An Iterative Algorithm for Graph De-anonymization

Jun Zhang[†] Youze Tang[†] Xiaokui Xiao[†] Yin Yang[‡] Zhenjie Zhang[‡] Marianne Winslett[‡§]

[†]Nanyang Technological University [‡]Advanced Digital Sciences Center, Singapore [§]University of Illinois, USA

[†]{jzhang027, s110013, xkxiao}@ntu.edu.sg

[‡]{yin.yang, zhenjie}@adsc.com.sg [§]winslett@illinois.edu

ABSTRACT

The availability of social network data is indispensable for numerous types of research. Nevertheless, data owners are often reluctant to release social network data, as the release may reveal the private information of the individuals involved in the data. To address this problem, several techniques have been proposed to *anonymize* social networks for privacy preserving publications. To evaluate the privacy protection of existing techniques, this paper presents an algorithm designed for de-anonymizing the anonymized graphs produced by the existing techniques. With experiments on a large set of anonymized graphs generated by the existing techniques, we demonstrate that our algorithm can re-identify a large portion of individuals in many anonymized graphs, which sheds light on their effectiveness and relative superiority.

1. INTRODUCTION

A social network is a graph where each node represents an individual, and each edge between two nodes indicates that a certain relationship exists between the two individuals corresponding to the two nodes. Social networks are an important type of data that provide invaluable information on how individuals interact with each other, and they have become indispensable for research in numerous areas. However, data owners are often reluctant to release social network data, as the release may disclose the private information of the individuals involved in the data. A straightforward solution for this problem is to first remove the explicit identifier (e.g., name) associated with each node in a social network, and then release the resulting social network. This solution, however, does not provide a high degree of privacy protection, as it cannot guard against adversaries with prior knowledge about the *structure* of the original social network [2]. For example, assume that an adversary knows in advance that Alice appears in a social network G, and the node in G corresponding to Alice is adjacent to 100 edges. Further assume that G contains only one node with 100 edges. Then, even if we remove the identifier of each node in G, the adversary can still easily re-identify the node corresponding to Alice by inspecting the degree of each node.

To address the above problem, several techniques [3–6, 8–11] have been proposed to more effectively *anonymize* a social network G, by modifying the structure of G. Among the existing techniques, the most notable ones are by Liu and Terzi [8], Bonchi et al. [3], and Tassa and Cohen [9]. In particular, Liu and Terzi [8] as well as Bonchi et al. [3] devise algorithms that heuristically delete edges from G and then insert artificial edges into G. On the other hand, Tassa and Cohen [9] propose to divide the nodes in G into several partitions, and then release only aggregate information about the partitions, i.e., the number of nodes and edges in each partition, as well as the number of edges between any two partitions. Each of these three techniques has been experimentally evaluated, and has been shown to provide privacy protection to some extend. However, there has not been any system comparison of the privacy guarantees provided by those techniques.

In this paper, we propose an algorithm for de-anonymizing the graphs generated by the aforementioned three techniques, and we employ it to evaluate and compare each technique's privacy guarantee. In particular, our algorithm takes as input (i) a social network G where each node and each edge does not contain any attribute, as well as (ii) an anonymized version G^* of G. For any node $v \in G$ and any node $v^* \in G^*$, the algorithm outputs the likelihood that v corresponds to v^*. In other words, our algorithm mimics a powerful adversary who knows whether or not there exists an edge between any two nodes in G, and it employs such background knowledge to re-identify each node of the anonymized graph G^*. With extensive experiments on real data, we demonstrate the effectiveness of our algorithm by showing that it can re-identify up to 43% of individuals in the anonymized graphs produced by the existing techniques [3, 8, 9]; in addition, our experimental results shed lights on their effectiveness and relative superiority.

2. PROBLEM STATEMENT

A social network G is an undirected graph with a set of vertices V and a set of edges E. Each vertex $v \in V$ corresponds to an individual, while each edge $e \in E$ between two nodes $v_1, v_2 \in V$ indicates the existence of certain relationship between v_1 and v_2 (e.g., v_1 and v_2 are knows each other). An *anonymized* version of G is graph G^* that contains the same set of nodes as in G; the set of edges in G^*, denoted as E^*, can be different from the one in G. Our objective is to infer, for any node $v \in G$ and any node $v^* \in G$, whether or not v corresponds to v^*. We consider that G^* is generated from any of the techniques proposed in previous work by Liu and Terzi [8], Bonchi et al. [3], and Tassa and Cohen [9], as will be reviewed in Section 3.

3. RELATED WORK

This section reviews the graph anonymization techniques by Liu and Terzi [8], Bonchi et al. [3], and Tassa and Cohen [9], which are the targets of our de-anonymization algorithm.

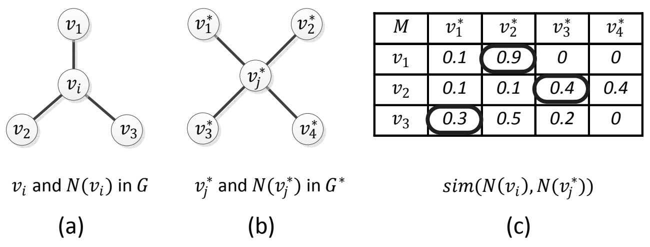
Liu and Terzi's approach [8]. Liu and Terzi propose to anonymize G by removing some edges in G and then adding some artificial edges back into G. The objective of their approach is to ensure that the anonymized graph G^* is k-*degree anonymouse*, i.e., any node in G^* has the same degree with at least $k-1$ other nodes in G^*. This guarantees that, even if the adversary knows in advance the degree of a node corresponding to an individual, the adversary still cannot accurately re-identify the individual from G^*.

Bonchi et al.'s approach [3]. Bonchi et al.'s propose two methods that also anonymize G by deleting and inserting edges in G. The first method works by randomly remove each edge in G with a probability p. The second method first randomly eliminate each edge in G with probability p, and then, for each two nodes v_1, v_2 that are not connected by an edge in G, insert an edge between v_1 and v_2 with a probability p'. The values of p and p' are decided in a manner such that the expected number of edges removed from G would be equal to the expected number of edges inserted.

Tassa and Cohen's approach [9]. Tassa and Cohen's algorithm first divides the nodes in G into several groups, such that each group contains at least k nodes. After that, the algorithm releases the information of node groups in an aggregate manner. In particular, for each node group C, it returns the number of nodes in C, as well as the number edges between the nodes in C. In addition, for any two node groups C_1 and C_2, it returns the number of edges whose two endpoints are in C_1 and C_2, respectively.

4. SOLUTION

Overview. Our algorithm (for inferring the correspondences between nodes in G and nodes in G^*) runs in an iterative manner. In particular, it maintains a $n \times n$ matrix M (referred to as the *correspondence matrix*, where n is the number of nodes in G, such that the entry e_{ij} at the i-th row and j-th column of M quantifies the likelihood that the i-th node v_i in G corresponds to the j-th node v_j^* in G^*. Initially, all entries in M are set to $\frac{1}{n}$, i.e., the mappings from a node in G to all nodes in G^* are deemed equally likely. After that, the algorithm iteratively refines M to adjust the value of each entry, and it terminates after the changes in entry values after an iteration is negligible. Specifically, the refinement performed by each iteration of the algorithm is based on heuristic method for evaluating the similarity between a node $v_i \in G$ and a node $v_j \in G^*$, as explained in the following.



M	v_1^*	v_2^*	v_3^*	v_4^*
v_1	0.1	0.9	0	0
v_2	0.1	0.1	0.4	0.4
v_3	0.3	0.5	0.2	0

Figure 1: An example of *sim* function

Evaluation of Node Similarity. Consider the correspondence matrix M given to a certain iteration of our algorithm. To refine M, we evaluate the similarity between any node v_i in G and any node v_j^* in G^*, by inspecting the entry e_{ij} in M, as well as (i) the set $N(v_i)$ of nodes in G that are the neighbors of v_i and (ii) the set $N(v_j^*)$ of nodes in G^* that are the neighbors of v_j^* in G^*. In particular, we try to match the nodes in $N(v_i)$ to those in $N(v_j^*)$. If there exists a matching that looks likely, then we increase the value of e_{ij}; otherwise, the value of e_{ij} is decreased. More formally, we update the value of e_{ij} based on the following equation:

$$e_{ij} = \frac{e_{ij} + sim(N(v_i), N(v_j^*))}{1 + \max(|N(v_i)|, |N(v_j^*)|)}, \tag{1}$$

where (i) $sim(N(v_i), N(v_j^*))$ denotes the similarity between $N(v_i)$ and $N(v_j^*)$ (as will be explained shortly), (ii) the value 1 in the denominator of the R.H.S. equals the maximum possible value of e_{ij}, and (iii) the term $\max(|N(v_i)|, |N(v_j^*)|)$ in the denominator of the R.H.S. equals the maximum possible value of $sim(N(v_i), N(v_j^*))$. After all entries in M are updated, we normalize the entries in each row of M, such that all entries in a row sums up to 1.

Computation of $sim(N(v_i), N(v_j^*))$. To compute $sim(N(v_i), N(v_j^*))$, we first construct a complete bipartite graph B consisting of the nodes in $N(v_i)$ and $N(v_j^*)$, such that there is no edge between any two nodes in $N(v_i)$ (resp. any two nodes in $N(v_j^*)$). In addition, for any edge in B connecting the α-th node v_α in G to the β-th node v_β^* in G^*, we set the weight of the edge to the value of the entry $e_{\alpha\beta}$ in M. In other words, the edge captures the likelihood of the correspondence between v_α and v_β^*, based on the current correspondence matrix M. After that, we compute a maximum matching in B using the classic Hungarian algorithm [7], and we set $sim(N(v_i), N(v_j^*))$ to the total weight of the edges in the maximum matching. Intuitively, the larger $sim(N(v_i), N(v_j^*))$ is, the more likely that the nodes in $N(v_i)$ can be matched to the nodes in $N(v_j^*)$.

For example, suppose that v_i and $N(v_i)$ are as illustrated in Figure 1a, while v_j^* and $N(v_j^*)$ are as illustrated in Figure 1b. Figure 1c shows the entries in M pertinent to the nodes in $N(v_i)$ and $N(v_j^*)$. The entries highlighted in Figure 1c constitute the maximum matching between $N(v_i)$ and $N(v_j^*)$. Since the total weight of the highlighted entries equal 1.6, we have $sim(N(v_i), N(v_j^*)) = 1.6$.

Dealing with Tassa and Cohen's method [9]. The algorithm describes above requires that, for each two nodes

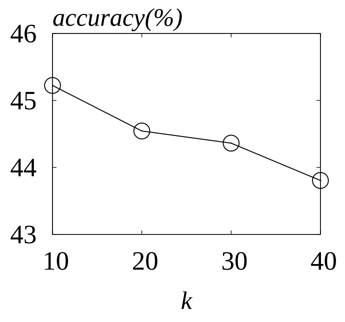


Figure 2: Accuracy of de-anonymization against Liu and Terzi's approach.

in the anonymized graph G^*, it is clearly defined whether there exists an edge between the two nodes. However, the anonymized graphs generated by Tassa and Cohen's method does not satisfy this requirement. In particular, it only releases (i) the number of edges in each node group and (ii) the number of edges between any two node groups, without specifying the two endpoints of each edge. To address this problem, we modify our algorithm as follows. Given the original graph G, we first divide the nodes into several groups using Tassa and Cohen's method. (Note that Tassa and Cohen's method is randomized, and hence, the node groups that we obtain are most likely different from those in G^*.)

After that, we estimate the similarity s between a node group C in G and to a node group C^* in G^*, using a slightly modified version of our iterative algorithm that regards each node group as a node. Once s is computed, we define s as the similarity between any node v in C and anonymized node group C^*. Then, we re-generate the node groups of G by re-running Tassa and Cohen's method on G, and we re-compute the similarity between v and C^*. The process is repeated multiple times, and the final similarity between v and C^* are computed as the average of all similarities between v and C^* estimated. Finally, we construct the correspondence matrix M based on the similarity between all pairs of v and C^*.

5. EXPERIMENT

We conduct experiments using a social network G provided by WSDM 2012 Data Challenge [1]. It contains $8,248$ nodes and $18,732$ edges, such that each node represents an author, and each edge between two scholars indicates that they have coauthored at least one paper published in a top data mining conference in the last five years. We implement all anonymization algorithms reviewed in Section 3, and employ them to generate a set of anonymized versions of G.

We employ our algorithm to de-anonymize each anonymized graph G^*, and we evaluate the accuracy of the de-anonymization. In particular, for the i-th node v_i in G, we inspect the i-th row in the correspondence matrix M produced by our algorithm, i.e., the row that indicate the similarity between v_i and each node in G^*. Suppose that e_{ij} is the largest entry in i-th row in M, then, the j-th node in G^* is deemed the one that most likely corresponds to v_i. Accordingly, we map v_i to v_j^*. This mapping process is re-

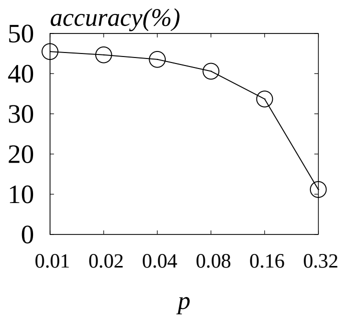


Figure 3: Accuracy against Bonchi et al.'s that only removes edges from G.

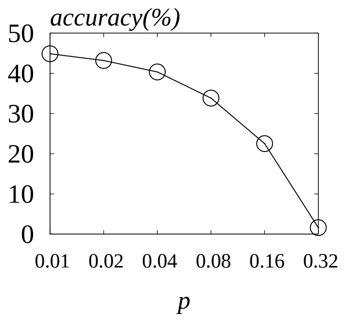


Figure 4: Accuracy against Bonchi et al.'s approach with perturbation.

peated for each node in G. (Note that two nodes in G might be mapped to the same node in G^*. After all mappings are produced, we count the number of correct mappings, and divide the count with the total number of nodes in G. We define the resulting fraction as the *accuracy* of the de-anonymization.

In the first experiment, we attack anonymized graphs generated by Liu and Terzi's approach [8] with parameter $k = 10, 20, 30, 40$, respectively. Figure 2 shows the accuracy of our de-anonymization as a function of k. Regardless of the value k, The accuracy of de-anonymization is above 43%, i.e., our algorithm correctly re-identifies over $3,500$ authors (out of $8,248$) from each anonymized graph. The mapping accuracy slightly decreases with the increase of k, as a larger k indicates a higher level of degree of privacy protection, leading to a higher difficulty for de-anonymization.

Our second experiment evaluates the performance of our algorithm in attacking one of the methods proposed by Bonchi et al. [3], i.e., the method that anonymizes G by removing edges from G. Figure 3 shows the accuracy of de-anonymization as a function of p, i.e., the probability that any edge in G is removed. When p is no more than 0.16, the accuracy of our algorithm is over 30%. The accuracy drops significantly when $p = 0.32$ since, under such a large value of p, Bonchi et al.'s approach would remove around 32% of the edges in G, which would significantly change the structure of the anonymized graph, making it difficult to re-identify nodes.

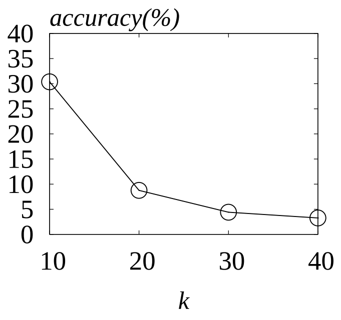


Figure 5: Accuracy against Tassa and Cohen's approach with perturbation

In the third experiment, we evaluate the performance of our algorithm when attacking another method proposed by Bonchi et al. [3], i.e., the method that anonymizes G by first removing edges and then inserting back a roughly equal amount of artificial edges. Figure 4 shows the accuracy of de-anonymization as a function of p, i.e., the probability of edge removal. As with the case in Figure 3, the de-anonymization accuracy decreases when p increases. Furthermore, the accuracy is lower than the case in Figure 3, since the artificial edges inserted into the anonymized graph make it more difficult to de-anonymize.

Finally, Figure 5 shows the accuracy of our algorithm when attacking Tassa and Cohen's method [9], with the parameter k varies from 10 to 40 (recall that k is the minimum size of each node group generated by Tassa and Cohen's method). The accuracy of our algorithm is above 30% when $k = 10$, but it drops considerably when k increases. This shows that Tassa and Cohen's method provides a high degree of privacy protection when k is large.

6. CONCLUSION

This paper presents a de-anonymization algorithm against several representative graph anonymization techniques. Experiments with real data show that our algorithm can re-identify up to 43% of the individuals from the anonymized graphs generating by the existing techniques, which demonstrates the effectiveness of our algorithm.

7. REFERENCES

[1] `http://www.wsdm2013.org/index.php/authors/data-challenge`.

[2] L. Backstrom, C. Dwork, and J. M. Kleinberg. Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography. In *WWW*, pages 181–190, 2007.

[3] F. Bonchi, A. Gionis, and T. Tassa. Identity obfuscation in graphs through the information theoretic lens. In *ICDE*, 2011.

[4] J. Cheng, A. W.-C. Fu, and J. Liu. K-isomorphism: privacy preserving network publication against structural attacks. In *SIGMOD*, pages 459–470, 2010.

[5] G. Cormode, D. Srivastava, T. Yu, and Q. Zhang. Anonymizing bipartite graph data using safe groupings. pages 115–139, 2010.

[6] M. Hay, G. Miklau, D. Jensen, D. F. Towsley, and C. Li. Resisting structural re-identification in anonymized social networks. pages 797–823, 2010.

[7] H. W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, 1955.

[8] K. Liu and E. Terzi. Towards identity anonymization on graphs. SIGMOD, pages 93–106, 2008.

[9] T. Tassa and D. Cohen. Anonymization of centralized and distributed social networks by sequential clustering. *TKDE*, 99(PrePrints), 2012.

[10] B. Zhou and J. Pei. Preserving privacy in social networks against neighborhood attacks. In *ICDE*, pages 506 –515, 2008.

[11] L. Zou, L. Chen, and M. T. özsu. K-automorphism: A general framework for privacy preserving network publication. pages 946–957, 2009.