Defending against Inference Attack in Online Social Networks

by

Jiayi Chen
B.Eng., Shanghai Jiao Tong University, China, 2015

A Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF APPLIED SCIENCE

in the Department of Electrical and Computer Engineering

Defending against Inference Attack in Online Social Networks

by

Jiayi Chen
B.Eng., Shanghai Jiao Tong University, China, 2015

Supervisory Committee

---

Dr. Lin Cai, Supervisor
(Department of Electrical and Computer Engineering)

---

Dr. Issa Traoré, Departmental Member
(Department of Electrical and Computer Engineering)

**Supervisory Committee**

---

Dr. Lin Cai, Supervisor
(Department of Electrical and Computer Engineering)

---

Dr. Issa Traoré, Departmental Member
(Department of Electrical and Computer Engineering)

## ABSTRACT

The privacy issues in online social networks (OSNs) have been increasingly arousing the public awareness since it is possible for attackers to launch several kinds of attacks to obtain users' sensitive and private information by exploiting the massive data obtained from the networks. Even if users conceal their sensitive information, attackers can infer their secrets by studying the correlations between private and public information with background knowledge. To address these issues, the thesis focuses on the inference attack and its countermeasures.

First, we study how to launch the inference attack to profile OSN users via relationships and network characteristics. Due to both user privacy concerns and unformatted textual information, it is quite difficult to build a completely labeled social network directly. However, both social relations and network characteristics can help attribute inference to profile OSN users. We propose several attribute inference models based on these two factors and implement them with Naïve Bayes, Decision Tree, and Logistic Regression. Also, to study network characteristics and evaluate the performance of our proposed models, we use a well-labeled Google employee social network extracted from Google+ for inferring the social roles of Google employees. The experiment results demonstrate that the proposed models are effective in social role inference with Dyadic Label Model performing the best.

Second, we model the general inference attack and formulate the privacy-preserving data sharing problem to defend against the attack. The optimization problem is to maximize the users' self-disclosure utility while preserving their privacy. We propose two privacy-preserving social network data sharing methods to counter the inference

attack. One is the efficient privacy-preserving disclosure algorithm (EPPD) targeting the high utility, and the other is to convert the original problem into a multi-dimensional knapsack problem (d-KP) which can be solved with a low computational complexity. We use real-world social network datasets to evaluate the performance. From the results, the proposed methods achieve a better performance when compared with the existing ones.

Finally, we design a privacy protection authorization framework based on the OAuth 2.0 protocol. Many third-party services and applications have integrated the login services of popular social networking sites, such as Facebook and Google+, and acquired user information to enrich their services by requesting user's permission. However, due to the inference attack, it is still possible to infer users' secrets. Therefore, we embed our privacy-preserving data sharing algorithms in the implementation of OAuth 2.0 framework and propose RANPriv-OAuth2 to protect users' privacy from the inference attack.

# Contents

# List of Tables

# List of Figures

# ACKNOWLEDGEMENTS

First and foremost, I would like to thank my supervisor Professor Lin Cai for her great support throughout my study and research at the University of Victoria in the past two years. Thanks to her patient guidance and inspiring advice, I can keep making progress in my research work. It is always my great honor to pursue the master's degree under the supervision of Professor Lin Cai. I would also like to thank Professor Issa Traoré and Professor Alex Thomo for taking time reviewing my thesis.

I am also very grateful to Professor Jianping Pan for his great help and valuable comments on my research papers. I would also like to thank Dr. Jianping He for sharing his research experience and improving my work without reservation. They helped me get through the difficulties at the beginning of my research life.

Besides, I would like to thank all my friends in the Communication Networks Lab, Dr. Yongming Zhang, Dr. Yi Chen, Zhe Wei, Haoyuan Zhang, Yue Li, Yuanzhi Ni, Mohammad Ghasemiahmadi, Wen Cui, and Hamed Mosavat. I really cherish the time spent with my dear friends on study, research, and recreation. Thanks to them, I have a wonderful and unforgettable time at the University of Victoria.

Last but certainly not least, I would like to thank my parents for their endless love. Thanks to their support, I have the opportunity to chase my dreams.

Jiayi Chen

# DEDICATION

*Dedicated to my family and friends.*

# Chapter 1

# Introduction

## 1.1 Privacy Issues in Online Social Networks

Along with the explosive development of online social networks (OSNs), an increasing number of people prefer establishing their own network on the Internet. For example, Facebook has 1.01 billion daily active users on average in September 2015 [13], and Google+, since its launch in 2011, has attracted more than 300 million active users. On one hand, the accounts in popular social networking sites have become people's second identity since they record users' detailed profile information (*attributes*) and interpersonal relationships (*social relations*). On the other hand, given the massive information and social characteristics, online social networks have played an important role in several research and industry areas, e.g., community study, criminal networks and demography. As a result, people have to face the privacy issues in OSNs.

From the perspective of OSN users, the privacy in OSNs is quite paradoxical. People are willing to share part of their personal information to find new friends with similar interests, which is called *self-disclosure* [27]. However, due to the *privacy concerns* [11], OSN users are reluctant to disclose their full set of personal information. OSN users' definitions of privacy and secrets vary from individual to individual, which may depend on several factors including the sensitiveness of user information, personal preferences, and the roles of the objects which want to access their OSN data. The common method for privacy settings in OSNs is that the social network service providers let users determine whether a specific field is open to the public or not. However, there are extensive works and surveys [21, 36, 39, 40] having shown that it might not be a good solution. Worse still, it is possible to infer the hidden and secret

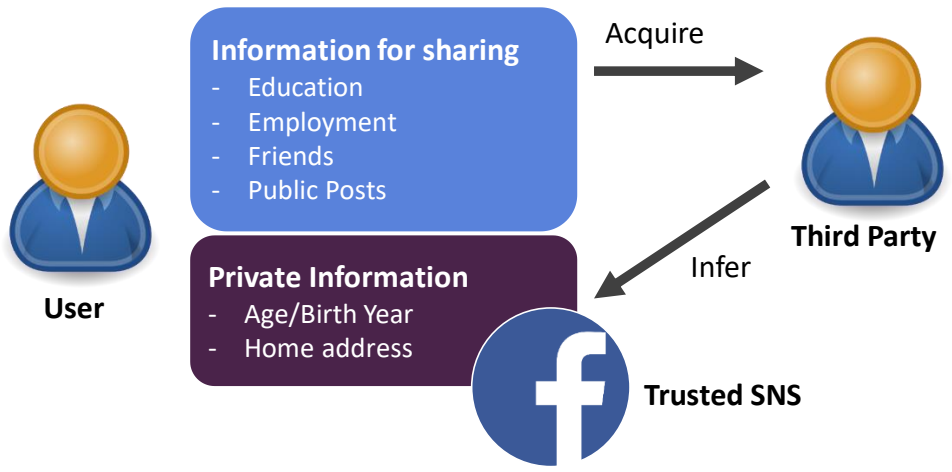



Figure 1.1: Inference attack launched by the third-party

information of OSN users with high accuracy by exploiting the public information, which is called *inference attack* [29]. Thus, unsafe self-disclosure may be followed by potential privacy leaks, leading to targeted spams, reputation damage and even property loss.

For many commercial or research purposes, the information of an individual in the social network is sometimes trivial. Instead, the statistical and structural information of the whole network is preferred. In this case, privacy-preserving data publishing has been studied to utilize the anonymized social networks without violating individual privacy and defend against the re-identification attack [22, 44]. However, the inference attack is quite different. People use data mining techniques to discover the knowledge from the anonymized social networks for marketing and analytics. It might be illegitimately overused for inference attack to infer the unauthorized information, especially when it comes to the customized or targeted services such as recommendations and advertising, where the identity and public information of the OSN users are exposed to the service providers (e.g., logging services with Facebook account information). The main privacy concern here is the inference attack on user secrets as shown in Figure 1.1. For various types of inference methods [9, 14, 43, 56], the principle is to find out the targeted secrets based on the information extracted from the published dataset and background knowledge of the attackers. For example, an attacker can train a classifier from the training dataset to predict whether a user has a certain secret. When a new social network is published, the attacker extracts

features from the public information of the targeted user as the input of the classifier and then infers the secret. In the whole process, the training dataset can be viewed as the background knowledge, and the extracted features can be regarded as the observation. Both the quality of the training set and the performance of the attacker's classifier affect the effectiveness of the inference attack.

## 1.2 Research Problems

In this thesis, we mainly study the inference attack in OSNs and its countermeasures to protect users' privacy. We try to address the following research problems.

- **How to launch the inference attack via social relations and network characteristics.** The typical inference attack mainly utilizes the textual information and meta information fetched from the databases of the social networking services. In the social networks, it is more preferable to use the structural information and network characteristics. However, the problem is quite challenging when the network information is not complete (for example, only a sub-graph of the original network is available). It drives us to look into the more basic components like dyadic and triadic relations in the social networks.

- **How to defend against the inference attack.** Since the adversary needs to observe the public information to make the inference, it is feasible to process (e.g., modify, obfuscate or mask) the public information to reduce the accuracy of the inference attack. However, when it comes to the non-anonymous situations, it is not suitable to add the noise and misleading information. Therefore, we need to design the masking/disclosure algorithms to counter the inference attack effectively and efficiently.

- **How to maximize the safe self-disclosure with privacy guarantee.** The nature of social networking services is to help users exchange information and establish relationships with others on the Internet. Masking public information for privacy protection will affect the utility of the social networking services and user experience. Therefore, we need to define the privacy constraints for the safe self-disclosure and make the trade-off between the self-disclosure and privacy protection.

- **How to implement the privacy protection module and make it compatible with the existing protocol.** Once we design the privacy-preserving disclosure algorithms, it is important to implement them in the social networking services. Since the algorithms are designed to control the shared information with others, we need to make the implementation compatible with the existing authorization protocol which is responsible for the access control.

## 1.3 Contributions

Although extensive efforts have been made to improve the inference attack and design the classifiers to recover the missing attributes and relations in a social network [9, 14, 17, 33, 38, 45, 47, 56], there are few works on how to prevent from the inference attack in OSNs. In this thesis, we aim to analyze the inference attack model, propose the possible solutions to protect users' secrets from being inferred, and design a novel and practical protection framework. The main contributions of this thesis are listed as follows.

- We observe and analyze a fully labeled real-world Google employee social network extracted from Google+ dataset, and then propose several inference attack models from two aspects, social relations and network characteristics, respectively. The proposed models can be also utilized to complete the missing information and profile the OSN users.

- We formulate the privacy-preserving social network data sharing problem as an optimization problem which aims to maximize the safe self-disclosure with privacy constraints. To solve the optimization problem, we propose two different masking algorithms, the EPPD algorithm and the d-KP algorithm, for different purposes.

- We conduct extensive experiments on two large real-world datasets to evaluate the performance of our proposed privacy-preserving data sharing algorithms and compare with the existing work. The results show that our algorithms perform better and counter the inference attack effectively.

- To implement our proposed algorithms, we also design a privacy-preserving authorization framework, RANPriv-OAuth, which enhances the privacy protection

of the instances of the popular authorization protocol OAuth 2.0. The framework enables flexible privacy settings and trust management to fulfill different privacy requirements and concerns.

In Chapters 4, 5, and 6, we will discuss the novelty and contributions in each aspect of our work in detail.

## 1.4 Thesis Outline

The rest of this thesis is organized as follows. In the next chapter, we summarize the related work on social network privacy and the differences between our work and the existing ones. In Chapter 3, we introduce the social network models and metrics used for this thesis. Chapter 4 focuses on the inference attack models via relationships and network characteristics with several experiments on the performance comparison. As the countermeasures against the inference attack, two masking/disclosure algorithms are proposed in Chapter 5. In Chapter 6, we implement and demonstrate a privacy-preserving online social network data sharing framework RANPriv-OAuth2 with the OAuth 2.0 authorization protocol. Finally, Chapter 7 concludes the whole thesis and describes the possible improvement for the future work.

# Chapter 2

# Related Work

## 2.1 Attribute Inference and Inference Attack

There have been extensive works on profiling online social network users with public information, and they can be grouped into two categories.

In the first category, some of the approaches focus on attribute inference via non-network features, such as textual and graphic information in user public profiles and posts [14, 32]. Even with location data, attackers can infer user demographics and sensitive information[33, 51]. For this kind of approaches, the correlation between public and latent attributes is the key factor to conduct inference.

The approaches in the second category try to utilize network characteristics and social relations to infer the latent attributes and links. McPherson et al. [42] pointed out that social relation based attribute inference is based on the phenomenon of homophily in social networks. This phenomenon is also observed in our Google+ dataset and utilized in our models. Besides inferring user profile, homophily also has some other applications (e.g., friend recommendation [52]). The basic idea of network characteristics based attribute inference is to find the correlation between these characteristics and latent attributes. In [9, 56], it is found that the reach of nodes plays an important role in inferring user demographics and social roles. However, it is observed in our work that the importance of a certain network feature varies in different social networks. Specifically, in the Google+ dataset, users of different classes may have so similar reach distributions that we can hardly use it to distinguish Google+ users.

There are also many efforts to combine both public attributes and social relations

to improve the performance of attribute inference. It has been shown in [43] that with a number of users with known attributes and the social network graph, the attacker can infer the attributes of other users. In addition, Zheleva et al. [57] indicated that not only friendship links can reveal sensitive attributes, but group membership information can also contribute to attribute inference. Another model called Social Attribute Networks (SAN) was used for link prediction and attribute inference. In [18], the SAN model can accurately reproduce the attribute structure of real social networks according to a variety of network metrics. And [17] showed that the SAN model can help to infer hidden attributes and predict social links. After first inferring missing attributes, the accuracy of the link-prediction algorithm can be improved greatly. The objective of our work is to profile users with complete social relations and no public attributes, which is more challenging and remains open.

When the adversary tries to infer the sensitive and secret attributes, it will become the inference attack. It is well studied to infer user demographics [9, 31], social roles [6, 56], hidden attributes [14, 43, 57], user activities [47], etc., in online social networks. Most of the inference attacks extract the features from the published data as the input of trained models to obtain the most probable secrets. To describe attackers' capabilities, Qian et al. [45, 46] used the knowledge graph where each relation connecting two entities is a piece of knowledge. In our work, the background knowledge of an attacker is captured by a social-attribute network [17, 18] where each piece of knowledge is a tree-like inference path involving at least 3 nodes.

To defend against the inference attack, Heatherly et al. [23] studied the inference attack based on Naive Bayes and introduced a correspondent protection approach. Different from our work, the principle of its masking algorithm is to remove the most highly indicative attributes and social relations without considering the correlation among public information, which makes it hard to satisfy a variety of utility metrics and the inference attacks based on other statistical learning methods. For our EPPD algorithm, we consider the correlations among public attributes or social relations, and consider customized utility values of them.

## 2.2 Privacy Protection in OSNs

### 2.2.1 Privacy-preserving Social Network Data Publishing

Privacy-preserving data publishing has been a well-studied research topic which studies how to publish useful data with protecting the data privacy. The whole data publishing process includes three roles, *record owner*, *data publisher*, and *data recipient* [16]. The data publisher collects data from the record owners and provides it for the data recipient. Since the raw data may contain sensitive information of individuals, it is necessary to anonymize it before the publication. The social network data publishing is similar to the typical one. The host social networking services provide the anonymized network or graph attributes for others to do the analytics and research. With massive network-centric data, social network data publication is vulnerable to various attacks including the re-identification attack and the inference attack [1].

There are extensive works on privacy-preserving social network data sharing and publishing, which mainly focused on anonymization techniques. Li et al. [34] proposed a graph-based privacy-preserving data publishing framework to construct several subgraphs and perform the existing anonymity operations independently for each subgraph. Wang et al. [49] studied how to outsource social networks with indistinguishability. The introduction of differential privacy [10] also provides a solid theoretical foundation for social network data publishing. Jorgensen et al. [26] involved differential privacy guarantees into attributed social graph publishing, and Day et al. [8] proposed a differential privacy based graph degree distribution publishing method. The scenario of this thesis is quite different from that of the typical privacy-preserving social network data publishing problem. On one hand, the users are not necessarily anonymized since the third-party also provides services to them. In this case, introducing misleading information like adding edges and edge swapping are not preferable. One the other hand, we not only consider the social network graph itself, but also take the profile attributes into account, while most related works for privacy-preserving social network data publishing focus on graph and graph statistics publishing only.

### 2.2.2 Controlled Information Sharing in OSNs

One of the typical security and privacy issues in OSNs is how to protect data from unauthorized access [4]. To address this issue, there are extensive works on the access

control models, which concentrate on how to share the social network data according to identity, relationship and data sensitivity [3, 7, 24]. They mainly studied the design of access control policies to secure the private social network information based on the trust among users. In addition, for the specific resources like photos, Xu et al. [53] proposed a distributed consensus-based method to control the photo sharing with friends by using an efficient facial recognition system. However, few of these works take the inference attack into account, which can be conducted via authorized access. Our work is from the novel perspective of the correlations between public and private information, which makes it possible to integrate with the existing access control models to defend against the inference attack.

## 2.3 OAuth 2.0 Protocol

OAuth 2.0 protocol is a new and popular industry-standard protocol for authorization in the web services [20]. It provides a web-based single sign-on (SSO) scheme for the OSN users. Different from the first generation, it simplifies the authorization process with security guarantee. There are four roles involved in the authorization process, resource owner, client, authorization server, and resource server, respectively. Among them, the authorization server and the resource server are located in the host social networking services. Similar to the roles in the social network data publishing, the host social networking services maintain the data of resource owner (OSN users) and provide data to the clients after authorization. However, the authorization process is emphasized in the protocol to satisfy the security requirements. Recent researches also focus on the security analysis and enhancement of OAuth 2.0 protocol and its implementations. Fett et al. [15] provided the first extensive formal analysis of OAuth 2.0 and proved the security of it. Sun et al. [48] and Chen et al. [5] examined the existing implementations of three OAuth identity providers and found several vulnerabilities in the implementations. In this thesis, we try to integrate our privacy-preserving social network data sharing algorithms with the implementation of OAuth 2.0 protocol in order to enhance the privacy protection in the post-authorization stages (resource access). In Chapter 6, we will introduce how the RANPriv-OAuth2 privacy protection framework works with the OAuth 2.0.

# Chapter 3

# Social Network Models & Metrics

## 3.1 Modeling Social Networks

We adopt different social network models for the inference attack (in Chapter 4) and its countermeasures (in Chapter 5). Since we mainly study how to launch the inference attack to profile OSN users by exploiting the relationships and network characteristics, we use the typical social network models to describe the social relations among OSN users only for this case. As for the countermeasures, we consider both the inference attack from the perspectives of profile information and structure information. Thus, we use a extended social-attribute network model to model both social relations and attribute links. Furthermore, we can also express the background knowledge of an adversary by means of the social-attribute network. Table 3.1 shows the symbols used in this chapter.

### 3.1.1 Social Network Model

We use a graph $G = \{V, E, L\}$ to describe a social network, where $V$ is the set of vertexes (social nodes), $E$ is the set of edges (social relations) and $L$ is the list of node labels. Each social node $v_i \in V$ represents an actor in the network labeled by $L_i \in C_0$, and each edge $e \in E$ connecting two nodes represents the relationship between them. There are two kinds of social relations, namely directed relations ($v_i$ follows $v_j$) and undirected relations ($v_i$ and $v_j$ are friends). In a directed social network graph, $e_{i,j} \in E$ means that node $v_i$ follows node $v_j$, and node $v_i$ and node $v_j$ are friends if and only if $e_{i,j}, e_{j,i} \in E$. Note that node $v_i$'s neighbor (friend) set is denoted as $\mathcal{N}_i$, a set of any node that $v_i$ has undirected relation with, i.e., $\mathcal{N}_i = \{v_j | e_{i,j}, e_{j,i} \in E\}$.

Table 3.1: Notation

| Symbol | Definition |
|---|---|
| $V_N$ / $V$ | Vertex set of social actors |
| $V_A$ | Vertex set of attributes |
| $V_L$ | labeled node set |
| $V_U$ | unlabeled node set |
| $E_N$ / $E$ | Edge set of social relations |
| $E_A$ | Edge set of attribute links |
| $v_i$ | a social node numbered $i$ |
| $C_0$ | the label/class set |
| $C_1$ | the label/class pair set |
| $L_i$ | label of $v_i$ |
| $L_{j,k}$ | label pattern of edge $e_{j,k}$ |
| $\mathcal{N}_u$ / $\mathcal{N}_a$ | Social actors connected with actor $u$/attribute $a$ |
| $\mathcal{N}_i$ | neighbor node set of $v_i$ |
| $\mathcal{N}_i^k$ | neighbor node set of $v_i$ with $C_0^k$ |
| $\mathcal{T}_i$ | social triad set of $v_i$'s ego network |
| $\mathcal{T}_i^k$ | social triad set of $v_i$'s ego network with $C_1^k$ |
| $\mathcal{A}_u$ | Attribute set of actor $u$ |
| $\mathcal{S}_u$ | Secret attributes of actor $u$ |
| $\mathcal{P}_u$ | Public attributes of actor $u$ |

In our work, we mainly focus on undirected social networks. From the perspective of each ego network, there are two levels of relationships, namely dyadic relationship (social tie) and triadic relationship (social triad) as shown in Figure 3.1. Dyadic relationship is the direct connection between the ego node and its neighbors while triadic relationship also involves the connections among neighbors. A social triad usually indicates stronger social relations, and a node included in multiple social triads is more likely to play an important role in the ego network [12].

Because a social triad $t_{i,j,k}$ in $v_i$'s ego network can be viewed as the ego node $v_i$ associating with a dyadic relation $e_{j,k}$, we define $\mathcal{T}_i = \{e_{j,k} | v_j, v_k \in \mathcal{N}_i\}$ as the set of all dyadic relations in node $v_i$'s neighborhood. We use $L_{j,k}$ to describe the label pattern of an edge $e_{j,k} \in \mathcal{T}_i$. With $|C_0|$ labels, the maximum number of possible label pattern excluding the ego node is $|C_1| = \binom{|C_0|+1}{2} = \frac{(|C_0|+1)|C_0|}{2}$.

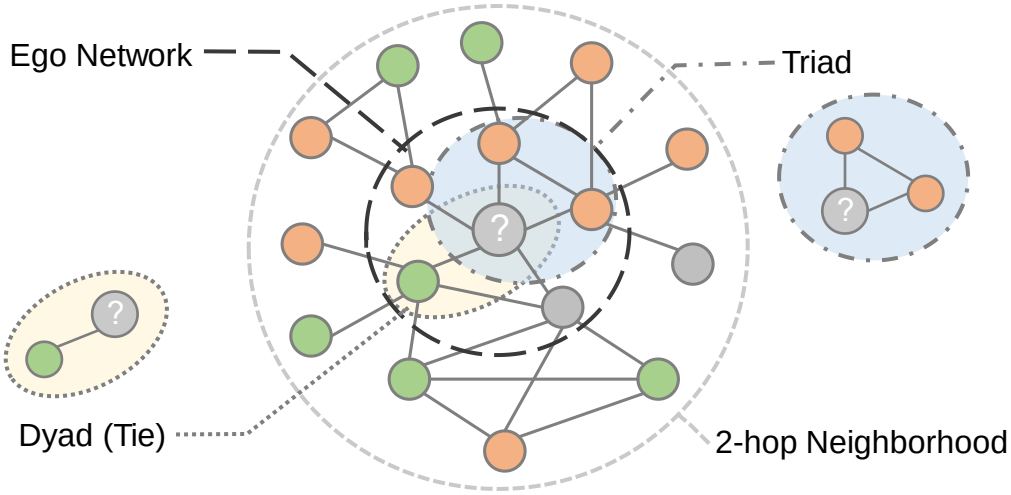



Figure 3.1: An example of social network model

### 3.1.2 Social-Attribute Network

A social network is usually described as a graph consisting of social actors and relationships where user attributes (e.g., profile information) are used to label or group social actors. To present both social actors and attributes together, we use a social-attribute network graph $G = \{V_N, V_A, E_N, E_A\}$, where $V_N$ is the social actor set, $V_A$ is the attribute node set, $E_N$ is the social relation set and $E_A$ is the attribute link set. The social-attribute network model is first proposed by Yin et al. [54], and it is widely used for social network analysis, link prediction and hidden attribute inference [18, 17]. There are two types of attributes: *categorical attribute* and *numerical attribute*. A categorical attribute belongs to a certain category in the user profile, where all candidates can be enumerated. If an attribute is described as a number, it can be regarded as a numerical attribute.

1. **Categorical Attribute**: A categorical attribute belongs to a certain category in the user profile, where all candidates can be enumerated. For example, "Engineer" belongs to the "Job" category.

2. **Numerical Attribute**: If an attribute is described as a number, it can be regarded as a numerical attribute. Age, height, weight, etc. are common numerical attributes.

In our model, in order to represent a numerical attribute with a node, it has to be converted into a categorical attribute by using the interval or ordinal variables. For

example, "Age: 26" can be shown as "Age: 20–29" or "Young". For simplicity, despite the fact that a category can be described with different granularities (e.g., location can be "Mountain View, CA" or only "CA"), we consider all attributes belonging to the same category are in the same level.

A social relation $(u, v) \in E_N$ means that $u$ and $v$ are friends in an undirected network, or $u$ follows $v$ in a directed network, while an attribute link $(u, v) \in E_A$ means that $u$ has the attribute $v$. We use an indicator function $t_u^G(v) \in \{0, 1\}$ to indicate the existence of edge $(u, v)$ in graph $G$. In a social-attribute network, there are two kinds of edges: *actor-to-attribute* and *actor-to-actor*. We can use the neighborhood information to form node sets where nodes have the same attributes or common friends. For a social actor $u$, the friend set (in undirected networks) of node $u$ is denoted as $\mathcal{N}_u = \{v | v \in V_N, (u, v) \in E_N\}$, and the attribute set of social actor $u$ is denoted as $\mathcal{A}_u = \{a | a \in V_A, (u, a) \in E_A\}$. Similarly, social actors sharing the same attribute $a$ are all involved in the set $\mathcal{N}_a = \{v | v \in V_N, (v, a) \in E_A\}$. Furthermore, we can obtain the social actor set with multiple common attributes and relationships by calculating the intersection of the corresponding neighborhood sets. For example, A's friends who are photographers can be expressed by $\mathcal{N}_A \cap \mathcal{N}_{\text{Photographer}}$.

## 3.2 Metrics & Network Characteristics

To explore the social structures and study the interaction among social actors, it is also necessary to study the correlation between network characteristics and user latent information such as demographics and social roles. Zhao et al. [56] indicated that network reach is the most significant factor related with social roles and statuses. Dong et al. [9] proposed the DFG algorithm with mixed network features to infer user demographics. Besides, there are several ways to evaluate the importance of a certain attribute or a social relation so that we can assign it with a weight for designing the disclosure algorithms in Chapter 5. Here, we consider the following social network characteristics and metrics.

### 3.2.1 Attribute Metrics

The number of its social actor neighbors shows how common the attribute is in the whole social network.

**Uniqueness**: OSN users prefer to show their differences from the others, and a

less common attribute usually provides more information. Here we use the inverse of the logarithm of the attribute node's degree centrality to calculate the uniqueness score.

$$p_U(a) = \frac{1}{\log(|\mathcal{N}_a|) + 1}. \tag{3.1}$$

**Commonness**: On the contrary, the third parties are more interested in the common attributes for analysis. We can simply use the normalized degree centrality in the whole network. To ensure the structure of community, we use the degree centrality in the ego network of the targeted node.

$$p_C(a, u) = \frac{|\mathcal{N}_u \cap \mathcal{N}_a|}{|\mathcal{N}_u|}. \tag{3.2}$$

### 3.2.2 Social Relation Metrics

An edge's value can be determined by the node similarity between two social actors connected. Intuitively, two similar social actors (sharing a lot of attributes in common) have a stronger tie between them.

**Jaccard Coefficient**: The normalized common neighbor metric describes the similarity of two social actors. The more common friends will bring a higher Jaccard coefficient.

$$p_J(e_{u,v}) = \frac{|\mathcal{N}_u \cap \mathcal{N}_v|}{|\mathcal{N}_u \cup \mathcal{N}_v|}. \tag{3.3}$$

**Adamic/Adar Score:** Adamic et al. [2] proposed this score to describe the similarity between two web pages. Different from the Jaccard coefficient, it considers the weight of each common feature. For our experiments, Adamic/Adar score is defined as follows.

$$p_A(e_{u,v}) = \sum_{k \in \mathcal{F}_u \cap \mathcal{F}_v} \frac{1}{\log |\mathcal{N}_k|}, \tag{3.4}$$

where the common feature with a smaller degree centrality weighs more.

### 3.2.3 Network Characteristics

The network characteristics are used to describe the structural features of the whole network or a single node in the network. In this thesis, we mainly focus on the node-level characteristics.

**Homophily**: The tendency that a node has relations with those individuals who have similar attributes [42]. We use the likelihood that a node $v_i$ connects to other

nodes with the same label to evaluate the extent of homophily.

$$H_i = \frac{|\{v_j | v_j \in \mathcal{N}_i, L_i = L_j\}|}{|\mathcal{N}_i|} \tag{3.5}$$

**Local Clustering Coefficient**: a measure of the degree to which a node $v_i$'s neighbors tend to be triads. It can be calculated as follows.

$$\begin{aligned} \text{LCC}_i &= \frac{2|\{e_{j,k} | v_j, v_k \in \mathcal{N}_i, e_{j,k} \in E\}|}{|\mathcal{N}_i|(|\mathcal{N}_i| - 1)} \\ &= \frac{2|\mathcal{T}_i|}{|\mathcal{N}_i|(|\mathcal{N}_i| - 1)} \end{aligned} \tag{3.6}$$

**Degree Centrality**: the number of links incident upon a node $v_i$. To obtain a normalized value, it is calculated as the fraction of nodes it is connected to.

$$\text{DC}_i = \frac{|\mathcal{N}_i|}{|V|} \tag{3.7}$$

**Average Neighbor Degree**: the average degree of the neighborhood of node $v_i$.

$$\text{AND}_i = \frac{\sum_{v_j \in \mathcal{N}_i} |\mathcal{N}_j|}{|\mathcal{N}_i|} \tag{3.8}$$

# Chapter 4

# Inference Attack via Relationships and Network Characteristics

## 4.1 Introduction

Attribute inference is quite important when the collected data is incomplete and not fully labeled. In order to conduct accurate analysis on OSN users, researchers first need to profile them and extract essential information for study. For example, if we want to explore the relationship between political tendency and social roles, we can obtain the former from posts and the latter from profiles by information extraction technologies. However, users are not always willing to disclose their personal information to the public so that some necessary profile attributes are usually incomplete and even missing. When it comes to the sensitive and private information, excessive inference will become the inference attack. To infer the hidden information accurately, it is necessary to profile users via all the available information.

What we can obtain from an online social network includes textual, graphic and relational/network information. One possible approach for attribute inference is to find the correlation between public textual information and hidden attributes [32]. Although raw textual information is usually unformatted, it can provide relatively precise results compared with photographic information. In [14], photographs can only help infer some basic demographics such as age and gender. For further inference, supplementary information from public profiles and posts are still necessary and essential.

Compared with textual and graphic information, relationships among users are

much easier to obtain and process. Due to homophily, a phenomenon where people are more likely to establish social relations with those who have something in common, users' friends can "honestly" reflect their latent attributes. For example, an individual with a group of photographer friends is very likely to be a photographer. A few existing works have used this feature to infer latent attributes [43, 57]. However, there remains many other relational and network characteristics that can help profile users. Traditional social network analysis methods often evaluate node connections, distributions and segmentations through such metrics as centrality, closeness and clustering coefficient [12]. It has also been observed that these metrics have correlation with social roles and demographics [9, 56]. It is worthy to study both relational and network characteristics in data sets and involve them in the inference attack model.

Therefore, in this chapter, we aim to develop inference models with various features and profile users in online social networks. To achieve our goal, there are several challenges as follows.

- **Unformatted Textual Information:** If we want to profile users accurately, textual information is no doubt the first choice. However, as mentioned above, in most situations, textual information is unformatted and incomplete. Therefore, for nodes with very little textual information, we have to use relational and network features alone to infer user latent attributes in a partially labeled social network.

- **Complex Networks:** A social network is a complex network with non-trivial topological features. In most studies, a raw network crawled needs to be scaled down into a suitable size (e.g., ego networks) according to a specific research objective, since redundant information may be of little importance and even lead to a decrement of inference performance.

- **Different Online Social Networks:** Actually, there are numerous online social networks serving different purposes, which results in different social structures. In fact, social structure based models that work well in a certain online social network may not always be feasible in a completely new one. For example, LinkedIn users prefer to establish relations with friends in real life while Google+ users are prone to follow others freely as long as they have the same interests. For this reason, network characteristics of these two networks are quite different according to our observation.

The main contributions of this work are three-fold. First, we extract a real-world Google employee social network from SNAP Google+ dataset [30] and label all nodes with both Google+ and LinkedIn profile as ground-truth. In this way, we can reduce errors in the original data set as much as possible. Second, in our dataset, we observe different network characteristics from the existing work [56] in a similar scenario, which is resulted from different online social networking purposes. Third, we propose several inference attack models via social relations and network characteristics to profile individuals in partially labeled online social networks, and then in the experiment part, we test the performance of our proposed models in the case study of social role inference.

### 4.1.1 Dataset

The dataset for our work is derived from SNAP Google+ dataset [30]. The original dataset consists of 132 ego networks with totally 107,614 nodes, and after combining all these ego networks, there are 30,494,866 directed edges. Because of the limitation of the ego network, the neighboring information of nodes is not always complete. In order to obtain an accurate observation, we extracted 1,638 Google employees in SNAP dataset with 50,864 undirected edges. In our work, these Google employees are classified into 3 groups by their job titles according to Google's official career classification [19], which are *Engineering & Design (E&D)*, *Sales & Services (S&S)* and *Executive (EXE)*. To ensure the correctness of labeling, all missing and unformatted profile information can be complemented by correspondent LinkedIn profiles. Fig. 4.1 presents an overview of the Google employee social networks generated by ForceAtlas2 [25]. Each color represents one kind of social role, and the size of a node reflects its degree. From the figure, we can see that the job titles of nodes are not uniformly distributed, where E&D nodes take the largest part in this social network.

### 4.1.2 Observation

We plot these network characteristics and metrics in our dataset as shown in Fig. 4.2. From the perspective of homophily, E&D nodes are more likely to link to other nodes with the same label, while S&S nodes, as the minority in the whole network, have little probability to link with the similar nodes. For local clustering coefficient, the neighbors of an S&S node are less likely to have relations with each other, which indicates that the ego networks of S&S nodes have a lower density of ties. In contrast,

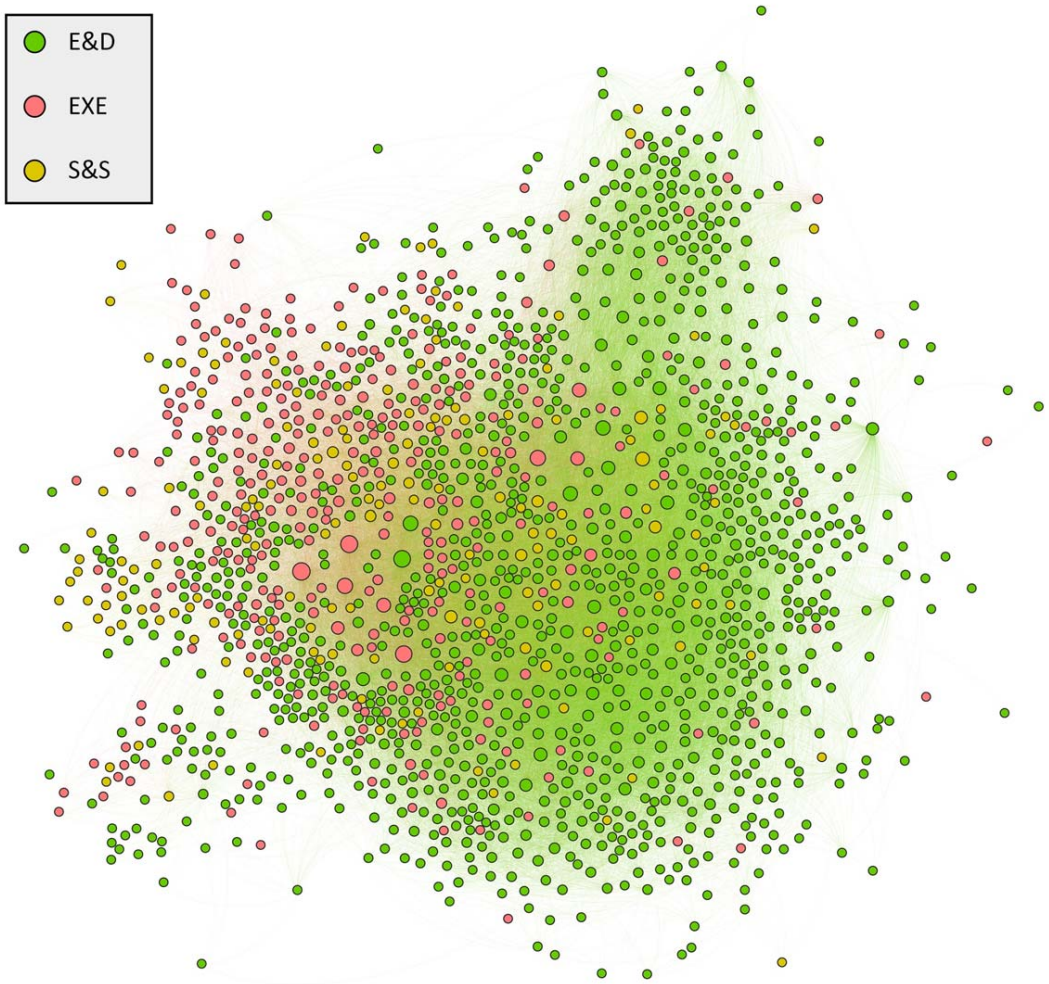



Figure 4.1: Google Employee Social Networks

E&D and EXE nodes have similar distribution of local clustering coefficient where the majority of nodes' local clustering coefficient is around 0.5 to 0.6. For the degree centrality and average neighbor degree, the observed results are quite different from those in the LinkedIn dataset [56]. In [56], the reach of the LinkedIn users (including degree centrality and average node degree) is regarded as an important factor to distinguish different groups of nodes. From our results, there is no significant different distribution of these two factors especially for average neighbor degree. The main reason is that user relationships in Google+ are not so tight as those in LinkedIn. Therefore, it is worth investigating whether these features can help profile users in Google+ dataset.

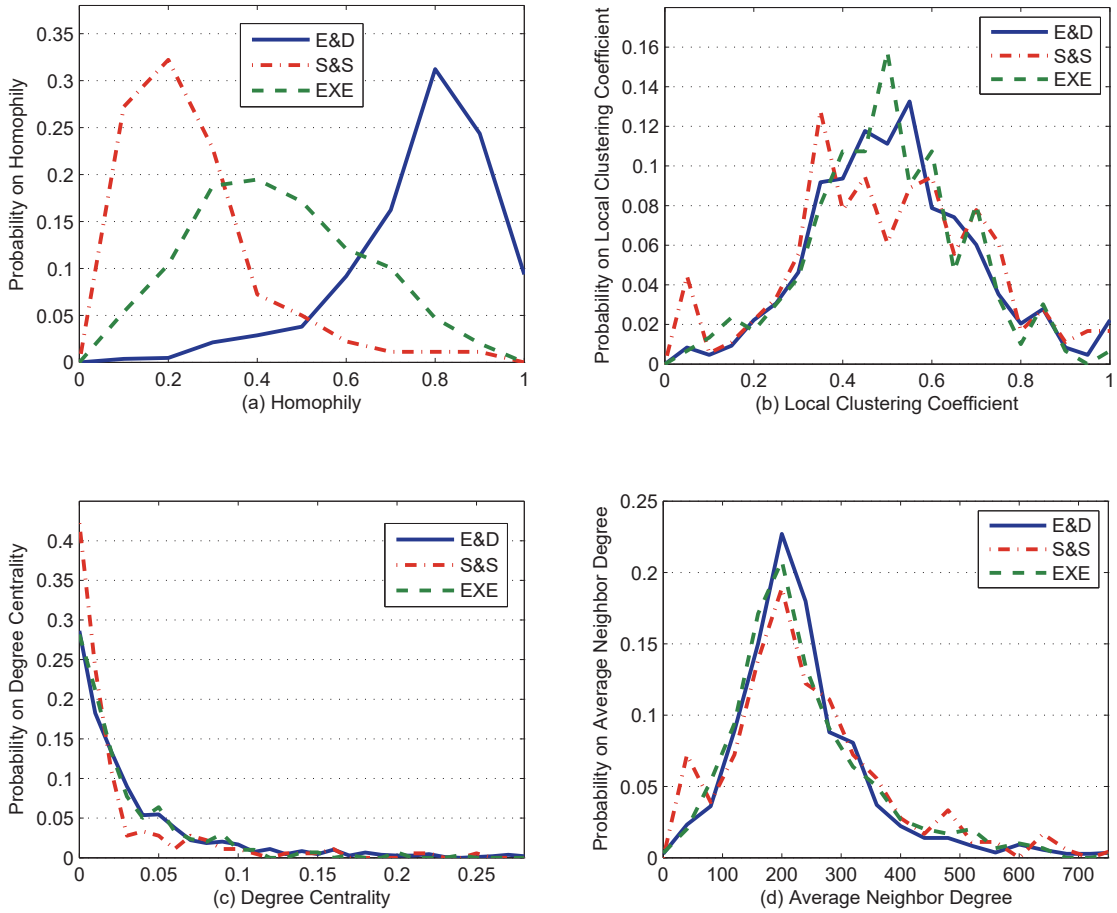



Figure 4.2: Social Network Characteristics for Google Employee Data Set

## 4.2 Inference Models

### 4.2.1 Problem Formulation

Traditional methods to profiling online social network users are mainly based on textual information including public profiles and posts [32, 23]. Considering incomplete textual information, it is also necessary to explore the potential correlation between user latent attributes and indirect information extracted from social network. It is quite common that there exist a large number of users whose profiles are not open to the public, but social relations are available. Thus, the objective of our work is to profile online social network users only via social relations and network characteristics.

We assume that users in the social network are partial labeled. Therefore, the problem can be formulated into the following form. Given a social network $G = \{V_L, V_U, E, L\}$ with labeled node set $V_L$ and unlabeled node set $V_U$, build a model

with $V_L, E, L$ to predict the labels of nodes in $V_U$. In our work, we start from the ego network of each node in the whole social network, and then obtain features from neighboring information (e.g., dyads and triads) and network metrics (e.g., Degree Centrality, Cluster Coefficient and Average Neighbor Degree).

### 4.2.2 Naïve Bayes Classification

Naïve Bayes Classification is simple to implement, and very efficient in document classification. For example, R. Heatherly et al. [23] used a Naïve Bayes classifier to predict a user's private information via public profile attributes. Similarly, we can formulate the classification problem into the following probabilistic models by replacing these profile attributes with neighboring information.

Suppose that all the nodes in a given social network can be classified into a certain class $C_0^k \in C_0$. To predict its most probable class, we have a list of neighboring information denoted by $X = \{x_1, x_2, \cdots, x_m\}$. Based on the assumption that each feature is conditionally independent from the others, the probability for a certain node $v_i$ to be classified into class $C_0^k$ can be calculated by the following expression.

$$\begin{aligned} P(C_0^k|x_1, x_2, \cdots, x_m) &= \frac{P(C_0^k) \cdot P(x_1, x_2, \cdots, x_m|C_0^k)}{P(x_1, x_2, \cdots, x_m)} \\ &= \frac{1}{Z} P(C_0^k) \cdot \prod_{j=1}^{m} P(x_j|C_0^k) \end{aligned} \tag{4.1}$$

where $Z$ is a constant scaling factor only related with $X$.

Based on this probabilistic model, we can obtain a simple Naïve Bayes classifier by choosing the most probable class label. The predicted label $\hat{y}_i$ is calculated as follows.

$$\hat{y}_i = \underset{k \in \{1,2,\cdots,|C_0|\}}{\arg\max} P(C_0^k) \cdot \prod_{j=1}^{m} P(x_j|C_0^k) \tag{4.2}$$

where the value of $\hat{y}$ is the sequence number of the most probable class for node $v_i$.

Here we propose 3 different Naïve Bayes classifier based models regarding dyadic and triadic level, namely two-hop label model, dyadic label model and triadic pattern model.

### Two-hop Label Model

In this classifier model, we consider two hop relationship for each node. We want to predict the label of a certain node from its neighbors' neighboring label distribution. The likelihood $P(N_j|C_0^k)$ can be obtained from the probability for a node linked with node $v_j$ labeled as $C_0^k$, $P(C_0^k|N_j)$, by Bayes theorem. The prediction function can be written in the following way.

$$\begin{aligned}\hat{y}_i &= \underset{k\in\{1,2,\cdots,|C_0|\}}{\arg\max} P(C_0^k)\cdot\prod_{v_j\in\mathcal{N}_i} P(N_j|C_0^k) \\ &= \underset{k\in\{1,2,\cdots,|C_0|\}}{\arg\max} P(C_0^k)\cdot\prod_{v_j\in\mathcal{N}_i} \frac{P(C_0^k|N_j)P(N_j)}{P(C_0^k)} \\ &= \underset{k\in\{1,2,\cdots,|C_0|\}}{\arg\max} P(C_0^k)\cdot\prod_{v_j\in\mathcal{N}_i} \frac{|\mathcal{N}_j^k|}{|V_L^k|} \end{aligned} \tag{4.3}$$

where $\mathcal{N}_j^k = \{n|n\in\mathcal{N}_j\bigcap V_L, L_j = C_0^k\}$ and $V_L^k = \{n|n\in V, L_j = C_0^k\}$. It seems that labels of the targeted node's neighbors are not very important because they are necessary in the prediction procedure.

### Dyadic Label Model

Different from the former model, dyadic label model concentrates on the label frequency of the targeted node's neighborhood. This model is based on the phenomenon of homophily. Because there are multiple nodes with the same label in the ego networks, we apply a multinomial Naïve Bayes classifier as is shown below. The frequency of each label appearing in the ego network is the key attribute for this model.

$$\begin{aligned}\hat{y}_i &= \underset{k\in\{1,2,\cdots,|C_0|\}}{\arg\max} P(C_0^k)\cdot\prod_{v_j\in\mathcal{N}_i} P(L_j|C_0^k) \\ &= \underset{k\in\{1,2,\cdots,|C_0|\}}{\arg\max} P(C_0^k)\cdot\prod_{l=1}^{|C_0|} P(C_0^l|C_0^k)^{|\mathcal{N}_i^l|} \end{aligned} \tag{4.4}$$

### Triadic Pattern Model

Similar with dyadic labels, triadic patterns in the ego network of a given node $v_i$ can used to infer its latent label. A triadic pattern is actually a label pair of an edge whose two ends are both connected with $v_i$. The likelihood in this model can be obtain by

calculating the probability of a node with label $C_0^k$ connecting to an edge $e_{m,n}$ with pattern $L_{m,n} \in C_1$. The triadic pattern model is also based on a multinomial Naïve Bayes classifier.

$$\hat{y}_i = \underset{k \in \{1,2,\cdots,|C_0|\}}{\arg\max} P(C_0^k) \cdot \prod_{l=1}^{|C_1|} P(C_1^l | C_0^k)^{|\mathcal{T}_i^l|} \tag{4.5}$$

These three models are all based on the neighboring information of the targeted node. The difference is that two-hop label model utilizes neighbor nodes' neighboring label distribution while dyadic label and triadic pattern models utilizes direct neighboring label/pattern distribution. Moreover, triadic pattern model considers additional third edges among neighbors, compared with dyadic label model.

### 4.2.3 Feature Based Model

The basic idea of feature based model is to extract all useful statistical and social network features, and then to distinguish nodes according to their feature vectors. To obtain the feature vectors, we need to find a feature generator function $\mathbf{x}_i = g(G, v_i)$. Given a certain machine learning algorithm $M$, we train the model with feature-label pairs $(\mathbf{x}_i, L_i)$ generated from nodes in $V_L$ to get a prediction function $f_M$. For a certain node $v_i \in V_U$, we can predict its label by $\hat{y} = f_M(x_i)$.

In the Chapter 3, we have mentioned four kinds of network characteristics, which can be used to build the feature vectors. To include more features, we extend the definition of Homophily into the label distribution in node $v_i$'s ego network.

$$H_i^k = \frac{|\{v_j | v_j \in \mathcal{N}_i, L_j = C_0^k\}|}{|\mathcal{N}_i|} \tag{4.6}$$

Then, the label distribution vector for node $v_i$ is $\mathbf{h}_i = [H_i^0, H_i^1, \cdots, H_i^{|C_0|}]^T$. Similarly, we can also involve the triadic pattern distribution as features in this model. Let $D_i^k$ denote the distribution of triad pattern $C_1^k$ in $\mathcal{T}_i$.

$$D_i^k = \frac{|\{e_{m,n} | e_{m,n} \in \mathcal{T}_i, L_{m,n} = C_1^k\}|}{|\mathcal{T}_i|} \tag{4.7}$$

where the triadic label distribution vector for node $v_i$ is $\mathbf{d}_i = [D_i^0, D_i^1, \cdots, D_i^{|C_1|}]^T$.

With the other normalized attributes, we can obtain the feature vector by the

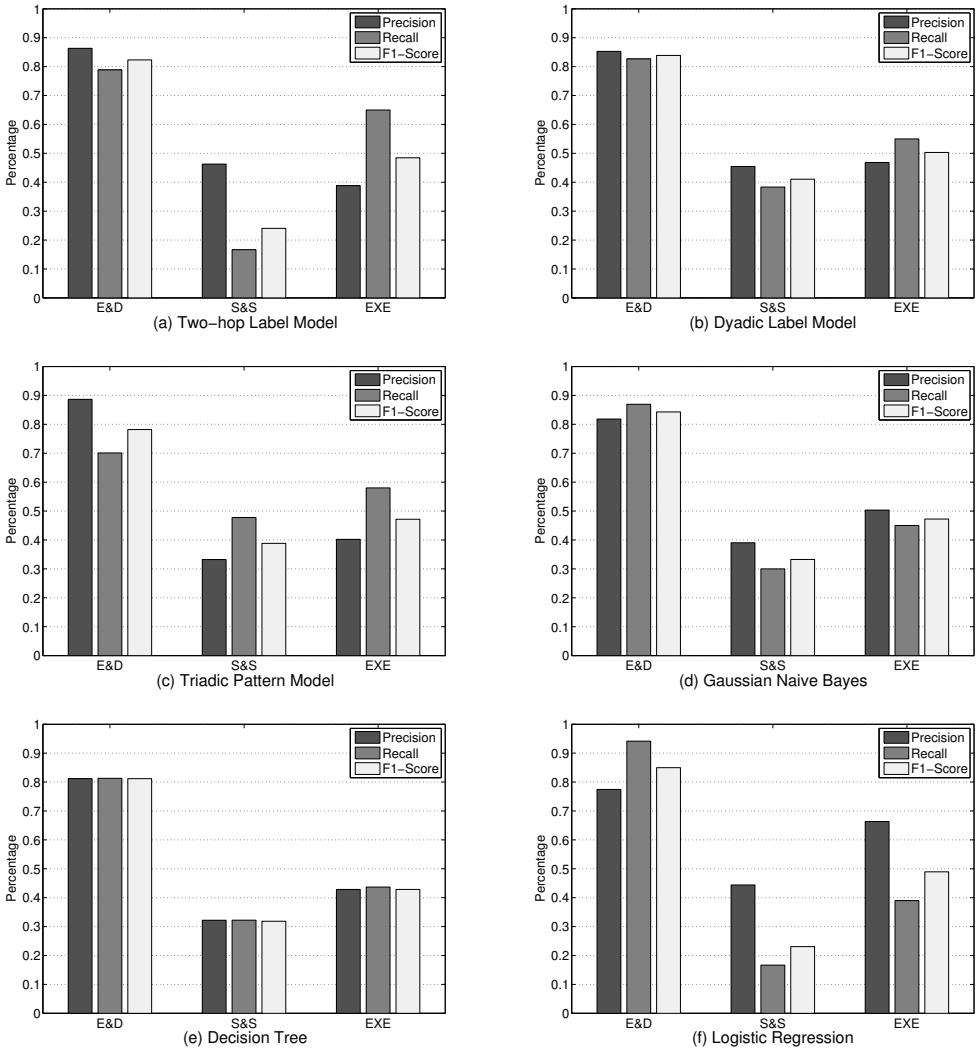



Figure 4.3: Performance on Different Inference Models

following expression.

$$\mathbf{x}_i = [\mathbf{h}_i^T, \mathbf{d}_i^T, \mathrm{LCC}_i, \mathrm{DC}_i, \mathrm{AND}_i]^T \tag{4.8}$$

For the experiments, we implement the feature based model with multiple machine learning algorithms, such as Gaussian Naïve Bayes, Decision Tree and Logistic Regression, to evaluate and compare its performance.

## 4.3 Experiments

In this section, we present the effectiveness of our proposed models by inferring the roles of Google employees in the company. For the evaluation of multi-class classification performance, we use precision (positive predictive value), recall (true positive rate) and f1-score (the harmonic mean of precision and recall).

### 4.3.1 Performance of Naïve Bayes Classifier Models

First, we test the performance of Naïve Bayes Classifier Models based on neighboring relationships. To obtain accurate results, we apply 10-fold cross-validation with the whole data set divided into 10 equal-sized subsets. At each time, one of the 10 subsets is chosen as testing set and the remaining 9 subsets are used for training. The process is repeated for 10 times with the final results calculated as the average of 10 results. The average precision, recall and f1-score of Two-hop Label Model, Dyadic Label Model and Triadic Pattern Model are shown in Fig. 4.3 (a)(b)(c), respectively.

For the overall performance, Dyadic Label Model performs the best out of three Naïve Bayes Classifier Models with f1-score 83.88% (E&D), 41.06% (S&S) and 50.28% (EXE). Compared with Dyadic Label Model, Two-hop Label Model can recall more EXE nodes while Triadic Pattern Model can recall more S&S with the decrement of precision. From the former observation in Section 3, E&D nodes are more likely to establish relations with E&D nodes than S&S and EXE nodes. That is also why the inference performance on E&D nodes is much better than S&S and EXE nodes. Moreover, due to the uneven label distribution, prediction results tend to the label in majority. Therefore, if we want to cover more minority label nodes, Triadic Pattern Model is a good choice.

### 4.3.2 Performance of Feature-based Models

Then, we test the performance of Feature-based Models implemented in three machine learning algorithms, Gaussian Naïve Bayes, Decision Tree and Logistic Regression. Similarly, we also apply 10-fold cross-validation on each machine learning algorithm. The results of these three algorithms are shown in Fig. 4.3 (d)(e)(f), respectively.

From the figure, the overall performance of Gaussian Naïve Bayes is slightly better than Decision Tree and Logistic Regression with f1-score 84.30% (E&D), 33.25% (S&S), 47.23% (EXE). For Logistic Regression, the precision of S&S and EXE nodes

is much higher than the other two algorithms, but the recall rates of these two nodes are very low. Comparing with the former Naïve Bayes Classifier Models, the results of Feature-based Model are slightly worse even with more network characteristic features. That is because these features are not very distinguishable among different roles just as our observations in the Google employee social network. Therefore, in this case, relations weigh more than network characteristics in social role inference.

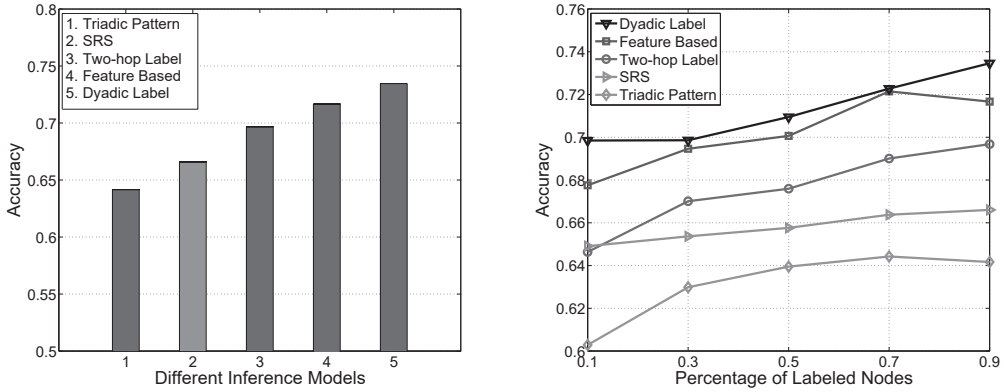



Figure 4.4: Overall Accuracy Comparison

Finally, we compare the overall performance of our models with the SRS Model proposed in [56] which is a probabilistic model based on various network metrics. As shown in Fig. 4.4, the overall accuracy of Dyadic Label Model, Feature-based Model (Gaussian Naïve Bayes) and Node Label Model is higher than the SRS model, while the SRS Model performs better than Triadic Pattern Model. Besides, when taking the percentage of labeled nodes into account, we find that the performance of Node Label Model is better than SRS Model except that with extremely few labeled nodes (10%), SRS Model has a better accuracy. And for the other models, the decrement of the labeled node size also results in a slight decrease of performance.

# Chapter 5

# Privacy-Preserving Online Social Network Data Sharing

## 5.1 Introduction

Nowadays, the third-party applications can access user profiles and relationships with user's authorization, so they can leverage the application-specific social networks and integrate their services with the existing OSNs. OAuth 2.0 [20], a common authorization protocol, has been designed to guarantee the authorization security and simplify the authorization process without sharing users' credentials. The authorization mechanism allows each user to know the resources that the third-party applications require and then to determine whether to grant the resource access permission. However, even if users have a full control of what to disclose, they have little knowledge of whether the third-party applications can exploit the disclosed information to infer their secrets. Meanwhile, the user experience with the OSNs and applications depend on safe self-disclosure. It is critical to ensure that users enjoy the benefits of the services free from the worries about privacy leakage.

A direct way of defending against the inference attack is to pre-process the released network data. There are several common operations to reduce the chance of inference attacks, such as *masking* (removing data), *obfuscation* (adding confusing data), and *generalization* (coarsening data) [50]. In this work, we prefer to mask the secret-related information rather than to add misleading information to maintain the data utility because misleading or fake information may bring users other troubles (e.g., inaccurate news feeds and friend recommendations) in non-anonymized social

networks. According to the surve conducted by Yong et al. [55] on privacy protection strategies on Facebook, the interviewed students were strongly against using fake information as a privacy protective method because of the confusion brought by the fake information. However, the excessive concealment of user attributes will reduce the self-disclosure utility of OSN users, which results in the degradation of user experience and application performance. Thus, we need to mask attributes effectively and efficiently with the consideration of both the self-disclosure utility and privacy concerns.

Therefore, our goal is to design the algorithms for releasing as much social network data as possible while satisfying the privacy guarantees according to different user concerns. To achieve the goal, there are three main challenges. i) Privacy Concerns: Different users may have different privacy concerns. A certain profile attribute may be a secret to one person while it may not be sensitive to another; ii) Attacker's Ability: It is necessary to design a general protection algorithm with regard to attackers' ability. But the background knowledge and capabilities of an attacker are usually unknown; and iii) Privacy and Utility Evaluation: To establish the privacy protection model, we need to quantify, evaluate and make a trade-off between the extent of self-disclosure and privacy leakage.

In this chapter, we first formulate the privacy-preserving online social network data sharing problem as a knapsack-like problem, and then propose two social network data disclosure methods, an efficient privacy-preserving disclosure (EPPD) algorithm and a multi-dimensional knapsack problem (d-KP) simplification based method, respectively. The main contributions are threefold.

- We use the social-attribute network model to describe both the social network data and attacker's knowledge, and propose the self-disclosure rate to quantify the leakage of user secret in the published network regardless of the attacker's knowledge.

- To defend the inference attack, we formulate a novel privacy-preserving social network data sharing problem, which maximizes the user self-disclosure utility with privacy guarantees. The optimization problem also takes different user concerns into account and enables a flexible self-disclosure evaluation in order to satisfy different user demands and scenarios.

- The two proposed social network data sharing methods are designed for different purposes and protection levels. The EPPD algorithm targets the high utility

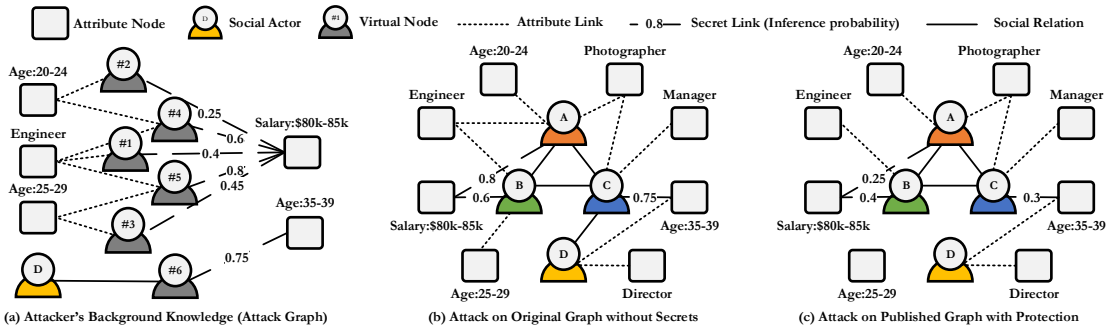



Figure 5.1: An example of the attribute inference attack in OSNs.

satisfying the formal privacy protection constraints, while the d-KP disclosure algorithm greatly reduces the computational complexity of solving the social relation disclosure problem. We use two real-world social networks to evaluate the performance and compare them with the existing work.

## 5.2 Attack Model and Problem Formulation

### 5.2.1 Privacy Inference Attack

In a social-attribute network, the privacy inference attack (sensitive attributes and relations inference) can be regarded as a special form of link prediction [17]. An attacker exploits both the public information from published social networks and the background knowledge to infer user privacy. Since the attacker can adopt a variety of techniques with different background knowledge, it is difficult to take all situations into consideration. However, it is feasible to model an attacker into a knowledge graph [34, 45, 46]. Similarly, we use another social-attribute network graph $G_A$ (attack graph) to describe the background knowledge of an attacker. An attack graph can contain the following information:

1. *Statistical Information.* It can be obtained from official statistics directly or from published data set. A piece of statistical information can be expressed as a conditional probability of owning a secret given several attributes.

2. *Node & Edge Information.* An attack graph can contain node and edge information not contained in the published social networks. It can involve part of the original social network, and additional edges from other social networks as well.

Table 5.1: Additional Notation

| Symbol | Definition |
|---|---|
| $G_A$ | Attack graph |
| $G_P$ | Published social network graph |
| $t_u(s)$ | Indicating whether actor $u$ has secret $s$ |
| $\Phi(u, s, G_P)$ | Self privacy disclosure of actor $u$'s secret $s$ in $G_P$ |
| $p_i$ | Value of the $i$th attribute or social relation |
| $\epsilon$ | Privacy budget |
| $\delta$ | Relaxation variable |
| $\theta$ | Privacy threshold determined by $\epsilon$ and $\delta$ |

Different from the social-attribute network for the original dataset, an attack graph translates each piece of statistical information into a tree-like inference path: *Attributes-Feature-Secret*, where *Feature* node is treated as a virtual social actor with several *Attribute* nodes connecting to it and a weighted edge to a *Secret* node. The weight is the conditional probability Pr(Secret|Attributes). For example, the statement "*People with attributes* $A1$ *and* $A2$ *have a probability of 90% to own secret* $S1$" can be expressed as path *A1,A2-F1-S1* where $F1$ is a virtual node representing social actors with attribute $A1$ and $A2$. Fig. 5.1 gives a toy example of the inference attack and a feasible defending method against it. According to #4 and #5 in the attack graph (a), the attacker infers that A and B in the original graph (b) have a high probability of earning \$80k–85k annually. According to #6, the attacker knows that around 75% friends of D are in the same age group, which implies that C, as D's friend, is probably around 35 to 39 years old. However, attackers can hardly make accurate inferences on processed graph (c) since only 25% of people in the age group of 20–24 and 40% of engineers have the salary of \$80k–85k. Also, for C, after removing the relation between C and D, it is hard to guess C's true age from other attributes. In this example, with sufficient information observed from the social-attribute network, an adversary can easily conduct the inference attack based on the background knowledge. After removing a few attributes and social relations, the probability of inferring the secrets of the targeted OSN user will greatly decrease.

### 5.2.2 Self Privacy Disclosure

Before introducing the formal privacy definition, we first list the additional symbols in Table 5.1. To quantify the privacy disclosure, Martin et al. [41] defined the disclosure risk as the likelihood of the most possible sensitive attribute assignment with respect to the background knowledge for privacy-preserving data sharing. In the case of social networks, the disclosure risk of a binary secret $s$ of the social actor $u$ in a published network graph $G_P$ given a certain background knowledge graph $G_A$ is:

$$\Pr(t_u(s) = 1|G_A, G_P) = \Pr(t_u^A(s) = 1|g(\mathcal{A}_u^P)), \tag{5.1}$$

where the function $\mathcal{A}_u^A = g(\mathcal{A}_u^P)$ maps the attribute set $\mathcal{A}_u^P$ onto $\mathcal{A}_u^A$ in the attack graph $G_A$.

However, since the background knowledge varies from attackers to attackers, we should consider a general privacy measurement without regard to a certain attack graph. Therefore, we introduce the *self privacy disclosure* to evaluate the disclosure rate of a certain social actor $n$'s secret $s$ in the graph $G$. Consider that the attacker graph is a complete anonymized version of the graph $G$. Then, the self privacy disclosure from the perspective of attributes, $\Phi_{\mathcal{A}}$, can be calculated by

$$\Phi_{\mathcal{A}}(u, s, G_P) \triangleq \Pr(t_u(s) = 1|\mathcal{A}_u \cap \mathcal{A}_u^P). \tag{5.2}$$

Also, we can process the social relations in a similar way to attributes. Let $\Phi_{\mathcal{N}}$ be the self privacy disclosure which is the likelihood of owning secret $s$ given the $\mathcal{N}_u$.

$$\Phi_{\mathcal{N}}(u, s, G_P) \triangleq \Pr(t_u(s) = 1|\mathcal{N}_u \cap \mathcal{N}_u^P). \tag{5.3}$$

Similarly to the existing definitions of privacy such as Bayes-optimal privacy [37], indistinguishable privacy [35] and differential privacy [10], we define the threshold for self privacy disclosure as the privacy guarantee. An operation is considered to be privacy-preserving if it the difference between the attacker's prior and posterior beliefs about the sensitive information is small enough, which can be expressed as follows.

$$\Phi(u, s, G_P) \leq \exp(\epsilon) \Pr(t_u(s) = 1) + \delta, \tag{5.4}$$

where $\Pr(t_u(s) = 1)$ is the prior probability of $u$'s secret $s$. Here, we use two param-

eters, $\epsilon$ and $\delta$, to control the privacy protection strength. $\epsilon$ is the privacy budget, a non-negative parameter to define how close the self privacy disclosure rate is to the prior probability. It determines the strict privacy threshold at $\delta = 0$, and therefore, it is usually set to a small number. $\delta$ controls the tolerance of privacy disclosure, and it should be numerically smaller than $1 - \exp(\epsilon) \Pr(t_u(s) = 1)$ since the range of self privacy disclosure is $[0, 1]$. $\delta$ enables a flexible way to relax the privacy constraint which allows to customize different privacy requirements for different situations.

### 5.2.3 Problem Formulation

The goal of our work is to mask part of edges in the social-attribute network so that users can disclose as much valuable personal information as possible with privacy guarantees. The social network data sharing problem is formulated as follows.

Given an original social network $G = \{V_N, V_A, E_N, E_A\}$ and user privacy concern matrix $C \in \mathbb{N}^{|V_N| \times |V_A|}$, obtain a protected social network $G_p = \{V_N, V_A, E'_N, E'_A\}$, where $E'_N \subset E_N$ and $E'_A \subset E_A$ such that the protected social network satisfies privacy requirements with maximized disclosure. In our case, the secrets of a user are regarded to be safe once its disclosure rate is lower than a certain threshold.

To clearly formulate our problem, we introduce some other symbols. The attributes of the node $u$ can be divided into two sets, the secret attributes $\mathcal{S}_u = \{v | v \in \mathcal{A}_u, C_{u,v} = 1\}$ and the public attributes $\mathcal{P}_u = \{v | v \in \mathcal{A}_u, C_{u,v} = 0\}$ respectively, where $\mathcal{A}_u = \mathcal{S}_u \cup \mathcal{P}_u, \mathcal{S}_u \cap \mathcal{P}_u = \emptyset$.

The masking operation starts with all edges involved and then removes one edge after another from the network. Here we process the network in the opposite way: at the beginning, the network includes all the nodes only, both social actors and attributes, and we add edges into it. For each attribute node, we decide whether to show it in the modified network. Then, we introduce an optimization problem as follows.

$$\max_{\mathbf{x}} \quad \sum_{i=1}^{|\mathcal{P}_u|} p_i x_i \tag{5.5}$$

$$\text{s.t.} \quad \Phi_{\mathcal{A}}(u, s_j, \mathbf{x}) \leq \theta_j, \qquad j = 1, \ldots, |\mathcal{S}_u|, \tag{5.6}$$

$$x_i \in \{0, 1\}, \qquad i = 1, \ldots, |\mathcal{P}_u|. \tag{5.7}$$

Each element $x_i$ in vector $\mathbf{x}$ indicates whether to disclose the corresponding public

attribute $a_i$ with utility value $p_i$ which can be defined as the value of the attribute for different purposes (see Section 3.2). $\Phi_{\mathcal{A}}(u, s_j, \mathbf{x})$ is the self privacy disclosure of node $u$'s secret $s_j$ according to vector $\mathbf{x}$, which is defined in (5.2). $\theta_j$ is the privacy protection threshold for secret $s_j$, which is equal to $\exp(\epsilon)\Pr(t_u(s_j) = 1) + \delta_j$. Note that the modified social network here is the subgraph of the original network. The self privacy disclosure is actually determined by selected attributes, and thus, we use $\Phi_{\mathcal{A}}(u, s_j, \mathbf{x})$ instead of $\Phi_{\mathcal{A}}(u, s_j, G_P)$.

As described above, the attribute disclosure problem for each social actor can be solved locally without the involvement of other social actors. However, a social relation involves two actors, and therefore, when we disclose a relation for a social actor, we should take into account the influence of this relation on the other actor. Hereby, we formulate the social relation disclosure problem as follows.

In the directed social networks, we mainly consider the successors of a social actor since users can determine who to follow, but not their followers. Besides, the removal of a directed social relation does not affect its reverse relation. Therefore, we can process the directed social relation disclosure problem in the similar way to the attribute disclosure problem by changing $\mathcal{P}_u$ into $\mathcal{N}_u$ and using the $\Phi_{\mathcal{N}}(u, s_j, G_P)$ as the constraints.

In the undirected social networks, the relation disclosure problem cannot be solved for each social actor, because the removal of one undirected edge will affect two social actors. Therefore, we need to address the relation disclosure problem as a whole, which is formulated as follows.

$$\max_{\mathbf{x}} \quad \sum_{i=1}^{|E_N|} p_i x_i \tag{5.8}$$

$$\text{s.t.} \quad \Phi_{\mathcal{N}}(u_k, s_{k,j}, \mathbf{x}) \leq \theta_{k,j}, \quad j = 1, \ldots, |\mathcal{S}_{u_k}|, k = 1, \ldots, |V_N|, \tag{5.9}$$

$$x_i \in \{0, 1\},\ i = 1, \ldots, |E_N|, \tag{5.10}$$

where $\theta_{k,j} = \exp(\epsilon)\Pr(t_{u_k}(s_{k,j}) = 1) + \delta_{k,j}$ and all protection constraints including all social actors and their secrets are considered together. Compared with the optimization problem formulated in (5.5), the main challenge is the complexity of the undirected social relation disclosure problem. Thus, to address this problem, we specifically propose a d-KP simplification method, which will be introduced in detail in the next section.

## 5.3 Privacy-Preserving Disclosure Algorithms

### 5.3.1 Overview

Our social network publishing problem is a knapsack-like combinatorial optimization problem. For each attribute or relation, we can regard the utility (e.g., self-disclosure) as its "profit" or "value", the contribution to secret disclosure as its "weight", and the privacy protection threshold as the "maximum weight". For the knapsack problem, there exists an exact solution by using dynamic programming (DP) or Branch and Bound [28]. Similarly, we can use DP to obtain a feasible solution to our proposed problem. The maximum value of selected attributes or social relations can be defined as the following general form:

$$p_{\max}(i, C) = \begin{cases} 0, & i \leq 1, i > n, \\ p_{\max}(i-1, C), & i > 1, \exists w > \theta, \\ \max(p_{\max}(i-1, C \cap N_i) + p_i, & \\ \qquad p_{\max}(i-1, C)), & \text{otherwise.} \end{cases} \tag{5.11}$$

where $i$ is the $i$th item (attribute or relation) with value $p_i$ and neighborhood social actor set $N_i$, $C$ is the intersection of selected items's social actor neighborhood sets and "$\exists w > \theta$" means that there is at least one secret's self privacy disclosure rate exceeding the threshold. The pseudo-code for DP algorithm is shown in Algorithm 1.

However, such a DP algorithm does not necessarily achieve the optimal solution, since the "weight" of each item is not fixed and even may become negative with different items selected. Consider the whole procedure of the DP algorithm as a tree. During the recursion, once the current combination (e.g. $\mathbf{x} = [1, 0, 1, 0, 0, \ldots, 0]$) exceeds the threshold, this branch will be cut because all the combinations derived from this branch are regarded to exceed the threshold as well. However, it is possible that there is a combination satisfying the constraints in a subtree due to the items with negative weight. Although the DP algorithm can cope with the negative weight situation in knapsack problems by preprocessing, it is hard to process "weights" in advance for our problem because of their inter-dependency.

Besides, the DP algorithm has an exponential time complexity. It is applicable when the data is of low scale only. However, a social actor may have hundreds and

**Algorithm 1** Dynamic Programming

**Input:** Item value list $\vec{p} = \{p_1, p_2, \ldots, p_n\}$, secret neighborhood set list $\vec{S} = \{S_1, S_2, \ldots, S_m\}$, item neighborhood set list $\vec{N} = \{N_1, N_2, \ldots, N_n\}$, secret threshold list $\theta = \{\theta_1, \theta_2, \ldots, \theta_m\}$
**Output:** Maximum value $p_{max}$

$C \leftarrow V_N, Sel \leftarrow \emptyset$
**for** $i \leftarrow n, n-1, \ldots, 1$ **do**
  **if** MaxValue$(i, C) \neq$ MaxValue$(i-1, C)$ **then**
    $Sel \leftarrow Sel \cup \{i\}, C \leftarrow C \cap N_i$
  **end if**
**end for**
**return** MaxValue$(n, V_N)$, $Sel$
**function** MaxValue$(i, C)$
  **if** i $= 0$ **then return** 0
  **end if**
  **for** $j \leftarrow 1, 2, \ldots, m$ **do**
    $w_j \leftarrow \frac{|C \cap N_i \cap S_j|}{|C \cap N_i|}$
  **end for**
  **if** $w_j <= \theta_j \forall j \in \{1, 2, \ldots, m\}$ **then**
    **return** max(MaxValue$(i-1, C)$,
      MaxValue$(i-1, C \cap N_i) + p_i$)
  **else**
    **return** MaxValue$(i-1, C)$
  **end if**
**end function**

even thousands of social relations, which make the DP algorithm fail to find out a solution within an acceptable time limit.

To design an efficient algorithm for social network data sharing, both protection performance and computational complexity should be taken into account. In our work, we first propose a heuristic method to obtain a feasible solution to the original combinatorial problem. In order to reduce the high complexity brought by the high-dimensional social relations, we then simplify the original social relation disclosure problem into a d-KP problem.

### 5.3.2 An Efficient Privacy-Preserving Disclosure Algorithm

For the knapsack problem and its variants, a great number of heuristic algorithms are proposed to obtain feasible solutions approaching the optimal one with relatively

low time complexity [28]. Among them, the greedy algorithm is the most intuitive with a reasonable performance. To obtain a feasible solution with polynomial time complexity, we propose the EPPD algorithm to solve the knapsack-like problem.

We define the total contribution of the selected items $T_{sel}$ to social actor $n$'s secret $s$ as the self privacy disclosure rate:

$$w_s(T_{sel}) = \Phi(u, s, T_{sel}) \tag{5.12}$$

$$= \frac{|\cap_{t' \in T_{sel}} \mathcal{N}_{t'} \cap \mathcal{N}_s|}{|\cap_{t' \in T_{sel}} \mathcal{N}_{t'}|}. \tag{5.13}$$

Accordingly, the contribution (weight) of a single item $t$ to social actor $u$'s secret $s$ with selected items $T_{sel}$ can be computed as the increment of the total contribution, i.e.,

$$w_{t,s}(T_{sel}) = \Phi(u, s, T_{sel} \cup \{t\}) - \Phi(u, s, T_{sel}). \tag{5.14}$$

The main idea of our algorithm is to compare all edges (attribute links or social relations) and find out the most suitable one at each time. If all constraints are satisfied after adding the selected item, then add it to the result list; otherwise, drop it and repeat the former steps until all edges are visited. Intuitively, an edge with a higher value and a lower contribution to each disclosure rate constraint is preferable. Here, we define the *efficiency* of an edge as its value-to-weight ratio. Since there are usually multiple constraints with different thresholds in our problem, it is not reasonable to add up all the "weights" for different secrets directly. Instead, we need to consider the magnitudes and importance of the constraints and compute the efficiency as the following general form:

$$\rho_i = \frac{p_i}{\sum_{j=1}^{m} b_j w_{i,j}}, \tag{5.15}$$

where $b_j$ is the importance coefficient of the $j$th constraint and $w_{i,j}$ is the total weight (contribution) with item $i$ to secret $j$. We use the total weight instead of the weight of each item defined in (5.14) so as to avoid the negative weight problem. In this way, the efficiency $e$ is always non-negative.

The coefficient $b_j$ can be changed for different purposes to reach a more suitable solution to the given scenario (e.g., a secret with a higher priority can have a smaller $b_j$). In the experiment shown in Section 5.4, we only take the magnitudes of con-

**Algorithm 2** EPPD Scheme

**Input:** Item value list $\vec{p} = \{p_1, p_2, \ldots, p_n\}$, secret neighborhood set list $\vec{S} = \{S_1, S_2, \ldots, S_m\}$, item neighborhood set list $\vec{N} = \{N_1, N_2, \ldots, N_n\}$, and secret threshold list $\theta = \{\theta_1, \theta_2, \ldots, \theta_m\}$

**Output:** Maximum value $p_{\max}$

$C \leftarrow V_N, Sel \leftarrow \emptyset, V_{\max} \leftarrow 0, \vec{l} \leftarrow [1, 2, \ldots, n]$
**while** $\vec{l} \neq \emptyset$ **do**
  $\rho_{\max} \leftarrow -1, s \leftarrow -1, w_{sel} \leftarrow \emptyset$
  **for** each $i \in \vec{l}$ **do**
    **for** $j \leftarrow 1, 2, \ldots, m$ **do**
      $w_j \leftarrow \frac{|C \cap N_i \cap S_j|}{|C \cap N_i|}$
    **end for**
    $\rho \leftarrow \frac{p_i}{\sum_{k=1}^{m} w_j/\theta_j}$
    **if** $\rho > \rho_{\max}$ **then**
      $s \leftarrow i, \rho_{\max} \leftarrow \rho, w_{sel} \leftarrow w$
    **end if**
  **end for**
  **if** $w_j <= \theta_j,\ \forall j \in \{1, 2, \ldots, m\}$ **then**
    $Sel \leftarrow Sel \cup \{s\}, C \leftarrow C \cap N_s, p_{\max} \leftarrow p_{\max} + p_s$
  **end if**
  $\vec{l}.remove(s)$
**end while**
**return** $p_{\max}, Sel$

straints into account and let $b_j = 1/\theta_j$. The pseudo-code is shown in Algorithm 2.

From the code, we can see that different from greedy heuristics for knapsack problems which requires to sort all items according to a certain criteria at the beginning, the proposed algorithm for the edge masking problem needs to find out the most suitable edge in each iteration, because the "weight" of each item changes after each iteration.

The greedy algorithm for the social relation masking problem in undirected networks is slightly different. Since the problem is solved as a whole, there may be hundreds of constraints. But actually, the existence of a certain edge only affects two nodes' constraints. With this property, each time we only need to calculate the related constraints and avoid much redundant computation. Therefore, the efficiency

of an undirected edge $e = (n_1, n_2)$ can be calculated as follows.

$$\rho_e = \frac{p_e}{\sum_{j=1}^{m_1} b_j w_{n_1,j} + \sum_{j=1}^{m_2} b_j w_{n_2,j}}, \tag{5.16}$$

where $m_1$ is the number of social actor $n_1$'s secrets and $m_2$ is that of social actor $n_2$.

**Computational Complexity:** In Algorithm 2, we calculate the efficiency of each item with set intersection operations. Assume that the time complexity of each set intersection operation is $O(|V_N|)$. In the attribute disclosure problem, the time complexity of the EPPD algorithm for **each node actor** is $O(n^2 m \cdot |V_N|)$, where $n$ is the number of attributes and $m$ is the number of secrets. In the practical attribute disclosure problem, the scale of $n$ (the attribute number of a social actor) is usually related to $|V_A|$, which is far less than the scale of $|V_N|$. In the undirected social relation disclosure problem, the time complexity of the EPPD algorithm for **the whole network** is $O(|S| \cdot |V_N| \cdot |E_N|^2)$ where $|S|$ is the total number of all constraints/secrets. When the social network becomes very large, it will cause scalability problem of social relation disclosure.

### 5.3.3 d-KP Simplification

In order to reduce the computation complexity of the undirected social relation disclosure problem, one possible solution is to simplify the original problem. Since the knapsack problem and its variants have been well-studied for decades, we try to transform the original relation disclosure problem into a d-KP problem. The key point is how to fix the weight of each item. We can rewrite the constraints (5.9) and assign a fixed weight to each public attribute. To determine the weight of each relation, we assume that the correlations among relations can be neglected (independent assumption), which is identical to the situation where the adversary only knows the secret distribution in each ego network and the degree of each social actor.

First, we consider the basic form of a single constraint. Given a node $u$, a secret $s$ and the social relation indicator vector $\mathbf{x}$, let $\vec{v} = \{v_1, v_2, \cdots, v_p\}$ denote the neighbor node of $u$ that whose corresponding social relation's indicator $x$ equals 1, and then

we have

$$\Phi_{\mathcal{N}}(u, s, \mathbf{x}) = \Pr(t_u(s) = 1 | v_1, \ldots, v_p) \tag{5.17}$$

$$= \Pr(t_u(s) = 1) \prod_{i=1}^{p} \frac{\Pr(v_i | t_u(s) = 1)}{\Pr(v_i)}, \tag{5.18}$$

where $\Pr(v_i)$ is the probability of connecting to node $v_i$.

If we substitute (5.18) into (5.9) and take the natural logarithm of both sides, we can obtain

$$\sum_{i=1}^{p} \ln \frac{\Pr(v_i | t_u(s) = 1)}{\Pr(v_i)} + \ln \Pr(t_u(s) = 1) \leq \ln \theta. \tag{5.19}$$

Then, for a node $u$, the weight of each social relation $e$ is now fixed. If $e$ is the edge connecting to $u$, the weight is equal to the mutual information between two events, "$u$ is connected to the other node $v$ of $e$" and "$u_k$ has the secrets of $s_{k,j}$" respectively; otherwise, it is equal to 0. Let $I(e; s_{k,j})$ denote the mutual information between social relation $e$ and node $u_k$'s secret $s_{k,j}$, i.e.,

$$I(e; s_{k,j}) = \begin{cases} \ln \frac{\Pr(v, t_{u_k}(s_{k,j})=1)}{\Pr(v) \Pr(t_{u_k}(s_{k,j})=1)} & u \in e \\ 0 & u \notin e \end{cases} \tag{5.20}$$

Note that the "mutual information" defined here is slightly different from that between two **random variables**. The former can be either non-negative or negative while the latter is always non-negative. A positive "mutual information" implies the contribution to inferring the secret, while a negative one implies the misguidance.

Then we obtain the d-KP simplified version of our proposed optimization problem as follows.

$$\max_{\mathbf{x}} \quad \sum_{i=1}^{|E_N|} p_i x_i \tag{5.21}$$

$$\text{s.t.} \quad \sum_{i=1}^{|E_N|} I(e_i; s_{k,j}) x_i \leq \theta'_{k,j},$$

$$j = 1, \ldots, |\mathcal{S}_{u_k}|, k = 1, \ldots, |V_N|, \tag{5.22}$$

$$x_i \in \{0, 1\},\ i = 1, \ldots, |E_N|, \tag{5.23}$$

where $\theta'_{k,j}$ is the new threshold which can be calculated by

$$\theta'_{k,j} = \ln(\exp(\epsilon) + \frac{\delta_{k,j}}{\Pr(t_{u_k}(s_{k,j}) = 1)}). \tag{5.24}$$

The intuition of d-KP transform is to use the information gain as the fixed weight of each relation, which has been widely adopted for feature selection in machine learning. The constraints are to limit the total information gain, which the adversary acquires about the targeted secrets from all disclosed relations, to a small number. Also, we can keep the original privacy constraints and adopt the greedy algorithm for d-KP as the disclosure strategy, selecting the edge $e = (m, n)$ with the minimum weight-price rate ($\rho_e^{-1}$) which is calculated by

$$\rho_e^{-1} = \frac{\sum_{j=1}^{|\mathcal{S}_m|} I(e; s_{m,j}) + \sum_{j=1}^{|\mathcal{S}_n|} I(e; s_{n,j})}{p_e}. \tag{5.25}$$

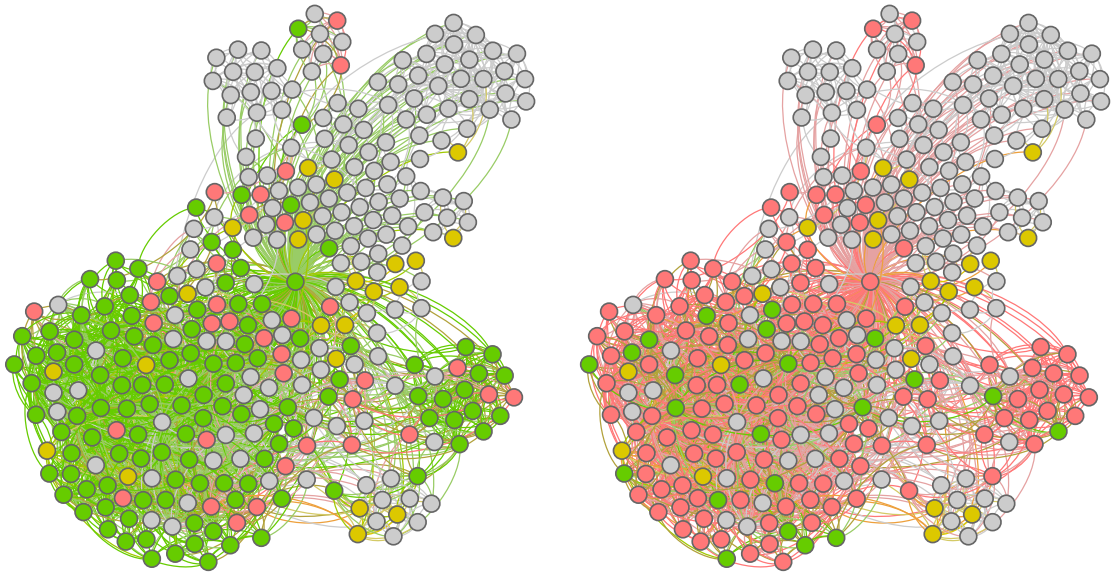
The reason for using $\rho_e^{-1}$ instead of $\rho_e$ is that the information gain can be non-positive in the d-KP simplified problem.

**Computational Complexity:** The common methods to solve d-KP can be applied to this problem. For our experiments, we use the greedy heuristics to solve the transformed d-KP whose time complexity is $O(|S| \cdot |E_N|)$ [28] with all items sorted in advance. Comparing to the EPPD algorithm, the d-KP disclosure algorithm computes the weights only once and does not need to compute them again during the iterations, which greatly decreases the time complexity.

On one hand, the d-KP disclosure algorithm naturally counters the inference attacks by machine learning. On the other hand, the fixed weight greatly reduces the time complexity with certain protection performance loss compared to directly solving the original relation disclosure problem. Therefore, the d-KP disclosure algorithm is suitable for strict privacy protection which requires a fast response.

## 5.4 Experiments

To evaluate the performance of our proposed social network publishing methods, we conduct several experiments on real-world social network datasets with different utility settings and a variety of inference attack algorithms.



(a) Before using EPPD (b) After using EPPD

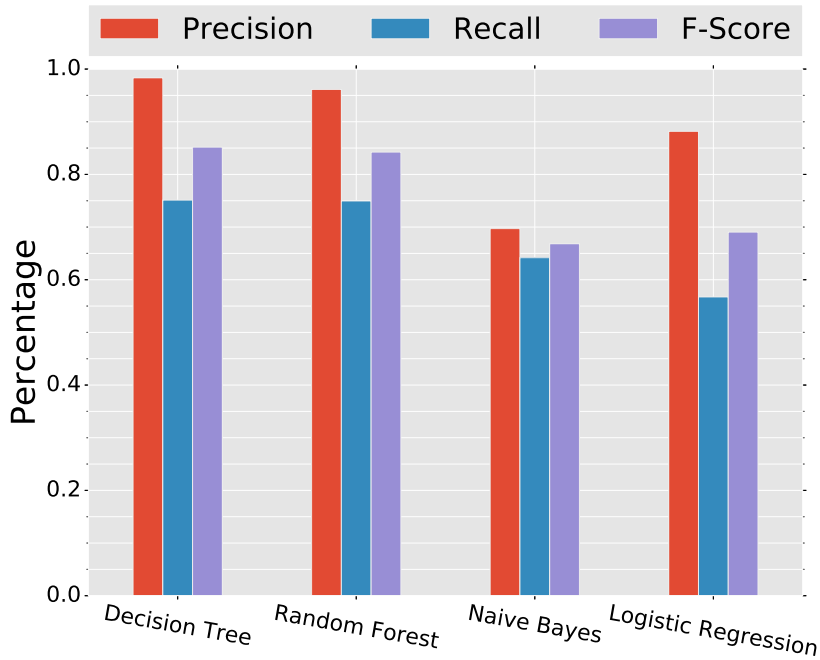
Figure 5.2: Results for inferring the social actors who attended the school 50 in a Facebook Ego Network before and after using the EPPD Algorithm for attribute disclosure. Green: True Positive, Grey: True Negative, Yellow: False Positive, and Red: False Negative.

### 5.4.1 Methodology

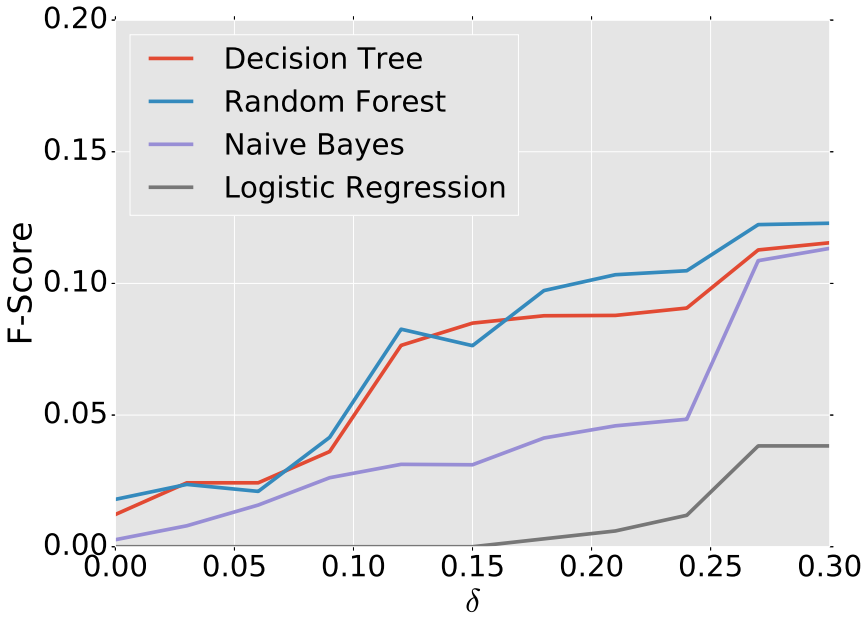
#### Datasets

We apply our algorithms in 2 real-world social network datasets published by Stanford Network Analysis Project (SNAP) [30]. One is the Facebook dataset, including 4,039 undirected social actors and 88,234 social relations, where user profiles consist of 11 categories (i.e., *gender*, *birthday*, *location*, *hometown*, *work*, *education*, etc.). For some categories, there are several sub-categories (e.g., *employer* and *start date* belong to *work* category). The other is the Google+ dataset, including 17,258 social actors, 1,344,555 directed social relations, where user profiles consist of 5 categories (i.e., *gender*, *institution*, *job title*, *location*, and *university*). Note that attributes in the same category are not necessarily mutually exclusive, as they may coexist in the same user profile (e.g., a user can speak two or more languages).

In the experiment, we select several attributes as the secrets with a moderate sample size (the attributes connected with around 5–10% of social nodes in the datasets) from different categories so that the inference attack can be successfully launched on the original data set with sufficient positive training data. The detailed secrets




(a) Performance of local classifiers before using EPPD




(b) F-Scores vs $\delta$ of local classifiers after using EPPD

Figure 5.3: Inference attacks via profile attributes on School 538

settings are shown in the Table 5.2.

Table 5.2: Secret Settings

| Dataset | Attribute name | Category | # of actors |
|---|---|---|---|
| Facebook | School 538 | School | 631 |
| | Birth year 5 | Birthday | 372 |
| | Location 84 | Hometown | 365 |
| | Concentration 14 | Concentration | 368 |
| Google+ | Google Inc. | Institution | 1,123 |
| | Manager | Job Title | 975 |
| | New York | Location | 976 |

**Disclosure Methods for Comparison**

In order to show the effectiveness of our proposed methods, we introduce some commonly used and benchmark methods for comparison. When we compare the utility metrics, all the algorithms are supposed to use the same privacy constraints introduced in Section 5.2.2. To determine $\epsilon$, the self privacy disclosure for a binary attribute connected with 10% social actors is supposed not to be larger than 0.5 (coin flipping) at $\delta = 0$, where $\epsilon$ should be smaller than $\ln(0.5/0.1) \approx 1.6$. To be specific, all $\epsilon$'s for different secrets are uniformly set to a moderate value 0.5, and we mainly study the relationship between relaxation variable $\delta$ and different metrics. Furthermore, the range of $\delta$ is from 0 to 0.3, since 0.3 is already quite high (loose) according to the sample size of the selected secrets.

- *Random Mask (RND)*: Randomly remove attribute or social relation until all of the protection constraints are satisfied. In our experiments, the results of random masking are the average of 100 runs.

- *Naive Bayes Mask (NB)*: Heatherly et al. [23] proposed a masking algorithm based on Naive Bayes classifier to defend against the inference attacks. The basic idea is to remove the attributes and social relations with higher likelihood.

- *d-KP*: In our work, the d-KP algorithm is mainly proposed to solve the undirected social relation disclosure problem. However, it can be also applied in the other disclosure problems with evaluating the information gain of the attributes and social relations as the disclosure criteria.

#### Algorithms for Inference Attacks

To test the protection effects, we use a variety of classification algorithms to conduct inference attack via either public attributes (local classifier) or social relations (relational classifier). The local classifiers include Decision Tree, Random Forest, Gaussian Naive Bayes and Logistic Regression with the default parameter settings in the Scikit-learn library[1]. The relational classifiers include class-distribution relational neighbor (cdrn), weighted-vote relational neighbor (wvrn) and network-only link-based classification (nolb) implemented in Netkit-SRL library[2] , which were proposed by S. A. Macskassy et al. [38]

To launch the inference attack, we have the following experiment settings. For local classifiers, the adversary is assumed to have the whole dataset as its training set, and then conduct the inference attack on the published dataset with the trained model. For relational classifiers, the adversary is assumed to have the half of the dataset and published network as its training set, and then conduct the inference attack on the remaining part of the dataset with the trained model. The different experiment settings are due to the different design principles. The relational classifiers rely on the identity of social actors while the local ones do not. It makes no sense for the relational classifier to train on the whole data set with all actors' secrets already known.

### 5.4.2 Attribute Disclosure Results

#### Protection Performance

The protection performance of our proposed disclosure algorithm depends on the self privacy disclosure constraint, which is defined in Section 5.2.3. As long as the self-privacy disclosure rate is smaller than the constraint threshold, we regard the published social network as a privacy-preserving one. In this part, we mainly study the protection performance of our EPPD algorithm under different $\delta$, because the self privacy disclosure rate of applying EPPD is closer to the constraint threshold compared to the other algorithms.

We first show an example of the inference attack based on the Decision Tree classifier on a small Facebook ego network with 348 users in Fig. 5.2, where the targeted

[1] `http://scikit-learn.org/stable/`

[2] `http://netkit-srl.sourceforge.net/`

secret is the education attribute School 50 (with 154 users having this attribute). Before we apply the EPPD algorithm, the adversary successfully infers 119 of 154 users with the secret (Precision: 83.80%, Recall: 77.27% and F-Score: 80.41%). After applying the EPPD algorithm ($\delta = 0.3$), the adversary fails to find out most positive instances with only 23 inferred correctly (Precision: 14.94%, Recall: 16.20% and F-Score: 15.54%). The performance of the inference attack decreases significantly, which means that the EPPD algorithm is able to defend against the inference attack effectively. For the experiment on the large dataset, we mainly study the F-Score of the classification results because it takes both precision and recall into account.

Fig. 5.3 shows the experiment on inferring the secret School 538 in the Facebook dataset. As shown in Fig. 5.3a, the original F-Score of 4 classifiers are 85.17% (Decision Tree), 84.24% (Random Forest), 66.83% (Naive Bayes), 69.05% (Logistic Regression) respectively, where random guess is around 15.62%. However, after applying the EPPD algorithm, the F-Scores of these four local classifiers are no larger than 15% even with a very loose $\delta = 0.3$. With critical attributes concealed, the published user information will mislead the local classifier to make an inaccurate prediction. Therefore, the local classifiers can hardly work on the processed dataset. When the privacy constraints are strict $\delta = 0$, almost every social actor with the targeted secret is well protected with all F-Scores around 1% only.

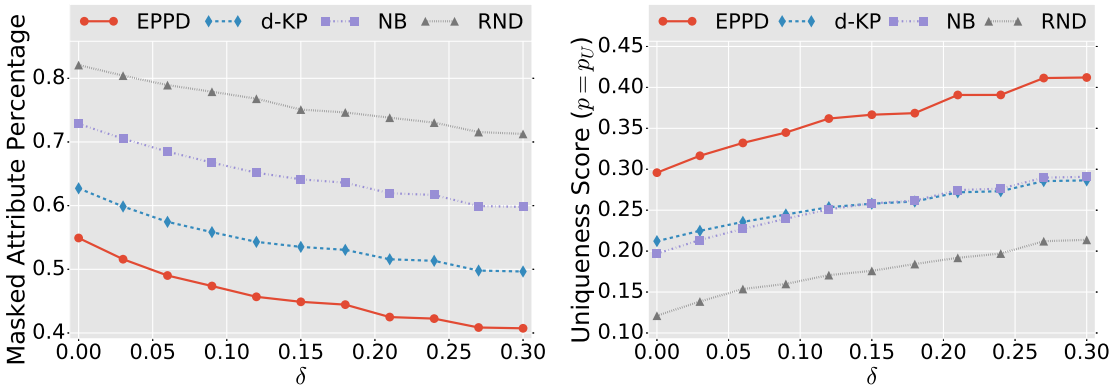
### Utility Comparison

To evaluate the performance of attribute disclosure, we mainly compare the utility scores and the percentage of masked attributes under different social network data sharing algorithms. In our experiment, we adopt the normalized value of each utility score $p$ (e.g., uniqueness score and commonness score) which is calculated as follows.

$$\mathcal{U} = \frac{\sum_{u \in V_N^*} \sum_{i=1}^{|\mathcal{P}_u|} p_i x_i}{\sum_{u \in V_N^*} \sum_{i=1}^{|\mathcal{P}_u|} p_i}, \tag{5.26}$$

where $V_N^* = \{u | u \in V_N, \mathcal{S}_u \neq \emptyset\}$ is the set of the affected social actors having privacy concerns. For an algorithm, a higher utility score indicates that more valuable non-sensitive attributes are shared.

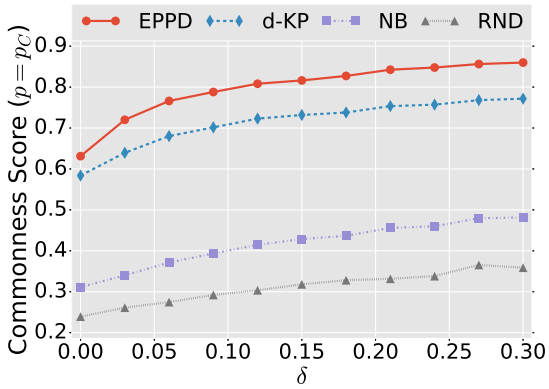
Fig. 5.4a and 5.4d show the attribute masking results of experiments on the Facebook and Google+ datasets. For all the four algorithms, the percentage of the masked attributes decreases with the increment of $\delta$. According to the results in



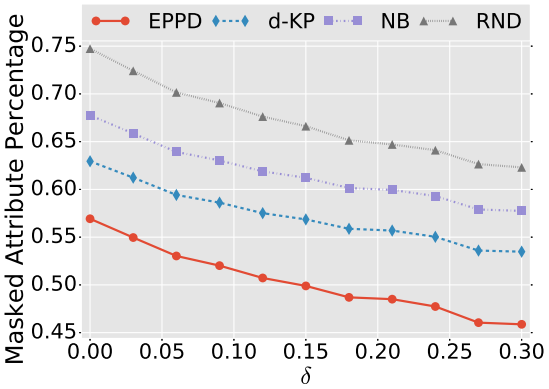

(a) Facebook: Masked Attributes - $\delta$



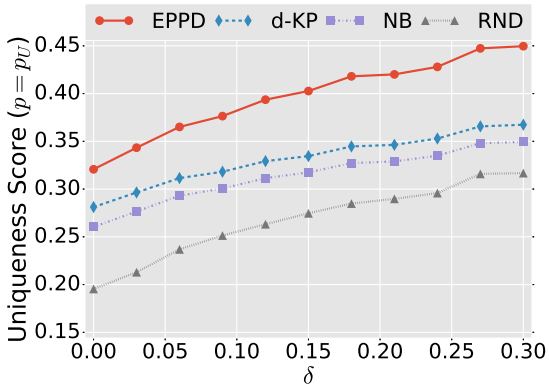

(b) Facebook: Uniqueness Score - $\delta$



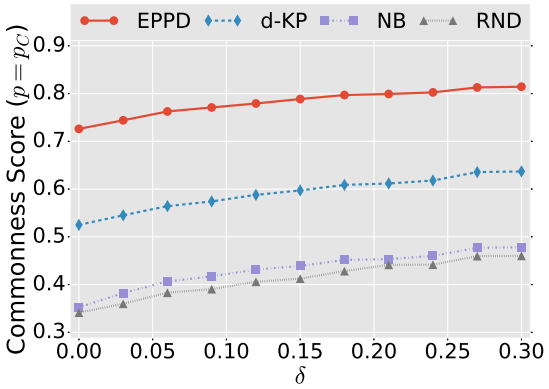

(c) Facebook: Commonness Score - $\delta$




(d) Google+: Masked Attributes - $\delta$




(e) Google+: Uniqueness Score - $\delta$




(f) Google+: Commonness Score - $\delta$

Figure 5.4: Profile attribute utility in Facebook and Google+ data sets

the Facebook dataset, the EPPD algorithm always has the best performance, where only 40.74% of the public attributes are masked at $\delta = 0.3$, while the other three algorithms need to mask 49.66% (d-KP), 59.79% (NB) and 71.27% (RND) of the public attributes. Even under the strict protection constraints at $\delta = 0$, the EPPD

algorithm only masks 55% of the public attributes. Besides, the d-KP algorithm is also better than the Naive Bayes Masking algorithm with fewer attributes masked. Similar results are also observed from the experiment on the Google+ dataset. Combining with results of protection performance experiments, the EPPD algorithm can realize a good protection effect by removing as few critical attributes as possible.

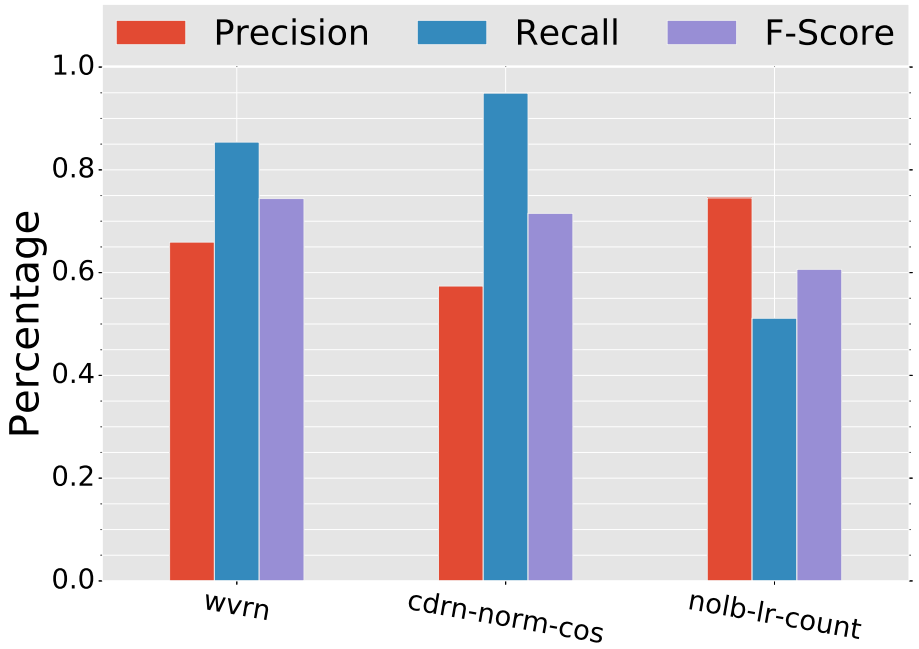
For the utility score experiments, we try the two different utility settings, namely the uniqueness score and the commonness score introduced in Section 3.2. Generally, the utility scores always increases with the increment of $\delta$ since a loose privacy constraint allows more information published. Fig. 5.4b and 5.4e compare the uniqueness scores of 4 algorithms in the Facebook and Google+ datasets. Intuitively, an attribute with a higher uniqueness score may imply more information about whether a social actor has a certain secret, which conflicts the idea of making a social actor's profile indistinguishable. Therefore, all the four algorithms have low uniqueness scores. Besides, the d-KP, an algorithm based on Information Gain, has a close performance to the Naive Bayes Masking in the Facebook dataset while it has a slightly higher score in the Google+ dataset. However, the EPPD algorithm has the best performance among the four algorithms with around 40-50% performance increase over the Naive Bayes Masking.

In contrast with the uniqueness score, the commonness score weighs more on the common attributes. As shown in Fig. 5.4c and 5.4f, both EPPD and d-KP algorithms have significantly higher scores than Naive Bayes Masking. In the Google+ dataset, the EPPD algorithm performs much better than the other three algorithms at any $\delta$ while the Naive Bayes Masking algorithm is just slightly better than Random Masking since it does not set the utility score as its objective. In summary, for the attribute disclosure problem, the EPPD algorithm is more desirable since it has a relatively higher utility score and mask fewer attributes with an acceptable time complexity.

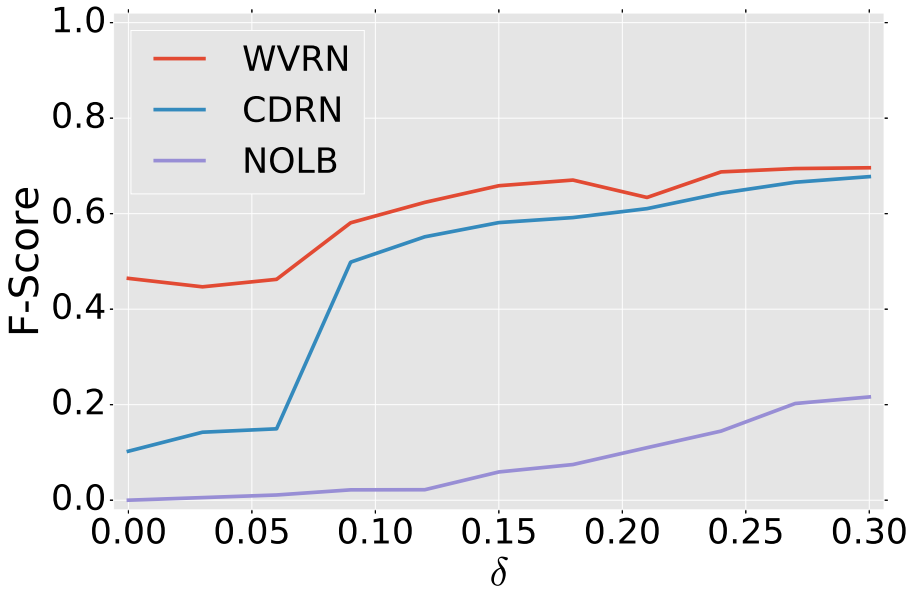
### 5.4.3 Social Relation Masking Results

#### Protection Performance

Similar to the attribute masking experiments, we first study the protection performance of the EPPD algorithm against three different relational classifiers, WVRN, CDRN and NOLB respectively. Although they all focus on the ego network of the targeted node, they have different classification principles which lead to different performances, where WVRN first assigns weights to the neighboring nodes then to get



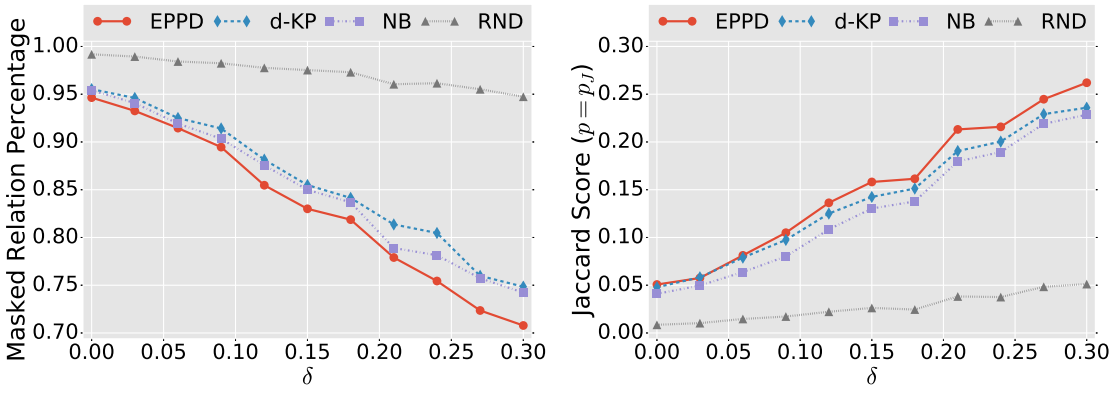

(a) Performance of relational classifiers before using EPPD




(b) F-Scores vs $\delta$ of relational classifiers after using EPPD

Figure 5.5: Inference attacks via social relations on School 538

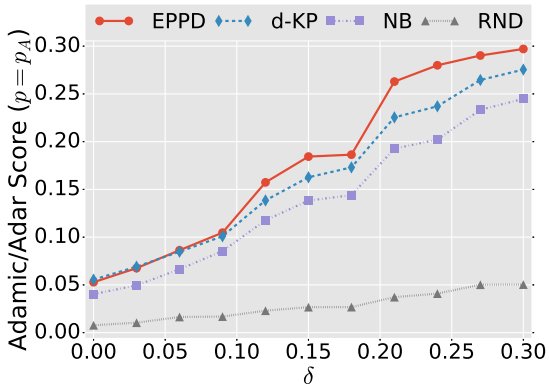
the weighted average probability of owning the secret, CDRN is based on the secret distribution in the ego network and NOLB uses logistic regression with labels of the neighboring nodes as feature vectors.



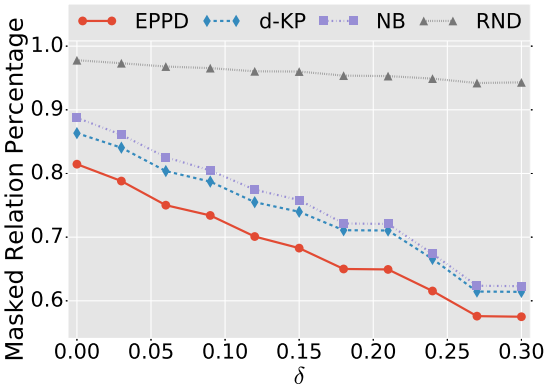

(a) Facebook: Masked Relations - $\delta$



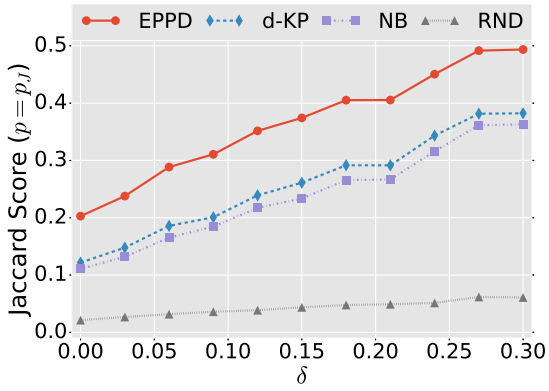

(b) Facebook: Jaccard - $\delta$



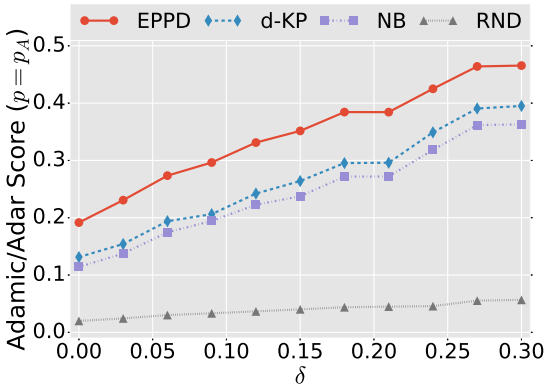

(c) Facebook: Adamic/Adar - $\delta$




(d) Google+: Masked Relations - $\delta$




(e) Google+: Jaccard - $\delta$




(f) Google+: Adamic/Adar - $\delta$

Figure 5.6: Social relation utility in Facebook and Google+ data sets

Fig. 5.5a shows the inference results of the three relational classifiers on School 538 before applying the EPPD algorithm. From the figure, we can see that WVRN has a slightly better performance with a higher F-Score 74.41%, compared with CDRN (71.53%) and NOLB (65.93%). CDRN features a high recall score 94.92% while

NOLB has a relatively high precision score 74.54%. Fig. 5.5b shows the change of F-Scores of the three relational classifiers with the increment of $\delta$. The performance of NOLB decreases significantly at any $\delta$. However, unlike the local classifiers, the F-Scores of WVRN and CDRN do not decrease dramatically at a loose $\delta = 0.3$ (WVRN: 69.61% and CDRN: 67.75%), and they are still even better than that of NOLB attacking on the original dataset. When $\delta \leq 0.06$, the F-Scores of the two classifiers are smaller than 0.5, making them ineffective in the inference attack.

The protection experiment results imply that a loose privacy constraint for social relation masking is still likely to expose the users' secrets to the threat of inference attacks by means of relational classifiers. Similar results were also observed in [23] where the removal of a number of social relations does not lead to a significant performance decrease for the relational classifiers. Thus, it is recommended to set a small value for the privacy threshold (both $\delta$ and $\epsilon$) to protect against social relation based inference attacks.

**Utility Comparison**

As discussed in Section 5.2.3, the directed and undirected social relation disclosure problems are quite different from each other. Here, we conduct both experiments with the Facebook dataset (undirected) and the Google dataset (directed). Besides the percentage of masked relations, we also use Jaccard coefficient and Adamic/Adar score introduced in Section 3.2 for performance evaluation. We also calculate the normalized value of each utility score $p$ as follows.

$$\mathcal{U} = \frac{\sum_{i=1}^{|E_N^*|} p_i x_i}{\sum_{i=1}^{|E_N^*|} p_i}, \tag{5.27}$$

where $E_N^* = \{e | e = (u, v) \in E_N, \mathcal{S}_u \neq \emptyset \text{ or } \mathcal{S}_v \neq \emptyset\}$ is the set of the affected edges with the social actors having privacy concerns.

Fig. 5.6a and 5.6d show the masked relation percentage the processed networks under different disclosure algorithms. Compared with the attribute masking, the social relation masking needs to mask much more edges, especially for the undirected one. As shown in the figure, when $\delta = 0$, nearly 95% affected social relations in the Facebook network need to be removed to satisfy the privacy guarantee while in the Google+ network, the EPPD algorithm can only retain about 18.54% affected social relations (d-KP: 13.67% and NB: 11.25%). Although the EPPD algorithm can

keep around 30% social relations at a loose $\delta = 0.3$, the performance of the inference attack does not decrease significantly. That is to say, to protect the privacy effectively, masking a great number of relations is essential, which leads to a limited operation space for privacy-preserving disclosure algorithms. For this reason, the performances of the three disclosure algorithms look similar. Nevertheless, the EPPD algorithm still has a slightly better performance over d-KP and NB.

Both Jaccard coefficient and Adamic/Adar score assign a weight to each social relation according to its importance based on the similarity between the two connected nodes. However, the former describes the similarity by the common friends (structure similarity) while the latter emphasizes the similarity between the two nodes' profiles (profile similarity). Fig. 5.6b and 5.6e illustrate the Jaccard results in both directed and undirected social networks. In the undirected Facebook network, the performances of EPPD and d-KP are very close and still slightly better than NB at a strict $\delta < 0.06$. And in the directed Google+ network, the EPPD algorithm has higher utility scores compared with the other two algorithms. Similar results are also observed in the Adamic/Adar experiments as shown in Fig. 5.6c and 5.6f. The utility performance of the EPPD algorithm is always good since it is designed to keep the important relations as many as possible. However, in the undirected social relation masking experiments, the d-KP algorithm, which simplifies the original disclosure problem and reduces the computational complexity, has a very similar performance with the EPPD algorithm at a strict $\delta$. In this case, we can use the d-KP algorithm to replace the EPPD algorithm for a faster response and fairly good results.

# Chapter 6

# Privacy Protection with OAuth 2.0 protocol

## 6.1 Introduction

OAuth 2.0 is an industry-standardized authorization protocol, which has been widely used in many social networking services, such as Facebook, Google+, LinkedIn, etc. It enables users to grant the third-party service providers (clients) access to the information without providing their secret credentials. Due to its convenience and security features, OAuth 2.0 is now supported by an increasing number of web and mobile client applications.

However, OAuth 2.0 mainly focuses on the security issues in the authorization process. Although users can control the social network data shared with the third party services, there still exist privacy risks due to the inference attack. Attackers can infer their secrets by exploiting the correlations among private and public information with background knowledge, even if users have hidden their secrets. To defend against the inference attack and enhance the privacy protection under OAuth 2.0 protocol, we design and implement a novel privacy-preserving data sharing framework, RANPriv-OAuth2, on the host server side.

RANPriv-OAuth2 aims to help users determine the data sharing strategy which maximizes the safe self disclosure of the users with the privacy guarantee. Taking the advantage of hosting massive data in the host server, the host social network service provider can easily find out secret-related information which might help the attackers launch the inference attack. By masking this kind of information, the attackers can

hardly make accurate inference on user secrets. Under certain privacy constraints, RANPriv-OAuth2 tries to maximize the social network data to be shared with the third-party services (minimize the masking information), which is related to user experience with these services. The masking algorithms used in RANPriv-OAuth2 are the EPPD algorithm and the d-KP algorithm proposed in Chapter 5 for attribute masking and social relation masking respectively. In addition, RANPriv-OAuth2 features the flexible and customized privacy settings. Users can set the sensitiveness on their secrets and allocate the trust level on the third-party services. Both trust and sensitiveness determine the strictness of privacy protection (privacy threshold).

In the demonstration, we implement RANPriv-OAuth2 on a demo host social network service server which uses real-world Facebook dataset. The demo shows the interaction among the user, the host social network service and the third-party service, and illustrate the whole process from the aspect of user device (e.g., Android phone).

## 6.2 System Design

### 6.2.1 Protocol Description

As mention in the previous section, RANPriv-OAuth2 is based on the OAuth 2.0 protocol, and it does not change the authorization process including the grant and verification of authorization code and access token. Instead, RANPriv-OAuth2 enhances the privacy protection against the inference attack by changing the prior and posterior process of the authorization. The only thing that needs to be indicated in the authorization part is the users' trust levels on the requester client. In this part, we will introduce the protocol flow of RANPriv-OAuth2 as shown in Figure 6.1. The followings are the main steps of the protocol flow.

(a) The authorization process starts from the interaction between the user (resource owner) and the third-party client. After the user request for the service from the client, the client finds that it requires the access to the user's information in the host social network service.

(b) Since the authorization process needs the involvement of the host social network service, the client is redirected to the authorization page with the URL containing the client information (e.g., client id, scopes, and redirect URIs)

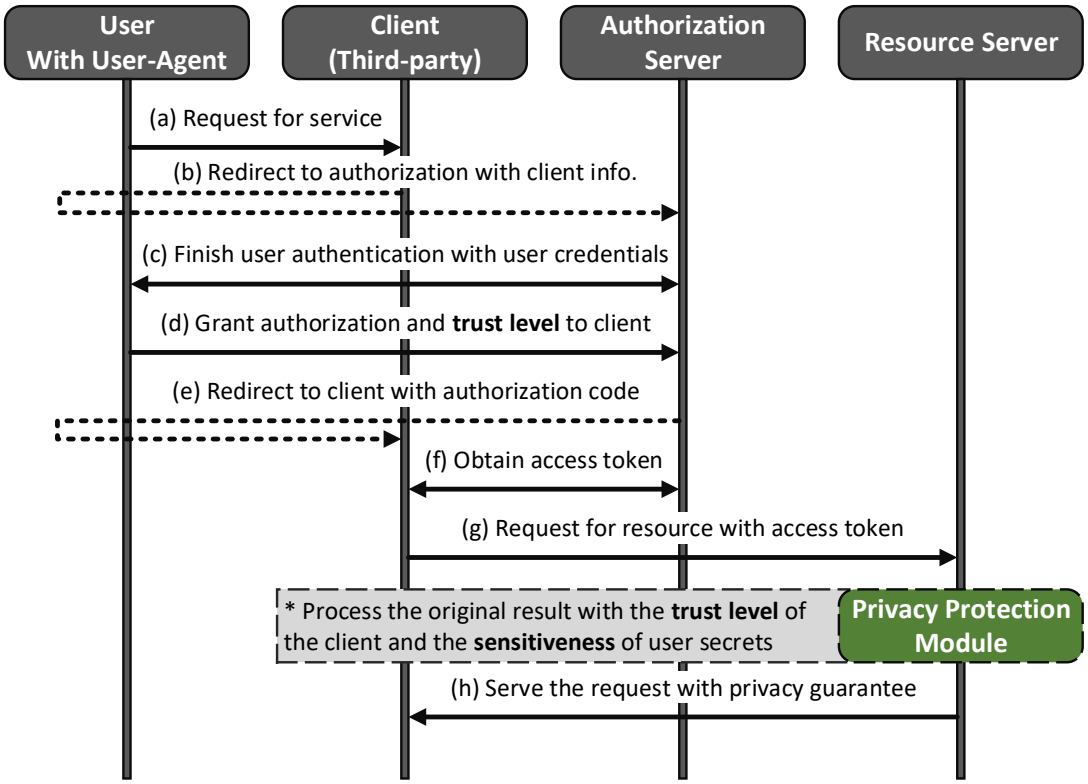



Figure 6.1: Changes in OAuth 2.0 Protocol

(c) Before granting the authorization, the user needs to finish the authentication process by verifying its identity. The common method is to login the host social networking services with a password.

(d) After authentication, the user is asked whether to grant the authorization or not. If so, the user needs to provide the trust level for the third-party client. Otherwise, the authorization process is canceled, and the client will not be allowed to access the user's information.

(e) The client is redirected to the client page with the authorization code (or other authorization grant types specified by the OAuth 2.0 protocol).

(f) The client uses the authorization code granted by the user to obtain the access token from the authorization server. In this step, the authorization server needs to verify the authorization code. After that, the client can request for the resource from the host social network services directly without the involvement of the user before the expiry time.

(g) The client requests for the protected resource of the user and provides the access token for authentication. It is also required to verify the scopes and redirect URIs to ensure the security. If any of the provided information does not match the record in the host social network service, the request will be regarded as an invalid one.

(h) The resource is not directly provided to the client. The privacy protection module on the resource server will process the original result according the trust level of the client and sensitiveness of the secrets set by the user. After processing, the request will be served with privacy guarantee. The whole process is finalized.

### 6.2.2 System Architecture

Since both the authorization server and the resource server belong to the host social network service, we put them together in this paper. Figure 6.2 shows the basic workflow of RANPriv-OAuth2 implemented in the host server, which includes three sub processes marked by different lines. For simplicity, we uses the authorization code grant as the grant type in the OAuth 2.0 protocol.

The dashed line is the privacy setting process. Users first need to set their secrets in the profile. Different from the typical privacy setting in the popular social networking sites, users are asked to allocate the privacy levels instead of just setting as “only me”, “friends” or “public” (the privacy setting in Facebook). The privacy settings are stored in the User Database.

The dotted line is the authorization granting process. It is similar to the original OAuth 2.0 authorization code granting process. However, users need to allocate the trust levels for the requesting clients when granting the authorization. After that, the client will be granted an access token for access to the user information on the host server.

The solid line is the protected resource access process. The clients send the request for the protected resources of users with the access token. After validating the token, the masking module is invoked to intercept the data query and process the original results by masking the secret-related information according to the privacy threshold obtained from the threshold calculator. Finally, the clients will obtain the processed results. The whole process ends.

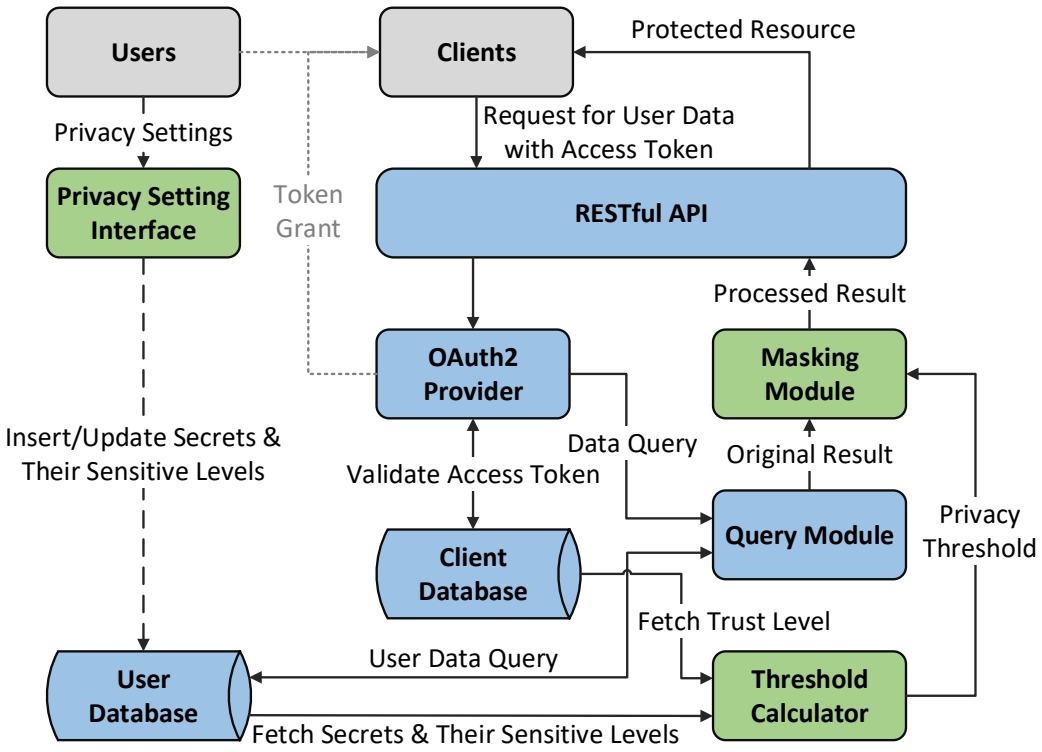



Figure 6.2: System architecture of RANPriv-OAuth2

### 6.2.3 Module Description

In Figure 6.2, The blue blocks are the existing modules in the host server, while the green ones are the new modules for privacy protection: the privacy setting interface, the threshold calculator and the masking module.

**Changes in the existing modules:** To be compatible with the RANPriv-OAuth2, it is required to add a new field in both **Grant** and **Token** tables (maintained in Client Database) to record the trust levels of the client services. A higher trust level indicates that a user can provide more information to the client. When users grant the authorization to the client services, the OAuth2 Provider will ask for the trust levels from users, bind them to the related grant objects and response with the authorization code. The client services use the authorization code to obtain the corresponding access tokens from the host servers where the trust levels are invisible to them.

**Privacy setting interface:** Most social networking sites allow users to customize their own privacy settings by controlling the visibility. In RANPriv-OAuth2, privacy setting interface enables users to determine their secrets (not open to pub-

lic) and assign the sensitiveness levels. A higher sensitiveness level indicates a more strict protection. After users submit the customized privacy settings, privacy setting interface will insert or update the secret records in User Database.

**Threshold calculator:** Based on both trust levels (clients) and sensitiveness levels (secrets), the privacy protection requirements can be calculated through threshold calculator. When the self secret disclosure rate is lower than the privacy threshold, the shared information can be regarded privacy-preserving. In RANPriv-OAuth2, the sensitiveness level determines the privacy budget $\epsilon$ and the trust level determines the relaxation variable $\delta$. The privacy threshold $\theta$ is calculated by $\theta = \exp(\epsilon)\Pr(s) + \delta$, where $\Pr(s)$ is the prior probability of having secret $s$ among all users.

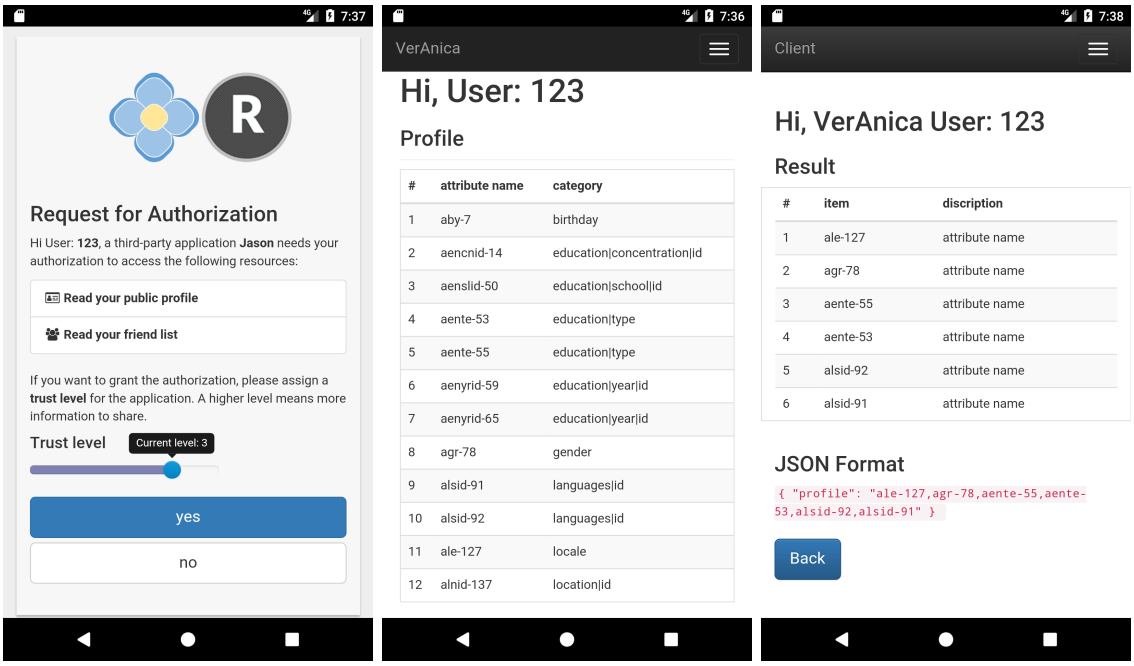
**Masking module:** Once the clients use the access tokens to read users' protected resources, the masking module will intercept the query operations and mask the secret-related information to make sure that the processed results are insufficient for the clients to infer the secrets. To make the trade-off between low latency (less processing time) and high utility (more information to share), RANPriv-OAuth2 adopts two kinds of privacy-preserving disclosure algorithms for attribute masking (profile) and social relation masking (friend/following list), which are EPPD algorithm and d-KP algorithm respectively. Note that the framework can also corporate with other masking algorithms by changing the algorithms in the masking module. Considering the clients may require the same resources for several times, it is also possible to cache the processed results for the low latency.

## 6.3 Demonstration

For the demonstration, we set up a host server running the host social networking service with SNAP Facebook data set [30] as its data source and a client server requesting user information in the host server. Both host and client web services are implemented in Python 3.6 with Flask microframework[1] on macOS Sierra. We use the Flask-OAuthlib[2] to implement the OAuth2 authorization provider, and embed RANPriv-OAuth2 on the host server. The user device is with Android operating system. The demo illustrates the authorization process, including user's granting authorization and client's fetching user data on the host server, from the aspect of the user device. Note that the data used in the demonstration has been anonymized

[1] http://flask.pocoo.org/

[2] https://flask-oauthlib.readthedocs.io/en/latest/




(a) Authorization (b) Original Results (c) Processed Results

Figure 6.3: RANPriv-OAuth2 DEMO on Android Phones

for the privacy concern, and both the trust level and the sensitiveness level are set to 5 levels, ranging from 0 to 4.

Figure 6.3 shows some snapshots on the user side. In Figure 6.3a, the authorization page contains the basic client information and requested user data, and allows users to set the trust level for the client. Figure 6.3b is the original user profile of User 123 stored in the host server. Due to the space limitation, we do not show the privacy setting page in Figure 6.3, and for our case, User 123 sets the education information (school "aenslid-50" and concentration "aencnid-14") as secrets. When the client uses the access token to fetch User 123's profile, it receives the user data as shown in Figure 6.3c where some attributes, besides the two secrets, in the profile are masked for the privacy protection.

# Chapter 7

# Conclusion and Future Work

## 7.1 Conclusion

In the thesis, we have introduced the inference attack and the possible methods to defend against it. For the inference attack model part, we have focused on how to profile users in partially labeled online social networks with relational and network characteristic features. We studied several network features in our Google+ data set and proposed 2 kinds of attribute inference models. To evaluate the effectiveness of our models, we have conducted several experiments on inferring social roles of Google employees in our data set. From the experiment results, the Dyadic Label Model performs the best among all the proposed models. Besides, the results have also shown that social relations including both dyadic and triadic relations play a more important role in attribute inference than network characteristics do.

For the countermeasures part, we investigated the online social network data sharing with defense against the inference attack and formulated the optimization problem which maximizes the utility with privacy guarantee and user privacy concerns. In the paper, we have proposed two different methods to address the problem. One is the EPPD algorithm which is based on the greedy heuristics to directly obtain a feasible and good solution to the original problem, and the other is the d-KP approximation algorithm which transforms and simplifies the undirected social relation disclosure problem into a d-KP problem. Extensive experiments on real-world datasets demonstrated that our proposed methods are efficient and effective to defend against the inference attack, and especially the EPPD algorithm substantially outperforms the existing masking techniques while the d-KP approximation algorithm has a lower

computational complexity, and it is applicable when dealing with the undirected social relation disclosure problem.

In addition, we designed a novel online social network privacy-preserving authorization framework, RANPriv-OAuth2, based on the OAuth 2.0 protocol. In the framework, we implemented the two privacy-preserving data sharing algorithms and provided the users with flexible and customized privacy settings for different privacy requirements and different third-party services.

## 7.2 Future Work

There are several research topics for the future work. We can further study the inference attack by improving the inference attack models as well as the protection methods.

### 7.2.1 Potential Improvements of Inference Models

In the thesis, we mainly discuss the inference attack models via the relationships and network characteristics to profile OSN users. Usually, the community structures contain more information than the basic dyadic or triadic relations. However, it is more difficult to discover and analyze them. In the future work, we will try to find out the potential community characteristics of the hidden information. Another limit for our work is that the social network model is a static model instead of a dynamic one. The social network structure always changes from time to time so that a dynamic social network model may imply more details and information about the hidden secrets. To improve the inference models, it is also a potential direction.

### 7.2.2 Potential Improvements of Protection Models

For the future work, we will further study the utility metrics to consider more complicated situations, such as non-linear utility objectives and attributes with different granularities. For the social relation masking, we will also investigate the correlations among structural or community information and user privacy, and makes the corresponding protection algorithms to defend against the inference attack based on these aspects.

# Bibliography

[1] Jemal H Abawajy, Mohd Izuan Hafez Ninggal, and Tutut Herawan. Privacy preserving social network data publication. *IEEE Communications Surveys & Tutorials*, 18(3):1974–1997, 2016.

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

[3] Barbara Carminati, Elena Ferrari, and Andrea Perego. Enforcing access control in web-based social networks. *ACM Transactions on Information and System Security (TISSEC)*, 13(1):6, 2009.

[4] Barbara Carminati, Elena Ferrari, and Marco Viviani. Security and trust in online social networks. *Synthesis Lectures on Information Security, Privacy, & Trust*, 4(3):1–120, 2013.

[5] Eric Y Chen, Yutong Pei, Shuo Chen, Yuan Tian, Robert Kotcher, and Patrick Tague. Oauth demystified for mobile application developers. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 892–903. ACM, 2014.

[6] Jiayi Chen, Jianping He, Lin Cai, and Jianping Pan. Profiling online social network users via relationships and network characteristics. In *Global Communications Conference (GLOBECOM), 2016 IEEE*, pages 1–6. IEEE, 2016.

[7] Yuan Cheng, Jaehong Park, and Ravi Sandhu. An access control model for online social networks using user-to-user relationships. *IEEE Transactions on Dependable and Secure Computing*, 13(4):424–436, 2016.

[8] Wei-Yen Day, Ninghui Li, and Min Lyu. Publishing graph degree distribution with node differential privacy. In *Proc. of ACM SIGMOD*, 2016.

[9] Yuxiao Dong, Yang Yang, Jie Tang, Yang Yang, and Nitesh V Chawla. Inferring user demographics and social strategies in mobile social networks. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 15–24. ACM, 2014.

[10] Cynthia Dwork. Differential privacy: A survey of results. In *Proc. of TAMC*. Springer, 2008.

[11] Catherine Dwyer, Starr Hiltz, and Katia Passerini. Trust and privacy concern within social networking sites: A comparison of facebook and myspace. *AMCIS 2007 proceedings*, page 339, 2007.

[12] David Easley and Jon Kleinberg. *Networks, crowds, and markets: Reasoning about a highly connected world*. Cambridge University Press, 2010.

[13] Inc. Facebook. Facebook statistics. `http://newsroom.fb.com/company-info/`. Accessed Jan 13, 2016.

[14] Quan Fang, Jitao Sang, Changsheng Xu, and M Shamim Hossain. Relational user attribute inference in social media. *Multimedia, IEEE Transactions on*, 17(7):1031–1044, 2015.

[15] Daniel Fett, Ralf Küsters, and Guido Schmitz. A comprehensive formal security analysis of oauth 2.0. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 1204–1215. ACM, 2016.

[16] Benjamin Fung, Ke Wang, Rui Chen, and Philip S Yu. Privacy-preserving data publishing: A survey of recent developments. *ACM Computing Surveys (Csur)*, 42(4):14, 2010.

[17] Neil Zhenqiang Gong, Ameet Talwalkar, Lester Mackey, Ling Huang, Eui Chul Richard Shin, Emil Stefanov, Elaine Runting Shi, and Dawn Song. Joint link prediction and attribute inference using a social-attribute network. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 5(2):27, 2014.

[18] Neil Zhenqiang Gong, Wenchang Xu, Ling Huang, Prateek Mittal, Emil Stefanov, Vyas Sekar, and Dawn Song. Evolution of social-attribute networks: measurements, modeling, and implications using google+. In *Proc. of ACM IMC*, 2012.

[19] Inc. Google. Teams and roles. `http://www.google.com/about/careers/teams/`. Accessed: Mar 20, 2016.

[20] Dick Hardt. The oauth 2.0 authorization framework. 2012.

[21] Eszter Hargittai et al. Facebook privacy settings: Who cares? *First Monday*, 15(8), 2010.

[22] Michael Hay, Gerome Miklau, David Jensen, Don Towsley, and Philipp Weis. Resisting structural re-identification in anonymized social networks. *Proceedings of the VLDB Endowment*, 1(1):102–114, 2008.

[23] Raymond Heatherly, Murat Kantarcioglu, and Bhavani Thuraisingham. Preventing private information inference attacks on social networks. *IEEE Transactions on Knowledge and Data Engineering*, 25(8):1849–1862, 2013.

[24] Hongxin Hu, Gail-Joon Ahn, and Jan Jorgensen. Multiparty access control for online social networks: model and mechanisms. *IEEE Transactions on Knowledge and Data Engineering*, 25(7):1614–1627, 2013.

[25] Mathieu Jacomy. Force-atlas graph layout algorithm. *URL: http://gephi.org/2011/forceatlas2-the-new-version-of-our-home-brew-layout*, 2009.

[26] Zach Jorgensen, Ting Yu, and Graham Cormode. Publishing attributed social graphs with formal privacy guarantees. In *Proc. of ACM SIGMOD*, 2016.

[27] Andreas M Kaplan and Michael Haenlein. Users of the world, unite! the challenges and opportunities of social media. *Business Horizons*, 53(1):59–68, 2010.

[28] Hans Kellerer, Ulrich Pferschy, and David Pisinger. *Introduction to NP-Completeness of knapsack problems*. Springer, 2004.

[29] John Krumm. Inference attacks on location tracks. *Pervasive computing*, pages 127–143, 2007.

[30] Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`, June 2014.

[31] Huaxin Li, Haojin Zhu, Suguo Du, Xiaohui Liang, and Xuemin Shen. Privacy leakage of location sharing in mobile social networks: Attacks and defense. *IEEE Transactions on Dependable and Secure Computing*, 2016.

[32] Jiwei Li, Alan Ritter, and Dan Jurafsky. Inferring user preferences by probabilistic logical reasoning over social networks. *arXiv preprint arXiv:1411.2679*, 2014.

[33] Muyuan Li, Haojin Zhu, Zhaoyu Gao, Si Chen, Le Yu, Shangqian Hu, and Kui Ren. All your location are belong to us: Breaking mobile social networks for automated user location tracking. In *Proceedings of the 15th ACM international symposium on Mobile ad hoc networking and computing*, pages 43–52. ACM, 2014.

[34] Xiang-Yang Li, Chunhong Zhang, Taeho Jung, Jianwei Qian, and Linlin Chen. Graph-based privacy-preserving data publication. In *Proc. of IEEE INFOCOM*, 2016.

[35] Jinfei Liu, Li Xiong, and Jun Luo. Semantic security: Privacy definitions revisited. *Transactions on Data Privacy*, 6(3):185–198, 2013.

[36] Yabing Liu, Krishna P Gummadi, Balachander Krishnamurthy, and Alan Mislove. Analyzing facebook privacy settings: user expectations vs. reality. In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, pages 61–70. ACM, 2011.

[37] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):3, 2007.

[38] Sofus A Macskassy and Foster Provost. Classification in networked data: A toolkit and a univariate case study. *Journal of Machine Learning Research*, 8(May):935–983, 2007.

[39] Michelle Madejski, Maritza Johnson, and Steven M Bellovin. The failure of online social network privacy settings. *Department of Computer Science, Columbia University, Tech. Rep. CUCS-010-11*, 2011.

[40] Michelle Madejski, Maritza Johnson, and Steven M Bellovin. A study of privacy settings errors in an online social network. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2012 IEEE International Conference on*, pages 340–345. IEEE, 2012.

[41] David J Martin, Daniel Kifer, Ashwin Machanavajjhala, Johannes Gehrke, and Joseph Y Halpern. Worst-case background knowledge for privacy-preserving data publishing. In *Proc. of IEEE ICDE*, 2007.

[42] Miller McPherson, Lynn Smith-Lovin, and James M Cook. Birds of a feather: Homophily in social networks. *Annual review of sociology*, pages 415–444, 2001.

[43] Alan Mislove, Bimal Viswanath, Krishna P Gummadi, and Peter Druschel. You are who you know: inferring user profiles in online social networks. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 251–260. ACM, 2010.

[44] Arvind Narayanan and Vitaly Shmatikov. De-anonymizing social networks. In *Security and Privacy, 2009 30th IEEE Symposium on*, pages 173–187. IEEE, 2009.

[45] J. Qian, X. Y. Li, C. Zhang, L. Chen, T. Jung, and J. Han. Social network de-anonymization and privacy inference with knowledge graph model. *IEEE Transactions on Dependable and Secure Computing*, PP(99):1–1, 2017.

[46] Jianwei Qian, Xiang-Yang Li, Chunhong Zhang, and Linlin Chen. De-anonymizing social networks and inferring private attributes using knowledge graphs. In *Proc. of IEEE INFOCOM*, 2016.

[47] Jonghyuk Song, Sangho Lee, and Jong Kim. Inference attack on browsing history of twitter users using public click analytics and twitter metadata. *IEEE Transactions on Dependable and Secure Computing*, 13(3):340–354, 2016.

[48] San-Tsai Sun and Konstantin Beznosov. The devil is in the (implementation) details: an empirical analysis of oauth sso systems. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 378–390. ACM, 2012.

[49] Guojun Wang, Qin Liu, Feng Li, Shuhui Yang, and Jie Wu. Outsourcing privacy-preserving social networks to a cloud. In *Proc. of IEEE INFOCOM*, 2013.

[50] Xintao Wu, Xiaowei Ying, Kun Liu, and Lei Chen. A survey of privacy-preservation of graphs and social networks. In *Managing and mining graph data*, pages 421–453. Springer, 2010.

[51] Qiuyu Xiao, Jiayi Chen, Le Yu, Huaxin Li, Haojin Zhu, Muyuan Li, and Kui Ren. Poster: Locmask: A location privacy protection framework in android system. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 1526–1528. ACM, 2014.

[52] Xing Xie. Potential friend recommendation in online social network. In *Green Computing and Communications (GreenCom), 2010 IEEE/ACM Int'l Conference on & Int'l Conference on Cyber, Physical and Social Computing (CPSCom)*, pages 831–835. IEEE, 2010.

[53] Kaihe Xu, Yuanxiong Guo, Linke Guo, Yuguang Fang, and Xiaolin Li. My privacy my decision: Control of photo sharing on online social networks. *IEEE Transactions on Dependable and Secure Computing*, 2015.

[54] Zhijun Yin, Manish Gupta, Tim Weninger, and Jiawei Han. Linkrec: a unified framework for link recommendation with user attributes and graph structure. In *Proc. of ACM WWW*, 2010.

[55] Alyson L Young and Anabel Quan-Haase. Information revelation and internet privacy concerns on social network sites: a case study of facebook. In *Proceedings of the fourth international conference on Communities and technologies*, pages 265–274. ACM, 2009.

[56] Yuchen Zhao, Guan Wang, Philip S Yu, Shaobo Liu, and Simon Zhang. Inferring social roles and statuses in social networks. In *Proc. of ACM SIGKDD*, 2013.

[57] Elena Zheleva and Lise Getoor. To join or not to join: the illusion of privacy in social networks with mixed public and private user profiles. In *Proc. of ACM WWW*, 2009.