Collective Data-Sanitization for Preventing Sensitive Information Inference Attacks in Social Networks

Zhipeng Cai, *Senior Member, IEEE*, Zaobo He, *Student Member, IEEE*, Xin Guan, *Member, IEEE*, and Yingshu Li, *Senior Member, IEEE*

Abstract—Releasing social network data could seriously breach user privacy. User profile and friendship relations are inherently private. Unfortunately, sensitive information may be predicted out of released data through data mining techniques. Therefore, sanitizing network data prior to release is necessary. In this paper, we explore how to launch an inference attack exploiting social networks with a mixture of non-sensitive attributes and social relationships. We map this issue to a collective classification problem and propose a collective inference model. In our model, an attacker utilizes user profile and social relationships in a collective manner to predict sensitive information of related victims in a released social network dataset. To protect against such attacks, we propose a data sanitization method collectively manipulating user profile and friendship relations. Besides sanitizing friendship relations, the proposed method can take advantages of various data-manipulating methods. We show that we can easily reduce adversary's prediction accuracy on sensitive information, while resulting in less accuracy decrease on non-sensitive information towards three social network datasets. This is the first work to employ collective methods involving various data-manipulating methods and social relationships to protect against inference attacks in social networks.

Index Terms—Inference attack, collective inference, rough set theory, attribute dependency, data sanitization

1 Introduction

Social networks provide a virtual stage for users to reveal themselves to their own societies or to the public. For example, Facebook users publish information regarding favorite books, popular songs, interesting movies, political views, etc. Users of ResearchGate [1], a professional network for scientists and researchers, publish information regarding research experiences, publications, academic activities and so on. Besides users, third party users may benefit from the huge amount of published data that can be easily and deliberately obtained from social networks. Third party users may refer to researchers, merchants, advertisers, and even adversaries. For instance, IMDb [2] may make use of the data released by Facebook to suggest proper movies and TV programs to target users. However, the data release scale is being restrained by the emerging privacy concerns. Facebook Beacon [3] is an unsuccessful example that reminds people to release anonymous and incomplete user data. Therefore, in addition to non-sensitive information, third party users are trying to mine sensitive information out of the released data.

- *Z. Cai, Z. He and Y. Li are with the Department of Computer Science, Georgia State University, Atlanta, GA 30303. E-mail: {zcai, yili}@gsu.edu, zhe4@student.gsu.edu.*
- *X. Guan is with the School of Information Science and Technology, Heilongjiang University, Harbin, Heilongjiang 100044, China. E-mail: guanxin.hlju@gmail.com.*

Manuscript received 28 Feb. 2016; revised 9 Sept. 2016; accepted 15 Sept. 2016. Date of publication 26 Sept. 2016; date of current version 6 July 2018. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below. Digital Object Identifier no. 10.1109/TDSC.2016.2613521

Primarily, two kinds of privacy concerns present in social networks: inherent-data privacy and latent-data privacy [4]. Inherent-data privacy focuses on released sensitive information contained in the user-submitted data profiles. For example, age and gender are necessary information in health services and usually most users are unwilling to release such information. De-anonymization towards anonymous data is an inherent-data privacy instance. For example, two New York Time journalists used to successfully identify personal information from the published search logs involving 650,000 users. The logs include the information of name, age, sex, location, etc., and such information is associated with a specific individual. Another well-known example is that individuals' medical visits were successfully identified based on the anonymized data made available by the Group Insurance Commission, and the former governor of Massachusetts was one of the victims. Latent-data privacy, on the other hand, focuses on unreleased sensitive information that can be inferred out of the released data or users' social relationships. For instance, Jenny does not publish her political opinions online, yet such information could be inferred by mining her friends' data as Jenny's social relationships may be public. Another illustrative example comes from ABCNews.com [5] and Boston Globe [6]. They reported that it is possible to determine the sexual orientations of some users by analyzing a subgraph from Facebook.

In this paper, we focus on latent-data privacy. We assume third party users may collect anonymous user data from social networks. Users may or may not release sensitive information [7]. However, third party users can carry

out de-anonymization actions and further infer sensitive information of users. We first investigate how to infer sensitive information hidden in the released data. Then, we propose some effective data sanitization strategies to prevent information inference attacks. On the other hand, the sanitized data obtained by these strategies should not reduce the valuable benefit brought by the abundant data resources, so that non-sensitive information can still be inferred and utilized by third party users.

To explore how to launch an inference attack by third party users, we investigate a typical inference attack called collective inference. We present a novel implementation method for collective inference. Collective inference mainly relies on iteratively propagating current predicting results throughout a network to improve prediction accuracy. Then we need to study how to best predict sensitive information in each iteration. Previous works primarily utilize the Naive Bayes classifier to infer sensitive information in each iteration. However, social network data are generally incomplete, inaccurate and uncertain, preventing the existing approaches from obtaining a precise learned model. Then the inference performance may be degraded. Our work considers the special features of social network data to investigate collective attacks in diverse large scale social networks.

The previous works for preventing inference attacks mainly have three deficiencies. First, users' released data and their friendship information are separately considered, degrading prediction accuracy possibly. Second, only a single type of manipulation method, such as filtering, perturbing, and adding, is considered at a time, incurring poor effectiveness performance. Third, data utility is not taken into full consideration, reducing the benefit brought by the abundant amount of data. Therefore, the previous works cannot reasonably balance privacy and data utility. In this work, we propose two strategies to prevent inference attacks. Our work can guarantee that necessary information cannot be obtained by third party users to accurately predict sensitive information. In addition, data utility can still be promoted by our work.

In this work, we focus on two concrete issues: (a) how exactly third party users launch an inference attack to predict sensitive information of users, and (b) are there effective strategies to protect against such an attack to achieve a desired privacy-utility tradeoff. Following is the summary of our contributions and improvements over the previous works:

1) Rather than considering users' attribute sets and friendship information separately, we present a novel implementation method for collective inference that can effectively predict users' sensitive information, with both attribute sets and friendship information comprehensively taken into account.
2) To hide sensitive information through manipulating attribute sets, rather than simply implementing perturbing methods through introducing various types of noises, we rationally identify the dependency relationship between sensitive information and non-sensitive information.
3) To hide sensitive information through manipulating friendship information, rather than simply adding or removing friendship links, we propose a novel concept that enables us to easily find the most representative links.
4) We further analyze the relationships between data utility and non-sensitive information. The identified relationships then support us to design a collective strategy to achieve a desired privacy-utility tradeoff. Our collective strategy takes advantages of various manipulating methods instead of only relying on one manipulating method.

The remainder of the paper is organized as follows. Section 2 addresses the related works. The investigated problem is formalized in Section 3. Section 4 introduces some preliminary knowledge. In Section 5, we investigate the working scenario of inference. Some data sanitization strategies are then proposed in Sections 6 and 7. The evaluation results are presented in Section 8. Section 9 concludes the paper.

2 Related Works

Anonymization and De-Anonymization. Privacy is typically protected by anonymization methods, i.e., removing information regarding name, religion, political view, etc. However, such network could be de-anonymized by utilizing background knowledge such as reference network. For example, De-anonymized approaches utilize 'network mapping' to map nodes from reference network to anonymized network.

In [8], the authors propose a community-enhanced de-anonymization approach to re-identify users, which first partitions the network into communities and then carries out a two-stage mapping: first mapping communities then the entire network. In [9], the authors consider a de-anonymization algorithm to re-identify the users in an anonymized social network based on network topology, namely, mapping the anonymous target graph and the aggregated graph from multiple social networks. Comparatively, our work attempts to protect against inferring sensitive information of users, rather than solely re-identifying users in an anonymized network. In [10], the authors propose a family of anonymization algorithms and consider the corresponding de-anonymization algorithms. However, their network model only consists of users and friendship links and the attackers are assumed to re-identify the users. Clearly, their studied problem is quite different from ours because they do not consider how to anonymize a network in order to protect against inference attacks on sensitive attributes. The work in [11] presents a systematic survey for the anonymization techniques for social network data. The anonymization techniques are mainly categorized into the clustering-based approaches and the graph modification approaches. Comparatively, our work takes advantage of various techniques to balance privacy and data utility.

Inference Attacks and Protecting Methods. There are many works investigating how to infer sensitive information of users. In [12], the authors demonstrate that users' sensitive information can be inferred via detecting communities based on the assumption that users in a community are more likely share common attributes. Similarly, the work in [13] indicates that users' sensitive information can be inferred based on friendship information and group memberships, and it also shows that disclosure of one user's hidden attribute would breach her friends privacy. In [14], the authors develop a Bayes network model to infer sensitive information based on

friendship links. Meanwhile, [14] takes a protection method that randomly hides friendship links and friends' attributes. Comparatively, our work studies which friendship link(s) and users own attribute(s) should be manipulated to protect privacy. Close to our work, [15] studies how to infer users' demographics (gender and age) depending on users' daily communication patterns. It novelly harness both the interaction between sensitive attributes and non-sensitive attributes, and the interaction among sensitive attributes (such as gender and age). Clearly, their method is quite different from ours because they do not consider the information from friendship relations that can be leveraged in order to infer sensitive information. Moreover, our work further studies how to protect against such inference attack deriving from collective information. Moreover, in [16], the authors consider the inference attacks to infer which shortened URLs clicked by a user in Twitter, only based on two public available information, twitter metadata and the click analytics information.

Note that sanitizing data prior to release is a popular method to balance a privacy-utility tradeoff. For example, both [17] and [18] sample a noise-data with differential privacy guarantees. Meanwhile, several works are proposed to protect other social properties, such as link privacy [19], degree distribution [20] and graph privacy [21]. Moreover, privacy preserving dynamic data streams attract people's attentions. For example, the works in [22] and [23] study how to effectively process distributed data streams while protecting personal data privacy.

Compared with the previous works, we not only explore collective attacks based on attribute sets and friendship information, but also propose some corresponding protecting strategies that can help with choosing the proper attributes and links to manipulate so that the privacy-utility tradeoff can be optimized.

3 Problem Statement

3.1 Social Network Model

We now present our network model.

Definition 3.1 (Social Network). *A social network is a graph $G(V, E, \mathcal{X})$ consisting of user set V, friendship link set E and the set of user attribute sets denoted by $\mathcal{X}$. For any user $u_i, u_j \in V$ $(1 \leq i, j \leq |V|)$, their friendship link $e_{i,j} \in E$ also indicates $e_{j,i} \in E$.*

Definition 3.2 (Attribute Set). *For an arbitrary user u_i, its attribute set is denoted by $\vec{X}_i \in \mathcal{X}$ $(1 \leq i \leq |V|)$. Each attribute $x_j \in \vec{X}_i$ $(1 \leq j \leq |\vec{X}_i|)$ is for a certain attribute category $h_r \in H$ $(1 \leq r \leq |H|)$, where H is the set of all the categories for a social network. We denote an attribute x_j as $x_j = \{h_r : l_1; \ldots; l_t\}$, which means x_j is for category h_r with value list $l_1; \ldots; l_t$ where $t \geq 1$.*

It is worth mentioning that for a particular category, the user input can be a single value or multiple values. For example, for category "Favorite movies", the input can be "The Terminator", "Titanic" and "Pianist". For category "Birthday", the input should be a single value. Moreover, there may be categories with no input values for some users, such as "Political view" and "Religion view". In specific applications, which categories are sensitive are determined by users. For example, Facebook users can directly hide their sensitive attribute "political view" or "sexual orientation" in their profiles.

We use $H_s \subseteq H$ to denote the set of the sensitive categories for a particular user. Any $x_j \in \vec{X}_i$ is a sensitive attribute of user u_i if x_j is for $h_r \in H_s$. Following is an example

$$
\begin{aligned}
H &= \{\text{Favorite movies, Favorite books, Religion view, Political view}\}\\
V &= \{u_1 = \text{Jack}, u_2 = \text{Emily}\}\\
\vec{X}_1 &= \{x_1 = \{\text{Favorite movies: Titanic}\}, x_2 = \{\text{Favorite books: Automata; Machine learning}\}\}\\
\vec{X}_2 &= \{x_1 = \{\text{Favorite movies: Pianist}\}, x_2 = \{\text{Political view: Conservative}\}\}\\
e_{1,2} &\in E, e_{2,1} \in E.
\end{aligned}
$$

In this example, there are four categories as shown in H. There are two users u_1 and u_2. u_1 publishes one favorite movie and two favorite books. Thus, for u_1, $H_s = \{\text{Religion view, Political view}\}$. u_2 publishes her political view, thus for u_2, $H_s = \{\text{Religion view}\}$. u_1 and u_2 are friends in the social network.

Each possible attribute value for an arbitrary attribute category $h_r \in H_s$ can be viewed as a class label when third party users predict sensitive attribute x_j for category h_r. For example, if h_r is category "Political view", we can consider two possible attribute values as our class labels: "Conservative" and "Liberal". Class label is formally defined as follows.

Definition 3.3 (Class Label). *We say that y_i $(i \geq 1)$ is one of the class labels for $h_r \in H_s$ if y_i is one of the attribute values for attribute category h_r.*

3.2 Utility and Privacy

We now formally define privacy and utility. The existing privacy definitions, such as differential privacy [24], k-anonymity [25], l-diversity [26], are only for inherent-data, and are not suitable for inference attacks. Meanwhile, most of the existing works evaluate data utility by only considering how much noise is added to the initial data. In this paper, we present a finer-grained utility definition.

The capability of third party users depends on how many disclosed sensitive attributes and how much background knowledge are known to them. Intuitively, we expect released data do not help with significantly improving prediction accuracy compared with the prediction accuracy based on prior knowledge.

Definition 3.4 (Prior Knowledge). *Prior knowledge is the information related to a data set but not necessarily obtained from the data set.*

For instance, prior knowledge can be users' movie viewing records, phone numbers, zip codes or the publicly available Voter Registration List. Such knowledge can be obtained from many ways rather than the data set itself. Then, privacy is formally defined as follows.

Definition 3.5 (Classifier Accuracy). *Classifier accuracy, denoted as $\Lambda_c^{h_r}(G)$, is the accuracy of classifier c trained on the*

available information of social graph G, and it is used to classify G to predict attributes for category $h_r \in H$.

Definition 3.6 (Privacy). *Given a social network G, prior knowledge $\mathcal{K}$ held by third party users, a set of classifiers denoted by $\mathcal{C}$, and a set of sensitive categories H_s, G is $(\Delta, \mathcal{C})$-private if for each attribute category $h_r \in H_s$, G satisfies*

$$\max_{c \in \mathcal{C}} \Lambda_c^{h_r}(G, \mathcal{K}) - \max_{c' \in \mathcal{C}} \Lambda_{c'}^{h_r}(\mathcal{K}) \leq \Delta,$$

Δ denotes the additional prediction accuracy gained by third party users by utilizing G. Clearly, $\Delta \geq 0$ since more related information are known by third party users. If $\Delta = 0$, it indicates that third party users do not gain additional prediction accuracy in predicting sensitive attributes for category $h_r \in H_s$. Note that Δ is specified by data publisher.

With respect to data utility, there are two factors to consider. First, the sanitized social graph should not deviate from the initial one by too much. Second, the sanitized social graph should guarantee a beneficiary can effectively infer the non-sensitive information of users. Then, we formally define it as follows:

Definition 3.7 (Utility). *Given social graph G, data dissimilarity measurer $\mathcal{M}$, prior knowledge known to third party users $\mathcal{K}$, classifier set $\mathcal{C}$, and non-sensitive category set $H - H_s$, the sanitized graph of G, denoted as G', satisfies (ϵ, δ)-utility if for each attribute category $h_r \in H - H_s$, the following conditions are satisfied:*

(i) $\mathcal{M}(G, G') \leq \epsilon$;

(ii) $\max_{c \in \mathcal{C}} \Lambda_c^{h_r}(G', \mathcal{K}) - \max_{c' \in \mathcal{C}} \Lambda_{c'}^{h_r}(\mathcal{K}) \geq \delta$.

δ denotes the additional prediction accuracy gained by third party users by utilizing G'. Clearly, $\delta \geq 0$. If $\delta = 0$, it indicates that the classifier does not gain additional classification accuracy by utilizing G' in predicting non-sensitive attributes for category $h_r \in H - H_s$. As well, both ϵ and δ are specified by data publisher.

Compared with the existing definitions, Definition 3.7 takes the inferred non-sensitive attributes into consideration (condition (ii)). That is, any sanitization strategy should guarantee a beneficiary of the sanitized data and could effectively infer the non-sensitive attributes.

3.3 Problem Definition

Based on the above privacy and utility definitions, given user-specified thresholds on privacy and utility, the sanitization social graph is expected to achieve the desired privacy-utility tradeoff:

Input:

(1) Social graph $G(V = V^k \cup V^U,\ E,\ \mathcal{X},\ Y = Y^K \cup Y^U,\ H_s)$ with user set V, friendship link set E, the set of user attribute sets $\mathcal{X}$, and the set of sensitive categories H_s. $y^i \in Y$ is a class label of u_i for an arbitrary category $h_r \in H_s$.

(2) Y^K is the set of known labels for users $u_i \in V^K$, where V^K is the set of users with known labels. Y^U is the set of unknown labels for users $u_i \in V^U$, where V^U is the set of users with unknown labels.

(3) User-specified privacy threshold Δ, and utility thresholds ϵ and δ.

Output:

Task 1: Prediction method that can predict Y^U for users $u_i \in V^U$, where $V^U = V - V^K$.

Task 2: Data publishing method with optimized tradeoff between privacy and utility.

The first task investigates how third party users launch an inference attack to predict sensitive attributes. A powerful inference method is expected. Since users have the option to publish no attributes for some categories, the attribute data are usually incomplete. Meanwhile, there are always dishonest users, so the attribute data may be inaccurate or uncertain. Therefore, we employ the Rough Set Theory (RST) as a building block to develop our inference method. RST is a mathematical tool that can be used to extract knowledge from incomplete, inaccurate and uncertain data sets. It allows us to easily analyze the large scale and diverse social network data. For the second task, RST helps us to easily distinguish the objective attributes to be manipulated to protect against inference attacks.

4 Preliminaries

In this section, several concepts of RST are introduced and some illustrative examples are given. We then describe how to use RST to extract decision rules from the attribute data. Last, we present how to determine the class label of an user based on friendship information.

4.1 Rough Set Theory

We only introduce several basic concepts of RST and more details can be found in [27]. Knowledge representation in RST is through an information system. Based on the information system, the decision rules can be extracted.

Definition 4.1 (Information System). *An information system is a pair $\Gamma = (V,\ H = C \cup D)$, where V is a finite set of users, and H is a nonempty finite set of attribute categories. H includes two subsets: the set of condition attribute categories C and the set of decision attribute categories D. For each attribute x_j for category $h_r \in H$, function $f_{x_j}(u) : V \xrightarrow{x_j} \Omega_{h_r}$ assigns an attribute value to x_j for user u, where Ω_{h_r} is the attribute value set for h_r.*

Example 4.1. A simple example of information system for a Facebook data set is presented in Table 1. As shown in Table 1, $V = \{u_1, u_2, \ldots u_8\}$, $C = \{h_1, h_2, h_3\}$, and $D = \{d\}$. Attribute "Favorite movies" of u_1 is assigned value "God's Not Dead".

Definition 4.2 (Indiscernibility Relation). *Given $H' \subseteq H$, any two users u_i and u_j having H'-indiscernibility relation is denoted by $IND_{H'}(u_i, u_j)$ where*

$$IND_{H'}(u_i, u_j) = \{(u_i, u_j) \in V^2 \mid \forall x_j \text{ for } H',\ f_{x_j}(u_i) = f_{x_j}(u_j)\}.$$

We denote the users whose attributes have the same values for H' as $[u]_{H'}$, called the equivalence class of H'-indiscernibility relation.

Example 4.2. Suppose $H' = \{h_2, h_3\}$ which is extracted from Table 1. Hence, both (u_1, u_3) and (u_2, u_5) have

TABLE 1
An Example Information System for a Facebook Data Set

V	h_1: Favorite musical	h_2: Favorite movies	h_3: Favorite books	d: Political view
u_1	Taylor Swift	God's Not Dead	Heaven Is For Real	Conservative
u_2	Carrie Underwood	Son of God	I Declare	Conservative
u_3	Carrie Underwood	God's Not Dead	Heaven Is For Real	Liberal
u_4	George Strait	The Fast and the Furious	Heaven Is For Real	Green
u_5	George Strait	Son of God	I Declare	Liberal
u_6	Taylor Swift	Transformers	The Hunger Games	Conservative
u_7	George Strait	Son of God	The Hunger Games	Liberal
u_8	Taylor Swift	Transformers	I Declare	Conservative

H'-indiscernibility relation. Table 1 also indicates $[u]_{H'} = \{\{u_1, u_3\}, \{u_2, u_5\}, \{u_4\}, \{u_6\}, \{u_7\}, \{u_8\}\}$.

Definition 4.3 (H'-Lower and H'-Upper Approximation of V'). *Given $V' \subseteq V$ and $H' \subseteq H$, V' can be approximated using only the information contained in H' by constructing H'-lower approximation and H'-upper approximation of V':*

$$\underline{H'}V' = \{u \mid [u]_{H'} \subseteq V'\}$$
$$\overline{H'}V' = \{u \mid [u]_{H'} \cap V' \neq \Phi\}.$$

Example 4.3. For the information system shown in Table 1, let $H' = \{h_2, h_3\}$ and $V' = \{u_1, u_2, u_6, u_8\}$. Hence, $\overline{H'}V' = \{u_1, u_2, u_3, u_5, u_6, u_8\}$ and $\underline{H'}V' = \{u_6, u_8\}$.

Definition 4.4 (Attribute Dependency). *Let $H' \subseteq H$ and $H'' \subseteq H$. We say that H'' depends on H' with degree k $(0 \leq k \leq 1)$, denoted by $H' \rightarrow^k H''$, if*

$$k = \gamma(H',\ H'') = \frac{|POS_{H'}(H'')|}{|V|}, \tag{1}$$

where $POS_{H'}(H'') = \bigcup_{X \in [x]_{H''}} \underline{H'}(X)$, called H'-positive region of H''.

In particular, if $k = 1$, we say that A'' totally depends on A'.

Example 4.4. For the information system shown in Table 1, let $H' = \{h_2, h_3\}$ and $H'' = d$. Since

$$[x]_{H'} = \{\{u_1, u_3\}, \{u_2, u_5\}, \{u_4\}, \{u_6\}, \{u_7\}, \{u_8\}\}$$
$$[x]_{H''} = \{\{u_1, u_2, u_6, u_8\}, \{u_4\}, \{u_3, u_5, u_7\}\}$$
$$\underline{H'}(\{u_1, u_2, u_6, u_8\}) = \{u_6, u_8\}$$
$$\underline{H'}(\{u_4\}) = \{u_4\}$$
$$\underline{H'}(\{u_3, u_5, u_7\}) = \{u_7\},$$

we can compute

$$POS_{H'}(H'') = \{u_6, u_8, u_4, u_7\}.$$

Hence,

$$k = \gamma(H',\ H'') = \frac{|POS_{H'}(H'')|}{|V|} = 4/8 = 1/2.$$

For an information system, there usually exist some redundant condition attributes that do not provide any additional knowledge for prediiction. Hence, RST defines a *reduct* for an information system as a minimum attribute set that keeps the indiscernibility relation. Furthermore, as would be discussed in Section 7, reduct can help us to find the privacy-dependent attributes and utility-dependent attributes, which is the foundation to balance the privacy-utility tradeoff.

Definition 4.5 (Reduct). *Given an information system $\Gamma = (V, H = C \cup D)$, any $R \subseteq C$ is a reduct of C if*

(i). $POS_R(D) = POS_C(D)$;
(ii). for any $h_r \in C, IND(R - h_r) \neq IND(C)$.

After removing the repetitive row, $(V, R \cup D)$ is called a reduct system.

Example 4.5. For the information system shown in Table 1, let $R_1 = \{h_1, h_2\}$, $R_2 = \{h_1, h_3\}$ and $R_3 = \{h_2, h_3\}$. We have

$$POS_C(D) = \{u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8\}$$
$$POS_{h_1}(D) = \{u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8\}$$
$$POS_{h_2}(D) = \{u_1, u_2, u_3, u_4, u_5, u_6, u_7, u_8\}$$
$$POS_{h_3}(D) = \{u_4, u_6, u_7, u_8\}.$$

Hence, we can conclude R_1 and R_2 are reducts of C since they also satisfy the second condition according to Definition 4.2. However, R_3 is not a reduct of C.

The first condition of Definition 4.5 indicates that the reduct retains the indiscernibility relation of the original attribute set. That is, any indiscernible pair of objects based on R is also indiscernible in A and vice versa. The second condition indicates that R is the minimum subset of A that keeps its indiscernibility.

4.2 Generating Decision Rules Based on an Attribute Set

We now introduce how the decision rules are generated based on the reduct system $(V, R \cup D)$. Suppose the equivalence class of the R-indiscernibility relation and the D-indiscernibility relation are $[u]_R = \{P_1, P_2, \ldots, P_m\}$ and $[u]_D = \{Q_1, Q_2, \ldots, Q_n\}$, respectively. Each P_i $(1 \leq i \leq m)$ and Q_j $(1 \leq j \leq n)$ is a user or a set of users. For example, for the information system shown in Table 2, $[u]_R = \{P_1 = \{u_1, u_3, u_9\}, P_2 = \{u_2, u_4\}, P_3 = \{u_5, u_6\}, P_4 = \{u_7, u_8\}\}$ if $R = \{h_1, h_2\}$, and $[u]_D = \{Q_1 = \{u_1, u_2, u_3, u_4, u_7, u_9\}, Q_2 = \{u_5, u_6, u_8\}\}$ if $D = \{d\}$.

Since both $[u]_R$ and $[u]_D$ partition V, each P_i is associated with a set $M_i = \{Q_j \mid P_i \cap Q_j \neq \Phi\}$. For example, P_4 is associated with $M_4 = \{Q_1, Q_2\}$.

Hence, for an arbitrary user u, we have:

If $u \in P_i$, then $u \in Q_{j_1}$ or $u \in Q_{j_2} \ldots$ or $u \in Q_{j_{|M_i|}}$.

TABLE 2
Information System for Generating Decision Rules

V	h_1: Favorite musical	h_2: Favorite movies	d: Political view
u_1	Taylor Swift	God's Not Dead	Conservative
u_2	Carrie Underwood	Son of God	Conservative
u_3	Taylor Swift	God's Not Dead	Conservative
u_4	Carrie Underwood	Son of God	Conservative
u_5	George Strait	Son of God	Liberal
u_6	George Strait	Son of God	Liberal
u_7	Taylor Swift	Transformers	Conservative
u_8	Taylor Swift	Transformers	Liberal
u_9	Taylor Swift	God's Not Dead	Conservative

According to Definition 4.1, we know that each P_i of $[u]_R$ corresponds to an attribute vector $\vec{X}(P_i) = \{x_1^i, x_2^i \dots x_{|R|}^i\}$, where an arbitrary user $u \in P_i$ if and only if $f_{x_1^i}(u) = v_{x_1^i}$ and $\dots$ and $f_{x_{|R|}^i}(u) = v_{x_{|R|}^i}$, where $v_{x_k^i}$ $(1 \le k \le |R|)$ is the attribute value of attribute x_k for the users in P_i. For example, P_1 corresponds to $\vec{X}(P_1) = \{$"Taylor Swift", "Gods Not Dead" $\}$.

Similarly, suppose there is a signal decision attribute d, i.e., $|D = 1|$, and each Q_j of $[u]_D$ corresponds to a decision attribute value v_{d_j}, where an arbitrary user $u \in Y_j$ if and only if $f_d(u) = v_{d_j}$. For example, any $u \in Y_j$ if and only if $f_d(u)$ = "Conservative".

Hence, the above rule can be rewritten as

if $f_{x_1^i}(u) = v_{x_1^i}$ and $\dots$ and $f_{x_{|R|}^i}(u) = v_{x_{|R|}^i}$, then $f_d(u) = v_{d_1}$ or $f_d(u) = v_{d_2}$, or $\dots$, or $f_d(u) = v_{d_{|M_i|}}$.

If $P_i \subseteq Q_j$, which indicates the class label of any user $u \in P_i$ is uniquely determined by d_j, we say P_i is a deterministic class. Otherwise, we call P_i as an indeterministic class.

Example 4.6. We extract decision rules from the reduct system $(V, R \cup D)$ shown in Table 2, where $R = \{h_1, h_2\}$ and $D = \{d\}$. Let $P_1 = \{u_1, u_3, u_9\}$, $P_2 = \{u_2, u_4\}$, $P_3 = \{u_5, u_6\}$, $P_4 = \{u_7, u_8\}$, $Q_1 = \{x_1, x_2, x_3, x_4, x_7, x_9\}$ and $Q_2 = \{x_5, x_6, x_8\}$. Based on the prior analysis, P_1, P_2 and P_3 are deterministic classes. Hence, the following decision rules are extracted:

if A_1 = "Taylor Swift" and A_2 = "God's Not Dead",
then, D = "Conservative";
if A_1 = "Carrie Underwood" and A_2 = "Son of God",
then, D = "Conservative";
if A_1 = "George Strait" and A_2 = "Son of God",
then, D = "Liberal".

4.3 Prediction Based on Friendship Information

Another significant knowledge that can be utilized to infer sensitive attributes is friendship information in social networks. However, it is inaccurate to extract decision rules based on friendship information directly, since there are relatively few links from users with known labels that connect to an arbitrary user u_i. Therefore, rather than directly extracting decision rules from the friendship links of u_i, we consider u_j's class, where $u_j \in N_i$ and N_i is the neighbor set of u_i. For clarity, u_i in class y_t is denoted by y_t^i.

For simplicity, the probability of u_i to be in class y_t, denoted as $P(y_t^i)$, is the average probability of its neighbors being in y_t:

$$P(y_t^i|N_i) = \frac{1}{|N_i|} \sum_{u_j \in N_i} P(y_t^j).$$

However, purely calculating the average probability of neighbors would incur overfitting. To prevent this, the weighted-vote Relational Neighbor algorithm (wvRN) [28] suggests to add a weight to each friendship link. There are many such methods and we adopt the ones with the assumption that the more public attributes shared by two friends, the more is the sensitive attributes that are shared by two friends. Then we introduce weight $W_{i,j}$ between u_i and u_j as follows:

$$W_{i,j} = \frac{|(x_1^i, \dots, x_m^i) \cup (x_1^j, \dots, x_n^j)|}{|\vec{X}_i|}. \quad (2)$$

Equation (2) calculates the total number of attributes shared by u_i and u_j divided by the number of u_i's attributes. Obviously, $W_{i,j} \neq W_{i,j}$. Then to determine y^i based on N_i becomes the following,

$$P(y_t^i|N_i) = \frac{1}{|N_i|} \sum_{n_j \in N_i} P(y_t^j) \frac{W_{i,j}}{\sum_{n_k \in N_i} W_{i,k}}. \quad (3)$$

5 Collective Inference

Unfortunately, the prediction methods described in the previous section have several problems. The attribute-based classifier (Section 4.2) just considers the attribute sets of the users it is classifying. Conversely, relation-based classifier (Section 4.3) only considers the friendship information of a user. However, third party users may launch an inference attack by exploiting all the publicly available information. Moreover, a major problem of relation-based classifier is that it requires that at least one of the neighbors of each unlabeled user to be located in the training set (i.e., the set of users with known labels, as shown in Equation (3)). Obviously, this strict requirement is hard to be satisfied by real-world data. Collective inference attempts to tackle the above two issues by considering both attribute-based classifier and relation-based classifier in a collaborative manner to improve prediction accuracy. Formally, we consider the following network prediction problem.

Definition 5.1 (Collective Inference). *Given social graph $G(V = V^k \cup V^U, E, \mathcal{X}, Y = Y^K \cup Y^U, H_s)$ with user set V, friendship link set E, the set of user attribute sets $\mathcal{X}$, and the set of sensitive categories H_s. $y^i \in Y$ is a class label of u_i for an arbitrary category $h_r \in H_s$. L^K is the known labels for users $u_i \in V^K$. Collective inference is to predict Y^U for users $u_i \in V^U$, where $V^U = V - V^K$.*

This problem is challenging as some of the user labels are unknown. A fundamental idea is to first predict a class label approximately and then refine the predicted result iteratively. Several collective classification algorithms have been proposed to increase accuracy when the network users are interrelated, such as the Iterative Classification Algorithm (ICA) [29] and Gibbs sampling (Gibbs) [30]. Many collective classification algorithms and variants, including ICA, use an attribute-based classifier M_A to predict the approximate class label at the bootstrap stage; then, they use both attribute and link based classifier, M_{AR}, to refine the results. The algorithms

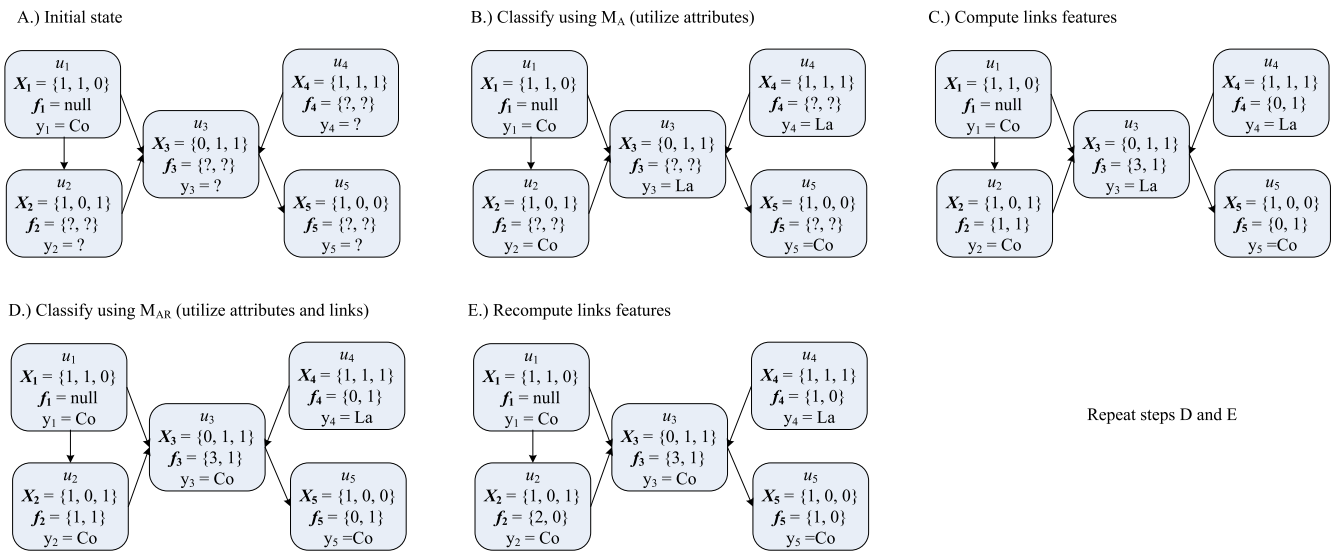

Fig. 1. An example for ICA-RST.

repeat these two operations until the class labels converge. We present an algorithm under the framework of ICA that takes RST as a local classifier (one that uses local information, e.g., attribute sets of users), denoted by ICA-RST.

ICA-RST is shown in Algorithm 1. It first learns an attribute-based classifier M_A based on the known labels Y^K (step 1), which is a set of RST decision rules. Then, by M_A, it predicts the labels of the users with unknown labels, V^U (steps 2-3). Step 5 stores the known labels Y^K and the predicted labels $\{y^i|u_i \in V^U\}$ in set Y^R. The known labels and the predicted labels are utilized to calculate link features for each user in V^U (step 7). Step 8 then learns a classifier M_{AR} based on all of the attributes and labels. Step 10 utilizes M_{AR} to predict unknown labels. Finally, Step 11 returns the predicted results.

Algorithm 1. ICA-RST

Input: V = users, E = links, $\mathcal{X}$ = attribute set, Y^K = labels of known users ($Y^K = \{y_i|u_i \in V^K\}$)
Output: Y^U = labels of unknown users ($Y^U = \{y_i|u_i \in V^U$, $V^U = V - V^K\}$)

1 $M_A = learn\_RST\_Rule(V^K, Y^K)$; // learn classifier M_A utilizing only attributes
2 **for** *each user* $u_i \in V^U$ **do**
3 $y_i \leftarrow M_A(\vec{X}_i)$; // predict the labels of the unknown users utilizing M_A
4 **for** $t = 1$ *to* n **do**
5 $Y^R \leftarrow Y^K \cup \{y_i|u_i \in V^U\}$; // store the known labels and the predicted labels in set Y^R
6 **for** *each user* $u_i \in V^U$ **do**
7 $\vec{f}_i = calReFeats(V, E, Y^R)$; // calculate link features utilizing known labels and the predicted labels
8 $M_{AR} = learn\_RST\_Rule(V, Y^R)$; // learn classifier M_{AR} utilizing all of the attributes and labels
9 **for** *each user* $u_i \in V^U$ **do**
10 $y_i = M_{AR}(\vec{X}_i, \vec{f}_i)$; // re-predict the unknown labels utilizing M_{AR}
11 return Y^U

Fig. 1 shows an example for ICA-RST, which is applied to political view inference attacks. Each step in Fig. 1 displays a social graph consisting of five users with the corresponding friendship links. The class label of each node is y_i, which takes value from label set $Y = \{Con, Lib\}$, representing *conservative party* and *liberal party*, respectively. Four users have unknown labels ($V^U = \{u_2, u_3, u_4, u_5\}$) and only one user has known labels ($V^K = \{u_1\}$). In step A, no labels y_i and link features $\vec{f}_i$ in V^U have been predicted, so they are marked with a question mark. In step B, attribute-based classifier M_A assigns a label to u_i in V^U using only attribute $\vec{X}_i$. Based on the predicted labels in step B, step C then computes the link features of each u_i. For instance, $\vec{f}_3 = \{3, 1\}$ in step C since u_3 has three links with label *Co* (i.e., u_1, u_2, u_5) and one link with label *La* (i.e., u_4). In step D, classifier M_{AR} reclassifies users in V^U using the attributes and link features, and it recomputes the link features. Repeat step D and step E until the labels of u_i in V^U converge to a stable state.

6 Hiding Sensitive Information

The existing privacy preservation techniques, such as differential privacy [24], k-anonymity [25], l-diversity [26] and so forth, are designed for inherent-data privacy only, and do not protect against inference attacks directly. For instance, differential privacy ensures that the aggregation results of a data set that operates the differential privacy algorithms are the same with or without one row. k-anonymity guarantees that third party users cannot distinguish real data from at least their nearest $k-1$ neighbors. Since our goal is to release social network data while preserving data utility and protecting against inference attacks, the above techniques are not competent.

To develop a sanitization strategy, there are three issues to be addressed concerning inference attacks. First, we should understand the relationship between sensitive attributes and the released data set. For instance, *Bryden* made Facebook analysis and found that conservatives with distinguished cultural tastes than other partisans [31]. Second, it is necessary to figure out which attribute or link manipulating method(s) should be carried out to achieve

the desired privacy-utility tradeoff. For example, we can add or modify an attribute or a link to add noises to the released data. Also, we can remove some attributes and links to anonymize the released data. However, which one of the above methods are better? Last, for a specific manipulating method, how to effectively carry it out to achieve the desired privacy-utility tradeoff? For example, which attributes and links should be removed to markedly decrease the prediction accuracy on sensitive attributes while resulting in less accuracy decrease on non-sensitive attributes. In the following, we address these three issues.

6.1 Choosing Attributes to Manipulate

One of the most significant aspects is the dependency relationships between non-sensitive attributes and sensitive attributes. Through analyzing dependence relationships, we can reveal which publicly available attributes dominate the prediction results on sensitive attributes. Namely, dependency relationship provides the theoretical basis to determine which attributes should be chosen to manipulate. For example, suppose *political view* depends on *activity* and *favorite movies*, which indicates that we can manipulate these two attributes to reduce the prediction accuracy on *political view*. We denote the attributes that dominate the classification results on sensitive attributes as privacy-dependent attributes.

As shown in Definition 4.4 in Section 4.1, given an arbitrary information system $\Gamma = (V, A = C \cup D)$, any decision attribute set $D' \subseteq D$ depending on condition attribute set $C' \in C$ with degree k can be calculated as $C' \rightarrow^k D'$. Here, C and D can be viewed as publicly available attributes and sensitive attributes, respectively.

To hide sensitive attributes, our idea is to manipulate the most dependent attributes with respect to each sensitive attribute: for an arbitrary user u_i with attribute set $\vec{X}_i$, and a sensitive attribute $x_j = \{h_r : l_t\}$, we can find the most dependent attribute $x_s \in C$ $(1 \leq s \leq |C|)$ for sensitive attribute x_j based on the following:

$$\text{argmax}_s\{k \mid x_s \rightarrow^k l_t\}.$$

In practice, we can find any n_t-most dependent attributes for sensitive attribute with attribute value l_t, after extracting the attributes with the largest n_t dependence degree.

However, simply manipulating privacy-dependent attributes may result in utility reduction if utility is not considered. Consider the scenario that IMDb makes use of the data released by Facebook to suggest proper movies and TV programs to target users. It may classify users considering different movie types to make recommendations, depending on users' attribute sets. However, movie types could also depend on a privacy-dependent attribute. For example, the possible movie types are closely related to the attribute of "favorite movies".

We denote the attributes dominating the classification results on non-sensitive attributes as utility-dependent attributes. Then the following statement determines our choice.

Problem 6.1. *Given social graph $G(V, E, \mathcal{X} = C \cup D)$ with publicly available attribute set C and sensitive attribute set D, determine the set of attributes $C' \subseteq C$ so that $G'(V, E, C' \cup D)$ has the most decrease in prediction accuracy in D, while preserving the utility of C.*

Hence, the double dependency relationships become a challenge for the attribute manipulating method.

6.2 Attribute Manipulating Method

Obviously, attributes can be manipulated in three manners: *adding* new attributes, *removing* existing attributes, and *perturbing* (substitute one attribute to another). Since both *adding* and *perturbing* decrease prediction accuracy on sensitive information by introducing different types of noises, they are collectively called the obfuscation method. *Removing*, however, can be viewed as an anonymization method. Taking which manipulating method(s) depends on data semantics, privacy and utility metrics and so on. For example, if users specify a set of attributes as sensitive and quantify utility as the expected number of released attributes, the *removing* method could be advisable.

Suppose we just release our data to the public and do not announce what the data is used for. For example, social graph G is released online for research purpose and x_p is a privacy-dependent attribute of G. In this case, we have no direct measurement to determine how to perturb x_p; namely, use what attribute to substitute x_p, since no applications are specified. In this case, the removing method may be a proper choice. We just need to remove the privacy-dependent attributes.

For example, consider two social graphs G_1 and G_2, which are sanitized graphs of G after applying the obfuscation and anonymization methods, respectively. When we consider G_1 in which there is an attribute "favorite movies: Titanic", based on the employed obfuscation method, the original attribute set may not have this attribute or have an obsoletely distinct one. Hence, utility cannot be guaranteed by an obfuscation method when the application is not specified.

However, if the data are released for a special purpose such as movie recommendation, we could evaluate the changing utility when manipulating the attributes. In this case, the perturbing method could be a proper choice since properly perturbing can guarantee the desired privacy-utility tradeoff. For example, when we consider G_2, it may sacrifice much utility if there exists intersection between privacy-dependent attributes and utility-dependent attributes. Due to these observations, we consider *removing* and *perturbing* separately.

6.3 Link Manipulating Method

Another option for protecting against inference attacks is to manipulate links. Unlike attribute, link manipulating methods only add new links and remove existing links. With the same reason, we only consider the link anonymization method in the case of releasing the data set to the public and without announcing what the data are used for. With the same goal, the manipulated links should reduce the prediction accuracy on sensitive attributes. Suppose that adding or removing a link renders the prediction results on sensitive attributes locating in each class with a same probability, and we call this link as *indistinguishable link*, which is formally defined as follows:

Definition 6.1 (Δ'-Indistinguishable Link). *Given social graph $G(V, E, \mathcal{X})$ and an arbitrary $u_i \in V$ with possible class labels $Y = \{y_1, y_2, \ldots\}$, and $P\{y_t^i\}$ is the probability of u_i with label y_t. Any link $f_j \in F_{i,j}$ is an indistinguishable link of u_i if*

removing f_j results that

$$Var\{P\{y_1^i\}, P\{y_2^i\}, \ldots, P\{y_{|Y|}^i\}\} \leq \Delta', \quad (4)$$

where $Var(S)$ is for valuating the variance of set S.

To hide sensitive attributes through removing links, our idea is to manipulate the most indistinguishable link with respect to each user. We can find the most indistinguishable link f_j for u_i based on the following:

$$\text{argmin}_j\{Var\{P\{y_1^i\}, P\{y_2^i\}, \ldots, P\{y_{|Y|}^i\}\} \mid \text{removing } f_j\}$$

7 Collective Method

To protect against inference attacks, we attempt to manipulate attributes by perturbing and removing separately in the respective situations. As mentioned in Section 6.2, these two methods must be restricted by the utility requirements. In this section, in order to achieve the desired privacy-utility tradeoff, we present how to utilize removing and perturbing in a collective manner.

Clearly, simply removing or perturbing Privacy-Dependent Attributes (PDAs) could reduce prediction accuracy on non-sensitive attributes. Hence, there should exist a compromise strategy for manipulating the PDAs to achieve the privacy-utility tradeoff. Therefore, rather than removing or perturbing PDAs directly, we analyze the relationship between PDAs and Utility-Dependent Attributes (UDAs) first.

For simplicity, we have the following collective method:

Algorithm 2. Collective Method

Input: G, PDAs, UDAs
Output: collective method
1: **if** PDAs $\cap$ UDAs = Φ **then**
2: **removing** PDAs;
3: **else then**
4: **removing** PDAs - *Core*;
5: **perturbing** *Core*

Algorithm 2 shows that if there are no shared attributes between PDAs and UDAs, we just need to remove the PDAs since they have no contributions on utility (Step 2). Conversely, with the same reason, we remove the difference set between PDAs and the shared attribute set *Core* (step 4). For the shared attributes, perturbing them to optimize the privacy-utility tradeoff (step 5).

The details of the perturbing method on *Core* in Algorithm 2 are presented in Section 7.1.

7.1 Perturbing

We formally define the shared attributes as a *Core*.

Definition 7.1 (Core). *Given an information system $\Gamma = (V, A = C \cup D), D = D_u \cup D_p$, where D_u and D_p are two decision attribute sets for utility and privacy, respectively. We say that $C' \subseteq C$ is a core of D_u and D_p if $C' \subseteq R_u$ and $C' \subseteq R_p$, where R_u and R_p are the reduct of C for $\Gamma = (V, A = C \cup D_u)$ and $\Gamma = (V, A = C \cup D_p)$, respectively.*

Our idea is to substitute each attribute in the *Core* with a generic attribute, which ensures that third party users cannot get specific information to increase prediction accuracy on sensitive attributes, while guarantees no significant accuracy reduction on data utility. Moreover, the higher level of generalization, the more preference to privacy for the utility-privacy tradeoff. Since there are different levels of generalization, the generic attributes can be organized into a hierarchy, which is formally defined as follows:

Definition 7.2 (Generic Attribute Hierarchy). *A Generic Attribute Hierarchy (GAH) is a finite hierarchical ordering. The first layer of the ordering is one of the privacy-dependent attributes, and each parent layer is a generic of the sublayer.*

Definition 7.2 indicates that the ancestor of the GAH is the highest level of generalization of initial attributes. Substituting one privacy-dependent attribute with the ancestor of the GAH would render the highest level of privacy. For example, if one attribute value in core is for category *favorite movies*, the corresponding GAH can be

$$\text{Star Wars} \rightarrow \text{Fantasy} \rightarrow \text{American film}.$$

This indicates that we can substitute original attribute "Star Wars" with "American film", in order to get the highest level of generalization. We could also substitute it with "Fantasy" to give more preference to utility for the utility-privacy tradeoff since "Fantasy" is more specific than "American film". Hence, GAH guarantees that we can programmatically determine which level of generic value should be chosen to optimize the privacy-utility tradeoff.

Algorithm 3 presents the generation process of the generic values for guaranteeing optimal ϵ-utility.

Algorithm 3. Generate Generic Value

Input: Core, ϵ = utility threshold
Output: GAH
1: **while** $\max_{c \in \mathcal{C}} \Lambda(G', \mathcal{K}, X_{non}) - \max_{c' \in \mathcal{C}} \Lambda(\mathcal{K}, X_{non}) \geq \epsilon$ **do**
2: further generate all the current attributes;
3: return *Perturbed Core*

8 Evaluation

8.1 Datasets

In our experiments, we investigate three different Facebook datasets. The first one is the SNAP Facebook dataset[1] which contains user friendships and a number of node attributes such as gender, birthday, position, employer, location, etc. The other two are the Facebook dataset containing all the Facebook friendships at Caltech and MIT in 2005, as well as a number of node attributes such as student/faculty status flag, gender, graduation year, academic major, etc.[2] For convenience, we denote these three datasets as SNAP, Caltech, and MIT, respectively. In Caltech and MIT, each attribute is specified by a numeric value and each of which indicates a corresponding attribute. However, in SNAP, each attribute is specified by a 0/1 value and each of which indicates the absence/presence of the corresponding attribute. For example, attribute "EducationDegree: undergraduate; master;

1. https://snap.stanford.edu/data/egonets-Facebook.html
2. http://www.michaelzimmer.org/2011/02/15/facebook-data-of-1-2-million-users-from-2005-released/

TABLE 3
General Statistics About the Three Datasets

Network property	SNAP	Caltech	MIT
Number of nodes	792	769	6440
Number of friendship links	14,024	16,656	251,252
Number of attributes for each user	20	7	7
Number of values for decision attribute	2	4	7
Number of components in the graph	10	4	18
Nodes in largest connected component	775	762	6,402
Edges in largest connected component	14,006	16,651	251,230
Diameter longest shortest path	10	6	8

PHD" with attribute value 010 means that the attribute value is master. For convenience, we map each attribute in SNAP into an unique numeric value in each attribute category. For example, the above attribute value 010 in Education degree is mapped to 2.

In Table 3, some general statistics about the three datasets are provided. It shows that all of the three graphs are almost fully connected.

8.2 Experiment Settings

In our experiments, we regard *gender* in SANP and *student/faculty status flag* (*flag* for short) in Caltech and MIT as sensitive attributes.

Table 3 shows that there are 2, 4, and 7 attribute values in SNAP, Caltech and MIT, respectively, which are regarded as class labels here.

We predict a sensitive attribute with the following attack models: 1) the attack model with absence of link information (AttrOnly), 2) the attack model with absence of attribute information (LinkOnly), and 3) the attack model based on collective inference (CC).

As mentioned in Section 5, a major issue is raised if directly executing LinkOnly requires that at least one of the neighbors of each unlabeled user locates in the training set (as shown in Equation (3)). Hence, in our experiments, we first predict the class label of those unlabeled nodes by classifying their attribute sets. Next, we predict the class label of any user u_i by calculating the weighted average probability of its neighbors with one class label (as calculated in Equation (3)).

Moreover, CC employs attribute based classifier to predict the approximate class label at the bootstrap stage. Then, it uses classifier that based on both attribute and link, M_{AR}, to refine the results. In our experiments, we employ the following M_{AR}

$$\alpha P_A\{y_t^i\} + \beta P_L\{y_t^i\}, \tag{5}$$

where $P\{y_t^i\}$ and $P_L\{y_t^i\}$ are the probabilities of u_i with label y_t, assigned by AttrOnly and LinkOnly, respectively. $\alpha + \beta = 1$, where α and β represent the ratio of AttrOnly and LinkOnly, respectively. The values of α and β are determined by dataset features. Specifically, α is larger than β iff

TABLE 4
Information of the Reduct Systems for SNAP, Caltech and MIT

Decision attribute	No. of condition attributes
Gender in SNAP	$19 \rightarrow 13$
Flag in Caltech	$6 \rightarrow 5$
Flag in MIT	$6 \rightarrow 5$

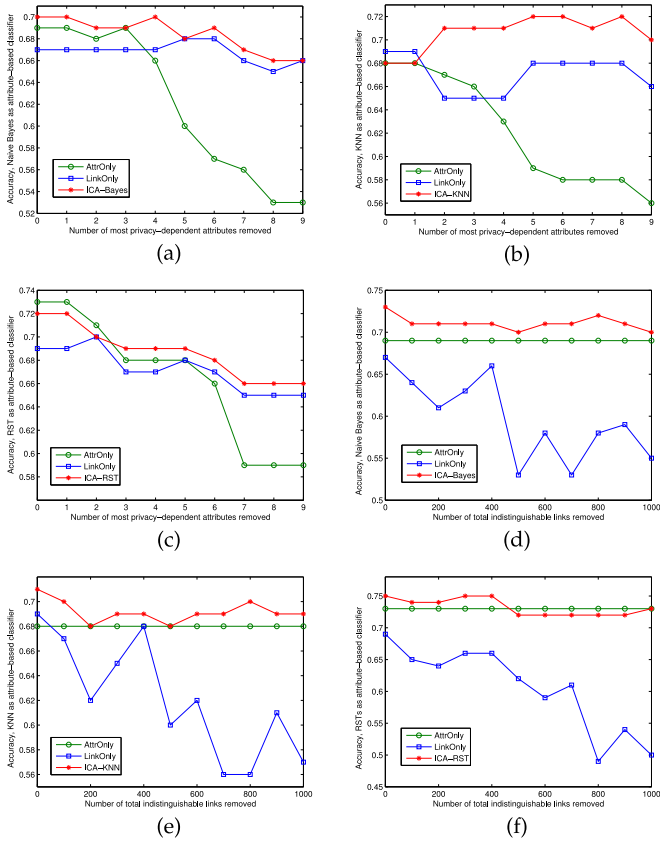

Fig. 2. Sensitive attribute prediction accuracy on SNAP with different attack models. With most privacy-dependent attributes removed, and (a) Bayes, (b) KNN, (c) RST as attribute-based classifier; With indistinguishable links removed, and (d) Bayes, (e) KNN, (f) RST as attribute-based classifier.

the node attributes are more indicative than node relations. To determine α and β, we study a set of experiments with multiple combinations and find the optimal one that renders the best prediction accuracy for CC. In Section 8.3, we set both α and β as 0.5; namely, an average prediction result assigned by AttrOnly and LinkOnly is expected. In Section 8.4, the utility and privacy under several pairs of α and β would be discussed.

For the attribute-based classifier utilized in AttrOnly, LinkOnly and CC, we carry it out with three techniques: RST, Navie Bayes and KNN [32]. Hence, with different attribute-based classifiers, AttrOnly can be further specified as: 1) RST, 2) Navie Bayes, 3) KNN; LinkOnly can be further specified as: 4) LinkOnly-RST, 5) LinkOnly-Bayes, 6) LinkOnly-KNN; and CC can be further specified as: 7) ICA-RST, 8) ICA-Bayes, 9) ICA-KNN.

8.3 Effect of Attribute-Removal and Link-Removal Methods on Inference Attacks

In this part, we aim to protect against inference attacks with the following sanitization methods: 1) Attribute removal: remove the most privacy dependent attributes, namely, the attributes in the reduct system (Section 6.1), and 2) Link removal: remove the distinguishable links (Section 6.3). To evaluate the effectiveness of these two data sanitization methods, we take prediction accuracy as the criterion.

Table 4 lists the information of the reduct systems for these three datasets. We can see that in Table 4, the number

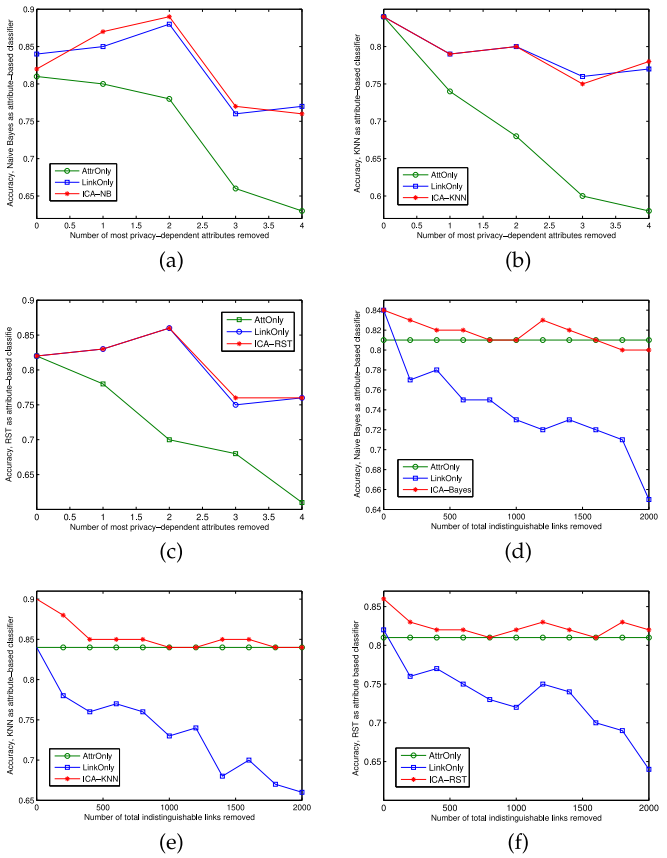

Fig. 3. Sensitive attribute prediction accuracy on Caltech with different attack models. With most privacy-dependent attributes removed, and (a) Bayes, (b) KNN, (c) RST as attribute-based classifier; With indistinguishable links removed, and (d) Bayes, (e) KNN, (f) RST as attribute-based classifier.

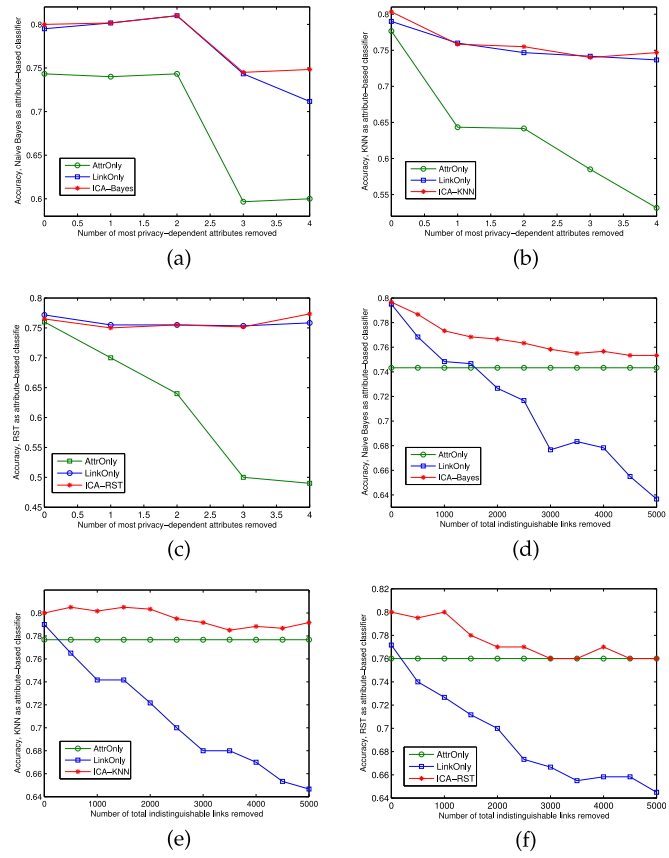

Fig. 4. Sensitive attribute prediction accuracy on MIT with different attack models. With most privacy-dependent attributes removed, and (a) Bayes, (b) KNN, (c) RST as attribute-based classifier; With indistinguishable links removed, and (d) Bayes, (e) KNN, (f) RST as attribute-based classifier.

of condition attributes is reduced from 19 to 13 in SNAP and from 6 to 5 in Caltech and MIT, respectively.

8.3.1 SNAP

Figs. 2a, 2b, and 2c show the prediction accuracy of different attack models on SNAP with the removal of the most privacy dependent attributes. Figs. 2d, 2e, and 2f show the prediction accuracy of different attack models on SNAP with the removal of the indistinguishable links. As we can see from Figs. 2a, 2b, and 2c, removing the most privacy dependent attributes is generally successful in reducing the prediction accuracy on sensitive attributes.

It is shown that there is a decrease in the prediction accuracy with more and more attributes being removed. Surprisingly, however, the accuracy of LinkOnly does not decrease significantly while we remove attributes. For LinkOnly, as discussed in Section 8.2, we first predict the class labels of those unlabeled nodes by classifying their attribute sets; hence, the accuracy decrease of attribute-based classifier should also render the accuracy decrease for LinkOnly. A possible explanation is that just a small part of the nodes need labels in the first step of LinkOnly through classifying attribute set, since most of the labeled nodes are in the training set. Hence, removing attributes do not have a significant influence. Clearly, we can see that CC generally outperforms AttrOnly and LinkOnly.

The results in Figs. 2d, 2e, and 2f show that removing the indistinguishable links is generally successful in reducing the prediction accuracy on sensitive attributes. However, we find a surprising phenomena in Fig. 2: a volatile prediction accuracy after the removal of a single attribute or link. Especially, a much more volatile prediction accuracy after the removal of a single link. For the volatility related to attribute, it is a result of large class size difference in the SNAP dataset. Since approximately 65 percent of the nodes in SNAP are "male" and there are no attributes that are highly dependent on gender, a small change of attributes can affect the prediction accuracy in uncontrollable ways. For the volatility related to links, it is a result of the local optimal link-removal strategy. Since the link-removal strategy always manipulates the most indistinguishable link with respect to each user, it cannot guarantee the removed link is globally optimal. Therefore, a small change of links can also affect the prediction accuracy in uncontrollable ways.

8.3.2 Caltech

Figs. 3a, 3b, and 3c show the prediction accuracy of different attack models on Caltech with the most privacy dependent attributes removed. Figs. 3d, 3e, and 3f show the prediction accuracy of different attack models on Caltech with the removal of the indistinguishable links. As we can see from Figs. 3a, 3b, and 3c, compared with the results on SNAP, there is a much more volatile prediction accuracy after the removal of a single attribute. This is a result of larger class size difference in Caltech than that of SNAP. Since approximately 72 percent of the nodes in Caltech have a same class

TABLE 5
Setting of Utility Attribute and Privacy Attribute

	Utility attribute	Privacy attribute
SNAP	education type	gender
Caltech	gender	flag
MIT	gender	flag

TABLE 6
Information for PDAs, UDAs and Core

Dataset	No. of UDAs	No. of PDAs - Core	No. of Core
SNAP	7	6	6
Catech	3	2	1
MIT	3	2	1

TABLE 7
Maximum Utility/Privacy Under Collective, Attribute Removal and Link Removal Methods with $\alpha = 0.5$, $\beta = 0.5$

Dataset	Collective	Attribute removal	Link removal
SNAP	1.1967	1.1639	1.1639
Caltech	1.5273	1.3433	1.3433
MIT	1.2636	1.1881	1.1931

label and there are no attributes that are highly dependent on flag, a small change of attributes can affect the prediction accuracy in uncontrollable ways.

8.3.3 MIT

Figs. 4a, 4b, and 4c show the prediction accuracy of different attack models on MIT with the most privacy dependent attributes removed. Figs. 4d, 4e, and 4f show the prediction accuracy of different attack models on MIT with the removal of the indistinguishable links. Fig. 4 shows that removing the most privacy dependent attributes or indistinguishable links is generally successful in reducing the prediction accuracy on sensitive attributes. As we can see from Figs. 4a, 4b, and 4c, compared with the results on Caltech, there is a less volatile prediction accuracy after the removal of a single attribute. This appears to be a result of larger class size difference in the Caltech dataset than that of the MIT. Approximately 67 percent of the nodes in MIT have a same class label and there are no attributes that are highly dependent of flag.

Figs. 1, 2, and 3 in the supplementary file, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TDSC.2016.2613521, show the prediction accuracy of different attack models on SNAP, Caltech and MIT, respectively, with the most privacy dependent attributes and indistinguishable links removed simultaneously. It shows that for SNAP, Caltech and MIT, the prediction accuracy is more sensitive to the removal of attributes than the removal of links. For example, as shown in Fig. 3a, the predication accuracy decreases from 0.8 to 0.7916 when indistinguishable links are removed. Comparatively, the predication accuracy decreases from 0.8 to 0.5667 when the most privacy dependent attributes are removed.

8.4 Effect of Collective Method on Inference Attacks

We further test the collective method to evaluate the effectiveness of our data sanitization. Since there are no utility and privacy expectation specified for each dataset, we select two attributes as privacy attribute and utility attribute, respectively. The selection of the above two attributes is listed in Table 5. We attempt to evaluate the effectiveness of our method in achieving a desired privacy/utility tradeoff: reducing the prediction accuracy on sensitive attribute while ensuring the prediction accuracy on utility atteibute.

TABLE 8
General Statistics About Priacy/Utility on SNAP with $\alpha = 0.5$, $\beta = 0.5$

L	Uti/pri	No. of R-Attr	Uti/pri	No. of R-Link	Uti/pri
5	1.1613	0	1.1639	0	1.1639
6	1.1803	3	1.0862	200	1.1500
7	1.1967	6	0.9524	400	1.1333
8	1.1967	9	0.9375	600	1.1148

TABLE 9
General Statistics About Priacy/Utility on Caltech with $\alpha = 0.5$, $\beta = 0.5$

L	Uti/pri	No. of R-Attr	Uti/pri	No. of R-Link	Uti/pri
5	1.4839	0	1.3433	0	1.3433
6	1.4918	1	1.1970	400	1.2464
7	1.5112	2	1.0274	800	1.2206
8	1.5273	3	0.9865	1,200	1.1690

Since each attribute has a numeric value, we cannot generate a generic value from the semantic view directly. However, we can map several attribute values to an interval and generalize them with an unique value in this interval. Algorithm 4 is used to generate generic attribute values. For each attribute category h_r in *Core*, Algorithm 4 first calculates the maximum and minimum attribute values of all the users for h_r (steps 2-3). Then, it calculates the range between MAX and MIN under generic level L (step 4). Finally, for each user i, Algorithm 4 maps its original attribute value $x_{i,r}$ to $\lfloor (X_{i,r} - MIN)/Range \rfloor$ (steps 5-7). In Algorithm 4, perturbing degree decreases with the increase of generalization level L.

Algorithm 4. Generate Generic Value

Input: *Core*, L = generalization level
Output: Generic attribute set with level L
1: **for** *each attribute category* $h_r \in Core$ **do**
2: $MAX_r = \max(x_{1,r}, x_{1,r}, \ldots, x_{|V|,r})$;
3: $MIN_r = \min(x_{1,r}, x_{1,r}, \ldots, x_{|V|,r})$;
4: $Range_r = \lfloor (MAX_r - MIN_r)/L + 1 \rfloor$;
5: **for** $i = 1$ to $|V|$ **do**
6: $x_{i,r} = \lfloor (x_{i,r} - MIN)/Range \rfloor$;
7: return $x_{i,r}$

According to Algorithm 2, the information for PDAs, UDAs and Core for SNAP, Caltech and MIT are shown in Table 6.

We test multiple levels of generalization (set generalization level as $L = 5, 6, 7, 8$) and compare the collective method with the data removal and link removal sanitization methods. We use utility/privacy as privacy-utility tradeoff criteria to evaluate the performance of these three data-sanitization methods.

Table 7 shows the maximum utility/privacy under these three methods, with $\alpha = 0.5$ and $\beta = 0.5$. From Table 7, we

TABLE 10
General Statistics About Priacy/Utility on MIT with $\alpha = 0.5$, $\beta = 0.5$

L	Uti/pri	No. of R-Attr	Uti/pri	No. of R-link	Uti/pri
5	1.2313	0	1.1881	300	1.1931
6	1.2425	1	1.0469	600	1.1901
7	1.2580	2	1.0342	900	1.1897
8	1.2636	3	0.9698	1,200	1.1798

observe that the collective method achieves the best privacy/utility tradeoff with ratio 1.1967, 1.5273 and 1.2636 in SNAP, Caltech and MIT, respectively. Tables 8, 9, and 10 show the utility/privacy under different generalization levels, and different numbers of removed attributes and links. In Tables 8, 9, and 10, "R-Attr", "R-Link" and "Uti/pri" represent "Number of Removed attribute", "Number of Removed link" and "Utility/privacy", respectively. As shown in Tables 8, 9, and 10, utility to privacy ratio decreases with the increase of perturbing degree (L from 5 to 8). Moreover, utility to privacy ratio decreases as well with more and more attributes and links being removed. Additionally, we observe that our proposed collective method generally outperforms attribute removal and link removal method.

Furthermore, we evaluate the maximum utility/privacy under different combinations of α and β: $\alpha = 0.1$, $\beta = 0.9$ and $\alpha = 0.9$, $\beta = 0.1$. The results are shown in Tables 1 and 2 all both in supplementary, available online. Table 7, Tables 1 and 2 in supplementary, available online, show that utility/privacy value of collective method is always better than that of attribute removal and link removal method, when an average prediction result are assigned by AttrOnly and LinkOnly, i.e., $\alpha = 0.5$ and $\beta = 0.5$.

9 Conclusion

We address two issues in this paper: (a) how exactly third party users launch an inference attack to predict sensitive information of users, and (b) are there effective strategies to protect against such an attack to achieve a desired privacy-utility tradeoff. For the first issue, we show that collectively utilizing both attribute and link information can significantly increase prediction accuracy for sensitive information. For the second issue, we explore the dependence relationships for utility/public attributes, and privacy/public attributes. Based on these results, we propose a Collective Method that take advantages of various data manipulating methods to guarantee sanitizing user data does not incur a bad impact on data utility. Using Collective Method, we are able to effectively sanitize social network data prior to release. The solutions for the two addressed issues are proven to be effective towards three real social datasets.

Acknowledgments

This work is partly supported by the US National Science Foundation under grant CNS-1252292. Zhipeng Cai is the corresponding author.

References

[1] (2016). [Online]. Available: https://www.researchgate.net/
[2] (2016). [Online]. Available: http://www.imdb.com/
[3] "Facebook beacon," http://www.nydailynews.com/life-style/gaydar-project-mit-attempts-predict-sexuality-based-facebook-profiles-article-1.404453, 2007.
[4] Z. He, Z. Cai, and Y. Li, "Customized privacy preserving for classification based applications," in *Proc. 1st ACM Workshop Privacy-Aware Mobile Comput.*, 2016, pp. 37–42.
[5] K. Heussner, "'Gaydar' on facebook: Can your friends reveal sexual orientation?" ABC News, New York, NY, USA, vol. 14, no. 10, p. 2, 2009.
[6] C. Johnson, "Gaydar," The Boston Blobe, Boston, MA, USA, 2009.
[7] (2013). [Online]. Available: http://www.pewinternet.org/2013/05/21/teens-social-media-and-privacy/
[8] S. Nilizadeh, A. Kapadia, and Y.-Y. Ahn, "Community-enhanced de-anonymization of online social networks," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, 2014, pp. 537–548.
[9] A. Narayanan and V. Shmatikov, "De-anonymizing social networks," in *Proc. 30th IEEE Symp. Security Privacy*, 2009, pp. 173–187.
[10] L. Backstrom, C. Dwork, and J. Kleinberg, "Wherefore art thou r3579x?: Anonymized social networks, hidden patterns, and structural steganography," in *Proc. 16th Int. Conf. World Wide Web*, 2007, pp. 181–190.
[11] B. Zhou, J. Pei, and W. Luk, "A brief survey on anonymization techniques for privacy preserving publishing of social network data," *ACM SIGKDD Explorations Newslett.*, vol. 10, no. 2, pp. 12–22, Dec. 2008.
[12] A. Mislove, B. Viswanath, K. P. Gummadi, and P. Druschel, "You are who you know: Inferring user profiles in online social networks," in *Proc. 3rd ACM Int. Conf. Web Search Data Mining*, 2010, pp. 251–260.
[13] E. Ryu, Y. Rong, J. Li, and A. Machanavajjhala, "Curso: Protect yourself from curse of attribute inference: A social network privacy-analyzer," in *Proc. ACM SIGMOD Workshop Databases Social Netw.*, 2013, pp. 13–18.
[14] J. He, W. W. Chu, and Z. V. Liu, "Inferring privacy information from social networks," in *Proc. 4th IEEE Int. Conf. Intell. Security Informat.*, 2006, pp. 154–165.
[15] Y. Dong, Y. Yang, J. Tang, Y. Yang, and N. V. Chawla, "Inferring user demographics and social strategies in mobile social networks," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2014, pp. 15–24.
[16] J. Song, S. Lee, and J. Kim, "Inference attack on browsing history of twitter users using public click analytics and twitter metadata," *IEEE Trans. Dependable Secure Comput.*, vol. 13, no. 3, pp. 340–354, May/Jun. 2014.
[17] Z. Jorgensen, T. Yu, and G. Cormode, "Publishing attributed social graphs with formal privacy guarantees," in *Proc. Int. Conf. Manage. Data*, 2016, pp. 107–122.
[18] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao, "PrivBayes: Private data release via bayesian networks," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2014, pp. 1423–1434.
[19] C. Liu and P. Mittal, "Linkmirage: Enabling privacy-preserving analytics on social relationships," in *Proc. 23nd Annu. Netw. Distrib. Syst. Security Symp.*, Feb. 21-24, 2016.
[20] W.-Y. Day, N. Li, and M. Lyu, "Publishing graph degree distribution with node differential privacy," *Proc. Int. Conf. Manage. Data*, 2016, pp. 123–138.
[21] P. Gundecha, G. Barbier, J. Tang, and H. Liu, "User vulnerability and its reduction on a social networking site," *ACM Trans. Knowl. Discovery Data*, vol. 9, no. 2, pp. 12:1–12:25, Sep. 2014.
[22] A. Friedman, I. Sharfman, D. Keren, and A. Schuster, "Privacy-preserving distributed stream monitoring," in *Proc. Netw. Distrib. Syst. Security Symp.*, 2014.
[23] Q. Li, G. Cao, and T. F. L. Porta, "Efficient and privacy-aware data aggregation in mobile sensing," *IEEE Trans. Dependable Secure Comput.*, vol. 11, no. 2, pp. 115–129, Mar. 2014.
[24] C. Dwork, "Differential privacy," in *Automata, Languages and Programming*, M. Bugliesi, B. Preneel, V. Sassone, and I. Wegener, Eds. Berlin, Germany: Springer, 2006, pp. 1–12.
[25] L. Sweeney, "K-anonymity: A model for protecting privacy," *Int. J. Uncertainty Fuzziness Knowl.-Based Syst.*, vol. 10, no. 5, pp. 557–570, Oct. 2002.
[26] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam, "L-diversity: Privacy beyond k-anonymity," *ACM Trans. Knowl. Discovery Data*, vol. 1, no. 1, Mar. 2007, Art. no. 3.
[27] Z. Pawlak, "Rough set theory and its applications to data analysis," *Cybern. Syst.*, vol. 29, no. 7, pp. 661–688, 1998.

[28] S. A. Macskassy and F. Provost, "Classification in networked data: A toolkit and a univariate case study," *J. Mach. Learn. Res.*, vol. 8, pp. 935–983, May 2007.
[29] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI Mag.*, vol. 29, no. 3, 2008, Art. no. 93.
[30] D. Jensen, J. Neville, and B. Gallagher, "Why collective inference improves relational classification," in *Proc. 10th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2004, pp. 593–598.
[31] (2015). [Online]. Available: http://www.cbc.ca/1.3154617
[32] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques*, 3rd ed. San Francisco, CA, USA: Morgan Kaufmann, 2011.

Zhipeng Cai received the BS degree from Beijing Institute of Technology and the MS and PhD degrees from the Department of Computing Science, University of Alberta. He is currently an assistant professor in the Department of Computer Science, Georgia State University. His research areas focus on networking, privacy and big data. He received an NSF CAREER Award. He served as the program chairs for WASA 2014, COCOON 2014, IPCCC 2013 and ISBRA 2013 and vice general chair for IPCCC 2014. He is now a steering committee co-chair of the International Conference on Wireless Algorithms, Systems, and Applications (WASA). He is an editor/guest editor of the *Algorithmica*, the *Theoretical Computer Science*, the *Journal of Combinatorial Optimization*, the *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, and the *International Journal of Sensor Networks*. He is a senior member of the IEEE.

Zaobo He received the MS degree in computer science from Shannxi Normal University, China, in 2013. He is working towrad the PhD degree in the Department of Computer Science, Georgia State University. His research interests include privacy preservation, social networking, and big data. He is a student member of the IEEE.

Xin Guan received the bachelor's degree from the School of Computer Science and Technology, Heilongjiang University, China, in 2001, the master's degree from Harbin Institute of Technology, China, in 2007, and the PhD degree from the Graduate School of Science and Technology, Keio University, Japan, in 2012. His research interests include topology control, performance evaluation and routing algorithm in wireless networks. Currently, he is an associate professor with Heilongjiang University, China. He served as a technical committee member of the IEEE-GLOBECOM, PIMRC, HPCC, IWCMC and so on. He served as a reviewer of *IEEE-Network*, the *KSII Transactions on Internet and Information Systems*, the *EURASIP Journal on Wireless Communications and Networking*, the *Security and Communication Networks* (Wiley), the *International Journal of Ad Hoc and Ubiquitous Computing*, the *ACM/Springer Mobile Networks & Applications (MONET)*, and so on. He is a member of the IEEE.

Yingshu Li received the BS degree from the Department of Computer Science and Engineering, Beijing Institute of Technology, China, and the MS and PhD degrees from the Department of Computer Science and Engineering, University of Minnesota-Twin Cities. She is currently an associate professor in the Department of Computer Science, Georgia State University. Her research interests include wireless networking, sensor networks, sensory data management, social networks, and optimization. She received an NSF CAREER Award. She is a senior member of the IEEE.