Sensitive Label Privacy Protection on Social Network Data

Yi Song*, Panagiotis Karras◊, Qian Xiao*, and Stéphane Bressan*

*School of Computing
National University of Singapore
{songyi, xiaoqian, steph}@nus.edu.sg
◊Rutgers Business School
Rutgers University
karras@business.rutgers.edu

Abstract. This paper is motivated by the recognition of the need for a finer grain and more personalized privacy in data publication of social networks. We propose a privacy protection scheme that not only prevents the disclosure of identity of users but also the disclosure of selected features in users' profiles. An individual user can select which features of her profile she wishes to conceal. The social networks are modeled as graphs in which users are nodes and features are labels. Labels are denoted either as sensitive or as non-sensitive. We treat node labels *both* as background knowledge an adversary may possess, *and* as sensitive information that has to be protected. We present privacy protection algorithms that allow for graph data to be published in a form such that an adversary who possesses information about a node's neighborhood cannot safely infer its identity and its sensitive labels. To this aim, the algorithms transform the original graph into a graph in which nodes are sufficiently indistinguishable. The algorithms are designed to do so while losing as little information and while preserving as much utility as possible. We evaluate empirically the extent to which the algorithms preserve the original graph's structure and properties. We show that our solution is effective, efficient and scalable while offering stronger privacy guarantees than those in previous research.

1 Introduction

The publication of social network data entails a privacy threat for their users. Sensitive information about users of the social networks should be protected. The challenge is to devise methods to publish social network data in a form that affords utility without compromising privacy. Previous research has proposed various privacy models with the corresponding protection mechanisms that prevent both inadvertent private information leakage and attacks by malicious adversaries. These early privacy models are mostly concerned with identity and link disclosure. The social networks are modeled as graphs in which users are nodes and social connections are edges. The threat definitions and protection

mechanisms leverage structural properties of the graph. This paper is motivated by the recognition of the need for a finer grain and more personalized privacy.

Users entrust social networks such as Facebook and LinkedIn with a wealth of personal information such as their age, address, current location or political orientation. We refer to these details and messages as features in the user's profile. We propose a privacy protection scheme that not only prevents the disclosure of identity of users but also the disclosure of selected features in users' profiles. An individual user can select which features of her profile she wishes to conceal.

The social networks are modeled as graphs in which users are nodes and features are labels[1]. Labels are denoted either as sensitive or as non-sensitive. Figure 1 is a labeled graph representing a small subset of such a social network. Each node in the graph represents a user, and the edge between two nodes represents the fact that the two persons are friends. Labels annotated to the nodes show the locations of users. Each letter represents a city name as a label for each node. Some individuals do not mind their residence being known by the others, but some do, for various reasons. In such case, the privacy of their labels should be protected at data release. Therefore the locations are either sensitive (labels are in red italic in Figure 1[2]) or non-sensitive.

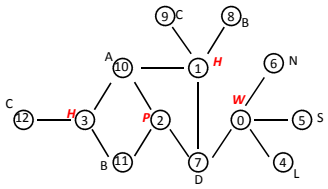


Fig. 1. Example of the labeled graph representing a social network

The privacy issue arises from the disclosure of sensitive labels. One might suggest that such labels should be simply deleted. Still, such a solution would present an incomplete view of the network and may hide interesting statistical information that does not threaten privacy. A more sophisticated approach consists in releasing information about sensitive labels, while ensuring that the identities of users are protected from privacy threats. We consider such threats as *neighborhood attack*, in which an adversary finds out sensitive information based on *prior knowledge* of the number of neighbors of a target node and the labels of these neighbors. In the example, if an adversary knows that a user has three friends and that these friends are in A (Alexandria), B (Berlin) and C (Copenhagen), respectively, then she can infer that the user is in H (Helsinki).

We present privacy protection algorithms that allow for graph data to be published in a form such that an adversary cannot safely infer the identity and

[1] Although modeling features in the profile as attribute-value pairs would be closer to the actual social network structure, it is without loss of generality that we consider atomic labels.

[2] W: Warsaw, H: Helsinki, P: Prague, D: Dublin, S:Stockholm, N: Nice, A: Alexandria, B: Berlin, C: Copenhagen, L: Lisbon

sensitive labels of users. We consider the case in which the adversary possesses both structural knowledge and label information.

The algorithms that we propose transform the original graph into a graph in which any node with a sensitive label is indistinguishable from at least $\ell-1$ other nodes. The probability to infer that any node has a certain sensitive label (we call such nodes *sensitive nodes*) is no larger than $1/\ell$. For this purpose we design ℓ-diversity-like model, where we treat node labels as *both* part of an adversary's background knowledge *and* as sensitive information that has to be protected.

The algorithms are designed to provide privacy protection while losing as little information and while preserving as much utility as possible. In view of the tradeoff between data privacy and utility [16], we evaluate empirically the extent to which the algorithms preserve the original graph's structure and properties such as density, degree distribution and clustering coefficient. We show that our solution is effective, efficient and scalable while offering stronger privacy guarantees than those in previous research, and that our algorithms scale well as data size grows.

The rest of the paper is organized as follows. Section 2 reviews previous works in the area. We define our problem in Section 3 and propose solutions in Section 4. Experiments and result analysis are described in Section 5. We conclude this work in Section 6.

2 Related Work

The first necessary anonymization technique in both the contexts of micro- and network data consists in removing identification. This nave technique has quickly been recognized as failing to protect privacy. For microdata, Sweeney et al. propose k-anonymity [17] to circumvent possible identity disclosure in naively anonymized microdata. ℓ-diversity is proposed in [13] in order to further prevent attribute disclosure.

Similarly for network data, Backstrom et al., in [2], show that naive anonymization is insufficient as the structure of the released graph may reveal the identity of the individuals corresponding to the nodes. Hay et al. [9] emphasize this problem and quantify the risk of re-identification by adversaries with external information that is formalized into structural queries (node refinement queries, subgraph knowledge queries). Recognizing the problem, several works [5, 11, 18, 20–22, 24, 27, 8, 4, 6] propose techniques that can be applied to the naive anonymized graph, further modifying the graph in order to provide certain privacy guarantee. Some works are based on graph models other than simple graph [12, 7, 10, 3].

To our knowledge, Zhou and Pei [25, 26] and Yuan et al. [23] were the first to consider modeling social networks as labeled graphs, similarly to what we consider in this paper. To prevent re-identification attacks by adversaries with immediate neighborhood structural knowledge, Zhou and Pei [25] propose a method that groups nodes and anonymizes the neighborhoods of nodes in the same group by generalizing node labels and adding edges. They enforce a *k-anonymity* privacy constraint on the graph, each node of which is guaranteed to have the same

immediate neighborhood structure with other $k-1$ nodes. In [26], they improve the privacy guarantee provided by *k-anonymity* with the idea of ℓ-diversity, to protect labels on nodes as well. Yuan et al. [23] try to be more practical by considering users' different privacy concerns. They divide privacy requirements into three levels, and suggest methods to generalize labels and modify structure corresponding to every privacy demand. Nevertheless, neither Zhou and Pei, nor Yuan et al. consider labels as a part of the background knowledge. However, in case adversaries hold label information, the methods of [25, 26, 23] cannot achieve the same privacy guarantee. Moreover, as with the context of microdata, a graph that satisfies a k-anonymity privacy guarantee may still leak sensitive information regarding its labels [13].

3 Problem Definition

We model a network as $G(V, E, L^s, L, \Gamma)$, where V is a set of nodes, E is s set of edges, L^s is a set of sensitive labels, and L is a set of non-sensitive labels. Γ maps nodes to their labels, $\Gamma : V \rightarrow L^s \cup L$. Then we propose a privacy model, *ℓ-sensitive-label-diversity*; in this model, we treat node labels *both* as part of an adversary's background knowledge, *and* as sensitive information that has to be protected. These concepts are clarified by the following definitions:

Definition 1. *The **neighborhood information** of node v comprises the degree of v and the labels of v's neighbors.*

Definition 2. *(ℓ-**sensitive-label-diversity**) For each node v that associates with a sensitive label, there must be at least $\ell - 1$ other nodes with the same neighborhood information, but attached with different sensitive labels.*

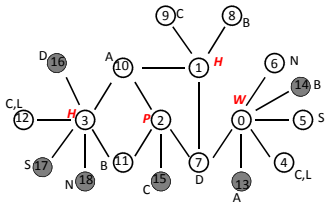


Fig. 2. Privacy-attaining network example

In Example 1, nodes 0, 1, 2, and 3 have sensitive labels. The neighborhood information of node 0, includes its degree, which is 4, and the labels on nodes 4, 5, 6, and 7, which are L, S, N, and D, respectively. For node 2, the neighborhood information includes degree 3 and the labels on nodes 7, 10, and 11, which are D, A, and B. The graph in Figure 2 satisfies 2-*sensitive-label-diversity*; that is because, in this graph, nodes 0 and 3 are indistinguishable, having six neighbors with label A, B, {C,L}, D, S, N separately; likewise, nodes 1 and 2 are indistinguishable, as they both have four neighbors with labels A, B, C, D separately.

4 Algorithm

The main objective of the algorithms that we propose is to make suitable grouping of nodes, and appropriate modification of neighbors' labels of nodes of each group to satisfy the *l-sensitive-label-diversity* requirement. We want to group nodes with as similar neighborhood information as possible so that we can change as few labels as possible and add as few noisy nodes as possible. We propose an algorithm, Global-similarity-based Indirect Noise Node (GINN), that does not attempt to heuristically prune the similarity computation as the other two algorithms, Direct Noisy Node Algorithm (DNN) and Indirect Noisy Node Algorithm (INN) do. Algorithm *DNN* and *INN*, which we devise first, sort nodes by degree and compare neighborhood information of nodes with similar degree. Details about algorithm *DNN* and *INN* please refer to [15].

4.1 Algorithm GINN

The algorithm starts out with group formation, during which all nodes that have not yet been grouped are taken into consideration, in clustering-like fashion. In the first run, two nodes with the maximum similarity of their neighborhood labels are grouped together. Their neighbor labels are modified to be the same immediately so that nodes in one group always have the same neighbor labels. For two nodes, v_1 with neighborhood label set (LS_{v_1}), and v_2 with neighborhood label set (LS_{v_2}), we calculate neighborhood label similarity (NLS) as follows:

$$NLS(v_1, v_2) = \frac{|LS_{v_1} \cap LS_{v_2}|}{|LS_{v_1} \cup LS_{v_2}|} \tag{1}$$

Larger value indicates larger similarity of the two neighborhoods.

Then nodes having the maximum similarity with any node in the group are clustered into the group till the group has ℓ nodes with different sensitive labels. Thereafter, the algorithm proceeds to create the next group. If fewer than ℓ nodes are left after the last group's formation, these remainder nodes are clustered into existing groups according to the similarities between nodes and groups.

After having formed these groups, we need to ensure that each group's members are indistinguishable in terms of *neighborhood information*. Thus, neighborhood labels are modified after every grouping operation, so that labels of nodes can be accordingly updated immediately for the next grouping operation. This modification process ensures that all nodes in a group have the same *neighborhood information*. The objective is achieved by a series of modification operations. To modify graph with as low information loss as possible, we devise three modification operations: *label union*, *edge insertion* and *noise node addition*. Label union and edge insertion among nearby nodes are preferred to node addition, as they incur less alteration to the overall graph structure.

Edge insertion is to complement for both a missing label and insufficient degree value. A node is linked to an existing nearby (two-hop away) node with that label. Label union adds the missing label values by creating super-values

shared among labels of nodes. The labels of two or more nodes coalesce their values to a single super-label value, being the union of their values. This approach maintains data integrity, in the sense that the true label of node is included among the values of its label super-value. After such edge insertion and label union operations, if there are nodes in a group still having different neighborhood information, noise nodes with non-sensitive labels are added into the graph so as to render the nodes in group indistinguishable in terms of their neighbors' labels. We consider the unification of two nodes' neighborhood labels as an example. One node may need a noisy node to be added as its immediate neighbor since it does not have a neighbor with certain label that the other node has; such a label on the other node may not be modifiable, as its is already connected to another sensitive node, which prevents the re-modification on existing modified groups.

Algorithm 1: Global-Similarity-based Indirect Noisy Node Algorithm

Input: graph $G(V, E, L, L^s)$, parameter l;
Result: Modified Graph G'

while $V_{left} > 0$ **do**
  if $|V_{left}| \geq l$ **then**
    compute pairwise node similarities;
    group $\mathcal{G} \leftarrow v_1, v_2$ with $Max_{similarity}$;
    Modify neighbors of $\mathcal{G}$;
    while $|\mathcal{G}| < l$ **do**
      $dissimilarity(V_{left}, \mathcal{G})$;
      group $\mathcal{G} \leftarrow v$ with $Max_{similarity}$;
      Modify neighbors of $\mathcal{G}$ without actually adding noisy nodes ;
  else if $|V_{left}| < l$ **then**
    for *each* $v \in V_{left}$ **do**
      $similarity(v, \mathcal{G}s)$;
      $\mathcal{G}_{Max\_similarity} \leftarrow v$;
    Modify neighbors of $\mathcal{G}_{Max\_similarity}$ without actually adding noisy nodes;
Add expected noisy nodes;
Return $G'(V', E', L')$;

In this algorithm, noise node addition operation that is expected to make the nodes inside each group satisfy ℓ-sensitive-label-diversity are *recorded*, but *not* performed right away. Only after *all* the preliminary grouping operations are performed, the algorithm proceeds to process the *expected node addition* operation at the final step. Then, if two nodes are expected to have the same labels of neighbors and are within two hops (having common neighbors), only one node is added. In other words, we merge some noisy nodes with the same label, thus resulting in fewer noisy nodes.

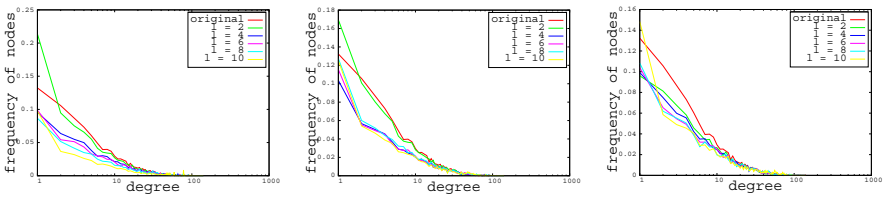
5 Experimental Evaluation

We evaluate our approaches using both synthetic and real data sets. All of the approaches have been implemented in Python. The experiments are conducted

on an Intel core, 2Quad CPU, 2.83GHz machine with 4GB of main memory running Windows 7 Operating System. We use three data sets. The first data set [1] is a network of hyperlinks between weblogs on US politics. The second data set that we use is generated from the Facebook dataset proposed in [14]. The third data set that we use is a family of synthetic graphs with varying number of nodes. The first and second datasets are used for the evaluation of effectiveness (utility and information loss). The third data set is used to measure runtime and scalability (running time). (Please refer to [15] for more information.)

5.1 Data Utility

We compare the data utilities we preserve from the original graphs, in view of measurements on degree distribution, label distribution, degree centrality [19], clustering coefficient, average path length, graph density, and radius. We show the number of the noisy nodes and edges needed for each approach.



(a) DNN (b) INN (c) GINN

Fig. 3. Facebook Graph Degree Distribution

Figure 3 shows the degree distribution of the Facebook graph both before and after modification. Each subfigure in Figure 3 shows degree distributions of graphs modified by one algorithm. We can see that the degree distributions of the modified graphs resemble the original ones well, especially when l is small.

To sum up, these measurements (for other results please refer to [15]) show that the graph structure properties are preserved to a large extent. The strong resemblance of the label distributions in most cases indicates that the label information, another aspect of graph information, is well maintained. They suggest as well that algorithm *GINN* does preserve graph properties better than the other two while these three algorithms achieve the same privacy constraint.

5.2 Information Loss

In view of utility of released data, we aim to keep information loss low. Information loss in this case contains both structure information loss and label information loss. We measure the loss in the following way: for any node $v \in V$, label dissimilarity is defined as: $\mathcal{D}(l_v, l'_v) = 1 - \frac{|l_v \cap l'_v|}{|l_v \cup l'_v|}$, where l_v is the set of v's original labels and l'_v the set of labels in the modified graph. Thus, for the modified graph including n noisy nodes, and m noisy edges, information loss is defined as

$$IL = \omega_1 n + \omega_2 m + (1 - \omega_1 - \omega_2) \sum \mathcal{D}(l_v, l'_v) \tag{2}$$

where ω_1 ,ω_2 and $1 - \omega_1 - \omega_2$ are weights for each part of the information loss. Figure 4 shows the measurements of information loss on the synthetic data set using each algorithm. Algorithm *GINN* introduces the least information loss.

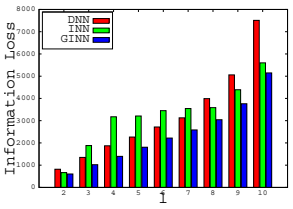


Fig. 4. Information Loss

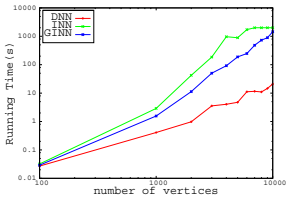


Fig. 5. Running Time

5.3 Algorithm Scalability

We measure the running time of the methods for a series of synthetic graphs with varying number of nodes in our third dataset. Figure 5 presents the running time of each algorithm as the number of nodes increases. Algorithm *DNN* is faster than the other two algorithms, showing good scalability at the cost of large noisy nodes added. Algorithm *GINN* can also be adopted for quite large graphs as follows: We separate the nodes to two different categories, with or without sensitive labels. Such smaller granularity reduces the number of nodes the anonymization method needs to process, and thus improves the overall efficiency.

6 Conclusions

In this paper we have investigated the protection of private label information in social network data publication. We consider graphs with rich label information, which are categorized to be either sensitive or non-sensitive. We assume that adversaries possess prior knowledge about a node's degree and the labels of its neighbors, and can use that to infer the sensitive labels of targets. We suggested a model for attaining privacy while publishing the data, in which node labels are *both* part of adversaries' background knowledge *and* sensitive information that has to be protected. We accompany our model with algorithms that transform a network graph before publication, so as to limit adversaries' confidence about sensitive label data. Our experiments on both real and synthetic data sets confirm the effectiveness, efficiency and scalability of our approach in maintaining critical graph properties while providing a comprehensible privacy guarantee.

References

1. L. A. Adamic and N. Glance. The political blogosphere and the 2004 U.S. election: divided they blog. In *LinkKDD*, 2005.
2. L. Backstrom, C. Dwork, and J. M. Kleinberg. Wherefore art thou R3579X?: anonymized social networks, hidden patterns, and structural steganography. *Commun. ACM*, 54(12), 2011.

3. S. Bhagat, G. Cormode, B. Krishnamurthy, and D. S. and. Class-based graph anonymization for social network data. *PVLDB*, 2(1), 2009.
4. A. Campan and T. M. Truta. A clustering approach for data and structural anonymity in social networks. In *PinKDD*, 2008.
5. J. Cheng, A. W.-C. Fu, and J. Liu. K-isomorphism: privacy-preserving network publication against structural attacks. In *SIGMOD*, 2010.
6. G. Cormode, D. Srivastava, T. Yu, and Q. Zhang. Anonymizing bipartite graph data using safe groupings. *PVLDB*, 19(1), 2010.
7. S. Das, Ö. Egecioglu, and A. E. Abbadi. Anonymizing weighted social network graphs. In *ICDE*, 2010.
8. A. G. Francesco Bonchi and T. Tassa. Identity obfuscation in graphs through the information theoretic lens. In *ICDE*, 2011.
9. M. Hay, G. Miklau, D. Jensen, D. Towsley, and P. Weis. Resisting structural re-identification in anonymized social networks. *PVLDB*, 1(1), 2008.
10. Y. Li and H. Shen. Anonymizing graphs against weight-based attacks. In *ICDM Workshops*, 2010.
11. K. Liu and E. Terzi. Towards identity anonymization on graphs. In *SIGMOD*, 2008.
12. L. Liu, J. Wang, J. Liu, and J. Zhang. Privacy preserving in social networks against sensitive edge disclosure. In *SIAM International Conference on Data Mining*, 2009.
13. A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam. ℓ-diversity: privacy beyond k-anonymity. In *ICDE*, 2006.
14. MPI. http://socialnetworks.mpi-sws.org/.
15. Y. Song, P. Karras, Q. Xiao, and S. Bressan. Sensitive label privacy protection on social network data. Technical report TRD3/12, 2012.
16. Y. Song, S. Nobari, X. Lu, P. Karras, and S. Bressan. On the privacy and utility of anonymized social networks. In *iiWAS*, pages 246–253, 2011.
17. L. Sweeney. K-anonymity: a model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5), 2002.
18. C.-H. Tai, P. S. Yu, D.-N. Yang, and M.-S. Chen. Privacy-preserving social network publication against friendship attacks. In *SIGKDD*, 2011.
19. O. Tore, A. Filip, and S. John. Node centrality in weighted networks: generalizing degree and shortest paths. *Social Networks*, 32(3), 2010.
20. W. Wu, Y. Xiao, W. Wang, Z. He, and Z. Wang. K-symmetry model for identity anonymization in social networks. In *EDBT*, 2010.
21. X. Ying and X. Wu. Randomizing social networks: a spectrum perserving approach. In *SDM*, 2008.
22. X. Ying and X. Wu. On link privacy in randomizing social networks. In *PAKDD*, 2009.
23. M. Yuan, L. Chen, and P. S. Yu. Personalized privacy protection in social networks. *PVLDB*, 4(2), 2010.
24. L. Zhang and W. Zhang. Edge anonymity in social network graphs. In *CSE*, 2009.
25. B. Zhou and J. Pei. Preserving privacy in social networks against neighborhood attacks. In *ICDE*, 2008.
26. B. Zhou and J. Pei. The k-anonymity and ℓ-diversity approaches for privacy preservation in social networks against neighborhood attacks. *Knowledge and Information Systems*, 28(1), 2010.
27. L. Zou, L. Chen, , and M. T. Özsu. K-automorphism: a general framework for privacy-preserving network publication. *PVLDB*, 2(1), 2009.