires fewer spoofing messages than O-Random when the attacker performs a given number of attack iterations. In other words, given the number of spoofing messages the attacker can send, the attacker should adopt O-Gradient to construct the attack orderings.

[5]The fewer users adopt the trustee-based social authentication, the more secure it is. An extreme case is that if no user adopts the system, then the forest fire attacks cannot be spreaded from the seed users at all.

Similar conclusions are applicable to other combinations of seed selection strategies and trustee selection strategies. Thus, we don't show the corresponding results for conciseness.

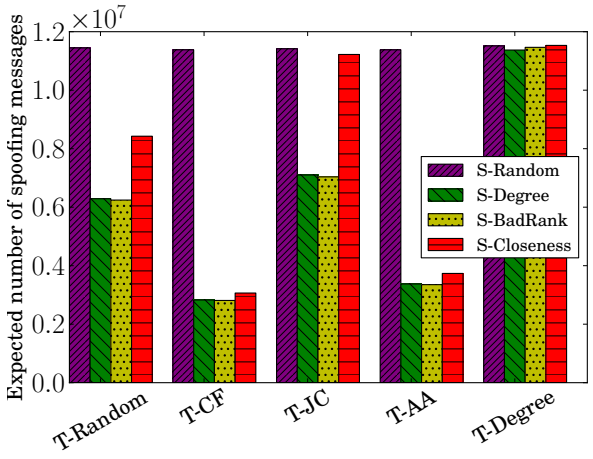


Figure 2.4: The expected number of required spoofing messages for different seed users selection strategies and trustee selection strategies on Flickr. We observe that our strategy T-Degree can increase the attacker's costs by a few times in some cases.

Impact of seed users selection strategies and trustee selection strategies: Figure 2.3 and Figure 2.4 show the expected number of compromised users and the expected number of required spoofing messages respectively for different seed selection strategies and trustee selection strategies. We can draw a few conclusions.

First, we find that forest fire attack is a potential big threat. For instance, when the seed users are appointed as trustees of many users (i.e., S-Degree) and the trustees are selected by T-CF, the attacker can compromise around 190,000, 660,000, and 3,760,000 users in the three social networks, respectively. This represents a growth of two to three orders of magnitude from the 1,000 seed users. However, our strategy T-Degree can decrease the expected number of compromised users by one to two orders of magnitude. For instance, the expected number of compromised users of T-Degree is 53 times smaller than that of T-CF on Twitter when the seed users selection strategy is S-Degree. Moreover, our strategy T-Degree can increase the costs for attackers by a few times in some cases. For instance, the cost of sending spoofing messages of T-Degree is 3 times bigger than that of T-CF and that of T-AA on Flickr when the seed users are appointed as trustees of many users. The reason is that the trustee networks constructed by T-Degree are more loosely connected, which makes it harder for forest fire attacks to propagate among them.

Second, even if the seed users are distributed among a social network uniformly at random (i.e., S-Random), the attacker can still compromise dozens of times more users. For instance, the attacker can still compromise 65 to 80 times more users in Twitter depending on how trustees are selected.

Third, T-JC works better than T-AA which performs better than T-CF for all seed users selection strategies except S-Random. We find that the outdegree distributions of the trustee networks constructed by T-CF are skewed towards high degrees the most while those constructed by T-JC are skewed towards low degrees the most. Thus, the seed users in the trustee networks

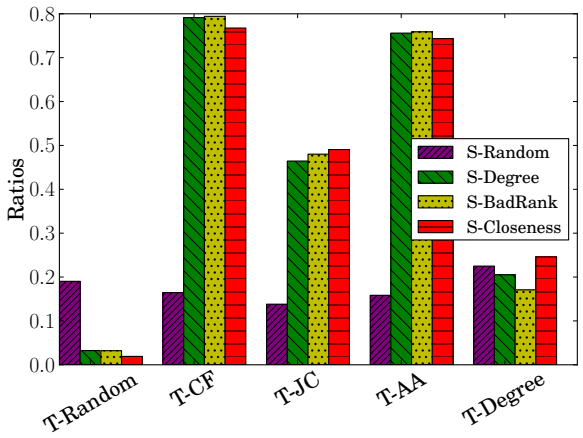


Figure 2.5: The ratios between the expected number of users that are compromised without spoofing attacks and those with spoofing attacks for different seed selection strategies and different trustee selection strategies on Flickr. We find that the expected number of compromised users only decreases by 20% to 25% when the attacker does not use spoofing attacks in some cases, which implies that spoofing attacks are optional.

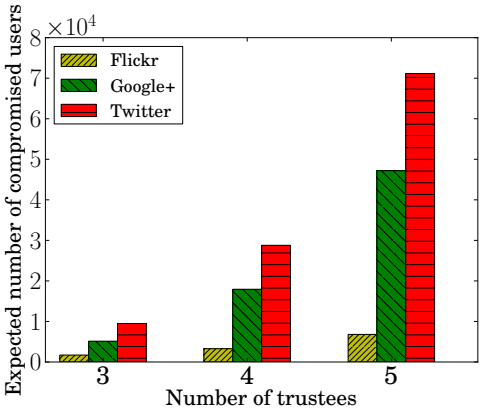
constructed by T-JC have lower outdegrees than those constructed by T-AA, which have lower outdegrees than those constructed by T-CF. As a result, T-JC performs better than T-AA and T-AA performs better than T-CF.

Impact of spoofing attacks: The attacker can choose not to send any spoofing message, which requires no costs of sending spoofing messages. Moreover, spoofing probabilities might decrease as the attacker performs more attack iterations because trustees become aware of the spoofing attacks, and we consider an extreme case where spoofing probabilities are zero in all attack iterations. That spoofing probabilities are zero is equivalent to that the attacker does not use spoofing attacks in terms of the number of compromised users.

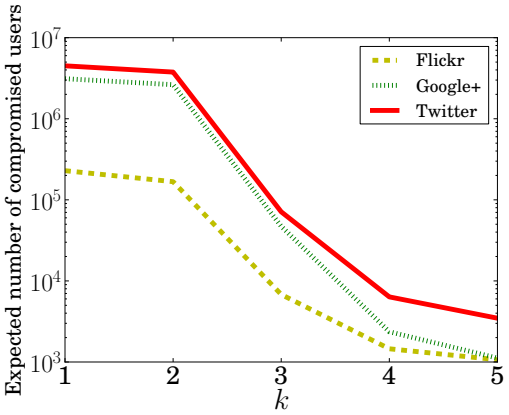
Figure 2.5 shows the ratios between the expected number of users that are compromised without spoofing attacks and those with spoofing attacks for different seed users selection strategies and trustee selection strategies on Flickr.

We find that, without spoofing attacks, the expected number of compromised users only decreases by 20% to 25% when the trustee selection strategy is T-CF or T-AA and the seed users selection strategy is not S-Random, which means that the attacker can still compromise orders of magnitude more users and that spoofing attacks are optional in some cases.

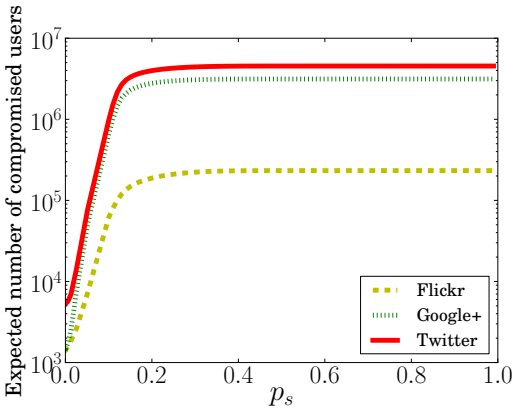
Impact of m_u, k, $p_s^{(t)}(v,u)$, n_s, and $p_r^{(t)}(u)$: Towards this end, we assume that the attacker and the defender are making their best choices, i.e., the seed selection strategy is S-Degree and the trustee selection strategy is T-Degree. For simplicity, we assume $m_u = m$ for all users u who adopt the trustee-based social authentication service and $p_s^{(t)}(v,u) = p_s$ for all edges (v,u) in the trustee network and attack iterations. Moreover, we consider zeroth-order approximation of recovery probabilities, i.e., we use the average recover probability to approximate recover probabilities of all users



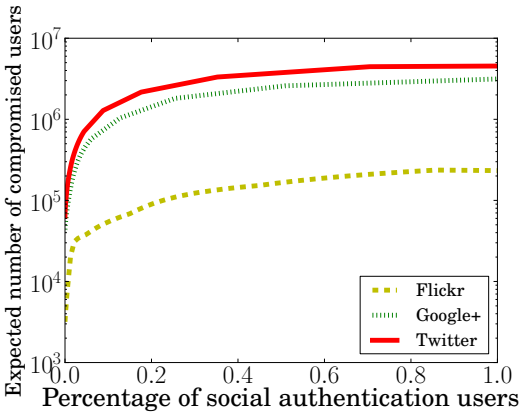
(a) Impact of m



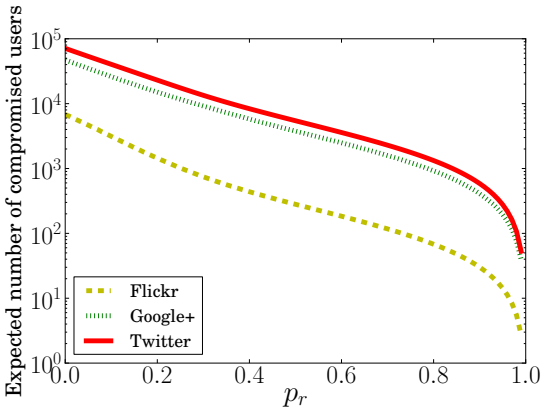
(b) Impact of k



(c) Impact of p_s



(d) Impact of n_s



(e) Impact of p_r

Figure 2.6: Impact of m, k, p_s, n_s, and p_r. We find that $k = 4$ is a good tradeoff between security and usability.

in different attack iterations. In other words, $p_r^{(t)}(u) = p_r$ for all users and attack iterations. p_r represents the average fraction of compromised users that realize their accounts are compromised and take actions to recover them in each attack iteration.

Figure 2.6 shows the impact of m, k, p_s, n_s, and p_r. Note that the x-axis in Figure 2.6d is the percentage of users that adopt the trustee-based social authentication service. We have a few observations.

Naturally, we observe that the expected number of compromised users decreases with k and p_r, and it increases with m, p_s, and n_s. Moreover, the trends are *qualitatively* similar on the three social networks for each of the five parameters, which indicates the generality of our results.

The parameters m and k balance between security and usability. Specifically, a smaller m or a bigger k makes the system more secure but less usable. Figure 2.6b shows that the expected number of compromised users decreases dramatically when k increases from 1 to 4. For instance, the expected number of compromised users decreases by around one order when k changes from 3 to 4. However, the decrease is not significant when k increases from 4 to 5. Existing trustee-based social authentication systems [40, 107] set k to be 3; our observations indicate that they could set k to be 4 to better balance between security and usability.

We find that the expected number of compromised users changes dramatically when p_s is around 0.2 on all the three social networks. Moreover, almost all users that have adopted the trustee-based social authentication service are compromised when p_s is bigger than around 0.3. Our results imply that it is urgent to mitigate the spoofing attacks via designing better user interfaces to remind users of not easily sharing the verification codes to others via messages.

The expected number of compromised users increases dramatically with n_s in the range where n_s is small. This implies that obtaining a few more seed users enables the attacker to compromise significantly more users when the number of seed users is small. Moreover, even if the number of seed users is 0, the attacker can still compromise thousands of nodes in Flickr and tens of thousands of nodes in Google+ and Twitter.

The expected number of compromised users significantly decreases with p_r. For instance, the number of compromised users decreases by one order in all the three social networks when around 40% of compromised users recover to be uncompromised in each attack iteration (i.e., $p_r = 0.4$). Our results imply that it is important for users to take actions to recover their accounts quickly when suspicious activities of their accounts are detected.

2.8 Summary of Results

We provide the first systematic study about the security of trustee-based social authentications. First, we introduce *forest fire attacks*. In these attacks, an attacker first obtains a small number of compromised seed users and then iteratively attacks the rest of users according to a priority ordering of them. Second, we construct a probabilistic model to formalize the threats of forest fire attacks and their costs for attackers. Third, we introduce a few strategies to select seed users and construct priority orderings, and we discuss various defense strategies. Fourth, via extensive evaluations using three real-world social network datasets, we find that forest fire attack is a potential big threat. For instance, with a small number (e.g., 1,000) of seed users, an attacker can further compromise two to three orders of magnitude more users in some scenarios with low (or even no) costs of sending

spoofing messages. However, our defense strategy, which guarantees that no users are trustees of too many other users, can decrease the number of compromised users by one to two orders of magnitude and increase the costs for attackers by a few times in some cases. Moreover, the recovery threshold should be set to be 4 to better balance between security and usability.

Chapter 3

Social Structure based Sybil Account Detection

In this chapter, we present our *SybilBelief*, a semi-supervised learning framework, to detect Sybil accounts in social networks using the social structure. This work was published in *IEEE Transactions on Information Forensics and Security, Vol.9, No.6*. This is a joint work with Mario Frank and Prateek Mittal.

3.1 Introduction

Sybil attacks, where a single entity emulates the behavior of multiple users, form a fundamental threat to the security of distributed systems [36]. Example systems include peer-to-peer networks, email, reputation systems, and online social networks. For instance, in 2013 it was reported that 10% of Twitter accounts were Sybils [1]. Sybil accounts in online social networks are used for criminal activities such as spreading spam or malware [117], stealing other users' private information [15, 43], and manipulating web search results via "+1" or "like" clicks [53].

Traditionally, Sybil defenses require users to present trusted identities issued by certification authorities. However, such approaches violate the open nature that underlies the success of these distributed systems [125]. Recently, there has been a growing interest in leveraging social networks to mitigate Sybil attacks [8, 26, 34, 120, 121, 125, 133, 139, 140]. These schemes are based on the observation that, although an attacker can create arbitrary Sybil users and social connections among themselves, he or she can only establish a limited number of social connections to benign users. As a result, Sybil users tend to form a community structure among themselves, which enables a large number of Sybil users to integrate into the system. Note that it is crucial to obtain social connections that represent trust relationships between users, otherwise the structure-based Sybil detection mechanisms have limited detection accuracy. See Section 3.2.1 for more discussions.

However, existing structure-based approaches suffer from one or more of the following draw-

backs: (1) they can bootstrap from either only known benign [34, 121, 139, 140] or known Sybil nodes [133], limiting their detection accuracy (see Section 3.6), (2) they cannot tolerate noise in their prior knowledge about known benign [26] or Sybil nodes [133], and (3) they are not scalable [34, 120, 121, 125, 139, 140].

To overcome these drawbacks, we recast the problem of finding Sybil users as a semi-supervised learning problem, where the goal is to propagate reputations from a small set of known benign and/or Sybil users to other users along the social connections between them. More specifically, we first associate a binary random variable with each user in the system; such random variable represents the label (i.e., benign or Sybil) of the user. Second, we model the social network between users in the system as a pairwise Markov Random Field, which defines a joint probability distribution for these binary random variables. Third, given a set of known benign and/or Sybil users, we infer the posterior probability of a user being benign, which is treated as the reputation of the user. For efficient inference of the posterior probability, we couple our framework with Loopy Belief Propagation [99], an iterative algorithm for inference on probabilistic graphical models.

We extensively evaluate the influence of various factors including parameter settings in the SybilBelief, the number of labels, and label noises on the performance of SybilBelief. For instance, we find that SybilBelief is relatively robust to parameter settings, SybilBelief requires one label per community, and SybilBelief can tolerate 49% of labels to be incorrect in some cases. In addition, we compare SybilBelief with state-of-the-art Sybil classification and ranking approaches on real-world social network topologies. Our results demonstrate that SybilBelief performs orders of magnitude better than previous Sybil classification mechanisms and significantly better than previous Sybil ranking mechanisms. Finally, SybilBelief proves to be more resilient to noise in our prior knowledge about known benign users and known Sybil users.

In summary, our work makes the following contributions:

- We propose SybilBelief, a semi-supervised learning framework, to perform both Sybil classification and Sybil ranking. SybilBelief overcomes a number of drawbacks of previous work.
- We extensively evaluate the impact of various factors including parameter settings in SybilBelief, the number of labels, and label noise on the performance of SybilBelief using synthetic social networks. For instance, we find that SybilBelief is relatively robust to parameter settings, SybilBelief requires one label per community, and SybilBelief can tolerate 49% of labels to be incorrect in some cases.
- We demonstrate, via evaluations on real-world social networks, that SybilBelief performs orders of magnitude better than previous Sybil classification mechanisms and significantly better than previous Sybil ranking mechanisms. Moreover, SybilBelief is more resilient to label noise, i.e., partially corrupted prior knowledge about known benign users and known Sybil users.

3.2 Problem Definition

We formally define the Sybil detection problem. Specifically, we first introduce the social network model. Then we introduce a few design goals.

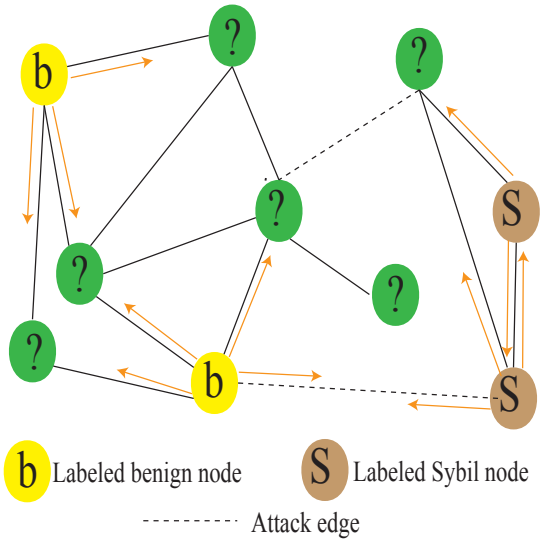


Figure 3.1: The propagation in SybilBelief. Given a set of labeled nodes, we want to infer the labels of the remaining nodes. SybilBelief iteratively propagates the label information from the labeled nodes to their neighbors.

3.2.1 Social Network Model

Let us consider an undirected social network $G = (V, E)$, where a node $v \in V$ represents a user in the system and an edge $(u, v) \in E$ indicates that the users $u \in V$ and $v \in V$ are socially connected. In an ideal setting, G represents a weighted network of trust relationships between users, where the edge weights represent the levels of trust between users [92]. Each node is either *benign* or *Sybil*.

Figure 3.1 illustrates a Sybil attack. We denote the subnetwork including the benign nodes and the edges between them as the *benign region*, denote the subnetwork including the Sybils and edges between them as the *Sybil region*, and denote the edges between the two regions as *attack edges*. Note that the benign region could consist of multiple communities and we will evaluate their impact on Sybil detections in Section 3.5.

An attacker could obtain attack edges via spoofing benign nodes to link to Sybils or compromising benign nodes, which turns the edges between the compromised benign nodes and other benign nodes to attack edges. Compromised benign nodes are treated as Sybils, and they could be those whose credentials are available to the attacker or *front peers* [129] who collude with Sybils.

One fundamental assumption underlying the structure-based Sybil detections is that the benign region and the Sybil region are sparsely connected (i.e., the number of attack edges is small), compared to the connections among themselves. We notice that this assumption is equivalent to assuming that the social networks follow *homophily*, i.e., two linked nodes tend to have the same label. For an extreme example, if the benign region and the Sybil region are isolated from each other, then the network has perfect homophily, i.e., every two linked nodes have the same label. As we will see in Section 3.3, the concept of homophily can better help us incorporate both known benign and Sybil nodes because it explicitly models labels of nodes.

Note that, it is crucial to obtain social networks that satisfy the homophily assumption. Otherwise the detection accuracies of structure-based approaches are limited. For instance, Yang et al. [135] showed that the *friendship network* in RenRen, the largest online social networking site in China, does not satisfy this assumption, and thus structure-based approaches should not be applied to such friendship networks. However, Cao et al. [26] found that the invitation-based friendship network in Tuenti, the leading online social network in Spain, satisfies the homophily assumption, and thus their Sybil ranking mechanism achieves reasonably good performance. In general, online social network operators can obtain social networks that satisfy homophily via two methods. One method is to approximate *trust* relationships between users through looking into user interactions [132], inferring tie strengths [46], and asking users to rate their social contacts [130]. The other method is to preprocess the networks so that they are suitable for structure-based approaches. In particular, operators could first detect and remove compromised benign nodes (e.g., front peers) [38, 129], which decreases the number of attack edges and increases the homophily. Moreover, Alvisi et al. [8] showed that some simple detectors might enforce the social networks to be suitable for structure-based Sybil defenses if the attack edges are established randomly [8].

3.2.2 Design Goals

Our goal is to detect Sybils in a system via taking a social network between the nodes in the system, a small set of known benign nodes, and (optionally) a small set of known Sybils as input. Specifically, we have the following design goals.

1. Sybil classification/ranking: Our goal is to design a mechanism that can either classify nodes into benign and Sybil or that can rank all nodes in descending order of being benign.

2. Incorporating known labels: In many settings, we already know that *some* users are benign and that some users are Sybil. For instance, in Twitter, verified users can be treated as known benign labels and users spreading spam or malware can be treated as known Sybil labels. To improve overall accuracy of the system, the mechanism should have the ability to incorporate information about both known benign and known Sybil labels. It is important that the mechanism should not *require* information about known Sybil labels, but if such information is available, then it should have the ability to use it. This is because in some scenarios, for example when none of the Sybils have performed an attack yet, we might not have known information about any Sybil node.

3. Tolerating label noise: While incorporating information about known benign or known Sybil labels, it is important that the mechanism is resilient to noise in our prior knowledge about these labels. For example, an adversary could compromise the account of a known benign user, or could get a Sybil user whitelisted. We target a mechanism that is resilient when a minority fraction of known labels are incorrect.

4. Scalability: Many distributed systems (e.g., online social networks, reputation systems) have hundreds of millions of users and billions of edges. Thus, for real world applicability, the computational complexity of the mechanism should be low, and the mechanism should also be parallelizable.

Requirements 2, 3, and 4 distinguish our framework from prior work. Sybil classification approaches such as SybilLimit [139] and SybilInfer [34] do not incorporate information about known

Sybil labels (limiting detection accuracy, as shown in Section 3.6), are not resilient to label noise[1], and are not scalable. Sybil ranking approaches such as SybilRank [26] and CIA [133] incorporate information about either known benign or known Sybil labels, but not both. They are also not resilient to label noise.

3.3 SybilBelief model

We introduce our approach SybilBelief, which is scalable, tolerant to label noise, and able to incorporate both known benign labels and Sybil labels.

3.3.1 Model Overview

To quantify the homophily in social networks, we first propose a new probabilistic local rule which determines the reputation score for a node v via aggregating its neighbors' label information. Then, we demonstrate that this local rule can be captured by modeling social networks as Markov Random Fields (MRFs). Specifically, each node in the network is associated with a binary random variable whose state could either be *benign* or *Sybil*, and MRFs define a joint probability distribution over all such random variables. Given a set of known benign labels and/or known Sybil labels, the posterior probabilities that nodes are benign are used to classify or rank them. We adopt Loopy Belief Propagation [94, 99] to approximate the posterior probabilities. Figure 3.1 illustrates how SybilBelief iteratively propagates the beliefs/reputations from the labeled nodes to the remaining ones.

3.3.2 Our Probabilistic Local Rule

Recall that we have a social network $G = (V, E)$ of the nodes in the system. Each node can have two states, i.e., benign or Sybil. Thus, we associate a binary random variable $x_v \in \{-1, 1\}$ with each node. $x_v = +1$ means that node v is a benign node and $x_v = -1$ indicates that node v is Sybil. In the following, we use x_A to represent the set of random variables associated with the nodes in the set A. Moreover, we use $\bar{x}_A$ to denote the observed values of these random variables.

There might exist some prior information about a node v independently from all other nodes in the system. Such prior information could be the content generated by v or its behavior. We model the prior belief of v being benign as follows:

$$P(x_v = +1) = \frac{1}{1 + \exp(-h_v)} , \tag{3.1}$$

where h_v quantifies the prior information about v. More specifically, $h_v > 0$ encodes the scenario in which v is more likely to be benign; $h_v < 0$ encodes the opposite scenario; $h_v = 0$ means prior information is not helpful to determine v's state.

[1]These Sybil classification mechanisms only incorporate one labeled benign node, which makes them not resilient to label noise.

We now introduce $\Gamma_v = \{u|(u,v) \in E\}$, the set of v's neighbors in the social network, and their respective states $\bar{x}_{\Gamma_v}$. When these states are known, the probability of v to be benign is modeled as

$$P(x_v = +1|\bar{x}_{\Gamma_v}) = \frac{1}{1 + \exp(-\sum_{u\in\Gamma_v} J_{uv}\bar{x}_u - h_v)} \ , \tag{3.2}$$

where J_{uv} is the coupling strength between u and v. Specifically, $J_{uv} > 0$ means u and v tend to have the same state; $J_{uv} < 0$ indicates u and v tend to have opposite states; and $J_{uv} = 0$ indicates that there is no coupling between them. In practice, these coupling strengths can encode trust levels between nodes.

Note that our local rule in Equation 3.2 incorporates the homophily assumption via setting $J_{uv} > 0$.

3.3.3 A Pairwise Markov Random Field

We find that the probabilistic local rule introduced in the previous section can be applied by modeling the social network as a pairwise Markov Random Field (MRF). A MRF defines a joint probability distribution for binary random variables associated with all the nodes in the network. Specifically, a MRF is specified with a *node potential* for each node v, which incorporates prior knowledge about v, and with an *edge potential* for each edge (u, v), which represents correlations between u and v. In the context of Sybil detection, we define a node potential $\phi_v(x_v)$ for the node v as

$$\phi_v(x_v) := \begin{cases} \theta_v & \text{if } x_v = 1 \\ 1 - \theta_v & \text{if } x_v = -1 \end{cases}$$

and an *edge potential* $\varphi_{uv}(x_u, x_v)$ for the edge (u, v) as

$$\varphi_{uv}(x_u, x_v) := \begin{cases} w_{uv} & \text{if } x_u x_v = 1 \\ 1 - w_{uv} & \text{if } x_u x_v = -1 \ , \end{cases}$$

where $\theta_v := (1+\exp\{-h_v\})^{-1}$ and $w_{uv} := (1 +\exp\{-J_{uv}\})^{-1}$. Then, the following MRF satisfies the probabilistic local rule.

$$P(x_V) = \frac{1}{Z} \prod_{v\in V} \phi_v(x_v) \prod_{(u,v)\in E} \varphi_{uv}(x_u, x_v) \ ,$$

where $Z = \sum_{x_V} \prod_{v\in V} \phi_v(x_v) \prod_{(u,v)\in E} \varphi_{uv}(x_u, x_v)$ is called the partition function and normalizes the probabilities.

The node potential $\phi_v(x_v)$ incorporates our prior knowledge about node v. Specifically, setting $\theta_v > 0.5$ assigns a higher probability to node v of being benign than Sybil. Setting $\theta_v < 0.5$ models the opposite and $\theta_v = 0.5$ is indifferent between the two states. This is the mechanism through which our framework can incorporate given benign or Sybil labels.

The edge potential $\varphi_{uv}(x_u, x_v)$ encodes the coupling strength of two linked nodes u and v. The larger w_{uv} is, the stronger the model favors u and v to have the same state. More precisely, $w_{uv} > 0.5$ means connected nodes tend to have the same state; $w_{uv} < 0.5$ means connected nodes tend to have opposite states; $w_{uv} = 0.5$ encodes the scenario where there is no coupling between u and v. Note that setting $w_{uv} > 0.5$ encodes our homophily assumption.

3.3.4 Detecting Sybils

First, we modify the above MRF to incorporate known labels. Then we use the modified model to perform both Sybil classification and Sybil ranking.

Leveraging given labels: Suppose we observe states for a set of nodes, and let us denote them as $L \subseteq V$. We use these labels to set the corresponding parameters in the MRF as follows. For any unlabeled node v, i.e., $v \notin L$, we set $\theta_v = 0.5$. Moreover, our inference model (i.e., Equation 4) does not rely on the prior beliefs of those labeled nodes. So we can set θ_v to be any positive value, where $v \in L$. The coupling strength parameter is set as $w_{uv} = w > 0.5$ for any edge (u, v) to model the homophily assumption. In principle, the coupling strength parameters can incorporate the trust levels between nodes, and thus they can be different for different edges. Moreover, the model parameters could also be learned from social networks with a large number of known benign and Sybil nodes. However, such a learning approach would adversely impact the computational complexity of the algorithm. In the following, we will use *natural parameter setting* to denote the setting that $\theta_v = \theta_l > 0$ for $v \in L$, $\theta_v = 0.5$ for $v \in V - L$, and $w_{uv} = w > 0.5$ for any edge (u, v).

To be convenient in our later analysis, we define *evidence potentials* using known labels as follows:

$$\phi_v^L(x_v) = \begin{cases} \phi_v(x_v)\delta_{x_v \bar{x}_v} & \text{if } v \in L \\ \phi_v(x_v) & \text{if } v \notin L \ , \end{cases} \tag{3.3}$$

where $\bar{x}_v$ is the known label of node v, and $\delta_{x_v \bar{x}_v}$ is the Kronecker delta function, i.e., $\delta_{x_v \bar{x}_v} = 1$ if $x_v = \bar{x}_v$, otherwise $\delta_{x_v \bar{x}_v} = 0$.

In our modified model, given the labeled node set L, we can compute the posterior distribution for each unlabeled node v. Specifically, we have

$$\begin{aligned} P(x_v|\bar{x}_L) &= \sum_{x_{V/v}} P(x_V|\bar{x}_L) \\ &= \frac{1}{Z^L} \sum_{x_{V/v}} \prod_{v \in V} \phi_v^L(x_v) \prod_{(u,v) \in E} \varphi_{uv}(x_u, x_v) \end{aligned} \tag{3.4}$$

The posterior probabilities $P(x_v = +1|\bar{x}_L)$ that v is benign given observed nodes L are used to *classify* or *rank* them.

Sybil classification: We map the Sybil classification problem as the following inference problem.

$$y_v = \operatorname*{argmax}_{i \in \{-1,1\}} P(x_v = i|\bar{x}_L)$$

where y_v is the inferred label of v, i.e., $y_v = 1$ indicates that v is benign, otherwise v is Sybil.

Sybil ranking: We use the posterior probabilities $P(x_v = +1|\bar{x}_L)$ to rank all the unlabeled nodes.

Boosting: We use a boosting strategy for our algorithm if either only labeled benign nodes or only labeled Sybil nodes are given. Since both cases are algorithmically equivalent, we take the first case as an example to illustrate our strategy.

We first sample some nodes uniformly at random from the entire system, and we treat them as labeled Sybil nodes. Then we compute the posterior distribution of every unlabeled node. In each

such process, we get a posterior distribution for every node. Furthermore, we repeat this process K times. Thus, we get K posterior distributions for every node, which are denoted as $P_i(x_v|\bar{x}_{L_i})$, where $i = 1, 2, \cdots, K$. We aggregate the K posterior distributions for every node as follows:

$$P(x_v = -1|\bar{x}_L) = \max_i P_i(x_v = -1|\bar{x}_{L_i})$$
$$P(x_v = +1|\bar{x}_L) = 1 - P(x_v = -1|\bar{x}_L)$$

The aggregated posterior distributions are then used to classify or rank nodes. This boosting strategy works because our model can update prior beliefs and therefore is robust to label noise (see Section 3.6 and 3.5). In each boosting trial, if some of the sampled nodes are true Sybil nodes, then this trial can detect a subset of Sybil nodes. Due to the robustness to label noise, even if some sampled nodes are actually benign, the propagation of Sybil beliefs among the benign region is limited once the number of such sampled nodes is smaller than the number of labeled benign nodes. Thus in our experiments, we limit the number of sampled Sybil nodes in the boosting process by the number of labeled benign nodes.

3.4 SybilBelief Learning Algorithm

Our Sybil classification and ranking mechanisms rely on the computation of the posterior distributions given in Equation 3.4. Generally, there are two major ways to infer such posterior distributions: *sampling* and *variational inference*. We adopt variational inference to learn the posterior distributions since it is more scalable than sampling approaches such as Gibbs sampling. Specifically, we adopt Loopy Belief Propagation (LBP) to calculate the posterior distributions for each node.

Loopy Belief Propagation (LBP) [99]: The basic step in LBP is to pass messages between neighboring nodes in the system. Message $m_{uv}^{(t)}(x_v)$ sent from u to v in the tth iteration is

$$m_{uv}^{(t)}(x_v) = \sum_{x_u} \phi_u^L(x_u)\varphi_{uv}(x_u, x_v) \prod_{k \in \Gamma(u)/v} m_{ku}^{(t-1)}(x_u)$$

Here, $\Gamma(u)/v$ is the set of all neighbors of u, except the receiver node v. This encodes that each node forwards a product over incoming messages of the last iteration and adapts this message to the respective receiver based on the coupling strength with the receiver.

For social networks without loops (i.e., for trees), LBP is guaranteed to converge and to compute the exact posterior distribution. For networks with loops, LBP approximates the posterior probability distribution without convergence guarantees. However, in practical applications and benchmarks in the machine learning literature [94], LBP has demonstrated good results and is, today, a widely used technique.

Stopping condition: The message passing iterations stop when the changes of messages become negligible (e.g., L1 distance of changes becomes smaller than 10^{-3}) or the number of iterations exceeds a predefined threshold. After stopping, we estimate the posterior probability distribution $P(x_v|\bar{x}_L)$ by

$$P(x_v|\bar{x}_L) \propto \phi_v^L(x_v) \prod_{k \in \Gamma(v)} m_{kv}^{(t)}(x_v)$$

Scalability: The complexity of one LBP iteration is $O(m)$, where m is the number of edges. So the total complexity is $O(m*d)$, where d is the number of LBP iterations. Note that social networks are often sparse graphs [52, 91]. Thus we have $O(m * d) = O(n * d)$, where n is the number of nodes. Moreover, we find that setting d to be 10 already achieves good results in our experiments. Furthermore, LBP can be easily parallelized. Specifically, we can distribute nodes in the system to multiple processors or computer nodes, each of which collects messages for nodes assigned to them.

Analytical solutions: It's hard to derive analytical solutions for our model in general settings. However, when the network is a tree, one labeled benign node (denoted as b) and one labeled Sybil node (denoted as s) are observed, and the parameters are chosen to be natural settings, we can analytically derive labels for all unlabeled nodes. This is formally stated as the following theorem.

Theorem 2. *For any unlabeled node v with distance d_{vb} to b and d_{vs} to s, according to our local inference model, v is labeled as benign if $d_{vb} < d_{vs}$; v is labeled as Sybil if $d_{vb} > d_{vs}$; and v is indistinguishable between benign and Sybil if $d_{vb} = d_{vs}$.*

The proof can be found in Appendix A.1.

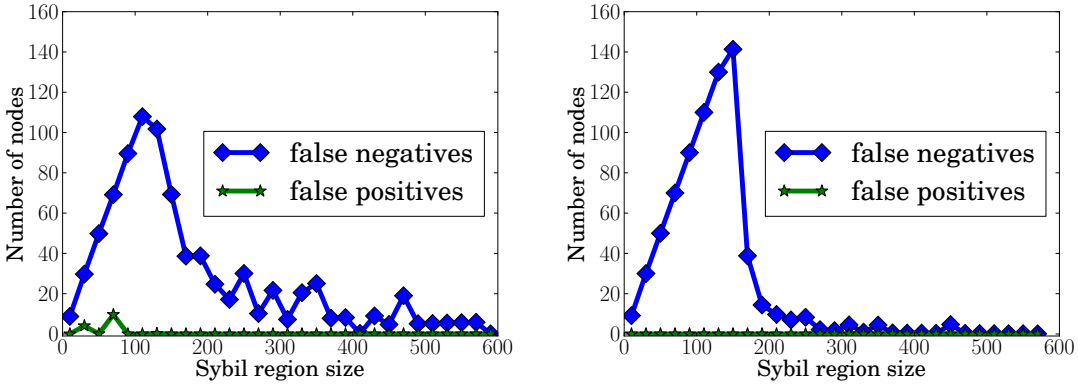
3.5 Evaluating SybilBelief

We evaluate the influence of various factors including parameter settings in SybilBelief, the number of labels, label sites, label noises, mixing time of the social networks, and scenarios where only labeled benign or Sybil nodes are observed, on the performance of SybilBelief. Since these experiments require social networks with various sizes, we will use well-known network generators (e.g., Erdos-Renyi model (ER) [39] and the Preferential Attachment (PA) model [11]) to synthesize both the benign region and the Sybil region. We will also study the impacts of different network generators. Furthermore, throughout these experiments, we view SybilBelief as a classification mechanism.

In the basic experimental setup, we adopt the PA model to generate both the benign region and the Sybil region with an average degree of 10, add 500 attack edges between them uniformly at random, and fix the benign region to have 1000 nodes; we set the SybilBelief model parameters as $\theta_v = 0.5$ for any node v, and $w_{uv} = w = 0.9$ for any edge (u, v); we assume one labeled benign node and one labeled Sybil node; the labeled benign (Sybil) nodes are uniformly sampled from the benign (Sybil) network. When we study the impact of one factor, we fix other factors to be the same as in the basic setup and vary the studied one. Moreover, all of our results reported in the following are averaged over 100 trials.

Impact of network generators: We first study the impacts of network generators on the performances of SybilBelief. To this end, we choose the ER network generator [39] and the PA network generator [11].

For each triple (benign region, Sybil region, attack edges) setting, we can compute the false negatives and false positives. Figure 3.2 shows false negatives and positives as a function of the Sybil region size. We find that both false negatives and false positives first increase and then decrease as we increase the Sybil region size. The reason for this sudden decrease phenomenon is that when the



(a) ER-ER

(b) PA-PA

Figure 3.2: The false negatives and false positives as a function of the Sybil region size. (a) Networks are synthesized by the ER model. (b) Networks are synthesized by the PA model. There exists an optimal strategy for the attackers.

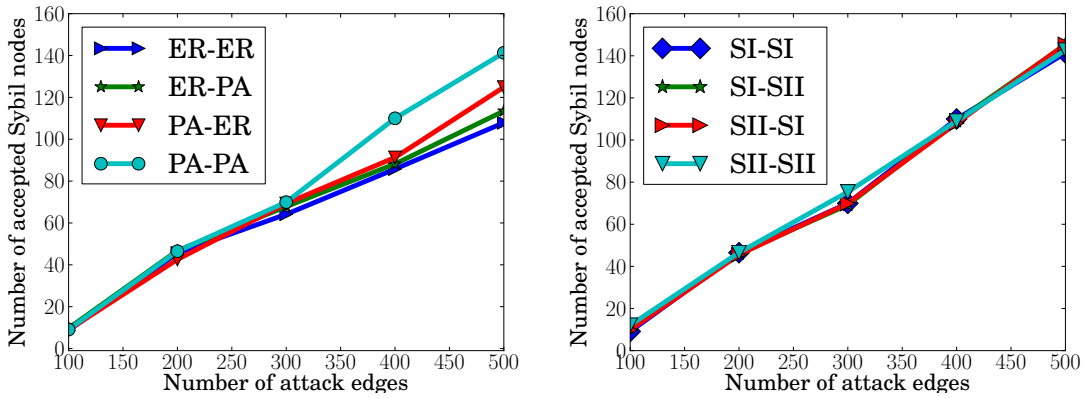
Sybil region size is bigger than some threshold, the homophily is strong enough so that SybilBelief can easily distinguish between the benign and Sybil regions. For a (benign region, attack edges) setting, we can search for the maximum false negatives and false positives via increasing the Sybil region size. We denote these maximum false negatives as *accepted Sybil nodes* and maximum false positives as *rejected benign nodes*, which will be used as the metrics to evaluate Sybil classification systems.

Figure 3.3a shows the accepted Sybil nodes as a function of the number of attack edges while fixing the benign region to be the setting in the basic setup for different network generator combinations. We find that PA-generated networks have more accepted Sybil nodes than ER-generated networks. Our findings imply that attackers should generate scale-free networks in order to inject more Sybils to the benign region. The rejected benign nodes are always less than 5 (i.e., the false positive rates are smaller than 0.5% since we have 1000 benign nodes) in these experiments. We don't show them due to the limited space.

Impact of label sites: In practice, some known labeled nodes might be end points of attack edges while others might be far away from attack edges. So one natural question is which ones to be selected as the input labels for SybilBelief. Specifically, we consider the following scenarios.

- **SI:** Labeled benign (Sybil) nodes are not end points of attack edges.
- **SII:** Labeled benign (Sybil) nodes are end points of attack edges. This could correspond to the scenario where sophisticated attackers obtain knowledge about the labels used in SybilBelief and establish targeted attacks.

Figure 3.3b shows the accepted Sybil nodes as a function of the number of attack edges for the four combinations of the label sites. We find that the label sites have no influence on the number



(a) Network generator impact

(b) Label sites impact

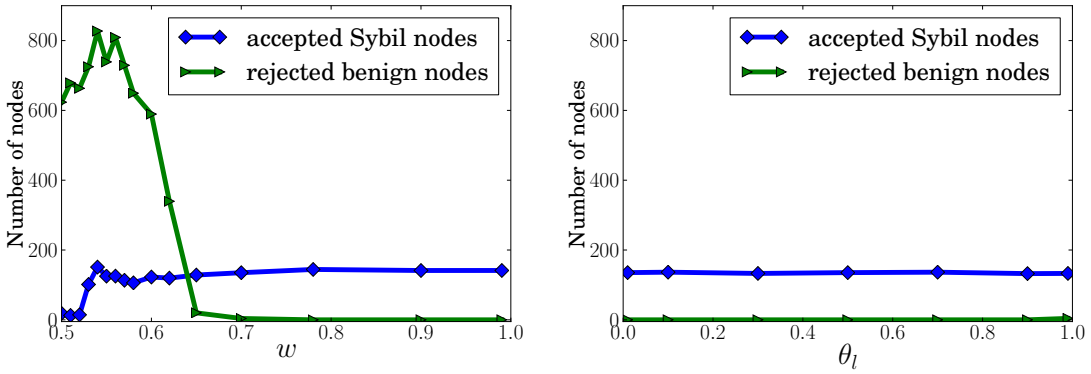
Figure 3.3: The accepted Sybil nodes as a function of the number of attack edges. (a) Impact of different network generators. The notation $M1 - M2$ means we use $M1$ to produce the benign region and $M2$ to synthesize the Sybil region. We observe that the attackers should design their Sybil regions to approach scale-free networks in order to inject more Sybils. (b) Impact of different combinations of label sites. Our algorithm SybilBelief is robust to label sites.

of accepted Sybil nodes. Again, the rejected benign nodes are always less than 5 (i.e., the false positive rates are smaller than 0.5%), which are not shown due to the limited space. Our results imply that the administrator could simply select benign/Sybil nodes uniformly at random as input labels for SybilBelief.

Impact of the model parameters: In SybilBelief, there exists one parameter w_{uv} corresponding to the homophily strength of the edge (u, v). We set $w_{uv} = w$ for all edges, and Figure 3.4 illustrates the impact of w on the performance of SybilBelief. To demonstrate that our model is independent with θ_v for $v \in L$, Figure 3.4 also shows the case in which we vary θ_l, where $\theta_l = \theta_v$ for $v \in L$. We observe that there exists a phase transition point w_0 (e.g., $w_0 \approx 0.65$ in these experiments) for w. When $w > w_0$, SybilBelief achieves a good tradeoff between accepted Sybil nodes and rejected benign nodes.

Impact of the number of labeled nodes: Figure 3.5 shows the influence of various number of labels on the performance of SybilBelief. We observe that the number of accepted Sybil nodes increases dramatically when the labeled benign and Sybil nodes are highly imbalanced, i.e., their ratio is bigger than 10 or smaller than 0.1. Again, the rejected benign nodes are always less than 5 (i.e., the false positive rates are smaller than 0.5%), which we don't show here due to the limited space. When the ratio is between 0.1 and 10, the number of accepted Sybil nodes is stable across various number of labeled benign nodes. Having more labeled Sybil nodes helps SybilBelief accept fewer Sybil nodes. However, these achieved margins are just the more labeled Sybils[2]. So we conclude that SybilBelief only needs one label for each community.

[2]We will not accept these labeled Sybil nodes.



(a) $\theta_l = 0.50$ (b) $w = 0.90$

Figure 3.4: The number of accepted Sybil nodes and rejected benign nodes as a function of the model parameters w and θ_l. (a) $\theta_l = 0.50$ and we vary w. (b) $w = 0.90$ and we vary θ_l. We observe that there exists a phase transition point w_0 (e.g., $w_0 \approx 0.65$ in our experiments) for the parameter w. SybilBelief is robust for $w > w_0$. Moreover, we confirm that SybilBelief performance is independent with θ_l once it's bigger than 0.

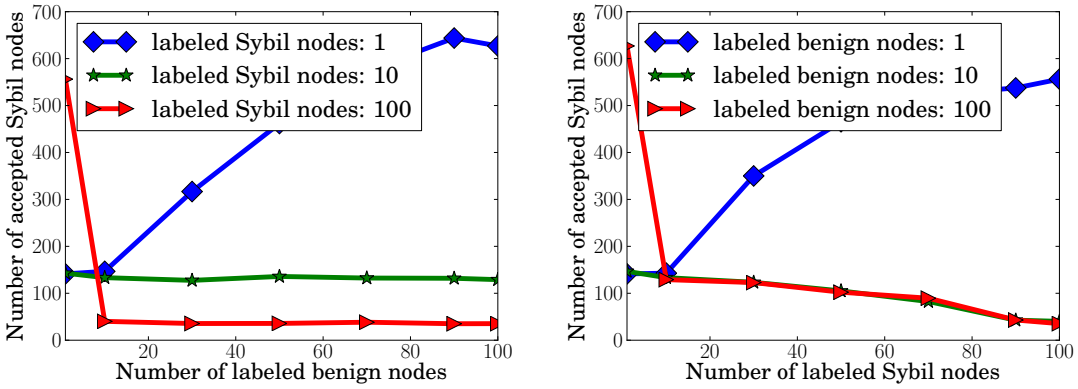
Impact of label noise: In practice, we could use a machine learning classifier or crowdsourcing system to obtain labels. For instance, Thomas et al. [117] used a classifier to assign labels for Twitter accounts. Wang et al. [128] proposed to label nodes via a crowdsourcing platform such as Amazon Mechanical Turk[3]. Unfortunately, labels obtained from these approaches often have noise. For example, labels got from crowdsourcing could have noise up to 35% [128]. Furthermore, an adversary could compromise a known benign node, or could get a Sybil node whitelisted. Thus, one natural question is how label noise affects SybilBelief's performance.

Figure 3.6a shows the influences of such label noise. Unsurprisingly, we find that SybilBelief accepts more Sybil nodes and rejects more benign nodes when a larger fraction of labels are incorrect. However, with even 49% noise[4], SybilBelief still performs very well, i.e., SybilBelief with 49% noise only accepts three times more Sybil nodes than SybilBelief without noise, and its rejected benign nodes are always less than 3 (i.e., the false positive rates are smaller than 0.3%). Furthermore, we find that our algorithm can detect the incorrect labels with 100% accuracy.

Impact of community structure: Most existing Sybil detection mechanisms [26, 34, 120, 121, 125, 133, 139, 140] rely on the assumption that the benign region is fast mixing. However, Mohaisen [93] showed that many real-world social networks may not be as fast-mixing as was previously thought. Furthermore, Viswanath et al. [125] showed that the performance of existing Sybil detection methods decrease dramatically when the benign region consists of more and more communities (i.e., the mixing time is larger and larger).

[3] `https://www.mturk.com/mturk/welcome`

[4] $\geq$ 50% noise makes any algorithm accept infinite number of Sybils.



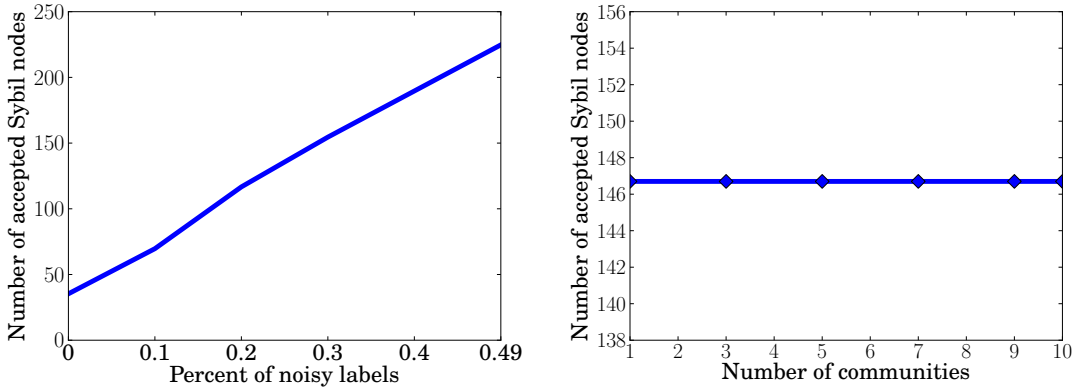
(a) (b)

Figure 3.5: The number of accepted Sybil nodes as a function of the number of labeled benign and Sybil nodes. (a) The number of Sybil nodes is fixed while varying the number of benign labels. (b) The number of labeled benign nodes is fixed while increasing the number of Sybil labels. We observe that SybilBelief only requires one label per community.

Similar to the experiment done by Viswanath et al., we study the impact of the community structure (i.e., the mixing time) on the performance of SybilBelief. Specifically, for the benign region, we use PA to generate k independent communities. For the ith community, we link it to the previous $i-1$ communities via 10 random edges. The Sybil region is one community generated by PA. We randomly sample 1 Sybil label from the Sybil region and 10 benign labels from the benign region such that each community has at least 1 labeled node. Figure 3.6b shows the number of accepted Sybil nodes as a function of the number of communities in the benign region. The number of rejected benign nodes is always close to 0 and not shown here due to limited space. We conclude that SybilBelief is robust to community structure in the benign region.

Boosting with only labeled benign or Sybil nodes: If we only observe labeled benign or Sybil nodes, we can boost our algorithm by sampling some nodes and treating them as labeled Sybil or benign nodes. Since only observing labeled benign nodes is symmetric to only observing labeled Sybil nodes in the algorithmic perspective, we take the former case as an example to illustrate our boosting strategy.

In these experiments, we assume 100 benign labels are observed but no Sybil labels are obtained. So we sample 10 nodes uniformly at random, and treat them as the Sybil labels. Other factors are fixed to be the natural settings. With a given benign region, a Sybil region, attack edges between them and the benign labels, we repeat this Sybil labels sampling process for k times, and ensemble their results in the way described in our algorithm section. We use SB-B-k to denote the case where we use boosting strategy and the number of boosting trials is k. Figure 3.7 compares SB-B-1, SB-B-10 and SB. In the setting of SybilBelief, we assume 100 benign labels and 10 Sybil labels. We conclude that with only partial labels, our boosting strategy can still achieve performance comparable to the scenario where both benign and Sybil labels are observed. Furthermore, the number of boosting trials balances between the accepted Sybil nodes and rejected benign nodes.



(a) Label noise impact

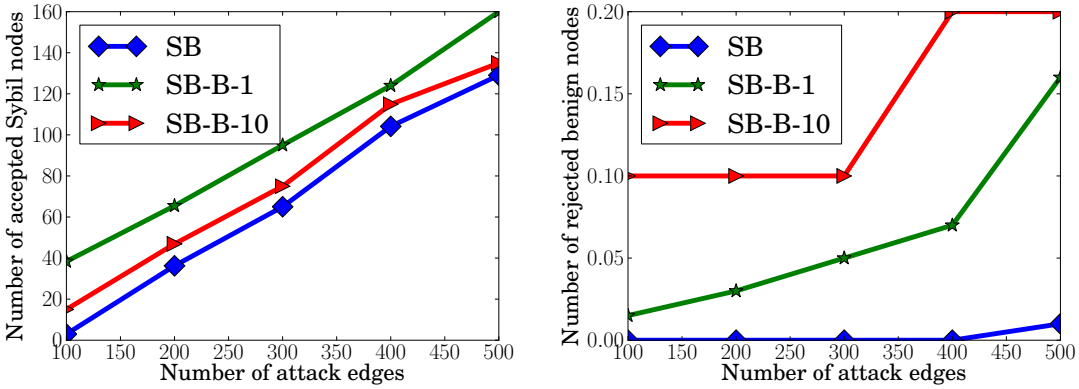
(b) Mixing time impact

Figure 3.6: Impact of label noise and community structure of the benign region on the number of accepted Sybil nodes. (a) We have 100 labeled benign and Sybil nodes. The x-axis is the percent of both noisy benign and Sybil labels. We find that SybilBelief can tolerate 49% of labels to be incorrect. (b) The benign region consists of multiple communities. We observe that SybilBelief is robust to community structures.

More specifically, boosting with more trials accepts fewer Sybil nodes but rejects more benign nodes.

Summary: We have the following observations:

- SybilBelief accepts more Sybil nodes in the PA-generated networks than in the ER-generated networks. This implies that attackers should design their Sybil regions to approach scale-free networks.
- SybilBelief is robust to label sites.
- There exists a phase transition point w_0 (e.g., $w_0 \approx 0.65$ in our experiments) for the parameter w. SybilBelief performance is robust for $w > w_0$.
- SybilBelief only requires one label per community.
- SybilBelief can tolerate 49% of labels to be incorrect. Moreover, SybilBelief can detect incorrect labels with 100% accuracy.
- SybilBelief is robust to community structures in the benign region.
- With only benign or Sybil labels, our boosting strategy can still achieve performances comparable to the case where both benign and Sybil labels are observed. Furthermore, the number of boosting trials can be used to balance between accepted Sybil nodes and rejected benign nodes.



(a) Accepted Sybil nodes

(b) Rejected benign nodes

Figure 3.7: Boosting with only labeled benign nodes. The numbers in the legend are the number of boosting trials. With only labeled benign nodes, SybilBelief (SB) with boosting can achieve performances comparable to the case where both benign and Sybil labels are observed. Furthermore, the number of boosting trials balances between the accepted Sybil nodes and rejected benign nodes.

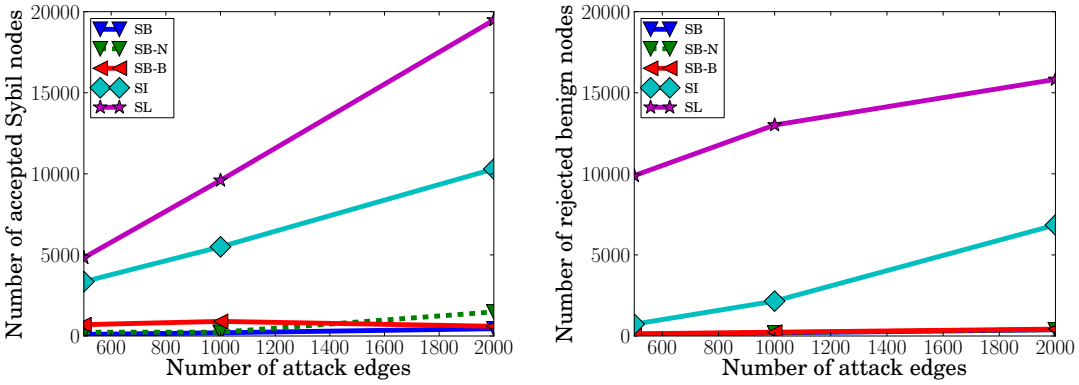
Table 3.1: Dataset statistics.

Metric	Facebook	Slashdot	Email
Nodes	43,953	82,168	224,832
Edges	182,384	504, 230	339,925
Ave. degree	8.29	12.27	3.02

3.6 Comparing SybilBelief with Previous Approaches

We compare our approach SybilBelief and its variants with previous Sybil classification and Sybil ranking mechanisms. The benign regions are real social networks while the Sybil regions are either generated by network generators such as Preferential Attachement [11] or duplicates of the benign regions[5]. We find that SybilBelief and its variants perform orders of magnitude better than previous Sybil classification systems and significantly better than previous Sybil ranking systems. Furthermore, in contrast to previous approaches, SybilBelief is robust to label noise.

[5]We acknowledge that one limitation of our work is that we didn't evaluate these approaches using real Sybil users. This is because it is hard for us to obtain a social network which represents trust relationships between users and which includes ground truth information about benign and Sybil users.



(a) Accepted Sybil nodes

(b) Rejected benign nodes

Figure 3.8: The number of accepted Sybil nodes and rejected benign nodes as a function of the number of attack edges. The benign region is the Facebook network, and the Sybil regions are synthesized by PA model. We observe that SB, SB-N, and SB-B all work an order of magnitude better than previous classification systems. Furthermore, we find that incorporating both benign and Sybil labels increases the performance of our algorithm.

3.6.1 Experimental Setups

Dataset description: We use three social networks representing different application scenarios. These three datasets are denoted as Facebook, Slashdot, and Email.

Facebook is an interaction graph from the New Orleans regional network [123]. In this graph, nodes are Facebook users and a link is added between two users if they comment on each other's wall posts at least once. The Email network was generated using email data from a large European research institution [75]. Nodes are email addresses and there exists a link between two nodes if they communicate with each other at least once. Slashdot is a technology-related news website, which allows users to tag each other as friends or foes. The Slashdot network thus contains friend/foe links between the users. We choose the largest connected component from each of them in our experiments. Table 3.1 summarizes the basic dataset statistics.

We note that some previous work removes nodes with degrees smaller than a threshold from the social networks. For instance, SybilLimit [139] removes nodes with degree smaller than 5 and SybilInfer [34] removes nodes with degree smaller than 3. Mohaisen et al. [93] found that such preprocessing will prune a large portion of nodes. Indeed, social networks often have a long-tail degree distribution (e.g., power-law degree distribution [30] and lognormal degree distribution [52]), in which most nodes have very small degrees. Thus, a large portion of nodes are pruned by such preprocessing.

Therefore, such preprocessing could result in high false negative rates or high false positive rates depending on how we treat the pruned nodes. If we treat all the pruned nodes whose degrees are smaller than a threshold as benign nodes, then an attacker can create many Sybil nodes with

degree smaller than the threshold, resulting in high false negative rates, otherwise a large fraction of benign nodes will be treated as Sybil nodes, resulting in high false positive rates. So we do not perform such preprocessing to the three social networks.

Metrics: Following previous work [34, 120, 121, 139], the metrics we adopt for evaluating Sybil classification mechanisms are the number of *accepted Sybil nodes* and the number of *rejected benign nodes*, which are the maximum number of accepted Sybils and the maximum number of rejected benign nodes for a given number of attack edges. To evaluate the Sybil ranking mechanisms, similar to previous work [26, 125], we adopt AUC, Area Under the Receiver Operating Characteristic (ROC) Curve. Given a ranking of a set of benign and Sybil nodes, AUC is the probability that a randomly selected benign node ranks before a randomly selected Sybil node. We compute the AUC in the manner described in [56].

Parameter settings: We set $\theta_v = 0.5$ for any unlabeled node, which means we don't distinguish a node between benign and Sybil if no prior information is available. For those labeled nodes, their states are fixed and thus their corresponding parameters θ_v don't influence our model. So we also set them to be 0.5 by simplicity. Furthermore, as we find that SybilBelief is robust to the parameter w_{uv} when it is bigger than the transition point, and thus we set $w_{uv} = w = 0.90$ for any edge (u, v) in all experiments. In the experiments of boosting our SybilBelief with only labeled benign nodes, the number of boosting trials is set to be 10.

3.6.2 Compared Approaches

We compare SybilBelief with two classical Sybil classification mechanisms, i.e., SybilLimit [139] and SybilInfer [34], and two recent Sybil ranking mechanisms, i.e., SybilRank [26] and Criminal account Inference Algorithm [133]. Table 3.2 summarizes the notations of these algorithms. Note that Viswanath et al. [125] showed that SybilLimit and SybilInfer are essentially also ranking systems. However, Cao et al. [26] showed that SybilRank outperforms them in terms of rankings. Thus, we will compare SybilBelief with SybilLimit and SybilInfer via treating them as Sybil classification mechanisms.

SybilLimit (SL): SL requires each node to run a few random routes starting from themselves. For each pair of nodes u and v, if v's random routes have enough intersections with u's and these intersections satisfy a balance constraint, then u accepts v as a benign node. In our experiments, we random sample enough such pairs of nodes to estimate the number of accepted Sybil nodes and the number of rejected benign nodes.

SybilInfer (SI): SI relies on a special random walk, i.e., the stationary distribution of this random walk is uniform. Given a set of random walk traces, SI infers the posterior probability of any node being benign. Note that SI can only incorporate one labeled benign node.

SybilRank (SR): SR performs random walks starting from a set of benign users. Specifically, with h labeled benign nodes, SR designs a special initial probability distribution over the nodes, i.e., probability $1/h$ for each of the labeled benign nodes and probability 0 for all other nodes, and SR iterates the random walk from this initial distribution for $\log(n)$ iterations, where n is the number of nodes in the system. It is well known that this random walk is biased to high-degree nodes. Thus, SybilRank normalizes the final probabilities of nodes by their degrees and uses the normalized probabilities to rank nodes. Note that SR can only incorporate benign labels.

Table 3.2: Notations of algorithms.

Notation	Description
SL	SybilLimit [139]
SI	SybilInfer [34]
SR	SybilRank [26]
SR-N	SybilRank [26] with label noise
CIA	Criminal account Inference Algorithm [133]
CIA-N	CIA with label noise
SB	SybilBelief
SB-N	SybilBelief with label noise
SB-B	SybilBelief with only labeled benign nodes
Random	Randomly assign a reputation score between 0 and 1 to each node

SybilRank-Noise (SR-N): We use this abbreviation to denote the case where the labels given to SybilRank are noisy, i.e., some of the labeled benign nodes are actually Sybils.

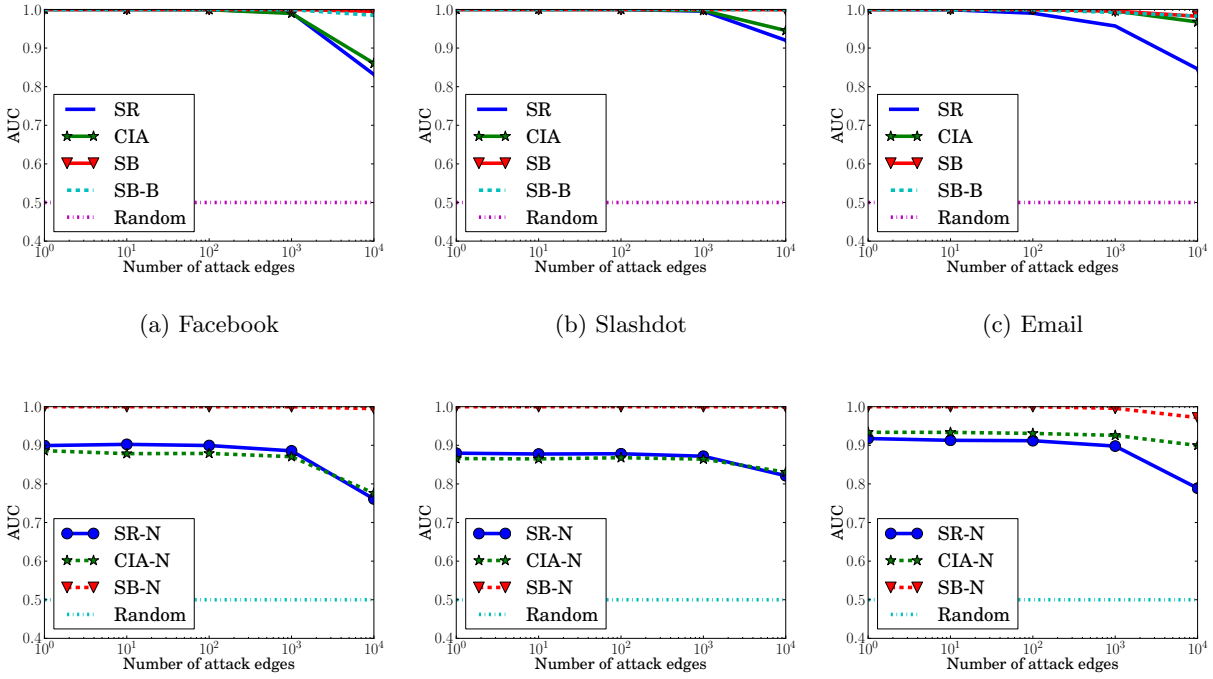
Criminal account Inference Algorithm (CIA): CIA is also based on a random walk with a special initial probability distribution, but it differs from SR in two major aspects. First, CIA starts the random walk from labeled Sybil nodes. Second, in each iteration, CIA restarts the random walk from the special initial probability distribution with probability $1 - \alpha$. Since the random walk is started from Sybil nodes, 1 - p_v is treated as the reputation score of node v, where p_v is the stationary probability of v. Then those reputation scores are used to rank nodes. As was proposed in the original paper [133], we set the restart parameter α to 0.85. Note that CIA can only incorporate labeled Sybil nodes.

Criminal account Inference Algorithm-Noise (CIA-N): Analogous to SR-N, we use this abbreviation to denote the case where the input labels are partially wrong.

We abbreviate variants of our method by *SybilBelief (SB), SybilBelief-Noise (SB-N) and SybilBelief-Boosting (SB-B)*. SB incorporates both benign and Sybil labels; SB-N indicates the scenario where some of the labeled benign and Sybil nodes are noise; SB-B means only benign labels are observed, and we sample some nodes uniformly at random from the entire network and treat them as Sybil labels.

3.6.3 Comparing with Sybil Classification Mechanisms

In order to calculate the number of *accepted Sybil nodes* and the number of *rejected benign nodes*, we need to evaluate these algorithms on Sybil regions with various sizes. Thus, we use a network generator to synthesize the Sybil region and add attack edges between it and the benign region uniformly at random. In our experiments, we adopt scale-free network generator Preferential Attachment (PA) [11]. However, we still use real social networks as the benign regions. We assume 100 labeled benign nodes, which are sampled from the benign region uniformly at random. Since SybilLimit and SybilInfer don't incorporate labeled Sybil nodes, we assume only one labeled Sybil



(a) Facebook (b) Slashdot (c) Email

(d) Facebook with label noise (e) Slashdot with label noise (f) Email with label noise

Figure 3.9: AUC as a function of the number of attack edges on different social networks. For each social network, we treat it as both the benign and Sybil regions, and add attack edges between them uniformly at random. The AUCs are averaged over 10 trials for each number of attack edges. We observe that both SB and SB-B outperform previous approaches. Furthermore, in contrast to previous approaches, SB is robust to label noise.

node for our approaches, which is sampled from the Sybil region uniformly at random. In the experiments with label noise, we assume 10 out of the labeled benign nodes are incorrect. In the boosting experiments, we randomly sample 10 nodes from the entire network and treat them as Sybil labels.

Figure 3.8 shows the comparison results on the Facebook dataset. We only show results on the Facebook network because SybilLimit and SybilInfer are not scalable to other social networks we consider. We have several observations. First, SybilBelief performs orders of magnitude better than SybilLimit and SybilInfer in terms of both the number of accepted Sybil nodes and the number of rejected benign nodes. Second, unsurprisingly, SB-N and SB-B don't work as well as SB, but the margins are not significant. Since SybilLimit and SybilInfer can only incorporate one labeled benign node, the gains of SB and SB-B over them come from (a) incorporating more labels and (b) the use of the Markov Random Fields and the Belief Propagation algorithm.

3.6.4 Comparing with Sybil Ranking Mechanisms

Following previous work [8, 26, 34, 125, 139], we synthesize networks as follows: we use each real-world social networks as both the benign region and the Sybil region, and then we add attack edges between them uniformly at random.

We assume that 100 labeled benign nodes and 100 labeled Sybil nodes are given. For the experiments with label noise, we assume 10 out of the 100 benign and Sybil labels are wrong. For SB-B, we randomly sample 100 nodes from the entire network including both benign and Sybil regions and treat them as Sybil labels.

Figure 3.9 shows the AUC of the rankings obtained by various approaches in the three different social networks. From these figures, we can draw several conclusions.

First, SB and SB-B consistently outperform SR and CIA across different social networks. Furthermore, the improvements are more significant as the number of attack edges becomes larger. This is because our proposal can incorporate both benign and Sybil labels.

Second, with label noise, performances of both SR and CIA degrade dramatically. However, SB's performance is almost unchanged. SB is robust to noise because SB incorporates labels probabilistically. Specifically, a node receives beliefs from all of its neighbors. Since the majority of labels are correct, the beliefs from the wrongly labeled nodes are dominated by those from the correctly labeled nodes.

Third, CIA consistently performs better than SR, which can be explained by the fact that CIA restarts the random walk from the special initial probability distribution with some probability in each iteration.

3.6.5 Summary

We have performed extensive evaluations to compare our approach SybilBelief and its variants with previous approaches on graphs with synthetic Sybil nodes. From the comparison results, we conclude that SybilBelief and its variants perform orders of magnitude better than previous Sybil classification systems and significantly better than previous Sybil ranking systems. Furthermore, in contrast to previous approaches, SybilBelief is robust to label noise.

3.7 Summary of Results

We propose SybilBelief, a semi-supervised learning framework, to detect Sybil nodes in distributed systems. SybilBelief takes social networks among the nodes in the system, a small set of known benign nodes, and, optionally, a small set of known Sybil nodes as input, and then SybilBelief propagates the label information from the known benign and/or Sybil nodes to the remaining ones in the system.

We extensively evaluate the influence of various factors including parameter settings in the SybilBelief, the number of labels, and label noises on the performance of SybilBelief. Moreover, we compare SybilBelief with state-of-the-art Sybil classification and ranking approaches on real-world social network topologies. Our results demonstrate that SybilBelief performs orders of magnitude

better than previous Sybil classification mechanisms and significantly better than previous Sybil ranking mechanisms. Furthermore, SybilBelief is more resilient to noise in our prior knowledge about known benign nodes and known Sybils.

Chapter 4

Attribute Inference and Link Prediction

In this chapter, we discuss the inference of private user attributes and hidden social connections between users in online social networks. This work was presented at the *6th ACM Workshop on Social Network Mining and Analysis, 2012*, at Beijing, China, and it also appeared in *ACM Transactions on Intelligent Systems and Technology, Vol.5, No.2*. This is a joint work with Ameet Talwalkar, Lester Mackey, Ling Huang, Richard Shin, Emil Stefanov, Elaine Shi, and Dawn Song.

4.1 Introduction

Online social networks (e.g., Facebook, Google+) have become increasingly important resources for interacting with people, processing information and diffusing social influence. Understanding and modeling the mechanisms by which these networks evolve are therefore fundamental issues and active areas of research.

The classical *link prediction problem* [78] has attracted particular interest. In this setting, we are given a snapshot of a social network at time t and aim to predict links (e.g., friendships) that will emerge in the network between t and a later time t'. Alternatively, we can imagine the setting in which some links existed at time t but are missing at t'. In online social networks, a change in privacy settings often leads to missing links, e.g., a user on Google+ might decide to hide her family circle between time t and t'. The missing link problem has important ramifications as missing links can alter estimates of network-level statistics [70], and the ability to infer these missing links raises serious privacy concerns for social networks. Since the same algorithms can be used to predict new links and missing links, we refer to these problems jointly as link prediction.

Another problem of increasing interest revolves around node attributes [142]. Many real-world networks contain rich categorical node attributes, e.g., users in Google+ have profiles with attributes including employer, school, occupation and places lived. In the *attribute inference problem*, we aim

to populate attribute information for network nodes with missing or incomplete attribute data. This scenario often arises in practice when users in online social networks set their profiles to be publicly invisible or create an account without providing any attribute information. The growing interest in this problem is highlighted by the privacy implications associated with attribute inference as well as the importance of attribute information for applications including people search [3, 4], collaborative filtering [86] and user identity resolution [13].

In this work, we simultaneously use network structure and node attribute information to improve performance of both the link prediction and the attribute inference problems, motivated by the observed interaction and homophily between network structure and node attributes. The principle of social influence [42], which states that users who are linked are likely to adopt similar attributes, suggests that network structure should inform attribute inference. Other evidence of interaction [68, 72] shows that users with similar attributes, or in some cases antithetical attributes, are likely to link to one another, motivating the use of attribute information for link prediction. Additionally, previous studies [42, 71] have empirically demonstrated those effects on real-world social networks, providing further support for considering both network structure and node attribute information when predicting links or inferring attributes.

However, the algorithmic question of how to simultaneously incorporate these two sources of information remains largely unanswered. The relational learning [88, 110, 116, 141], matrix factorization and alignment [87, 109] based approaches have been proposed to leverage attribute information for link prediction, but they suffer from scalability issues. More recently, Backstrom and Leskovec [9] presented a Supervised Random Walk (SRW) algorithm for link prediction that combines network structure and edge attribute information, but this approach does not fully leverage node attribute information as it only incorporates node information for neighboring nodes. For instance, SRW cannot take advantage of the common node attribute San Francisco of u_2 and u_5 in Fig. 4.1 since there is no edge between them.

Yin et al. [137, 138] proposed the use of *Social-Attribute Network* (SAN) to gracefully integrate network structure and node attributes in a scalable way. They focused on generalizing Random Walk with Restart (RWwR) algorithm to the SAN model to predict links as well as infer node attributes. In this work, we generalize several leading supervised and unsupervised link prediction algorithms [58, 78] to the SAN model to both predict links and infer missing attributes. We evaluate these algorithms on a novel, large-scale Google+ dataset, and demonstrate performance improvement for each of them. Moreover, we make the novel observation that inferring attributes could help predict links, i.e., link prediction accuracy is further improved by first inferring missing node attributes.

4.2 Problem Definition

In our problem setting, we use an undirected[1] graph $G = (V, E)$ to represent a social network, where edges in E represent interactions between the $N = |V|$ nodes in V. In addition to network structure, we have categorical attributes for nodes. For instance, in the Google+ social network, nodes are users, edges represent friendship (or some other relationship) between users, and node attributes are derived from user profile information and include fields such as employer, school, and hometown. In this work we restrict our focus to categorical variables, though in principle other

[1] Our model and algorithms can also be generalized to directed graphs.

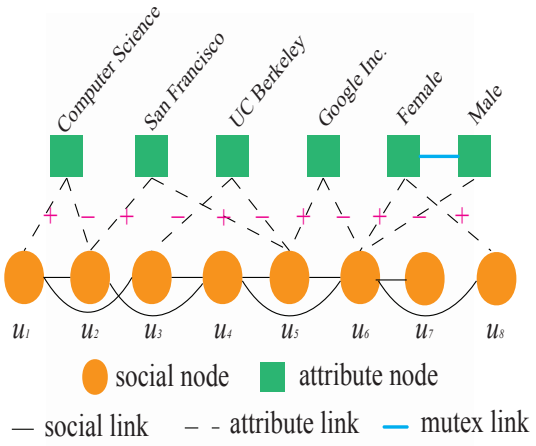


Figure 4.1: Illustration of a Social-Attribute Network (SAN). The link prediction problem reduces to predicting social links while the attribute inference problem involves predicting attribute links.

types of variables, e.g., live chats, email messages, real-valued variables, etc., could be clustered into categorical variables via vector quantization, or directly discretized to categorical variables.

We use a binary representation for each categorical attribute. For example, various employers (e.g., Google, Intel and Yahoo) and various schools (e.g., Berkeley, Stanford and Yale) are each treated as separate binary attributes. Hence, for a specific social network, the number of distinct attributes M is finite (though M could be large). Attributes of a node u are then represented as a M-dimensional trinary column vector $\vec{a}_u$ with the i^{th} entry equal to 1 when u has the i^{th} attribute (*positive attribute*), -1 when u does not have it (*negative attribute*) and 0 when it is unknown whether or not u has it (*missing attribute*). We denote by $A = [\vec{a}_1\ \vec{a}_2 \cdots \vec{a}_N]$ the attribute matrix for all nodes. Note that certain attributes (e.g. Female and Male, age of 20 and 30) are mutually exclusive. Let L be the set of all pairs of mutually exclusive attributes. This set constrains the attribute matrix A so that no column contains a 1 for two mutually exclusive attributes.

We define the link prediction problem as follows:

Definition 1 (Link Prediction Problem)**.** *Let $T_i = (G_i, A_i, L_i)$ and $T_j = (G_j, A_j, L_j)$ be snapshots of a social network at times i and j. Then the link prediction problem involves using T_i to predict the social network structure G_j. When $i < j$, new links are predicted. When $i > j$, missing links are predicted.*

We work with three snapshots of the Google+ network crawled at three successive times, denoted $T_1 = (G_1, A_1, L_1)$, $T_2 = (G_2, A_2, L_2)$ and $T_3 = (G_3, A_3, L_3)$. To predict new links, we use various algorithms to solve the link prediction problem with $i = 2$ and $j = 3$ and first learn any required hyperparameters by performing grid search on the link prediction problem with $i = 1$ and $j = 2$. Similarly, to predict missing links, we solve the link prediction problem with $i = 2$ and $j = 1$ and learn hyperparameters via grid search with $i = 3$ and $j = 2$.

For any given snapshot, several entries of A will be zero, corresponding to missing attributes.

The attribute inference problem, which involves only a single snapshot of the network, is defined as follows:

Definition 2 (Attribute Inference Problem)**.** *Let $T = (G, A, L)$ be a snapshot of a social network. Then the attribute inference problem is to infer whether each zero entry of A corresponds to a positive or negative attribute, subject to the constraints listed in L.*

Our goal is to design scalable algorithms leveraging both network structure and rich node attributes to address these problems for real-world large-scale networks.

4.2.1 Algorithms

4.3 Model and Algorithms

4.3.1 Social-Attribute Network Model

Social-Attribute Network was first proposed by Yin et al. [137, 138][2] to predict links and infer attributes. However, their original model didn't consider negative and mutually exclusive attributes. In this section, we review this model and extend it to incorporate negative and mutex attributes.

Given a social network G with M distinct categorical attributes, an attribute matrix A and mutex attributes set L, we create an augmented network by adding M additional nodes to G, with each additional node corresponding to an attribute. For each node u in G with positive or negative attribute a, we create an undirected link between u and a in the augmented network. For each mutually exclusive attribute pair (a, b), we create an undirected link between a and b. This augmented network is called the *Social-Attribute Network* (SAN) since it includes the original social network interactions, relations between nodes and their attributes and mutex links between attributes.

Nodes in the SAN model corresponding to nodes in G are called *social nodes*, and nodes representing attributes are called *attribute nodes*. Links between social nodes are called *social links*, and links between social nodes and attribute nodes are called *attribute links*. Attribute link (u, a) is a *positive attribute link* if a is a positive attribute of node u, and it is a *negative attribute link* otherwise. Links between mutually exclusive attribute nodes are called *mutex links*. Intuitively, the SAN model explicitly describes the sharing of attributes across social nodes as well as the mutual exclusion between attributes, as illustrated in the sample SAN model of Fig. 4.1. Moreover, with the SAN model, the link prediction problem reduces to predicting social links and the attribute inference problem involves predicting attribute links.

We also place weights on the various nodes and edges in the SAN model. These social node weights could describe the individual tendencies (e.g., user activeness) of issuing social links and the attribute node weights could be used to balance the influence of social nodes versus attribute nodes. Furthermore, the edge weights of social links could describe their tie strengths and the attribute

[2]Note that they name this model as *Augmented Graph*. We call it as *Social-Attribute Network* because it's more meaningful.

Table 4.1: Summary of various (a) link prediction, (b) attribute inference and (c) iterative link and attribute inference algorithms.

(a) Link prediction

Algorithm	References
Unsupervised	
Common Neighbor (CN)	[78]
Common Neighbor (CN-SAN)	us
Adamic-Adar (AA)	[7, 78]
Adamic-Adar (AA-SAN)	us
Low-rank Approximation (LRA)	[78]
Low-rank Approximation (LRA-SAN)	us
CN + Low-rank Approximation (CN+LRA)	[78]
CN + Low-rank Approximation (CN+LRA-SAN)	us
AA + Low-rank Approximation (AA+LRA)	us
AA + Low-rank Approximation (AA+LRA-SAN)	us
Random Walk with Restart (RWwR)	[22, 98]
Random Walk with Restart (RWwR-SAN)	[138]
Supervised	
SLP-I	[58, 79]
SLP-II	[58]
SLP-SAN-III/SLP-SAN-VI	us

(b) Attribute inference

Algorithm	References
Unsupervised	
Common Neighbor (CN-SAN)	us
Adamic-Adar (AA-SAN)	us
Low-rank Approximation (LRA-SAN)	us
CN + Low-rank Approximation (CN+LRA-SAN)	us
AA + Low-rank Approximation (AA+LRA-SAN)	us
Random Walk with Restart (RWwR-SAN)	[138]
Supervised	
Supervised Attribute Inference (SAI-SAN)	us

(c) Iterative link and attribute inference

Algorithm	References
Iterative Link and Attribute Inference	us

link weights could balance the influence of social links versus attribute links. We use $w(u)$ and $w(u, v)$ to denote the weight of node u and the weight of link (u, v), respectively. Additionally, for a given social or attribute node u in the SAN model, we denote by $\Gamma_+(u)$ and $\Gamma_{s+}(u)$ respectively the set of *all neighbors* and the set of *social neighbors* connected to u via social links or positive

attribute links. We define $\Gamma_{-}(u)$ and $\Gamma_{s-}(u)$ in a similar fashion. This terminology will prove useful when we describe our generalization of leading link prediction algorithms to the SAN model in the next section.

The fact that no social node can be linked to multiple mutex attributes is encoded in the *mutex property*, i.e., there is no triangle consisting of a mutex link and two positive attribute links in any social-attribute network, which enforces a set of constraints for all attribute inference algorithms.

In this work, we focus primarily on node attributes. However, we note that the SAN model can be naturally extended to incorporate *edge attributes*. Indeed, we can use a function (e.g., the logistic function) to map a given set of attributes for each edge (e.g., edge age) into the real-valued edge weights of the SAN model. The attributes-to-weight mapping function can be learned using an approach similar to the one proposed by Backstrom and Leskovec [9].

Link prediction algorithms typically compute a probabilistic score for each candidate link and subsequently rank these scores and choose the largest ones (up to some threshold) as putative new or missing links. In the following, we extend both unsupervised and supervised algorithms to the SAN model. Furthermore, we note that when predicting attribute links, the SAN model features a post-processing step whereby we change the lowest ranked putative positive links violating the mutex property to negative links. Table 4.1 summarizes the various link prediction and attribute inference algorithms as well as their references.

Unsupervised Link and Attribute Inference

Liben-Nowell and Kleinberg [78] provide a comprehensive survey of unsupervised link prediction algorithms for social networks. These algorithms can be roughly divided into two categories: local-neighborhood-based algorithms and global-structure-based algorithms. In principle, all of the algorithms discussed in [78] can be generalized for the SAN model. In this work we focus on representative algorithms from both categories and we describe below how to generalize them to the SAN model to predict both social links and attribute links. We add the suffix '-SAN' to each algorithm name to indicate its generalization to the SAN model. In our presentation of the unsupervised algorithms, we only consider positive attribute links, though many of these algorithms can be extended to signed networks [114].

Common Neighbor (CN-SAN): This is a local algorithm that computes a score for a candidate social or attribute link (u, v) as the sum of weights of u and v's common neighbors, i.e. $score(u, v) = \sum_{t \in \Gamma_+(u) \cap \Gamma_+(v)} w(t)$. Conventional CN only considers common social neighbors.

Adamic-Adar (AA-SAN): This is also a local algorithm. For a candidate social link (u, v) the AA-SAN score is

$$score(u, v) = \sum_{t \in \Gamma_+(u) \cap \Gamma_+(v)} \frac{w(t)}{\log |\Gamma_{s+}(t)|} .$$

Conventional AA, initially proposed in [7] to predict friendships on the web and subsequently adapted by [78] to predict links in social networks, only considers common social neighbors. AA-SAN weights the importance of a common neighbor proportional to the inverse of the log of social degree. Intuitively, we want to downweight the importance of neighbors that are either i) social

nodes that are social hubs or ii) attribute nodes corresponding to attributes that are widespread across social nodes. Since in both cases this weight depends on the social degree of a neighbor, the AA-SAN weight is derived based on social degree, rather than total degree.

In contrast, for a candidate attribute link (u, a), the attribute degree of a common neighbor does influence the importance of the neighbor. For instance, consider two social nodes with the same social degree that are both common neighbors of nodes u and a. If the first of these social nodes has only two attribute neighbors while the second has 1000 attribute neighbors, the importance of the former social node should be greater with respect to the candidate attribute link. Thus, AA-SAN computes the score for candidate attribute link (u, a) as

$$score(u, a) = \sum_{t \in \Gamma_{s+}(u) \cap \Gamma_{s+}(a)} \frac{w(t)}{\log |\Gamma_+(t)|}.$$

Low-rank Approximation (LRA-SAN): This algorithm takes advantage of global structure, in contrast to CN-SAN and AA-SAN. Denote X_S as the $N \times N$ weighted social adjacency matrix where the $(u, v)th$ entry of X_S is $w(u, v)$ if (u, v) is a social link and zero otherwise. Similarly, let X_A be the $N \times M$ weighted attribute adjacency matrix where the $(u, a)th$ entry of X_A is $w(u, a)$ if (u, a) is a positive attribute link and zero otherwise. We then obtain the weighted adjacency matrix X for the SAN model by concatenating X_S and X_A, i.e., $X = [X_S \ X_A]$. The LRA-SAN method assumes that a small number of latent factors (approximately) describe the social and attribute link strengths within X and attempts to extract these factors via low-rank approximation of X, denoted by $\hat{X}$. The LRA-SAN score for a candidate social or attribute link (u, t) is then simply $\hat{X}_{ut}$, or the $(u, t)th$ entry of $\hat{X}$. LRA-SAN can be computed efficiently via truncated Singular Value Decomposition (SVD).

CN + Low-rank Approximation (CN+LRA-SAN): This is a mixture of local and global methods, as it first performs CN-SAN using a SAN model and then performs low-rank approximation on the resulting score matrix. After performing CN-SAN, let S_S be the resulting $N \times N$ score matrix for all social node pairs and S_A be the resulting $N \times M$ score matrix for all social-attribute node pairs. By virtue of the CN-SAN algorithm, note that S_S includes attribute information and S_A includes social interactions. CN+LRA-SAN then predicts social links by computing a low-rank approximation of S_S denoted $\hat{S}_S$, and each entry of $\hat{S}_S$ is the predicted social link score. Similarly, $\hat{S}_A$ is a low-rank approximation of S_A, and each entry of $\hat{S}_A$ is the predicted score for the corresponding attribute link.[3]

AA + low-rank Approximation(AA+LRA-SAN): This is identical to CN+LRA-SAN but with the score matrices S_S and S_A generated via the AA-SAN algorithm.

Random Walk with Restart (RWwR-SAN) [138]: This is a global algorithm. In the SAN model, a Random Walk with Restart [22, 98] starting from u recursively walks to one of its neighbors t with probability proportional to the link weight $w(u, t)$ and returns to u with a fixed restart probability α. The probability $P_{u,v}$ is the stationary probability of node v in a random walk with restart initiated at u. In general, $P_{u,v} \neq P_{v,u}$. For a candidate social link (u, v), we compute

[3] An alternative method for combining CN-SAN and LRA-SAN under the SAN model that was not explored in this work involves defining $S = [S_S \ S_A]$, approximating S with $\hat{S}$ and using the $(u, t)th$ entry of $\hat{S}$ as a score for link (u, t).

$P_{u,v}$ and $P_{v,u}$ and let $score(u, v) = (P_{u,v} + P_{v,u})/2$. Note that RWwR for link prediction in previous work [78] computes these stationary probabilities based only on the social network. For a candidate attribute link (u, a), RWwR-SAN only computes $P_{u,a}$, and $P_{u,a}$ is taken as the score of (u, a).

We finally note that for predicting social links, if we set the weights of all attribute nodes and all attribute links to zero and we set the weights of all social nodes and social links to one, then all the algorithms described above reduce to their standard forms described in [78].[4] In other words, we recover the link prediction algorithms on pure social networks.

Supervised Link and Attribute Inference

Link prediction can be cast as a binary classification problem, in which we first construct features for links, and then use a classifier such as SVMs or Logistic Regression. In contrast to unsupervised attribute inference, negative attribute links are needed in supervised attribute inference.

Supervised Link Prediction (SLP-SAN): For each link in our training set, we can extract a set of topological features F (e.g. CN, AA, etc.) computed from pure social networks and the similar features F_SAN computed from the corresponding social-attribute networks. We explored 4 feature combinations: i) SLP-I uses only topological features F computed from social networks; ii) SLP-II uses topological features F as well as an aggregate feature, i.e., the number of common attributes of the two endpoints of a link; iii) SLP-SAN-III uses topological features F_SAN; and iv) SLP-SAN-VI uses topological features F and F_SAN. SLP-SAN-III and SLP-SAN-VI contain the substring 'SAN' because they use features extracted from the SAN model. SLP-I and SLP-II are widely used in previous work [58, 79].

Supervised Attribute Inference (SAI-SAN): Recall that attribute inference is transformed to attribute link prediction with the SAN model. We can extract a set of topological features for each positive and negative attribute link. Moreover, the positive attribute links are taken as positive examples while the negative attribute links are taken as negative examples. Hence, we can train a binary classifier for attribute links and then apply it to infer the missing attribute links.

Iterative Link and Attribute Inference

In many real-world networks, most node attributes are missing. Fig. 4.2 shows the fraction of users as a function of the number of node attributes in Google+ social network. From this figure, we see that roughly 70% of users have no observed node attributes. Hence, we will also investigate an iterative variant of the SAN model. We first infer the top attributes for users without any observed attributes. We then update the SAN model to include these predicted attributes and perform link prediction on the updated SAN model. This process can be performed for several iterations.

[4] For LRA-SAN this implies that X_A is an $N \times M$ matrix of zeros, so the truncated SVD of X is equivalent to that of X_S except for M zeros appended to the right singular vectors of X_S.

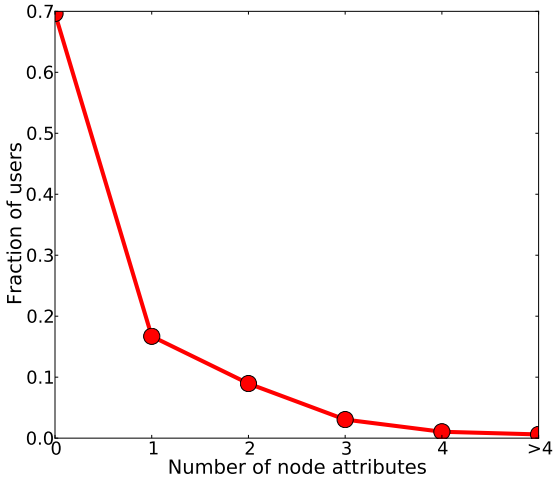


Figure 4.2: The fraction of users as a function of the number of node attributes in the Google+ social network.

Scalability

The local unsupervised algorithms CN-SAN and AA-SAN only concern about the hop-2 neighborhoods, and thus are scalable. The low-rank approximation based unsupervised algorithms (i.e., LRA-SAN, CN+LRA-SAN and AA+LRA-SAN) can be computed via truncated Singular Value Decomposition (SVD), which was shown to be scalable [115]. RWwR-SAN is based on a random walk with restart, which was also shown to be scalable [119]. Moreover, there are many papers/libraries making classifiers (e.g., SVM in our experiments.) scalable. For instance, Joachims [63] and LIBLINEAR[5].

4.4 Google+ Data

Google launched its new social network service named Google+ in early July 2011. We crawled three snapshots of the Google+ social network and their users' profiles on July 19, August 6 and September 19 in 2011. They are denoted as JUL, AUG and SEP, respectively. We then pre-processed the data before conducting link prediction and attribute inference experiments.

Preprocessing Social Networks: In Google+, users divide their social connections into circles, such as a family circle and a friends circle. If user u is in v's circle, then there is a directed edge (v, u) in the graph, and thus the Google+ dataset is a directed social graph. We converted this dataset into an undirected graph by only retaining edges (u, v) if both directed edges (u, v) and (v, u) exist in the original graph. We chose to adopt this filtering step for two reasons: (1) Bidirectional edges represent mutual friendships and hence represent a stronger type of relationship

[5] `http://www.csie.ntu.edu.tw/~cjlin/liblinear/`

Table 4.2: Statistics of social-attribute networks.

	#soci links	#all soci links	#soci nodes	#pos attri links	#attri nodes
JUL4	7062	7062	5200	24690	9539
AUG4	7430	7813			
SEP4	7422	8100			
JUL2	287906	287906	170002	442208	47944
AUG2	328761	339059			
SEP2	332398	354572			

that is more likely to be useful when inferring users' attributes from their friends' attributes (2) We reduce the influence of spammers who add people into their circles without those people adding them back. Spammers introduce fictitious directional edges into the social graph that adversely influence the performance of link prediction algorithms.

Collecting Attribute Vocabulary: Google+ profiles include short entries about users such as Occupation, Employment, Education, Places Lived, and Gender, etc. We use Employment and Education to construct a vocabulary of attributes because our previous work [51, 52] showed that these two attribute types are most powerful for forming links between people. We treat each distinct employer or school entity as a distinct attribute. Google+ has predefined employer and school entities, although users can still fill in their own defined entities. Due to users' changing privacy settings, some profiles in JUL are not found in AUG and SEP, so we use JUL to construct our attribute vocabulary. Specifically, from the profiles in JUL, we list all attributes and compute frequency of appearance for each attribute. Our attribute vocabulary is constructed by keeping attributes with frequency of at least 3.

Constructing Social-Attribute Networks: In order to demonstrate that the SAN model leverages node attributes well, we derived social-attribute networks in which each node has some positive attributes from the above Google+ social networks and attribute vocabulary. Specifically, for an attribute-frequency threshold k, we chose the largest connected social network from JUL such that each node has at least k distinct positive attributes. We also found the corresponding social networks consisting of these nodes in snapshots AUG and SEP. Social-attribute networks were then constructed with the chosen social networks and the attributes of the nodes. Specifically, we chose $k = \{2, 4\}$ to construct 6 social-attribute networks whose statistics are shown in Table 4.2. Each social-attribute network is named by concatenating the snapshot name and the attribute-frequency threshold. For example, 'JUL4' is the social-attribute network constructed using JUL and $k = 4$. These names are indicated in the first column of the table.

In the crawled raw networks, some social links in JULi are missing in AUGi and SEPi, where $i = 2, 4$. These links are missing due to one of two events occurring between the JUL and AUG or SEP snapshots: 1) users block other users, or 2) users set (part of) their circles to be publicly invisible after which point they cannot be publicly crawled. These missed links provide ground truth labels for our experiments of predicting missing links. However, these missing links can alter estimates of network-level statistics, and can have unexpected influences on link prediction algorithms [70]. Moreover, it is likely in practice that companies like Facebook and Google keep records of these missing links, and so it is reasonable to add these links back to AUGi and SEPi

for our link prediction experiments. The third column in Table 4.2 is the number of all social links after filling the missing links into AUGi and SEPi. The second column *#soci links* is used for experiments of predicting missing links, and column *#all soci links* is used for the experiments of predicting new links.

From these two columns, the number of new links or missing links can be easily computed. For example, if we use AUG2 as training data and SEP2 as testing data for link prediction, the number of new links is $354572 - 339059 = 15513$, which is computed with entries in column *#all soci links*. If we use AUG2 as training data and JUL2 as testing data in predicting missing links, the number of missing links is $339059 - 328761 = 10298$, which is computed with corresponding entries in column *#soci links* and *#all soci links*.

4.5 Experiments

4.5.1 Experimental Setup

In our experiments, the main metric used is AUC, Area Under the Receiver Operating Characteristic (ROC) Curve, which is widely used in the machine learning and social network communities [9, 29]. AUC is computed in the manner described in [57], in which both positive and negative examples are required. In principle, we could use new links or missing links as positive examples and all non-existing links as negative examples. However, large-scale social networks tend to be very sparse, e.g., the average degree is 4.17 in SEP2, and, as a result, the number of non-existing links can be enormous, e.g., SEP2 has around 2.9×10^{10} non-existing links. Hence, computing AUC using all non-existing links in large-scale networks is typically computationally infeasible. Moreover, the majority of new links in typical online social networks close triangles [9, 74], i.e., are hop-2 links. For instance, we find that 58% of the newly added links in Google+ are hop-2 links. We thus evaluate our large network experiments using hop-2 link data as in [9], i.e., new or missing hop-2 links are treated as positive examples and non-existing hop-2 links are treated as negative examples.

In a social-attribute network, there are two categories of hop-2 links: 1) those with two endpoints sharing at least one common social node, and 2) those with two endpoints sharing only common attribute nodes. Local algorithms applied to the original social network are unable to predict hop-2 links in the second category. Thus, we evaluate only with respect to hop-2 links in the first category, so as not to give unfair advantage to algorithms running on the social-attribute network. To better understand whether the AUC performance computed on hop-2 links can be generalized to performance on any-hop links, we additionally compute AUC using any-hop links on the smaller Google+ networks.

In general, different nodes and links can have different weights in social-attribute networks, representing their relative importance in the network. In all of our experiments, we set all weights to be one and leave it for future work to learn weights.

We compare our link prediction algorithms with Supervised Random Walk (SRW) [9], which leverages edge attributes, by transforming node attributes to edge attributes. Specifically, we compute the number of common attributes of the two endpoints of each existing link. As in [9], we also use the number of common neighbors as an edge attribute. We adopt the Wilcoxon-

Mann-Whitney (WMW) loss function and logistic edge strength function in our implementations as recommended in [9].

We compare our attribute inference algorithms with two algorithms, BASELINE and LINK, introduced by Zheleva and Getoor [142]. Using only node attributes, BASELINE first computes a marginal attribute distribution and then uses an attribute's probability as its score. LINK trains a classifier for each attribute by flattening nodes as the rows of the adjacency matrix of the social networks.[6] Zheleva and Getoor [142] found that LINK is the best algorithm when group memberships are not available.

We use SVM as our classifier in all supervised algorithms. For link prediction, we extract six topological features (CN-SAN, AA-SAN, LRA-SAN, CN+LRA-SAN, AA+LRA-SAN and RWwR-SAN) from both pure social networks and social-attribute networks. Hence, SLP-I, SLP-II, SLP-SAN-III and SLP-SAN-VI use 6, 7, 6 and 12 features, respectively. For attribute inference, we extract 9 topological features for each attribute link. We adopt two ranks (detailed in 4.5.2) for each low-rank approximation based algorithms, thus obtaining 6 features. The other three features are CN-SAN, AA-SAN and RWwR-SAN. To account for the highly imbalanced class distribution of examples for supervised link prediction and attribute inference we downsample negative examples so that we have equal number of positive and negative examples (techniques proposed in [35, 79] could be used to further improve the performance).

We use the pattern *dataset1-dataset2* to denote a train-test or train-validation pair, with *dataset1* a training dataset and *dataset2* a testing or validation dataset. When conducting experiments to predict new links on the AUGi-SEPi train-test pair, SRW, classifiers and hyperparameters of global algorithms, i.e., ranks in LRA-SAN, CN+LRA-SAN, and AA+LRA-SAN and the restart probability α in RWwR-SAN, are learned on the JULi-AUGi train-validation pair. Similarly, when predicting missing links on train-test pair AUGi-JULi, they are learned on train-validation pair SEPi-AUGi, where $i = 2, 4$.

The CN-SAN and AA-SAN algorithms are implemented in Python 2.7 while the RWwR-SAN algorithm and Supervised Random Walk (SRW) are implemented in Matlab, and all of them are run on a desktop with a 3.06 GHz Intel Core i3 and 4GB of main memory. LRA-SAN, CN+LRA-SAN and AA+LRA-SAN algorithms are implemented in Matlab and run on an x86-64 architecture using a single 2.60 Ghz core and 30GB of main memory.

4.5.2 Experimental Results

In this section we present evaluations of the algorithms on the Google+ dataset. We first show that incorporating attributes via the SAN model improves the performance of both unsupervised and supervised link prediction algorithms. Then we demonstrate that inferring attributes via link prediction algorithms within the SAN model achieves state-of-the-art performance. Finally, we show that by combining attribute inference and link prediction in an iterative fashion, we achieve even greater accuracy on the link prediction task.

[6]The original LINK algorithm [142] trained a distinct classifier for each attribute type. In our setting, an attribute type (e.g., Education) can have multiple values, so we train a classifier for each binary attribute value.

Table 4.3: Results for predicting new links. (a)AUC of hop-2 new links on the train-test pair AUG4-SEP4. (b)AUC of hop-2 new links on the train-test pair AUG2-SEP2. (c) (d) AUC of any hop new links on the train-test pair AUG4-SEP4. The numbers in parentheses are standard deviations.

(a)

Alg	w/o Attri	With Attri
Random	0.5000	0.5000
CN-SAN	0.6730	0.7315
AA-SAN	0.7109	0.7476
LRA-SAN	0.6003	0.6262
CN+LRA-SAN	0.6969	**0.7671**
AA+LRA-SAN	0.7118	0.7471
RWwR-SAN	0.6033	0.6143

(b)

Alg	w/o Attri	With Attri
Random	0.5000	0.5000
CN-SAN	0.6936	0.7508
AA-SAN	0.7638	**0.7895**
LRA-SAN	0.6410	0.6385
CN+LRA-SAN	0.5642	0.6373
AA+LRA-SAN	0.6032	0.6557
RWwR-SAN	0.6788	0.6912

(c)

Alg	w/o Attri	With Attri
Random	0.5000	0.5000
CN-SAN	0.7482	0.8298
AA-SAN	0.7483	0.8324
LRA-SAN	0.8075	0.8237
CN+LRA-SAN	0.7857	0.8651
AA+LRA-SAN	0.8193	0.8552
RWwR-SAN	0.9363	**0.9548**

(d)

Alg	AUC
SLP-I	0.9128(0.0140)
SLP-II	0.9580(0.0017)
SLP-SAN-III	0.9450(0.0007)
SLP-SAN-VI	**0.9706(0.0004)**
SRW	0.9383

Link Prediction

To demonstrate the benefits of combining node attributes and network structure, we run the SAN-based link prediction algorithms described in Section 4.2.1 both on the original social networks and on the corresponding social-attribute networks (recall that the SAN-based unsupervised algorithms reduce to standard unsupervised link prediction algorithms when working solely with the original social networks).

Predicting New Links: Table 4.3 shows the AUC results of predicting new links for each of our datasets. We are able to draw a number of conclusions from these results. First, the SAN model improves every unsupervised learning algorithm on every dataset, save for LRA-SAN on AUG2-SEP2. The reason that LRA-SAN's performance decreases on AUG2-SEP2 could be that we searched the ranks in LRA-SAN up to 3000 in the training and validation phase due to the limited computing resources available to us[7]. Second, Table 4.3d shows that attributes also improve supervised link prediction performance since SLP-SAN-VI, SLP-SAN-III and SLP-II outperform

[7]Note that LRA-SAN is scalable with more computing resources, which was shown by Talwalkar et al. [115]

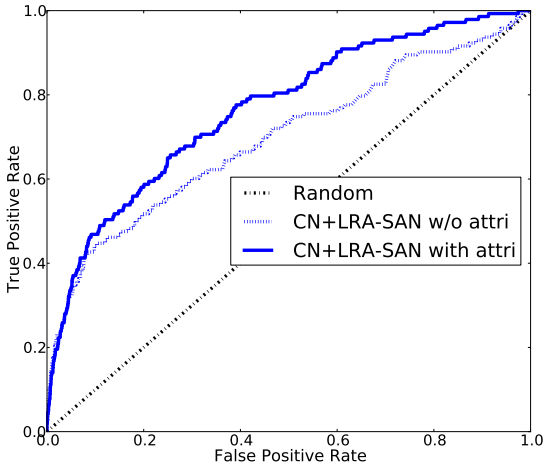


Figure 4.3: ROC curves of the CN+LRA-SAN algorithm for predicting new links. AUG4-SEP4 is the train-test pair. JUL4-AUG4 is the train-validation pair.

SLP-I. Moreover, SLP-SAN-VI, which adopts features extracted from both social networks and social-attribute networks, achieves the best performance, thus demonstrating the power of the SAN model. Third, comparing RWwR-SAN in Table 4.3c and SRW in Table 4.3d, we observe that the SAN model is better than SRW at leveraging node attributes since RWwR-SAN with attributes outperforms SRW. This result is not surprising given that SRW is designed for edge attributes and when transforming node attributes to edge attributes, we lose some information. For instance, as illustrated in Fig. 4.1, nodes u_2 and u_5 share the attribute San Francisco. When transforming node attributes to edge attributes, this common attribute information is lost since u_2 and u_5 are not linked.

Fig. 4.3 shows the ROC curves of the CN+LRA-SAN algorithm. We see that curve of CN+LRA-SAN with attributes dominates that of CN+LRA-SAN without attributes, demonstrating the power of the SAN model to effectively incorporate the additional predictive information of attributes. The results of other algorithms except LRA-SAN are similar and thus are not shown here.

Predicting Missing Links: Missing links can be divided into two categories: 1) links whose two endpoints have some social links in the training dataset. 2) links with at least one endpoint that has no social links in the training dataset. Category 1 corresponds to the scenarios where users block users or users set a part of their friend lists (e.g. family circles) to be private. Category 2 corresponds to the scenario in which users hide their entire friend lists. Note that all hop-2 missing links belong to Category 1. In addition to performing experiments to show that the SAN model improves missing link prediction, we also perform experiments to explore which category of missing links is easier to predict. Table 4.4 shows the results of predicting missing links on various datasets. As in the new-link prediction setting, the performance of every algorithm is improved by the SAN model, except for LRA-SAN on AUG4-JUL4 and RWwR-SAN on AUG4-JUL4 for hop-2 missing links.

Table 4.4: Results for predicting missing links. (a) AUC of hop-2 missing links on the train-test pair AUG4-JUL4. (b) AUC of hop-2 missing links on the train-test pair AUG2-JUL2. (c)-(f) AUC of any-hop missing links on the train-test pair AUG4-JUL4. Missing links in both categories 1 and 2 are used in (c) and (e). Missing links in Category 1 are used in (d) and (f). The numbers in parentheses are standard deviations.

(a)

Alg	w/o Attri	With Attri
Random	0.5000	0.5000
CN-SAN	0.7180	0.7925
AA-SAN	0.7437	0.7697
LRA-SAN	0.6569	0.6237
CN+LRA-SAN	0.7147	**0.7986**
AA+LRA-SAN	0.7410	0.7668
RWwR-SAN	0.5731	0.5676

(b)

Alg	w/o Attri	With Attri
Random	0.5000	0.5000
CN-SAN	0.6938	0.7309
AA-SAN	0.7633	**0.7796**
LRA-SAN	0.6044	0.6059
CN+LRA-SAN	0.5816	0.6266
AA+LRA-SAN	0.6212	0.6569
RWwR-SAN	0.6595	0.6706

(c)

Alg	w/o Attri	With Attri
Random	0.5000	0.5000
CN-SAN	0.5460	0.7012
AA-SAN	0.5460	0.7033
LRA-SAN	0.5495	0.6177
CN+LRA-SAN	0.5547	0.7048
AA+LRA-SAN	0.5640	0.7325
RWwR-SAN	0.2000	**0.7619**

(d)

Alg	w/o Attri	With Attri
Random	0.5000	0.5000
CN-SAN	0.7329	0.7765
AA-SAN	0.7330	0.7784
LRA-SAN	0.7316	0.7401
CN+LRA-SAN	0.7515	0.7510
AA+LRA-SAN	0.8104	0.8116
RWwR-SAN	0.7797	**0.8838**

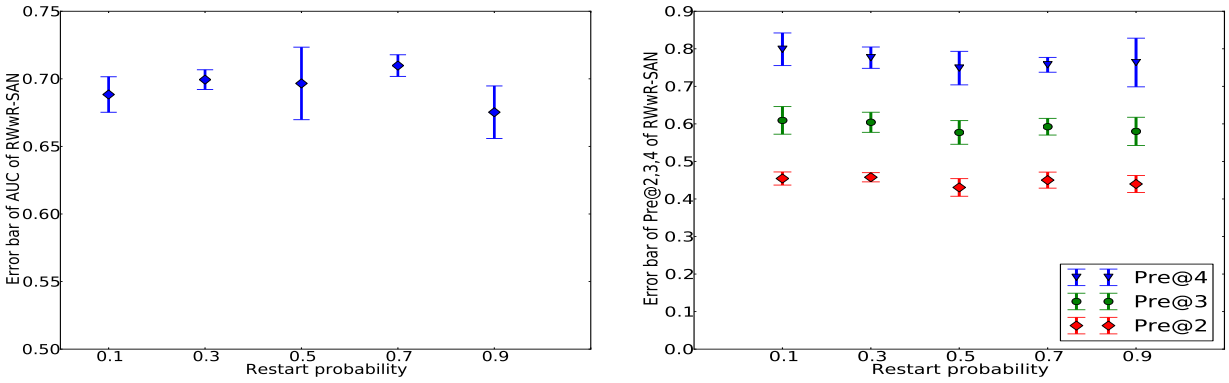
(e)

Alg	AUC
SLP-I	0.5453(0.0120)
SLP-II	0.6991(0.0065)
SLP-SAN-III	0.7161(0.0030)
SLP-SAN-VI	**0.8481(0.0022)**

(f)

Alg	AUC
SLP-I	0.8023(0.0088)
SLP-II	0.8403(0.0033)
SLP-SAN-III	0.8620(0.0080)
SLP-SAN-VI	**0.8854(0.0324)**

When comparing Tables 4.4c and 4.4d or Tables 4.4e and 4.4f, we conclude that the missing links in Category 2 are harder to predict than those in Category 1. RWwR-SAN without attributes performs poorly when predicting any-hop missing links in both categories (as indicated by the entry with 0.2000 in Table 4.4c). This poor performance is due to the fact that RWwR-SAN without attributes assigns zero scores for all the missing links in Category 2 (positive examples) and positive scores for most non-existing links (negative examples), making many negative examples rank higher than positive examples and resulting in a very low AUC.



(a) AUC under ROC curves

(b) Pre@2,3,4

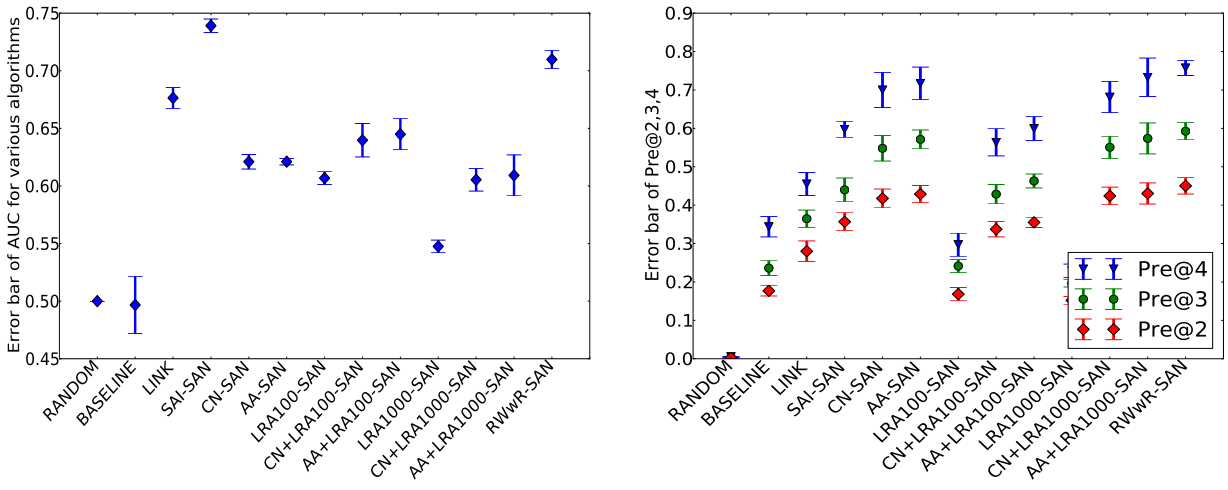
Figure 4.4: Impact of the restart probability on the performance of RWwR-SAN for attribute inference on SEP4. (a) AUC under ROC curves. (b) Pre@2,3,4.

Attribute Inference

In this section, we focus on inferring attributes using the SAN model. In our next set of experiments in Section 4.5.2, we use the results of these attribute inference algorithms to further improve link prediction, and the results of this iterative approach further validate the performance of the SAN model for attribute inference. Since the first step of iterative approach of Section 4.5.2 involves inferring the top attributes for each node, we employ an additional performance metric called Pre@K in our attribute inference experiments. Compared to AUC, Pre@K better captures the quality of the top attribute predictions for each user. Specifically, for each sampled user, the top-K predicted attributes are selected, and (unnormalized) Pre@K is then defined as the number of positive attributes selected divided by the number of sampled users. We address score ties in the manner described in [85]. Since most Google+ users have a small number of attributes, we set $K = 2, 3, 4$ in our experiments.

When evaluating algorithms for the inference of missing attributes, we require ground truth data. In general, ground truth for node attributes is difficult to obtain since it is often not possible to distinguish between negative and missing attributes. However, for most users the number of attributes is quite small, and so we assume that users with many positive attributes have no missing attributes. Hence, we evaluate attribute inference on users that have at least 4 specified attributes, i.e., we work with users in SEP4 and assume that each attribute link in SEP4 is either positive or negative.

In our experiment, we sample 10% of the users in SEP4 uniformly at random, remove their attribute links from SEP4, and evaluate the accuracy with which we can infer these users' attributes. All removed positive attribute links are viewed as positive examples, while all the negative attribute links of the sampled users are treated as negative examples. We run a variety of algorithms for attribute inference, and for each algorithm we average the results over 10 random trials. As noted above, we evaluate the performance of attribute inference using both AUC and Pre@K. Note that some attribute nodes in SEP4 only have one positive attribute link. As a result, if these positive attribute links are sampled as test data, it's hard to correctly infer them in the test phase because



(a) AUC under ROC curves

(b) Pre@2,3,4

Figure 4.5: Performance of various algorithms on attribute inference on SEP4. (a) AUC under ROC curves. (b) Pre@2,3,4.

there are no positive training examples for the corresponding attribute nodes. Therefore, we further remove the attribute nodes with only one positive attribute link and their corresponding attribute links from SEP4. Moreover, in the test phase, we only use the users among the sampled ones that have at least K attributes to compute the Pre@K to avoid the influence of users with too few positive attributes.

For the low-rank approximation based algorithms, i.e., LRA-SAN, CN+LRA-SAN and AA+LRA-SAN, we report results using two different ranks, 100 and 1000, and indicate which was used by the number following the algorithm name in Fig. 4.5. We choose these two small ranks for computational reasons and also based on the fact that low-rank approximation methods assume that a small number of latent factors (approximately) describe the social-attribute networks. Fig. 4.4 shows the impact of the restart probability on the performance of RWwR-SAN. We find that $\alpha = 0.7$ achieves the best AUC and comparable Pre@K. Thus we set the restart probability α to be 0.7.

Fig. 4.5 shows the attribute inference results for various algorithms. Several interesting observations can be made from this figure. First, under both metrics, all SAN-based algorithms perform better than BASELINE, save LRA100-SAN and LRA1000-SAN under Pre@2,3,4 metric, which indicates that the SAN model is good at leveraging network structure to infer missing attributes. Second, we find that AUC and Pre@K provide inconsistent conclusions about relative algorithm performance. For instance, the mean AUC values suggest that SAI-SAN beats all other algorithms. However, several unsupervised algorithms outperform SAI-SAN with respect to Pre@2,3,4. The inconsistencies between the two metrics are expected since AUC is a global measurement while Pre@K is a local one. Our SAI-SAN algorithm dominates LINK under both AUC and Pre@2,3,4 metrics, thus demonstrating the power of mapping attribute inference to link prediction with the SAN model.

Table 4.5: Results for iteratively inferring attributes and predicting links. (a) on the AUG4-SEP4 train-test pair. (b) on the AUG4-JUL4 train-test pair. Results are averaged over 10 trials. The numbers in parentheses are standard deviations.

(a)

Alg	w/o Attri	With Attri	With Inferred Attri
Random	0.5000(0)	0.5000(0)	0.5000(0)
CN-SAN	0.6730(0)	0.7174(0.0077)	0.7291(0.0063)
AA-SAN	0.7109(0)	0.7408(0.0063)	0.7440(0.0026)
LRA-SAN	0.6003(0)	0.6274(0.0052)	0.6320(0.0055)
CN+LRA-SAN	0.6969(0)	0.7497(0.0134)	**0.7534**(0.0084)
AA+LRA-SAN	0.7111(0)	0.7373(0.0050)	0.7442(0.0032)

(b)

Alg	w/o Attri	With Attri	With Inferred Attri
Random	0.5000(0)	0.5000(0)	0.5000(0)
CN-SAN	0.7180(0)	0.7780(0.0173)	0.7856(0.0100)
AA-SAN	0.7437(0)	0.7626(0.0100)	0.7661(0.0045)
LRA-SAN	0.6569(0)	0.6189(0.0105)	0.6134(0.0157)
CN+LRA-SAN	0.7147(0)	0.7838(0.0256)	**0.7969**(0.0059)
AA+LRA-SAN	0.7410(0)	0.7591(0.0118)	0.7673(0.0051)

Iterative Attribute and Link Inference

Section 4.5.2 demonstrated that knowledge of a user's attributes can lead to significant improvements in link prediction. However, in real-world social networks like Google+, the vast majority of user attributes are missing (see Fig. 4.2). To increase the realized benefits of social-attribute networks with few attributes, we propose first inferring missing attributes for each user whose attributes are missing and then performing link prediction on the inferred social-attribute networks. Recall that SAI-SAN achieves the best AUC, RWwR-SAN achieves the best Pre@K in inferring attributes (see Fig. 4.5) and AA-SAN achieves comparable Pre@K results while being more scalable. Thus, in the following experiments, we use AA-SAN to first infer the top-K missing attributes for users, and subsequently perform link prediction using various methods.

In our experiments, when we are working on the pair *train-test*, we sample 10% of the users of *train* uniformly at random and remove their attributes. We then run three variants of link prediction algorithms: i) without attributes, ii) with only the remaining attributes, and iii) with the remaining attributes along with the inferred attributes. The top-4 attributes are inferred for each sampled user by AA-SAN. We report the results averaged over 10 trials. The hyperparameters of the global algorithms are the same as those in (Section 4.5.2), which are learned from the corresponding train-validation pair.

Table 4.5a shows the results of first inferring attributes and then predicting new links on the AUG4-SEP4 train-test pair. Table 4.5b shows the results of first inferring attributes and then

predicting missing links on the AUG4-JUL4 train-test pair. We see that the inferred attributes improve the performance of all algorithms except LRA-SAN on predicting missing links. Again, the bad performance of LRA-SAN could be explained by the fact that we searched the ranks in LRA-SAN up to 3000 in the training and validation phase due to the limited computing resources available to us. The AUCs obtained with inferred attributes for all other algorithms are very close to those obtained with all positive attributes as shown in Table 4.3a, which further demonstrates that AA-SAN is an effective algorithm for attribute inference.

4.6 Summary of Results

We comprehensively evaluate the *Social-Attribute Network* (SAN) model in terms of link prediction and attribute inference. More specifically, we adapt several representative unsupervised and supervised link prediction algorithms to the SAN model to both predict links and infer attributes. Our evaluation with a large-scale novel Google+ network dataset demonstrates performance improvement for each of these generalized algorithm on both link prediction and attribute inference. Moreover, we demonstrate a further improvement of link prediction accuracy by using the SAN model in an iterative fashion, first to infer missing attributes and subsequently to predict links.

Chapter 5

Related Work

5.1 Trusted Friends based Secure Account Recovery

5.1.1 Social Authentications

Depending on how friends are involved in the authentication process, social authentications can be classified into *trustee-based* and *knowledge-based* social authentications. In trustee-based social authentications [21], the selected friends aid the user in the authentication process. Knowledge-based social authentication, however, asks the user questions about his or her selected friends, and thus friends are not directly involved.

Trustee-based social authentication systems: Authentication is traditionally based on three factors: *something you know* (e.g., a password), *something you have* (e.g., a RSA SecurID), and *something you are* (e.g., fingerprint).

Brainard et al. [21] proposed to use the fourth factor, i.e., *somebody you know*, to authenticate users. We call the fourth factor trustee-based social authentication. Originally, Brainard et al. combined trustee-based social authentication with some other factor as a two-factor authentication mechanism. It was later adapted to be a backup authenticator [40, 41, 107]. For instance, Schechter et al. [2] designed and built a prototype of trustee-based social authentication system which was integrated into Microsofts Windows Live ID system. Moreover, Facebook designed Trusted Friends in October, 2011 [41], and it was improved to be Trusted Contacts [40] in May, 2013.

Knowledge-based social authentication systems: Such social authentications are still based on *something you know*. Yardi et al. [136] proposed a knowledge-based authentication system based on photos to test if a user belongs to the group (e.g., interest groups in Facebook) that he or she tries to access. Facebook recently launched a similar photo-based social authentication system [102], in which Facebook shows a few photos of a friend of a user and asks the user to name the friend. Such system essentially relies on the knowledge that the user knows the person in the shown photos. However, recent work has shown, via theoretical modeling [67] and empirical evaluations [101], that photo-based social authentications are not resilient to various attacks such as automatic face

recognition techniques, questioning their use as a backup authentication mechanism. As a defense, Polakis et al. [100] recently proposed to transform faces and show distorted faces in the photos. They showed that these distorted friend faces, while easy for a user to recognize, are robust against face recognition attacks and image comparison attacks where attackers collect publicly available photos to compare and identify the individuals in displayed photos. In conclusion, photo-based social authentication constantly finds itself in arms race with face recognition algorithms, which are fast improving. Jain et al. [62] explored a much larger space of social knowledge for social authentication. They found that questions based on user interactions (e.g., private messages, pokes) have higher rates of applicability and recall than questions based on photos and other user attributes such as schools.

5.1.2 Diffusion Models

Our forest fire attacks essentially describe diffusion processes in a trustee network. We review a few representative diffusion models from different research areas and discuss the differences between them and our work.

Updates propagation models: Malkhi et al. [82] proposed the l-Tree propagation model to diffuse updates among a large distributed system of data replicas, some of which might exhibit Byzantine failures. Their model assumes a point-to-point communication for each pair of nodes. A node that already receives the update is called *active*, otherwise it is called *inactive*. Initially, a small set of nodes are active. Each active node is associated with a candidate set of nodes. In each iteration, each active node is allowed to send the update to at most F nodes which are selected from the corresponding candidate set uniformly at random. An inactive node becomes active if it receives the update from at least k other nodes.

There are two key differences between our forest fire attacks and the l-Tree propagation model. First, an uncompromised (i.e., inactive) node can receive verification codes (i.e., updates) from uncompromised trustees via spoofing attacks in forest fire attacks while an inactive node can only receive updates from active nodes in the l-Tree model. Second, in each iteration, each compromised node sends verification codes to *all* nodes that select it as a trustee in forest fire attacks while an active node can only send the update to at most F nodes in the l-Tree model.

Information propagation models: Models for how products and innovations propagate on a social network have been studied in various domains such as viral marketing and the spread of technological innovations. These models can be divided into two categories [66]: *linear threshold model* and *independent cascade model*. Again, a node is said to be *active* if it already adopts the corresponding product or innovation, otherwise it is *inactive*. In both models, a small set of nodes are active initially.

Linear threshold model: In this model, each node u is associated with a threshold, which indicates the fraction of u's friends that must become active before u becomes active. The propagation proceeds deterministically in discrete iterations: in the tth iteration, an inactive node becomes active if its fraction of active friends is no less than u' activation threshold.

Independent cascade model: Different from the linear threshold model, the independent cascade model proceeds in discrete iterations according to the following randomized rule: when a node u first becomes active in the tth iteration, it has a *single* chance to activate each of its currently inactive friend v and succeed with some probability which encodes the influence of u to v.

The key difference is that verification codes can be propagated from an uncompromised (i.e., inactive) node to another uncompromised node via spoofing attacks in forest fire attacks while an inactive node can only be activated by active nodes in the linear threshold model and the independent cascade model. Moreover, an active node only has a single chance to activate its friends in the independent cascade model while a compromised node can send verification codes to the uncompromised nodes that select it as a trustee as many times as it wants.

Epidemic propagation models: Epidemic propagation models describe the propagations of various contagious diseases such as sexually transmitted diseases, inuenza, and measles.

One popular epidemic model is the so-called SIR model (Chapter 21 of [37]). Different from the above reviewed models in which each node can be either active or inactive, SIR model assumes that each node can be in one of the three states, i.e., *susceptible*, *infectious*, and *removed*. Initially, a set of nodes are infectious and all other nodes are susceptible. Each infectious node u remains infectious for a fixed number of iterations I. In each of the I iterations, u has some probability of passing the disease to each of its susceptible neighbors. After I iterations, u becomes removed, which means that u cannot catch nor propagate the disease any more.

Again, a susceptible (i.e., uncompromised) node can only be influenced by infectious (i.e., compromised) nodes in the SIR model, which is different from the forest fire attacks. Moreover, the state *removed* is not meaningful in the context of forest fire attacks since a node could be compromised again even if it recovers the account and resets the password.

Summary: The key difference between forest fire attacks and previous diffusion models lies in the spoofing attacks. Specifically, an uncompromised node can obtain verification codes from its uncompromised trustees via spoofing attacks in the forest fire attacks while an inactive or susceptible node can only be influenced by its active or infectious neighbors in previous diffusion models.

5.2 Social Structure based Sybil Account Detection

5.2.1 Structure-based Sybil Defenses

Most existing structure-based Sybil defenses are based on either random walks or community detections. We refer readers to a recent survey [8] for more details.

Random walk based Sybil classification: SybilGuard [140] and SybilLimit [139] were the first schemes to propose Sybil detection using social network structure. SybilLimit relies on the insight that social networks are relatively well connected, and thus short random walks starting from benign users can quickly reach all other benign users [1]. Thus if two benign users perform $\sqrt{m}$ random walks (where m is the number of edges between benign users), then they will have an intersection with high probability, using the birthday paradox. The intersection of random walks is used as a feature by the benign users to validate each other. On the other hand, short random walks from Sybil users do not reach all benign users (due to limited number of attack edges), and thus do not intersect with the random walks from benign users.

[1] More precisely, SybilGuard and SybilLimit use a variant of random walks called random routes. Please see [139, 140] for more detail.

SybilInfer [34] aims to directly detect a bottleneck cut between benign and Sybil users. SybilInfer relies on random walks and uses a combination of Bayesian inference and Monte-Carlo sampling techniques to estimate the set of benign and Sybil users. Similar to SybilGuard, SybilLimit and SybilInfer, Gatekeeper [121] and SybilDefender [130] also leverage random walks. These mechanisms make additional assumptions about the structure of benign and Sybil nodes, and even the size of the Sybil population [130]. Moreover, they also require the benign regions to be fast mixing, which was shown to be unsatisfied by Mohaisen et al. [93].

In contrast to the above approaches, SybilBelief does not use random walks, and relies instead on the Markov Random Fields and Loopy Belief Propagation. SybilBelief is able to incorporate information about known benign and known Sybil nodes. Our experimental results show that SybilBelief performs an order of magnitude better than SybilLimit and SybilInfer. Moreover, SybilBelief is scalable to large scale social networks, unlike above mechanisms.

Random walk based Sybil ranking: SybilRank [26] performs random walks starting from a set of benign users. Specifically, with h labeled benign nodes, SybilRank designs a special initial probability distribution over the nodes, i.e., probability $1/h$ for each of the labeled benign nodes and probability 0 for all other nodes, and iterates the random walk from this initial distribution for $\log(n)$ iterations, where n is the total number of nodes in the network. It is well known that this random walk is biased to high-degree nodes. Thus, SybilRank normalizes the final probabilities of nodes by their degrees and uses the normalized probabilities to rank nodes. SybilRank scales to very large social networks and has shown good performance on the Tuenti network. However, SybilRank has two major limitations: (a) it does not tolerate label noise, and (b) it does not incorporate Sybil labels. Boshmaf et al. [19] improved upon SybilRank via detecting victims who are benign nodes but link to Sybil nodes. In particular, they extract features from user profiles and use them to learn a binary classifier to determine the probability that each node is a victim, and then they use these probabilities to influence the random walk used in SybilRank.

CIA is also based on a random walk with a special initial probability distribution, but it differs from SybilRank in two major aspects. First, CIA starts the random walk from labeled malicious nodes. Second, in each iteration, CIA restarts the random walk from the special initial probability distribution with some probability. Since the random walk is started from malicious nodes, 1 - p_v is treated as the reputation score of node v, where p_v is the stationary probability of v. Then those reputation scores are used to rank nodes. CIA scales well to large OSNs. However, CIA is unable to incorporate known benign labels and thus is fundamentally inapplicable in settings where a set of Sybil labels are unavailable. We found that CIA is also not resilient to label noise.

Alvisi et al. [8] studied Personalized PageRank (PPP) as a Sybil ranking mechanism. Unlike SybilRank, PPP incorporates only one benign label by initiating a random walk from the labeled benign node and returning back to it in each step with some probability. Similar to SybilRank, the normalized stationary probability distribution of PPP is used to rank all the users. SybilRank is better than PPP because PPP only incorporates one benign label.

These random walk based Sybil ranking approaches have complexity of $O(n\log n)$ [26], where n is the number of nodes in the social network. Our SybilBelief has a complexity of $O(nd)$, where d is the number of iterations. In practice, $\log n$ and d are very similar. In our experiments, we compared SybilBelief with SybilRank and CIA. We found that their computation times are similar.

Community detection based Sybil classification: Viswanath et al. [125] showed that the Sybil detection problem can be cast as a community detection problem. In their experimental evaluation, the authors found that using a simple local community detection algorithm proposed

by Mislove et al. [89] had equivalent results to using the state-of-art Sybil detection approaches. However, their scheme has computational complexity as high as $O(n^2)$, and thus does not scale to multi-million node social networks. Their approach is also unable to simultaneously incorporate both known benign and Sybil nodes. Moreover, Alvisi et al. [8] showed that their local community detection algorithm is not robust to advanced attacks by constructing such an attack.

Cai and Jermaine [25] proposed to detect Sybils using a latent community detection algorithm. With a hierarchical generative model for the observed social network, detecting Sybils is mapped to an Bayesian inference problem. Cai and Jermaine adopted Gibbs sampling, an instance of Markov chain Monte Carlo (MCMC) method, to perform the inference. However, it is well known in the machine learning community that the MCMC method is not scalable.

Defense in depth: Orthogonal to the above approaches, another active line of researches treat Sybil detection as a binary classification problem. Specifically, they extract features from social structure [135], and non-structural information such as user-generated contents (e.g., user profiles, tweets, and posts) [104, 113, 118, 133], user behaviors (e.g., the frequency of sending messages/tweets, likes) [27, 111, 124], and users' clickstreams (i.e., a sequence of HTTP requests made by users) [127]. Then they train off-the-shelf machine learning classifiers using these features.

We recently proposed to combine these feature-based approaches with SybilBelief to provide defense in depth [45]. The key intuition is that if an attacker tries to evade feature-based detections, the resulting social structure is appropriate for structure-based detections. Specifically, classifiers (e.g., Support Vector Machines [32], Logistic Regression [64]) that are learned in the feature-based approaches output a continuous score for each node, which is then used to determine the corresponding label. These scores can be used to initialize the node potentials (i.e., prior knowledge of being benign or malicious) of the nodes in our SybilBelief. We demonstrated that, via evaluations on a large-scale Twitter dataset with real Sybils, this defense in depth approach significantly outperforms previous structure-based and feature-based approaches.

5.2.2 Trust Propagation

Another line of research, which is closely related to Sybil detection, is the trust propagation problem. Several approaches have been proposed to propagate trust scores or reputation scores in file-sharing networks or auction platforms (e.g., [54, 65, 103]). Similar to SybilRank [26] and CIA [133], these approaches are variants of the PageRank algorithm [23]. In principle, these approaches can be applied to detect Sybils. Specifically, nodes with low trust scores could be classified as Sybils [26].

5.2.3 Markov Random Fields and Belief Propagation

The Markov Random Fields (MRF) has many applications in electrical engineering and computer science such as computer vision [33] and natural language processing [83]. However, the application of MRFs to the security and privacy area is rather limited.

Computing posterior distributions in probabilistic graphical models is a central problem in probabilistic reasoning. It was shown that this problem is NP-hard on general graphs [31]. Pearl proposed belief propagation algorithm for exact inference on trees, and he also noted that, on

graphs with loops, this algorithm leads to oscillations that prohibit any convergence guarantees [99]. Nevertheless, belief propagation on loopy graphs has often been used in practical applications and has demonstrated satisfying approximate performance [94].

5.3 Attribute Inference and Link Prediction

5.3.1 Attribute Inference

Friend-based attribute inference: He et al. [59] transformed attribute inference to Bayesian inference on a Bayesian network that is constructed using the social links between users. They evaluated their method using a LiveJournal social network dataset with *synthesized* user attributes. Moreover, it is well known in the machine learning community that Bayesian inference is not scalable. Lindamood et al. [80] modified Naive Bayes classifier to incorporate social links and other attributes of users to infer some attribute. For instance, to infer a user's major, their method used the user's other attributes such as employer and cities lived, the user's social friends and their attributes. However, their approach is not applicable to users that share no attributes at all. Heatherly et al. [60] further studied how to prevent such inference attacks.

Zheleva and Getoor [142] studied various approaches to consider both social links and groups that users joined to perform attribute inference. They found that, with only social links, the approach LINK achieves the best performance. LINK represents each user as a binary feature vector, and a feature has a value of 1 if the user is a friend of the person that corresponds to the feature. Then LINK learns classifiers for attribute inference using these feature vectors.

Mislove et al. [89] proposed to identify a local community in the social network by taking some seed users that share the same attribute value, and then they predicted all users in the local community to have the shared attribute value. Their approach is not able to infer attributes for users that are not in any local communities. Moreover, this approach is data dependent since detected communities might not correlate with the attribute value. For instance, Trauda et al. [122] found that communities in a MIT male network are correlated with residence but a female network does not have such property.

Our approaches are friend-based. We transform the attribute inference problem into a link prediction problem with the SAN model. Therefore, any link prediction algorithm can be used to infer missing attributes. More importantly, we demonstrate that attribute inference can in turn help link prediction with the SAN model.

Behavior-based attribute inference: Weinsberg et al. [131] investigated the inference of gender using the rating scores that users gave to different movies. In particular, they constructed a feature vector for each user; the ith entry of the feature vector is the rating score that the user gave to the ith movie if the user reviewed the ith movie, otherwise the ith entry is 0. They compared a few classifiers including Logistic Regression (LG) [64], SVM [32], and Naive Bayes [84], and they found that LG outperforms the other approaches. Bhagat et al. [14] studied attribute inference in an active learning framework. Specifically, they investigated which movies we should ask users to review in order to improve the inference accuracy the most. However, this approach might not be applicable in real-world scenarios because users might not be interested in reviewing the selected movies.

Chaabane et al. [28] used the information about the musics users like to infer attributes. They augmented the musics with the corresponding Wikipedia pages and then used LDA to identify the latent similarities between musics. A user is predicted to share attributes with those that like similar musics with the user. Kosinski et al. [69] tried to infer various attributes based on the list of pages that users liked on Facebook. Similar to the work performed by Weinsberg et al. [131], they constructed a feature vector from the Facebook likes and used Logistic Regression to train classifiers to distinguish users with different attributes. Luo et al. [81] constructed a model to infer household structures using users' viewing behaviors in Internet Protocol Television (IPTV) systems, and they showed promising results.

Other approaches: Otterbacher [97] studied the inference of gender using sources of information such as users' writing styles and comments associated with movie reviews. Gupta et al. [55] tried to infer interests of a Facebook user via sentiment-oriented mining on the Facebook pages that were liked by the user. Zhong et al. [143] demonstrated the possibility of inferring user attributes using the list of locations where the user has checked in. These studies are orthogonal to ours since they exploited information sources other than the social structures and behaviors that we focus on.

5.3.2 Link Prediction

A wide range of link prediction methods have been developed. For instance, models of complex networks, such as Preferential Attachment [12], SAN model [52] and Hierarchical model [29] can be viewed as models for predicting links. Liben-Nowell and Kleinberg [78] surveyed a set of unsupervised link prediction algorithms. Li [76] proposed Maximal Entropy Random Walk (MERW). Lichtenwalter et al. [79] proposed the PropFlow algorithm which is similar to RWwR but more localized. However, none of these approaches leverage node attribute information.

Link prediction methods leveraging attribute information first appear in the relational learning community [16, 88, 116, 141]. However, these approaches suffer from scalability issues. For instance, the largest network tested in [116] has about $3K$ nodes. Recently, Backstrom and Leskovec [9] proposed the Supervised Random Walk (SRW) algorithm to leverage edge attributes. However, SRW does not handle the scenario in which two nodes share common attributes (e.g., nodes u_2 and u_5 in Fig. 4.1), but no edge already exists between them. Mapping link prediction to a classification problem [35, 58, 79] is another way to incorporate attributes. We have shown that classifiers using features extracted from the SAN model perform very well. Yang et al. [134] proposed to jointly predict links and propagate node interests (e.g., music interest). Their algorithm relies on the assumption that each node interest has a set of explicit attributes. As a result, their algorithm cannot be applied to our scenario in which it's hard (if possible) to extract explicit attributes for our node attributes.

Chapter 6

Conclusion

Online social networking services are among the most popular web services. They have become increasingly important resources for interacting with people, processing information, and diffusing social influence. However, they are also vulnerable to both classical and new security and privacy risks. In this thesis, we discuss our proposals to make online social networking services secure against compromised or lost user accounts and Sybil attacks, and we introduce new attacks to the privacy of social networking users, which have implications for the design of privacy-preserving online social networking services. In particular, we first show that trusted friends can be leveraged to design secure account recovery methods. Second, we demonstrate that probabilistic graphical model techniques can be adapted to prevent Sybil attacks. Third, we show that diverse private information can be inferred with high accuracies from public data on social networking sites.

Bibliography

[1] "http://technicsandtime.com/2013/11/26/1-in-10-twitter-accounts-is-fake-say-researchers/, retrieved in march 2015."

[2] "http://www.alexa.com/topsites, retrieved in march 2015."

[3] "http://www.peekyou.com/."

[4] "http://www.spokeo.com/."

[5] "http://www.trustedreviews.com/news/instagram-now-has-more-users-than-twitter, retrieved in march 2015."

[6] "http://investor.fb.com/releasedetail.cfm?ReleaseID=893395, retrieved in march 2015."

[7] L. A. Adamic and E. Adar, "Friends and neighbors on the web," *Social Network*, vol. 25, no. 3, pp. 211–230, 2003.

[8] L. Alvisi, A. Clement, A. Epasto, S. Lattanzi, and A. Panconesi, "SoK: The evolution of Sybil defense via social networks," in *IEEE S & P*, 2013.

[9] L. Backstrom and J. Leskovec, "Supervised random walks:predicting and recommending links in social networks," in *WSDM*, 2011.

[10] BadRank, "http://pr.efactory.de/e-pr0.shtml."

[11] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, 1999.

[12] ——, "Emergence of scaling in random networks," *Science*, vol. 286, pp. 509–512, 1999.

[13] S. Bartunov, A. Korshunov, S.-T. Park, W. Ryu, and H. Lee, "Joint link-attribute user identity resolution in online social networks," in *SNA-KDD*, 2012.

[14] S. Bhagat, U. Weinsberg, S. Ioannidis, and N. Taft, "Recommending with an agenda: Active learning of private attributes using matrix factorization," in *RecSys*, 2014.

[15] L. Bilge, T. Strufe, D. Balzarotti, and E. Kirda, "All your contacts are belong to us: Automated identity theft attacks on social networks," in *WWW*, 2009.

[16] M. Bilgic, G. Namata, and Getoor, "Combining collective classification and link prediction," in *ICDM Workshops*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 381–386.

[17] J. Bonneau, E. Bursztein, I. Caron, R. Jackson, and M. Williamson, "Secrets, lies, and account recovery: Lessons from the use of personal knowledge questions at google," in *WWW*, 2015.

[18] J. Bonneau and S. Preibusch, "The password thicket: technical and market failures in human authentication on the web," in *WEIS*, 2010.

[19] Y. Boshmaf, D. Logothetis, G. Siganos, J. Lería, J. Lorenzo, M. Ripeanu, and K. Beznosov, "Integro: Leveraging victim prediction for robust fake account detection in osns," in *NDSS*, 2015.

[20] D. M. Boyd and N. B. Ellison, "Social network sites: Definition, history, and scholarship," *Journal of Computer-Mediated Communication*, vol. 13, no. 1, 2007.

[21] J. Brainard, A. Juels, R. Rivest, M. Szydlo, and M. Yung, "Fourth-factor authentication: Somebody you know," in *CCS*, 2006.

[22] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," *Computer Networks and ISDN Systems*, vol. 30, no. 1-7, pp. 107–117, 1998.

[23] ——, "The anatomy of a large-scale hypertextual web search engine," *Computer Networks and ISDN Systems*, vol. 30, no. 1-7, 1998.

[24] J. P. J. Bunnell and R. Henderson, "Cost-effective computer security: Cognitive and associative passwords," in *OZCHI*, 1996.

[25] Z. Cai and C. Jermaine, "The latent community model for detecting Sybils in social networks," in *NDSS*, 2012.

[26] Q. Cao, M. Sirivianos, X. Yang, and T. Pregueiro, "Aiding the detection of fake accounts in large scale social online services," in *NSDI*, 2012.

[27] Q. Cao, X. Yang, J. Yu, and C. Palow, "Uncovering large groups of active malicious accounts in online social networks," in *CCS*, 2014.

[28] A. Chaabane, G. Acs, and M. A. Kaafar, "You are what you like! information leakage through users' interests," in *NDSS*, 2012.

[29] A. Clauset, C. Moore, and M. E. J. Newman, "Hierarchical structure and the prediction of missing links in networks," *Nature*, vol. 453, no. 7191, pp. 98–101, May 2008.

[30] A. Clauset, C. R. Shalizi, and M. E. J. Newman, "Power-law distributions in empirical data," *SIAM Review*, no. 51, 2009.

[31] G. F. Cooper, "The computational complexity of probabilistic inference using bayesian belief networks," *Artificial Intelligence*, vol. 42, no. 2-3, 1990.

[32] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, 1995.

[33] G. Cross and A. Jain, "Markov random field texture models," *IEEE Trans. PAMI*, vol. 5, 1983.

[34] G. Danezis and P. Mittal, "SybilInfer: Detecting Sybil nodes using social networks," in *NDSS*, 2009.

[35] J. R. Doppa, J. Yu, P. Tadepalli, and L. Getoor, "Learning algorithms for link prediction based on chance constraints," in *ECML/PKDD*, 2010, pp. 344–360.

[36] J. R. Douceur, "The Sybil attack," in *IPTPS*, 2002.

[37] D. Easley and J. Kleinberg, *Networks, Crowds, and Markets: Reasoning About a Highly Connected World*. Cambridge University Press, 2010.

[38] M. Egele, G. Stringhini, C. Kruegel, and G. Vigna, "Compa: Detecting compromised accounts on social networks," in *NDSS*, 2013.

[39] P. Erdős and A. Rényi., "On random graphs i," *Publ. Math. Debrecen*, vol. 6, 1959.

[40] Facebook's Trusted Contacts, "goo.gl/xHmVHA."

[41] Facebook's Trusted Friends, "goo.gl/KdyYXJ."

[42] T. L. Fond and J. Neville, "Randomization tests for distinguishing social influence and homophily effects," in *WWW*. New York, NY, USA: ACM, 2011, pp. 601–610.

[43] P. L. Fong, "Preventing Sybil attacks by privilege attenuation: A design principle for social network systems," in *IEEE S & P*, 2011.

[44] H. Gao, J. Hu, C. Wilson, Z. Li, Y. Chen, and B. Zhao., "Detecting and characterizing social spam campaigns," in *IMC*, 2010.

[45] P. Gao, N. Z. Gong, S. Kulkarni, K. Thomas, and P. Mittal, "Sybilframe: A defense-in-depth framework for structure-based sybil detection," in *arXiv*, 2015.

[46] E. Gilbert and K. Karahalios, "Predicting tie strength with social media," in *CHI*, 2009.

[47] N. Z. Gong, M. Frank, and P. Mittal, "Sybilbelief: A semi-supervised learning approach for structure-based sybil detection," *IEEE TIFS*, vol. 9, no. 6, 2014.

[48] N. Z. Gong, A. Talwalkar, L. Mackey, L. Huang, E. C. R. Shin, E. Stefanov, E. R. Shi, and D. Song, "Joint link prediction and attribute inference using a social-attribute network," *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2014.

[49] N. Z. Gong, A. Talwalkar, L. Mackey, L. Huang, E. C. R. Shin, E. Stefanov, E. Shi, and D. Song, "Jointly predicting links and inferring attributes using a social-attribute network (san)," in *SNA-KDD*, 2012.

[50] N. Z. Gong and D. Wang, "On the security of trustee-based social authentications," *IEEE TIFS*, vol. 9, no. 8, 2014.

[51] N. Z. Gong and W. Xu, "Reciprocal versus parasocial relationships in online social networks," *Springer Social Network Analysis and Mining (SNAM)*, vol. 4, no. 1, 2014.

[52] N. Z. Gong, W. Xu, L. Huang, P. Mittal, E. Stefanov, V. Sekar, and D. Song, "Evolution of social-attribute networks: Measurements, modeling, and implications using google+," in *IMC*, 2012.

[53] Google Explores +1 Button To Influence Search Results, "http://www.tekgoblin.com/2011/08/29/google-explores-1-button-to-influence-search-results/."

[54] R. Guha, R. Kumar, P. Raghavan, and A. Tomkins, "Propagation of trust and distrust," in *WWW*, 2004.

[55] P. Gupta, S. Gottipati, J. Jiang, and D. Gao, "Your love is public now: Questioning the use of personal information in authentication," in *AsiaCCS*, 2013.

[56] D. J. Hand and R. J. Till, "A simple generalisation of the area under the roc curve for multiple class classification problems," *Machine Learning*, vol. 45, 2001.

[57] ——, "A simple generalisation of the area under the roc curve for multiple class classification problems," *Machine Learning*, vol. 45, pp. 171–186, 2001.

[58] M. A. Hasan, V. Chaoji, S. Salem, and M. Zaki, "Link prediction using supervised learning," in *SIAM Workshop on Link Analysis, Counterterrorism and Security*, 2006.

[59] J. He, W. W. Chu, and Z. V. Liu, "Inferring privacy information from social networks," in *IEEE Intelligence and Security Informatics (ISI)*, 2006.

[60] R. Heatherly, M. Kantarcioglu, and B. Thuraisingham, "Preventing private information inference attacks on social networks," *TKDE*, vol. 25, no. 8, 2013.

[61] P. Jaccard, "Étude comparative de la distribution florale dans une portion des alpes et des jura," *Bulletin de la Société Vaudoise des Sciences Naturelles*, 1901.

[62] S. Jain, N. Z. Gong, S. Basuroy, J. Lang, D. Song, and P. Mittal, "New directions in social authentication," in *USEC*, 2015.

[63] T. Joachims, "Making large-scale SVM learning practical," in *Advances in Kernel Methods - Support Vector Learning*, 1999.

[64] D. W. H. Jr and S. Lemeshow, *Applied logistic regression*. John Wiley & Sons, 2004.

[65] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, "The Eigentrust algorithm for reputation management in P2P networks," in *WWW*, 2003.

[66] D. Kempe, J. Kleinberg, and E. Tardos, "Maximizing the spread of influence through a social network," in *KDD*, 2003.

[67] H. Kim, J. Tang, and R. Anderson, "Social authentication: Harder than it looks," in *FC*, 2012.

[68] M. Kim and J. Leskovec, "Modeling social networks with node attributes using the multiplicative attribute graph model," in *UAI*, 2011.

[69] M. Kosinski, D. Stillwell, and T. Graepel, "Private traits and attributes are predictable from digital records of human behavior," *Proceedings of the National Academy of Sciences*, 2013.

[70] G. Kossinets, "Effects of missing data in social networks," *Social Networks*, vol. 28, pp. 247–268, 2006.

[71] G. Kossinets and D. Watts, "Empirical analysis of an evolving social network," *Science*, 2006.

[72] R. Kumar, J. Novak, P. Raghavan, and A. Tomkins, "Structure and evolution of blogspace," *Communications of the ACM*, vol. 47, no. 12, pp. 35–39, 2004.

[73] H. Kwak, C. Lee, H. Park, and S. Moon, "What is twitter, a social network or a news media?" in *WWW*, 2010.

[74] J. Leskovec, L. Backstrom, R. Kumar, , and A. Tomkins, "Microscopic evolution of social networks," in *KDD*. ACM, 2008, pp. 462–470.

[75] J. Leskovec, J. M. Kleinberg, and C. Faloutsos, "Graphs over time: densification laws, shrinking diameters and possible explanations," in *KDD*, 2005, pp. 177–187.

[76] R.-H. Li, J. X. Yu, and J. Liu, "Link prediction: the power of maximal entropy random walk," in *CIKM*, 2011.

[77] D. Liben-Nowell and J. Kleinberg, "The link prediction problem for social networks," in *CIKM*, 2003.

[78] ——, "The link prediction problem for social networks," in *CIKM*, 2003, pp. 556–559.

[79] R. N. Lichtenwalter, J. T. Lussier, and N. V. Chawla, "New perspectives and methods in link prediction," in *KDD*, 2010.

[80] J. Lindamood, R. Heatherly, M. Kantarcioglu, and B. Thuraisingham, "Inferring private information using social network data," in *WWW*, 2009.

[81] D. Luo, H. Xu, H. Zha, J. Du, R. Xie, X. Yang, and W. Zhang, "You are what you watch and when you watch: Inferring household structures from iptv viewing data," *IEEE Transactions on Broadcasting*, 2014.

[82] D. Malkhi, Y. Mansour, and M. K. Reiter, "Di usion without false rumors: on propagating updates in a byzantine environment," *Theoretical Computer Science*, 2003.

[83] C. D. Manning and H. Schutze, *Foundations of Statistical Natural Language Processing*, 1999.

[84] A. McCallum and K. Nigam, "A comparison of event models for naive bayes text classification," in *AAAI*, 1998.

[85] F. McSherry and M. Najork, "Computing information retrieval performance measures efficiently in the presence of tied scores," in *ECIR*, 2008.

[86] P. Melville and V. Sindhwani, "Recommender systems," *Encyclopedia of Machine Learning, Springer*, 2010.

[87] A. K. Menon and C. Elkan, "Link prediction via matrix factorization," in *ECML/PKDD*, 2011.

[88] K. T. Miller, T. L. Griffiths, and M. I. Jordan, "Nonparametric latent feature models for link prediction," in *NIPS*, 2009.

[89] A. Mislove, B. Viswanath, K. Gummadi, and P. Druschel, "You are who you know: Inferring user profiles in online social networks," in *Proceedings of the 3rd ACM International Conference of Web Search and Data Mining*, 2010.

[90] A. Mislove, H. S. Koppula, K. P. Gummadi, P. Druschel, and B. Bhattacharjee, "Growth of the flickr social network," in *WOSN*, 2008.

[91] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, and B. Bhattacharjee, "Measurement and analysis of online social networks," in *IMC*, 2007.

[92] A. Mohaisen, N. Hopper, and Y. Kim, "Keep your friends close: Incorporating trust into social network-based Sybil defenses," in *INFOCOM*, 2011.

[93] A. Mohaisen, A. Yun, and Y. Kim, "Measuring the mixing time of social graphs," in *IMC*, 2010.

[94] K. Murphy, Y. Weiss, and M. I. Jordan, "Loopy belief-propagation for approximate inference: An empirical study," in *UAI*, 1999.

[95] Node Centrality, "https://en.wikipedia.org/wiki/Centrality."

[96] K. Okamoto, W. Chen, and X.-Y. Li, "Ranking of closeness centrality for large-scale social networks," in *FAW*, 2008.

[97] J. Otterbacher, "Inferring gender of movie reviewers: exploiting writing style, content and metadata," in *CIKM*, 2010.

[98] J.-Y. Pan, H.-J. Yang, C. Faloutsos, and P. Duygulu., "Automatic multimedia cross-modal correlation discovery," in *KDD*, 2003.

[99] J. Pearl, *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann Publishers Inc., 1988.

[100] I. Polakis, P. Ilia, F. Maggi, M. Lancini, G. Kontaxis, S. Zanero, S. Ioannidis, and A. D. Keromytis, "Faces in the distorting mirror: Revisiting photo-based social authentication," in *CCS*, 2014.

[101] I. Polakis, M. Lancini, G. Kontaxis, F. Maggi, S. Ioannidis, A. D. Keromytis, and S. Zanero, "All your face are belong to us: Breaking facebook's social authentication," in *ACSAC*, 2012.

[102] A. Rice, "Facebook's knowledge-based social authentication. http://blog.facebook.com/blog.php?post=486790652130."

[103] M. Richardson, R. Agrawal, and P. Domingos, "Trust management for the semantic web," in *ISWC*, 2003.

[104] G. S. S. Yardi, D. Romero and D. Boyd, "Detecting spam in a Twitter network," *First Monday*, vol. 15(1), 2010.

[105] G. Sabidussi, "The centrality index of a graph," *Psychometrika*, 1966.

[106] S. Schechter, A. J. B. Brush, and S. Egelman, "It's no secret: Measuring the security and reliability of authentication via 'secret' questions," in *IEEE S & P*, 2009.

[107] S. Schechter, S. Egelman, and R. W. Reeder, "It's not what you know, but who you know," in *CHI*, 2009.

[108] J. Scripps, P.-N. Tan, F. Chen, and A.-H. Esfahanian, "A matrix alignment approach for link prediction," in *ICPR*, 2008.

[109] ——, "A matrix alignment approach for collective classification," in *ASONAM*, Athens, Greece, 2009.

[110] A. P. Singh and G. J. Gordon, "Relational learning via collective matrix factorization," in *KDD*, 2008.

[111] J. Song, S. Lee, and J. Kim, "Spam filtering in Twitter using sender-receiver relationship," in *RAID*, 2011.

[112] Spam messages, "http://en.wikipedia.org/wiki/Botnet."

[113] G. Stringhini, C. Kruegel, and G. Vigna, "Detecting spammers on social networks," in *ACSAC*, 2010.

[114] P. Symeonidis, E. Tiakas, and Y. Manolopoulos, "Transitive node similarity for link prediction in social networks with positive and negative links," in *RecSys*, 2010.

[115] A. Talwalkar, S. Kumar, and H. Rowley, "Large-scale manifold learning," in *CVPR*, 2008, pp. 273–297.

[116] B. Taskar, M.-F. Wong, P. Abbeel, and D. Koller, "Link prediction in relational data," in *NIPS*, 2003.

[117] K. Thomas, C. Grier, J. Ma, V. Paxson, and D. Song, "Design and evaluation of a real-time url spam filtering service," in *IEEE S & P*, 2011.

[118] K. Thomas, C. Grier, and V. Paxson, "Adapting social spam infrastructure for political censorship," in *LEET*, 2012.

[119] H. Tong, C. Faloutsos, and J.-Y. Pan, "Fast random walk with restart and its applications," in *ICDM*, 2006.

[120] D. N. Tran, B. Min, J. Li, and L. Subramanian, "Sybil-Resilient online content rating," in *NSDI*, 2009.

[121] N. Tran, J. Li, L. Subramanian, and S. S. Chow, "Optimal Sybil-resilient node admission control," in *INFOCOM*, 2011.

[122] A. L. Trauda, P. J. Muchaa, and M. A. Porter, "Social structure of facebook networks," *Physica A: Statistical Mechanics and its Applications*, vol. 391, no. 16, 2012.

[123] B. Viswanath, A. Misolve, M. Cha, and K. P. Gummadi, "On the evolution of user interaction in Facebook," in *WOSN*, 2009.

[124] B. Viswanath, M. A. Bashir, M. Crovella, S. Guha, K. P. Gummadi, B. Krishnamurthy, and A. Mislove, "Towards detecting anomalous user behavior in online social networks," in *USENIX Security*, 2014.

[125] B. Viswanath, A. Post, K. P. Gummadi, and A. Mislove, "An analysis of social network-based Sybil defenses," in *SIGCOMM*, 2010.

[126] C. Wang, V. Satuluri, and S. Parthasarathy, "Local probabilistic models for link prediction," in *ICDM*, 2007.

[127] G. Wang, T. Konolige, C. Wilson, and X. Wang, "You are how you click: Clickstream analysis for sybil detection," in *Usenix Security*, 2013.

[128] G. Wang, M. Mohanlal, C. Wilson, X. Wang, M. Metzger, H. Zheng, and B. Y. Zhao, "Social turing tests: Crowdsourcing Sybil detection," in *NDSS*, 2013.

[129] Y. Wang and A. Nakao, "Poisonedwater: An improved approach for accurate reputation ranking in p2p networks," *Future Generation Computer Systems (FGCS)*, vol. 26, no. 8, 2010.

[130] W. Wei, F. Xu, C. Tan, and Q. Li, "SybilDefender: Defend against Sybil attacks in large social networks," in *INFOCOM*, 2012.

[131] U. Weinsberg, S. Bhagat, S. Ioannidis, and N. Taft, "Blurme: Inferring and obfuscating user gender based on ratings," in *RecSys*, 2012.

[132] C. Wilson, B. Boe, A. Sala, K. P. Puttaswamy, and B. Y. Zhao, "User interactions in social networks and their implications," in *Eurosys*, 2009.

[133] C. Yang, R. Harkreader, J. Zhang, S. Shin, and G. Gu, "Analyzing spammer's social networks for fun and profit," in *WWW*, 2012.

[134] S. H. Yang, B. Long, A. Smola, N. Sadagopan, Z. Zheng, and H. Zha, "Like like alike — joint friendship and interest propagation in social networks," in *WWW*, 2011, pp. 537–546.

[135] Z. Yang, C. Wilson, X. Wang, T. Gao, B. Y. Zhao, and Y. Dai, "Uncovering social network Sybils in the wild," in *IMC*, 2011.

[136] S. Yardi, N. Feamster, and A. Bruckman, "Photo-based authentication using social networks," in *WOSN*, 2008.

[137] Z. Yin, M. Gupta, T. Weninger, and J. Han, "Linkrec: a unified framework for link recommendation with user attributes and graph structure," in *WWW*, 2010, pp. 1211–1212.

[138] ——, "A unified framework for link recommendation using random walks," in *ASONAM*, 2010.

[139] H. Yu, P. B. Gibbons, M. Kaminsky, and F. Xiao, "SybilLimit: A near-optimal social network defense against Sybil attacks," in *IEEE S & P*, 2008.

[140] H. Yu, M. Kaminsky, P. B. Gibbons, and A. Flaxman., "SybilGuard: Defending against Sybil attacks via social networks," in *SIGCOMM*, 2006.

[141] K. Yu, W. Chu, S. Yu, V. Tresp, and Z. Xu, "Stochastic relational models for discriminative link prediction," in *NIPS*, 2006.

[142] E. Zheleva and L. Getoor, "To join or not to join: The illusion of privacy in social networks with mixed public and private user profiles," in *WWW*, 2009.

[143] Y. Zhong, N. J. Yuan, W. Zhong, F. Zhang, and X. Xie, "You are where you go: Inferring demographic attributes from location check-ins," in *WSDM*, 2015.

[144] M. Zviran and W. J. Haga, "User authentication by cognitive passwords: an empirical assessment," in *JCIT*, 1990.

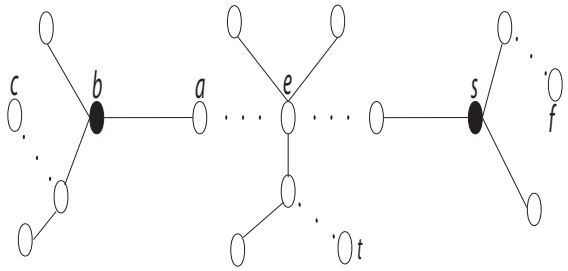


Figure A.1: Illustration of a Pairwise Markov Random Fields on a tree.

Appendix A

Proof of Theorems

A.1 Proof of Theorem 2

Figure A.1 illustrates a case where Theorem 2 holds. We will prove the theorem via analytically deriving the posterior distribution for each unlabeled node.

Since the network is a tree, there exists only one path between b and s. We denote the set of nodes on the path between nodes u and v as the set P_{uv}. Without loss of generality, we assume there are other nodes (e.g., e in Figure A.1) on the path P_{bs}. We divide the unlabeled nodes into 4 categories and derive posterior distributions for them one by one. Intuitively, in Figure A.1, these four categories are : I) nodes (e.g., e) on the path P_{bs}, II) nodes (e.g., t) on the sides of P_{bs}, III) nodes (e.g., c) on the left side of b, and IV) nodes (e.g., f) on the right side of s. Mathematically, we can define these four categories using their paths to b and s. Specifically, for any unlabeled node u with path P_{ub} to b and path P_{us} to s, we classify it as follows:

- *Category I*: $P_{ub} \subset P_{bs}$ and $P_{us} \subset P_{bs}$.
- *Category II*: $|P_{ub} - P_{bs}| > 0$, $|P_{us} - P_{bs}| > 0$, $|P_{ub} \cap P_{bs}| \geq 2$ and $|P_{us} \cap P_{bs}| \geq 2$.
- *Category III*: $P_{bs} \subset P_{us}$.
- *Category IV*: $P_{bs} \subset P_{ub}$.

In the following analysis of the four categories of nodes, instead of the flooding protocol [94] used in our empirical evaluations, we use serialized message-passing protocol [99] because it makes

our analysis easier. In this protocol, node u will send a message to its neighbor v only if u has received messages from all other neighbors. As a result, the message in each direction of each edge is passed only once. Furthermore, we normalize each message to have sum 1. Note that this normalization has no influence on the final posterior distributions. After all the messages are passed, belief propagation converges and exact posterior distributions for all nodes can be inferred. We let $P_{+1}(u) = P(x_u = +1|x_{\{b,s\}} = y_{\{b,s\}})$ and $P_{-1}(u) = P(x_u = -1|x_{\{b,s\}} = y_{\{b,s\}})$ in the following.

A.1.1 Category I

Category I nodes are all on the path P_{bs}. For any such node e, it will receive messages from its two neighbors e_l and e_r on the path P_{bs} and possibly from some nodes in Category II. Obviously, however, messages from Category II nodes have no influence on the posterior distribution of e. As a result, e's posterior distribution is only determined by the two messages from e_l and e_r. We call the message from the neighbor who is closer to b as the *benign* message and denote it as $m_b^{(d_{eb})}$, and call the other one as the *malicious* message and denote it as $m_s^{(d_{es})}$, where d_{eb} and d_{es} are e's distances to b and s respectively. Note that each message has two dimensions, and we use subscript +1 and -1 to represent them. Moreover, +1 corresponds to Sybil. In order to calculate the posterior distribution of e, we first prove the following lemma.

Lemma 3. *For any Category I node e with distance d_{eb} to b and d_{es} to s, its benign and malicious messages satisfy the following iterative equations:*

$$\begin{cases} m_{b,+1}^{(d_{eb})} + m_{b,-1}^{(d_{eb})} = 1 \\ m_{b,+1}^{(d_{eb})} - m_{b,-1}^{(d_{eb})} = -(2w-1)^{d_{eb}} \end{cases}$$

$$\begin{cases} m_{s,+1}^{(d_{es})} + m_{s,-1}^{(d_{es})} = 1 \\ m_{s,+1}^{(d_{es})} - m_{s,-1}^{(d_{es})} = (2w-1)^{d_{es}} \end{cases}$$

Proof. By symmetry, we only need to prove the iterative equations for benign messages. And we do so by mathematical induction. When $d_{eb} = 1$, node e is a neighbor of b, i.e., $e = a$ in Figure A.1. According to the serialized message passing protocol and our model, b needs to collect messages from other neighbors and combine them with its edge potential $\varphi_{ba}(x_b, x_a)$ before sending the message $m_b^{(1)}$ to a. In our simplified model, however, the messages from all other neighbors have no influence on the message $m_b^{(1)}$. Therefore, $m_b^{(1)}$ can be calculated using $\varphi_{ba}(x_b, x_a)$, i.e., $m_{b,+1}^{(1)} = 1 - w$ and $m_{b,-1}^{(1)} = w$. It's straightforward to verify this message satisfies the iterative equations. Furthermore, we can show that these iterative equations are satisfied for $d_{eb} + 1$ if they are satisfied for d_{eb}. Thus we have proved Lemma 3 as desired. □

The posterior probability of e being Sybil is $P_{+1}(e) = \frac{m_{b,+1}^{(d_{eb})} m_{s,+1}^{(d_{es})}}{(m_{b,+1}^{(d_{eb})} m_{s,+1}^{(d_{es})} + m_{b,-1}^{(d_{eb})} m_{s,-1}^{(d_{es})})}$. With Lemma 3, we have:

$$\frac{P_{+1}(e)}{P_{-1}(e)} = \frac{(1-(2w-1)^{d_{eb}})(1+(2w-1)^{d_{es}})}{(1+(2w-1)^{d_{eb}})(1-(2w-1)^{d_{es}})}$$

$$\frac{P_{+1}(e) - P_{-1}(e)}{P_{-1}(e)} = \frac{2((2w-1)^{d_{es}} - (2w-1)^{d_{eb}})}{(1+(2w-1)^{d_{eb}})(1-(2w-1)^{d_{es}})}$$

Thus, under the natural parameter settings, i.e., $0.5 < w < 1$, we obtain that $P_{+1}(e) > P_{-1}(e)$ if $d_{eb} > d_{es}$, $P_{+1}(e) < P_{-1}(e)$ if $d_{eb} < d_{es}$, and $P_{+1}(e) = P_{-1}(e)$ if $d_{eb} = d_{es}$. So we have proved that Theorem 2 holds for Category I nodes.

A.1.2 Category II

For any node t from Category II, without loss of generality, we assume the first intersection node between the paths P_{ts} and P_{bs} to be e, which is illustrated in Figure A.1. In our simplified model, the posterior distribution of t is only determined by the message from the neighbor that is on the path P_{te}. We denote this message as $m^{(d_{te})}$. Again, by mathematical induction, we have the following iterative equations:

$$\begin{cases} m_{+1}^{(d_{te})} + m_{-1}^{(d_{te})} = 1 \\ m_{+1}^{(d_{te})} - m_{-1}^{(d_{te})} = (P_{+1}(e) - P_{-1}(e))(2w-1)^{d_{te}} \end{cases}$$

. Since the posterior distribution of t is only determined by the message $m^{(d_{te})}$, we have $P_{+1}(t) = m_{+1}^{(d_{te})}$ and $P_{-1}(t) = m_{-1}^{(d_{te})}$. Thus, $P_{+1}(t) - P_{-1}(t) = (P_{+1}(e) - P_{-1}(e))(2w-1)^{d_{te}}$. So the label of t is the same as that of e. Since $d_{tb} = d_{te} + d_{eb}$ and $d_{ts} = d_{te} + d_{es}$, we conclude that Theorem 2 also holds for Category II nodes.

A.1.3 Category III and IV

For any Category III node, e.g., c in Figure A.1, its posterior distribution is only determined by the message from the neighbor that is on the path P_{cb}. We denote this message as $m^{(d_{cb})}$, where d_{cb} is the distance between c and b. Again, we can derive the following iterative equations by mathematical induction:

$$\begin{cases} m_{+1}^{(d_{cb})} + m_{-1}^{(d_{cb})} = 1 \\ m_{+1}^{(d_{cb})} - m_{-1}^{(d_{cb})} = -(2w-1)^{d_{cb}} \end{cases}$$

Under our natural parameter settings, we get $m_{+1}^{(d_{cb})} < m_{-1}^{(d_{cb})}$ for all c. Furthermore, $P_{+1}(c) = m_{+1}^{(d_{cb})}$ and $P_{-1}(c) = m_{-1}^{(d_{cb})}$. Thus all Category III nodes are labeled as benign. And we also have $d_{cb} < d_{cs}$ for any Category III node c. So we have proved that Theorem 2 also holds for Category III nodes. By symmetry, we can show that Theorem 2 holds for Category IV nodes