Community-Enhanced De-anonymization of Online Social Networks

Shirin Nilizadeh Apu Kapadia Yong-Yeol Ahn
School of Informatics and Computing
Indiana University Bloomington
Bloomington, IN, USA
{shirnili, kapadia, yyahn}@indiana.edu

ABSTRACT

Online social network providers have become treasure troves of information for marketers and researchers. To profit from their data while honoring the privacy of their customers, social networking services share 'anonymized' social network datasets, where, for example, identities of users are removed from the social network graph. However, by using external information such as a reference social graph (from the same network or another network with similar users), researchers have shown how such datasets can be de-anonymized. These approaches use 'network alignment' techniques to map nodes from the reference graph into the anonymized graph and are often sensitive to larger network sizes, the number of seeds, and noise — which may be added to preserve privacy.

We propose a divide-and-conquer approach to strengthen the power of such algorithms. Our approach partitions the networks into 'communities' and performs a two-stage mapping: first at the community level, and then for the entire network. Through extensive simulation on real-world social network datasets, we show how such community-aware network alignment improves de-anonymization performance under high levels of noise, large network sizes, and a low number of seeds. Even when nodes cannot be explicitly mapped, the community structure can be mapped between both networks, thus reducing the anonymity of users. For example, for our (real-world) Twitter dataset with 90,000 nodes, 20% noise, and 16 seeds, the state-of-the-art technique reduces anonymity by 0 bits, whereas our approach reduces anonymity by 9.71 bits (with 40% of nodes mapped).

Categories and Subject Descriptors

K.6.5 [**Management of Computing and Information Systems**]: Security and Protection

Keywords

social network de-anonymization; community detection

CCS'14, November 3–7, 2014, Scottsdale, Arizona, USA.

http://dx.doi.org/10.1145/2660267.2660324.

1. INTRODUCTION

Online social networks have exploded in popularity. For example, Facebook has over a billion active users [14], Google+ has over 350 million active users, and Twitter has close to 300 million active users [18]. Social network providers host vast amounts of personal information and relationship information between their users and have become a treasure trove sought by marketers and researchers alike [37]. The availability of rich social data has led to various applications, such as targeted online advertisements, the study of human behaviors (computational social science), and health care [33, 37]. Because of its value across various applications, such social data is often shared or sold to academic researchers or third-party companies. To protect the privacy of their users while exploiting the value of this data, social networking services attempt to 'anonymize' social network data before selling or sharing such information. For example, the services may provide social-network structure but remove people's identities and try to add some 'noise' by modifying relationships and attributes to a certain extent.

Various approaches have been proposed to 'de-anonymize' users in the dataset using external knowledge about the users [5, 27, 37, 53]. For example, an attacker may have access to the social-network structure with real identities (but not other sensitive attributes) by crawling publicly available relationships on the same site or a different social networking site with a similar customer base. Narayanan and Shmatikov have shown how the social structure from one site, such as Flickr, can be used to re-identify anonymized users on another site, such as Twitter [37]. In general these approaches perform a 'network alignment' between the two networks, mapping the nodes in the anonymized network to the nodes with identities in the reference network. Researchers have also looked at stronger adversarial models where the attacker has attribute information from the reference network [29, 36], and in such cases de-anonymization is easier because of the additional information. In this work we study the problem of de-anonymization using network alignment without access to additional attributes, but where the two networks have a high level of overlap, either by crawling the same network or another network with a similar user base.

The problem of aligning two networks is closely related to the graph isomorphism problem. While this problem cannot yet be efficiently solved in the general case [16], the above-mentioned techniques are reasonably effective when aligning social-network graphs because they use heuristics that exploit the unique structural properties of real-world networks such as heavy-tailed degree distributions and the presence of large cliques. Most techniques start from known common identities in the two graphs — 'seeds' — and then grow these mapped regions by comparing and matching the

local (microscopic) properties in each graph. As we show, these techniques require a high number of seeds and are often sensitive to high levels of noise (e.g., where a certain fraction of edges is 'rewired'). Furthermore, as the networks grow, microscopic structures are increasingly replicated across the entire network, making it hard to map nodes based solely on local properties.

Our contributions. In this paper we leverage 'mesoscopic'[1] properties of social networks for enhanced de-anonymization that is more robust to noise and a low number of seeds, and scales easier with large network size. Our approach leverages 'community detection' techniques to partition the networks into 'communities', i.e. dense subgraphs that capture social structure [15, 17]. Our proposed approach divides the problem into smaller sub-problems that can be solved by leveraging existing network alignment methods recursively on multiple levels. First, our approach maps the community structure of two graphs (which may overlap imperfectly) by considering the community structure as a coarse-grained graph. It then applies the network mapping technique to the nodes inside each community (along with a 'seed enrichment' phase) and finally to the entire graph. Through extensive simulations on real social network datasets, we suggest that our 'community boosting' technique generically provides a significant improvement to microscopic mapping algorithms, such as the one by Narayanan and Shmatikov (subsequently we refer to their algorithm as the "NS" algorithm) [37], and enhance their performance in a way that is more robust to noise and large network sizes.

Another major contribution is our analysis of the 'degree of anonymity' of users in the graph. Even when the explicit mapping of *nodes* is far from complete, we show that the mapping of *communities* may markedly reduce the degree of anonymity of users, since the probability distributions of potential mappings results in less uncertainty thanks to the mappings between communities. For example, we show that for our Twitter dataset with 90,000 nodes, 15% edge noise, and 16 seeds that the NS technique reduces anonymity by 2.6 bits (with 33% of nodes explicitly mapped), whereas our approach reduces anonymity by 13.17 bits (with 65% of nodes mapped). For the same dataset, with 20% edge noise and 16 seeds, the NS technique reduces anonymity by 0.0 bits (with almost no node explicitly mapped), whereas our approach reduces anonymity by 9.71 bits (with 40% of nodes mapped).

2. DEFINITIONS AND ATTACK MODELS

In general, de-anonymization is defined as "a data mining strategy in which anonymous data is cross-referenced with other data sources to re-identify the anonymous data source".[2] The idea is to collect enough information about an anonymous individual and enrich his/her profile. This profile can then be linked to other public information to attach an identity to the data. Social network de-anonymization usually concerns the problem of cross-referencing two or more social graphs to enrich anonymous users' profiles and re-identify them. Some attacks use only network structures, while others exploit user attributes such as user names and group memberships [29, 37, 53]. We now formalize the models and definitions used in our work.

2.1 Definitions and assumptions

[1]"Mesoscopic" is a term in physics used to refer to a granularity between microscopic and macroscopic.

[2]`http://whatis.techtarget.com/definition/de-anonymization-de-anonymization`

We interchangeably use the terms "network", "node", and "link" with "graph", "vertex", and "edge", respectively. All networks we use in this paper are undirected.

DEFINITION 1. *A graph, $G\langle V, E\rangle$ is a set of vertices V that represents the users in the network and a set of undirected edges $E \subseteq \{e = (u, v) : u, v \in V\}$ that represents links between users. In a social network, for example, edges would correspond to social relationships. We denote the degree of a node by k_v. Let $N = |V|$ be the total number of nodes in G.*

DEFINITION 2. *A graph G's community structure (C) is a disjoint partition of vertices in G, namely $C = \{c_1, c_2, \ldots, c_k\}$, where $c_i \neq \varnothing$ and $c_i \cap c_j = \varnothing$ if $i \neq j$ for $i, j \in \{1, 2, \ldots, k\}$. While there are many alternative definitions of communities [4, 15, 40], in this paper communities are defined by Infomap algorithm [45], which finds a partition that minimizes the average number of bits per step required to describe trajectories of random walkers.*

2.2 Attack model

Online social network providers release anonymized social networks to third-parties for various purposes including targeted advertising, developing new applications, academic research, and public competition [20, 43]. We assume the recipient of this data, if malicious, may try to de-anonymize the social network by explicitly mapping nodes in the extreme case and/or reducing the uncertainty of mappings to the greatest extent possible.

We assume the adversary has access to two networks, $G\langle V, E\rangle$ and $G'\langle V', E'\rangle$, where $V \cap V \neq \varnothing$, and $E \cap E' \neq \varnothing$. We focus on the cases where $V \approx V'$ and $E \approx E'$, i.e. where the vertices and edges are approximately the same. The difference in these sets is characterized more formally by a 'noise' parameter (see Section 6.2).

One of these networks is anonymized and contains sensitive private information associated with the (anonymized) nodes in the graph. The goal of an attacker is to align the anonymized network with the other, 'reference' network, re-identify anonymized users, and reveal the private information obtained from the anonymized network. If both networks are anonymized, the problem changes from re-identification to profile enrichment where the attacker tries to align two networks and collect more data about the anonymous users.

3. BACKGROUND

3.1 Re-identification algorithm by Narayanan and Shmatikov (NS)

Our algorithm is designed to leverage existing mapping methods. For our evaluation, our algorithm is built upon the re-identification algorithm by Narayanan and Shmatikov [37]. Their algorithm runs in two stages: 'seed detection' and 'propagation'. In the seed-detection step the algorithm maps a small number of users (seeds) between two networks by searching for unique subgraphs. The propagation step expands the set of matched users by incrementally comparing and mapping the neighbors of the previously mapped seeds.

3.1.1 Seed identification

Narayanan *et al*. [35, 37] have proposed a seed identification algorithm that randomly samples a subset of k-cliques from the reference graph and finds the corresponding cliques in the other graph. For a chosen clique, the algorithm examines the degree sequence of

the k nodes in the given clique and the number of common neighbors between each of $\binom{k}{2}$ pairs of users. For each candidate clique in the other graph, the algorithm compares the two sequences and decides based on an error parameter, θ, whether they are the same people or not. We use a similar approach for identifying initial seeds, which also helps in mapping communities (see Section 4.1).

3.1.2 Propagation

In the propagation step, the algorithm expands the set of identified seeds. In each iteration, the algorithm randomly picks an already-mapped node pair $(u, u') \in M$, where $u \in V$, $u' \in V'$, and M is the set of mappings. From the set of u's unmapped neighbors, it picks a random node v then compares it with each unmapped node (v') in the set of u''s unmapped neighbors. The similarity $\mathcal{S}$ between v and v' is defined as the number of v's neighbors that are already mapped to the neighbors of v', divided by the square root of its degree, namely

$$\mathcal{S}(v, v') = \frac{|\{(w, w') : w \in \mathcal{N}(v); w' \in \mathcal{N}(v'); \text{and } (w, w') \in M\}|}{\sqrt{k_v k_{v'}}},$$

where $\mathcal{N}(v)$ is the set of v's neighbors. After calculating $\mathcal{S}$ for all potential candidates (the unmapped neighbors of u') and creating an ordered list of the scores (L), the algorithm identifies the best (v'_1) and the second-best candidate (v'_2) that have the highest scores. v'_1 is accepted as the counterpart of v if its score is sufficiently better than that of v'_2. The uniqueness of the best candidate is measured by 'eccentricity' as defined by

$$ecc(L) = \frac{\mathcal{S}(v, v'_1) - \mathcal{S}(v, v'_2)}{\sigma(L)},$$

where L is the ordered list of the similarity scores ($\mathcal{S}$) of the unmapped neighbors of u', and $\sigma(L)$ is the standard deviation of the values in L.

3.2 Community detection

Community detection (or graph partitioning) has received much attention from various fields, because community structure is a common characteristic of a wide variety of networks across domains and communities usually correspond to important subunits of the systems. For instance, the communities in social networks correspond to social circles and those in biological networks correspond to functional modules. Originally, graph partitioning was introduced to solve the problem of optimal allocation of processes in a distributed computing context [24]. Since then, graph partitioning and community detection has been actively studied across disciplines [15, 42, 46]. Although there is no concrete definition of a community that is agreed upon, communities usually refer to groups of nodes (people) that are densely connected to each other while having lesser connections to nodes residing outside of the community. A large number of community detection methods have been developed and they are widely applied to many domains of science [4, 15, 44].

Although it is known that communities often overlap [4, 40], we use disjoint, non-overlapping communities to simplify the problem. Among the variety of community detection methods, we employ the Infomap algorithm [1, 45] here because it is one of the most widely accepted disjoint community detection algorithm; it was shown to excel in tests using synthetic benchmark networks [31, 32]. Also note that our goal is slicing the network into smaller, dense chunks, which may not correspond to meaningful social groups. In principle, myriad other community detection (or graph partitioning) approaches can be adopted to our framework, although we leave such exploration to future work.

3.3 Degree of anonymity

Pfitzmann and Kohntopp [41] defined "anonymity" as the state of being not identifiable within a set of subjects, the anonymity set. Chaum [10] first characterized the anonymity set as the measure of anonymity. This measurement has been used by several networks that provide anonymity for senders or receivers of messages [8, 25, 39]. Anonymity set size, however, does not take into account that different members may be more or less likely to send or receive messages. Based on a particular attack, also, these probabilities may differ. Serjantov and Danezis [47] and also Diaz *et al*. [12] used entropy to define degree of anonymity achieved by the users of a system towards particular attacker. This measurement depends on the distribution of probabilities and not simply the size of the anonymity set. The entropy of the system after the attack is compared against the maximum entropy, in which all N users are likely to be the originator of the message with equal probability $\frac{1}{N}$. The entropy of the system is defined as:

$$H(X) = -\sum_{i=1}^{N} p_i \log p_i,$$

where $H(X)$ is the entropy of the network, N is the number of nodes in the network, and p_i is the probability associated with node i. As the maximal entropy is $H_{\max} = \log N$, the attacker's information gain is $H_{\max} - H(X)$, and thus the degree of anonymity is defined as the normalized entropy of the system:

$$A(X) := 1 - \frac{H_{\max} - H(X)}{H_{\max}} = \frac{H(X)}{\log N},$$

where $0 \leq A(X) \leq 1$. This value quantifies the amount of information the system is hiding. For instance, $A(X) = 1$ indicates that the users in the network are completely anonymous. $A(X) = 0.5$ indicates that 'half the bits' of privacy are lost compared to the uniform distribution, which corresponds to $\log N$ bits of privacy.

4. OUR APPROACH: COMMUNITY-ENHANCED DE-ANONYMIZATION

The key notion of our community-based de-anonymization framework is that network communities provide an effective way to divide-and-conquer the problem of de-anonymization, particularly because communities are known to capture meaningful, mesoscopic structural relationships even in the presence of noise [31]. After dividing the reference and anonymized graphs into communities, these communities can be mapped first, and then users can be matched and re-identified for each corresponding community. In particular, our approach can employ an existing 'community-blind' mapping algorithm such as the NS algorithm for *community- and node-level* mapping. Our approach is thus a 'community-aware' mapping algorithm built upon community-blind mapping algorithms.

Figure 1 illustrates our proposed network alignment framework. Our algorithm has four steps: 1) community detection, 2) community mapping, 3) seed enrichment, and 4) global propagation. These steps are explained next.

4.1 Community mapping

The first step of our algorithm is detecting communities. As mentioned in Section 3.2, we use Infomap to slice both the reference and the anonymized networks into smaller, denser chunks. Nevertheless, any community detection (or graph partitioning) approach can be adopted to our framework. The next step is mapping communities that are found in the previous step. Our approach uses

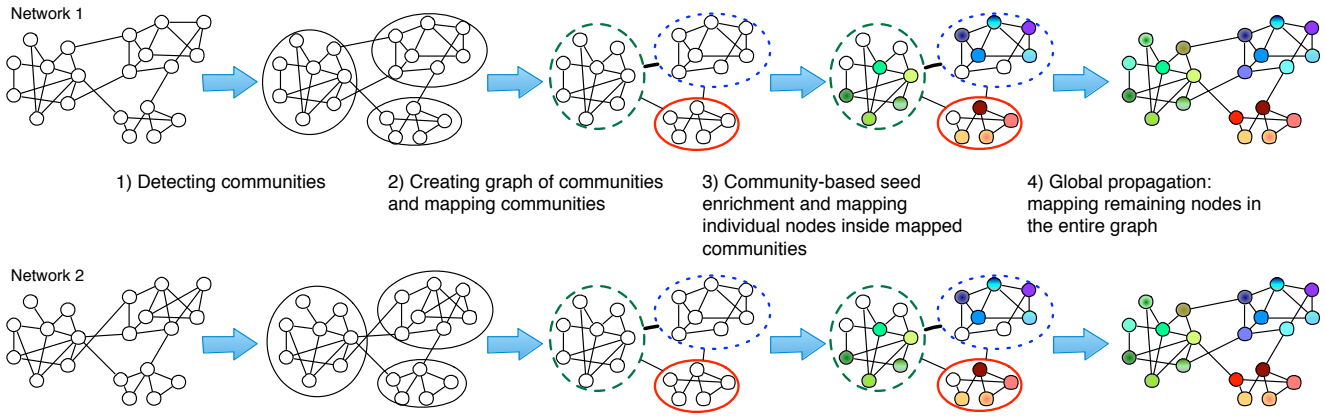


Figure 1: An overview of our approach where 1) each of two social graphs is divided to smaller partitions — communities; 2) communities of these two graphs are mapped; 3) nodes inside mapped communities are matched; and 4) NS propagation algorithm runs on the whole network to map remained unmapped nodes.

two strategies to map communities: (1) using already-known seeds and (2) using the network of communities. Once some communities have been mapped (forming seeds at the community level), the community-blind propagation algorithm is applied to the community graph to expand the set of mapped communities.

4.1.1 *Identifying seed communities*

Before communities can be mapped, the propagation algorithm needs some pre-identified seed mappings. After detecting communities in the two networks, communities associated with seed nodes can be mapped to each other. However, conflicts are possible when two seeds with two different communities in the first network are mapped to one community in the other network. Conflicts are minimized by simply counting the number of times that two communities are mapped together. For each community in the reference network, all possible mappings are listed based on counts in descending order and this community is mapped to the community on top of the list. There are some cases where one community is mapped to two different communities with the same scores. In these cases, a mapping is picked at random.

4.1.2 *Mapping communities by creating a network of communities*

The community structure itself can be considered as a high-level, coarse-grained graph; we consider each community as a node and the connection between communities as edges. This perspective allows us to directly reuse community blind mapping algorithms such as the NS algorithm to map communities.

Given a graph G, we create a weighted undirected graph of communities, G^*, where each community is a node and a weighted edge between two communities represents the number of connections between nodes in two communities.

In our framework, a community-blind mapping algorithm is run on the network of communities and is fed with some seed communities. Since we use NS in our evaluation, we propose a slight improvement to the NS propagation algorithm to exploit the weights in the graph of communities. As in the original NS algorithm, our "weighted propagation" algorithm starts with two graphs G_1^* and G_2^*. At each iteration the algorithm randomly picks a neighbor (μ^*) of already-mapped seeds ($\mathcal{U}^*$). We modify the similarity score function so that it includes the weight of edges in the weighted

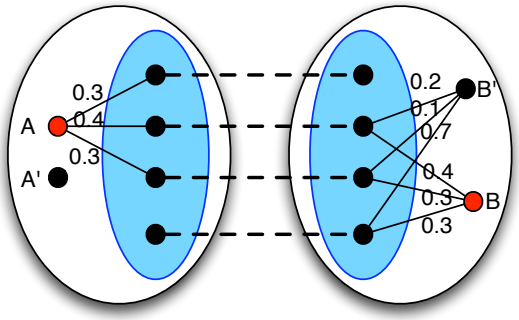


Figure 2: An example of mapping nodes of two weighted undirected networks using the 'weighted propagation' algorithm. The numbers on the edges show the edge weights.

graphs. We tested different similarity functions and found the following to be more effective:

$$\tilde{\mathcal{S}}(\mu^*, \nu^*) = \frac{\sum_{(p^*,q^*)\in\mathcal{N}(\mu^*,\nu^*)} (1 - |\sqrt{w_{(\mu^*,p^*)}} - \sqrt{w_{(q^*,\nu^*)}}|)}{\sqrt{d(\mu^*)d(\nu^*)}} \quad (1)$$

where $\mathcal{N}(\mu^*, \nu^*)$ is the set of already mapped pairs among the neighbors of μ^* and ν^*. $w_{(\mu^*,p^*)}$ is the weight of the edge between μ^* and p^*.

Figure 2 illustrates an example where the mapping algorithm tries to align node A in the left graph to a node in the right graph. Three of A's neighbors are already mapped. The algorithm starts with these and computes the similarity score for each neighbor of the mapped nodes in the right graph, i.e. $\tilde{\mathcal{S}}(A, B)$ and $\tilde{\mathcal{S}}(A, B')$. Using the score function of the original NS propagation algorithm, $\tilde{\mathcal{S}}(A, B) = \tilde{\mathcal{S}}(A, B')$ and the algorithm cannot map A to any node in the right graph. By contrast, here $\tilde{\mathcal{S}}(A, B)$ would be larger than $\tilde{\mathcal{S}}(A, B')$, and thus A is correctly mapped to B (with a suitable eccentricity threshold).

4.2 Seed enrichment and local propagation

One of the major benefits from the community decomposition and mapping is that additional seeds can be identified. Seeds are usually identified based on their uniqueness at a global level; communities offer a much more narrow search space for seeds, which may have otherwise not looked unique at the global scale. We call this step of finding more seeds leveraging community information "seed enrichment". Following seed enrichment, the community-blind mapping algorithm is applied to each pair of matched communities using the enriched set of seeds.

We propose the following approach to identify seeds at the community level, which is based on two distance metrics defined over nodes' degrees (d), and the clustering coefficients (cc):

$$D_d(v_i, v_j) = \frac{|d(v_i) - d(v_j)|}{\max(d(v_i), d(v_j))}$$

$$D_{cc}(v_i, v_j) = \frac{|cc(v_i) - cc(v_j)|}{\max(cc(v_i), cc(v_j))}$$

The clustering coefficient is a property of a node in a network and quantifies how close its neighbors are to being a clique. It can be quantified as the fraction of pairs of the node's neighbors that are connected to each other by edges. The clustering coefficient is between zero and one; if the neighborhood is fully connected, it is 1 and if there are few connections in the neighborhood, its value is close to 0.

These two metrics are computed and tested between each pair of nodes across the mapped communities. These nodes are matched and identified as seeds if either their degree or their clustering coefficients are similar enough and above a certain eccentricity threshold.

For each pair of mapped communities, the community-blind mapping algorithm is performed locally — only considering the nodes inside these communities. This algorithm takes two subgraphs (communities) $G_{c_1}\langle V_{c_1}, E_{c_1}\rangle$ and $G_{c_2}\langle V_{c_2}, E_{c_2}\rangle$ from two networks G_1 and G_2 and the set of seeds in these communities. This algorithm can also be run in parallel on each pair of mapped communities to increase performance.

4.3 Global propagation

The last step in our framework applies the community-blind mapping algorithm to the whole network using all the currently mapped nodes as seeds. This step is necessary because all communities may not be correctly mapped or mapped at all; therefore, some nodes are not chosen to be re-identified. Running the community-blind mapping algorithm globally expands these mappings. In short, community-blind mapping algorithms run only this global propagation step, while our approach adds the previous steps as intermediate steps resulting in a community-aware mapping algorithm.

5. DEGREE OF ANONYMITY

In this section, we propose a method of estimating the degree of anonymity of users in an anonymized network, given an error (noise) model that describes how the reference graph would differ from the anonymized graph. Throughout this section, we assume that we are given an anonymized graph and an error model. We compute the degree of anonymity by observing that the *community structure may reveal information about true mappings* even when nodes cannot be mapped by de-anonymization algorithms.

Consider two graphs $G(V, E)$ (reference) and $G'(V', E')$ (anonymized). $u \sim u'$ means that the mapping between u and u' is *true*. The set of true mappings is:

$$M_t = \{(u, u') : u \in V; u' \in V'; \text{and } u \sim u'\}.$$

We denote the *algorithmically detected mapping* between u and u' as $u \leftrightarrow u'$. The set of mappings is:

$$M_a = \{(u, u') : u \in V; u' \in V'; \text{and } u \leftrightarrow u'\}.$$

We denote the community mapping between c and c' as $c \leftrightarrow c'$. The set of community mappings is:

$$M_c = \{(c, c') : c \in C; c' \in C'; \text{and } c \leftrightarrow c'\}$$

where C and C' are the sets of communities in G and G'.

Here, we simplify the problem by ignoring all other information we can potentially obtain from the graphs, focusing only on the vertex sets and mappings. Given two graphs and mappings, we define the anonymity for a user $u \in V$ as the entropy over the probability distribution of potential mappings being true for user u:

$$H(u) = -\sum_{u' \in V'} P(u \sim u'|M_a) \log P(u \sim u'|M_a), \quad (2)$$

where $P(u \sim u')$ is the marginal probability that a node $u' \in V'$ is actually the true mapping with a given node $u \in V$. If $M_a = \varnothing$ or does not provide any information, we assume that $P(u \sim u'|M_a) = \frac{1}{N}$ for every u' and $H(u)$ reaches the maximum value of $\log N$. On the other hand, if we know that the algorithm works perfectly, namely $u \leftrightarrow u' \iff u \sim u'$, then $P(u \sim u'|M_a) = 1$ when $u \leftrightarrow u'$ and 0 otherwise. In this case $H(u)$ becomes zero.

Likewise, given M_a and M_c, we define the anonymity for a user $u \in V$ as the entropy over the probability distribution of potential mappings being true for user u:

$$H(u) = -\sum_{u' \in V'} P(u \sim u'|M_a, M_c) \log P(u \sim u'|M_a, M_c). \quad (3)$$

We define the normalized degree of anonymity for user u as:

$$A(u) := \frac{H(u)}{H_{max}}, \quad (4)$$

where $H_{max} = \log N$ is the maximum entropy.

Finally, we define the degree of anonymity for the whole system by averaging the degree of anonymity of all users in the network:

$$A(G) := \frac{\sum_{u \in V} A(u)}{N}. \quad (5)$$

Next, we show how the degree of anonymity can be estimated in practice. Our approximations and simplifications are focused on estimating the upper bound of the degree of anonymity.

5.1 Degree of anonymity of community-blind de-anonymization algorithm

We define V'_- (resp. V_-) as the set of nodes in V' (resp. V) that have not been mapped by the algorithm:

$$V'_- = \{u' \in V' : \nexists u \in V, \quad u \leftrightarrow u'\}.$$

If a community-blind algorithm is employed for de-anonymizing users, $P(u \sim u'|M_a)$ for a given $u \in V$ can be assigned values for all $u' \in V'$ based on the following cases.

(1) If u is mapped by the algorithm to z', the vertices $u' \in V'$ can be partitioned as:

a) The mapped node z', i.e. $(u, z') \in M_a$. In this case we need to compute $P(u \sim z' | u \leftrightarrow z')$ and this probability, given a graph and an error model, can be estimated by measuring how often a claimed mapping is correct in simulations. Let this value be p_{map}.

b) The remaining nodes that were not mapped to u, i.e. y' such that $(u, y') \notin M_a$. We need to compute $P(u \sim y' | u \leftrightarrow z', z' \neq y')$. This value can be estimated as $\frac{1-p_{map}}{|V'|-1}$, which assumes that any node in this set is the correct mapping with uniform probability.

(2) If u is not mapped by the algorithm, i.e. $(u, u') \notin M_a$, we consider the correct mapping to be within the entire vertex set V' with the same probability. That is, $P(u \sim u' | u \in V_-) = 1/|V'|$.

5.2 Degree of anonymity of community-aware de-anonymization algorithm

The community mapping can reveal additional information about the true mapping, and thus several more cases need to be considered as compared to the community-blind algorithm. We define C'_- (resp. C_-) as the set of communities in G' (resp. G) that have not been mapped. $c \leftrightarrow c'$ represents that the algorithm has mapped community c in G to community c' in G'. If a community-aware algorithm is employed for de-anonymizing users $P(u \sim u' | M_a, M_c)$ can be assigned values for all $u' \in V'$ based on the following cases (again this analysis presents a simpler case analysis as described earlier):

(1) If u is mapped to some node z' by the algorithm, and the community c of u is also mapped to the community c' of z', the vertices V' can be partitioned as:

a) The mapped node z'. In this case we need to compute $P(u \sim z' | u \leftrightarrow z', c \leftrightarrow c', u \in c, z' \in c')$, i.e. how often a claimed mapping is correct (in this circumstance). This probability can be estimated through simulation. Let this value be $p_{map,1a}$, where the second subscript "1a" refers to Case 1a.

b) The remaining nodes y' within c' that were not mapped to u. In this case, we need to compute $P(u \sim y' | u \leftrightarrow z', c \leftrightarrow c', y' \neq z', u \in c, z' \in c', y' \in c')$. The probability that a node is mapped to *any* other node in the same community estimated through simulation as $p_{map,1b}$. Thus $P(u \sim y' | u \leftrightarrow z', c \leftrightarrow c', y' \neq z', u \in c, z' \in c', y' \in c') = \frac{p_{map,1b}}{|c'|-1}$.

c) The remaining nodes r' that are not in c' (i.e. in $G' \setminus c'$). In this case, we need to compute $P(u \sim r' | u \leftrightarrow z', c \leftrightarrow c', r' \notin c', u \in c, z' \in c')$. The probability that a node is mapped to *any* other node not in community estimated through simulation as $p_{map,1c}$. Thus $P(u \sim r' | u \leftrightarrow z', c \leftrightarrow c', r' \notin c', u \in c, z' \in c') = \frac{p_{map,1c}}{|G \setminus c'|}$.

We note that $p_{map,1a} + p_{map,1b} + p_{map,1c} = 1$.

(2) If u is mapped to some node z' by the algorithm, and the community c of u is mapped to some community c' which is different from community z', the vertices u' can be partitioned as:

a) The mapped node z'. In this case we need to compute $P(u \sim z' | u \leftrightarrow z', c \leftrightarrow c', u \in c, z' \notin c')$, i.e. how often a claimed mapping is correct (in this circumstance). This probability can be estimated through simulation. Let this value be $p_{map,2a}$.

b) The nodes y' within c'. In this case, we need to compute $P(u \sim y' | u \leftrightarrow z', c \leftrightarrow c', u \in c, z' \notin c', y' \in c')$, the probability that a node is mapped to *any* node in the mapped community. This value can be estimated through simulation as $p_{map,2b}$. Thus $P(u \sim y' | u \leftrightarrow z', c \leftrightarrow c', u \in c, z' \notin c', y' \in c') = \frac{p_{map,2b}}{|c'|}$.

c) The remaining nodes r' that are not in c' (i.e. in $G' \setminus c'$) and are not mapped to u. In this case, we need to compute $P(u \sim r' | u \leftrightarrow z', c \leftrightarrow c', r' \neq z', u \in c, r' \notin c', z' \notin c')$, the probability that a node is mapped to *any* other node not in community. This value can be estimated through simulation as $p_{map,2c}$. Thus $P(u \sim r' | u \leftrightarrow z', c \leftrightarrow c', r' \neq z', u \in c, r' \notin c', z' \notin c') = \frac{p_{map,2c}}{|G \setminus c'|-1}$.

We note that $p_{map,2a} + p_{map,2b} + p_{map,2c} = 1$.

(3) If u is mapped to some node z' by the algorithm, but the community c of u is not mapped to any community in G', the vertices V' can be partitioned as:

a) The mapped node z'. In this case we need to compute $P(u \sim z' | u \leftrightarrow z', c \in C_-, u \in c)$, i.e. how often a claimed mapping is correct (in this circumstance). This probability can be estimated through simulation. Let this value be $p_{map,3a}$.

b) The remaining nodes that were not mapped to u, i.e. r' such that $(u, r') \notin M_a$. We need to compute $P(u \sim r' | u \leftrightarrow z', z' \neq r', u \in c, c \in C_-)$. Following a mapping of the anonymized graph, This value can be estimated as $\frac{1-p_{map,3a}}{|V'|-1}$, which assumes that any node in this set is the correct mapping with uniform probability.

(4) If u is not mapped to any node in G' by the algorithm, and the community c of u is also not mapped to any community in G', the vertices V' can be partitioned as:

a) We consider the correct mapping to be within the entire vertex set V' with the same probability. That is, $P(u \sim u' | u \in V_-, u \in c, c \in C_-) = 1/|V'|$.

(5) If u is not mapped to any node in G' by the algorithm, but the community c of u is mapped to a community c', the vertices u' can be partitioned as:

a) The nodes y' within c'. In this case, we need to compute $P(u \sim y' | u \in V_-, u \in c, c \leftrightarrow c', y' \in c')$, the probability that a node is mapped to *any* node in the mapped community. This value can be estimated through simulation as $p_{map,5a}$. Thus $P(u \sim y' | u \in V_-, u \in c, c \leftrightarrow c', y' \in c') = \frac{p_{map,5a}}{|c'|}$.

b) The remaining nodes r' that are not in c' (i.e. in $G' \setminus c'$). In this case, we need to compute $P(u \sim r' | u \in V_-, u \in c, c \leftrightarrow c', r' \notin c')$, the probability that a node is mapped to *any* other node not in community. This value can be estimated as $\frac{1-p_{map,5a}}{|G \setminus c'|}$, which assumes that any node in this set is the correct mapping with uniform probability.

5.3 Caveats

The key aim of our measure is *estimating the upper bound* of the degree of anonymity, to quantify the minimum possible damage that may be caused by the de-anonymization attacks. Thus one may need to devise more sophisticated and realistic methods to accurately calculate the degree of anonymity that is closer to reality. We simplified the problem by introducing several approximations. First of all, we ignored prior information that can be obtained from network structure. Although we may be able to condition the probabilities of mappings given prior knowledge of the network structure, we decided this approach was overly complex for our main goal of showing the relative performance of algorithms. Another

caveat is that we approximate the real probability for each case by running simulations that assume a specific ensemble of networks (e.g. an ensemble of networks with the same number of nodes, edges, noise level, and the type of noise). Our ensemble-simulation approach provides a way to estimate the breach of privacy with just one network, but at the same time, the parametrization of the reference graph may not coincide with the actual error model in real-life, and our estimation may not be accurate.

In addition, more fine-grained case analysis can be applied for a better estimate of the degree of anonymity. For example, Cases (1).b and (2) can be further split into nodes that were mapped to some other node or not mapped to any node at all. We have computed the degree of anonymity under this model and observed that there is only a slight difference between the two methods. We stick to the simpler model in this paper for ease of exposition (especially as it blows up the number of cases described in Section 5.2), and thus in Section 7 we show results only for the simpler model. Also note that the simpler model provides the upper bound of the degree of anonymity in our framework.

6. EVALUATION

We perform simulation-based experiments using real-world network datasets. For each experiment, we prepare a copy of the original network, partially alter its structure, and compare the network alignment performance of two approaches — community-aware and community-blind — using the networks. We also perform experiments on partially overlapping networks where some percentage of users from each network — original network and the edge-altered noisy network — are removed and thus two networks have partially different user sets.

6.1 Data sets

We use three real-world networks: (i) A collaboration network [38], which is a network of coauthorships between scientists who have posted preprints on the arXiv Condensed Matter E-Print Archive. In this network two authors are connected if they wrote at least one paper together. The network is constructed from all preprints posted between January 1, 1995 and March 31, 2005. This network has 36,458 nodes and 171,735 edges; (ii) A Twitter mention network [52], which captures the connections between users who mutually mentioned each other at least once between March 24th, 2012 and April 25th, 2012. We first extract the largest connected component from this graph and partition it into four graphs using the METIS graph partitioning algorithm [2] to obtain a smaller, more manageable network. We use one of the graph partitions with 90,332 nodes and 377,588 edges; and (iii) Using the same Twitter mention network, we also partition it into nine graphs using the METIS graph partitioning algorithm to obtain a much smaller network with 9,745 nodes and 50,164 edges. We show the impact of size on our approach using these two networks.

6.2 Experimental setup

6.2.1 Generating noisy anonymized networks

We replicate the original network and assume that it is anonymized. First, we assume that two networks have the same set of nodes while having different but overlapping sets of edges. We prepare an array of networks with different levels of noise to investigate the impact of noise on the performance of the algorithms. We use a common edge-rewiring method [59], which we describe in Algorithm 1. Briefly, the level of noise Θ is the portion of edges that are rewired. For instance, $\Theta = 0.10$ means that 10% of the edges are rewired.

Algorithm 1 Adding noise through edge rewiring

Input: $G_1\langle V_1, E_1\rangle$ **and mixing parameter** Θ
Output: $G_2\langle V_2, E_2\rangle$**: A noisy version of** G_1 **where** $V_1 = V_2$ **but** $E_1 \neq E_2$
copy G_1 to G_2
while $num\_rewired\_edges \leq \Theta \times |E_1|$ **do**
  randomly choose an edge $e_1 \in E_1$
  find $e_2 = (u, v) \in E_2$ which is $e_1 = (u, v)$'s corresponding edge in E_2
  remove $e_2 = (u, v)$ from E_2: $E_2 \leftarrow E_2 \setminus e_2$
  randomly choose a non-existent edge $e = (u, v)$ to be added: $E_2 \leftarrow E_2 \cup e$
end while
return $(G_2\langle V_2, E_2\rangle)$

We use the following levels of noise: $\{0.001, 0.01, 0.05, 0.1, 0.15, 0.2, 0.3, 0.4\}$. While we do examine scenarios with high noise, our results suggest that noise levels greater than 20% seem to set fundamental limitation on the possibility of de-anonymization through structure alone. The range of noise in our simulations is greater than the previously observed noise in previous simulation study [36, 37].

For each noise level, we generate an ensemble of 10 networks for each of the real-world networks. We run the InfoMap community detection algorithm [44] on every graph to detect the community structure.

We also conduct experiments for when the two networks are not identical to each other and may have different sets of nodes and edges. After generating noisy networks using the 'edge-rewiring method', we remove some percentage of the nodes randomly from both the original and the noisy networks so that if the noisy network has been generated by re-wiring 10% of edges, we additionally remove 5% of the nodes from each of the original and the noisy networks. Thus, the resulting networks would have different and overlapping user and edge sets.

6.2.2 Setup for calculating degree of anonymity

The attacker needs an estimate of the performance of our de-anonymization technique to compute the degree of anonymity. We mimic this process by performing several experiments (10 runs) on each data set with a specific level of noise and number of seeds and obtained the overall performance of the de-anonymization algorithm on that particular data set and settings. Then, we perform 10 new de-anonymization experiments on each data set and use the success probabilities from the previous experiments to calculate the degree of anonymity for the data set. Finally, we average the degree of anonymity values for these 10 experiments. We emphasize that the simpler version of the degree of anonymity calculation requires less prior knowledge about the algorithm's performance and it provides an upper bound for the degree of anonymity.

6.2.3 Eccentricity thresholds

For the node-mapping algorithms, we set the eccentricity threshold to 0.1 for all experiments. However, for community mapping, we set this threshold to 0, because we observed that having more mapped communities always gives more correctly mapped nodes. Consequently, this threshold also results in more false positives. However, the effect of false positives in community mapping is limited.

6.2.4 Initial seeds

We assume the attacker has some prior knowledge about a small number of nodes. To simplify the problem we provide the same set of initial seeds for both community-blind (original NS) and

community-aware algorithms instead of running a sophisticated seed detection algorithm. We identify four cliques in both networks (original and perturbed) and randomly choose some of them as seeds. To investigate the sensitivity to the initial seeds, we choose 4, 8, 16, 32, 64, 128 nodes as seeds (they correspond to 1, 2, 4, 8 and 16 cliques respectively). The NS propagation algorithm uses these seeds to perform global propagation while our algorithm employs them to map communities, then performs community-based seed enrichment before running the global propagation step.

6.3 Measuring performance

Although degree of anonymity is the better metric for measuring performance of de-anonymization algorithms, we also measure the number of correctly mapped nodes and incorrectly mapped nodes (as done by Narayanan and Shmatikov [37]), normalized by the total number of nodes in the networks, which we define below.

The *success rate* P_s is defined as the percentage of correctly re-identified users in the network.

DEFINITION 3. Success rate of de-anonymization. *Given graphs $G\langle V, E\rangle$ and $G'\langle V', E'\rangle$, the set of detected mappings M_a, and the true mapping M_t, the success rate P_s is*

$$P_s = \frac{|M_a \cap M_t|}{|V \cap V'|}.$$

Similarly, the *error rate* P_e is defined as the percentage of incorrectly mapped users.

DEFINITION 4. Error Rate of de-anonymization. *Given the same G, G', M, and M_t as in the definition of the success rate,*

$$P_e = \frac{|M_a \setminus M_t|}{|V \cup V'|}.$$

7. RESULTS

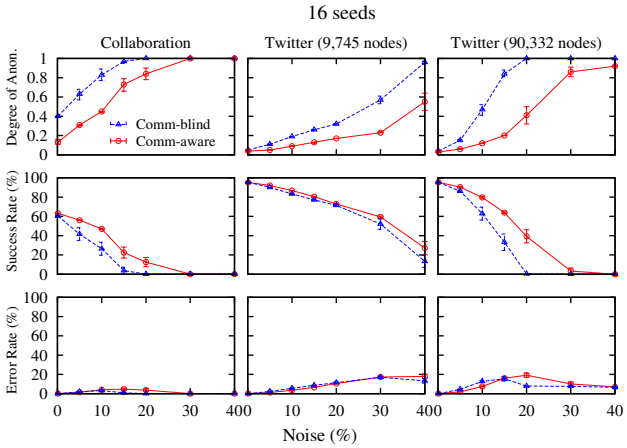
We now report the performance of two algorithms (community-blind NS global propagation vs. our community-aware algorithm). Our results demonstrate that our community-based method can boost the performance of de-anonymization, particularly when (1) there are fewer number of initial seeds, (2) the system size is large, and (3) the noise level is high.

7.1 Impact of noise, seed size, and network size on overall performance

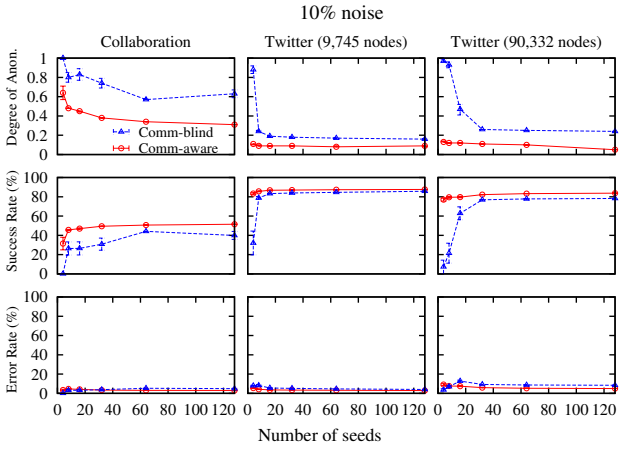
7.1.1 Impact of noise

Figure 3 shows the degree of anonymity $A(G)$ (top row), success rate P_s (middle row), and error rate P_e (bottom row) in terms of noise, number of seeds, and network size.[3] We can see that community-aware algorithm is much more effective in decreasing the anonymity of users in all the networks. For example, in the collaboration network, for all the levels of noises from 0 to 20% (Figure 3(a), left column), the decrease in degree of anonymity when using the community-aware algorithm is about twice as much as that when using community-blind algorithm. Specifically, for 10% noise and 16 seeds, $A(G)$ is 0.45 and 0.83 (or anonymity is 6.81 and 12.57 bits) when using community-aware and community-blind algorithms, respectively. Note that for a collaboration network with 36,458 nodes, the maximum anonymity is equal to 15.15

[3]The percentage of unmapped nodes is simply the difference between 100% and the sum of the success and error rates.



(a) Number of seeds is set to 16



(b) Level of noise is set to 10%

Figure 3: Performance of community-aware and community-blind algorithms on Collaboration, and Twitter mention networks.

bits. As noise increases, both algorithms are less successful in re-identifying users. However, the community-aware algorithm tolerates more noise than the community-blind algorithm. For example, in the collaboration network, with 20% noise, the community-aware algorithm is able to correctly map about 15% of users while community-blind algorithm can barely re-identify any user. The degree of anonymity is 0.84 and 1 (or anonymity is 12.72 and 15.15 bits) when using community-aware and community-blind algorithms, respectively. For 30% and 40% of noise, both algorithms perform poorly and the degree of anonymity is 1.

The same observations can be seen in the Twitter network with 90,332 nodes. In this network, the decrease is not uniform over different levels of noise (Figure 3(a), right column). Both algorithms are highly successful in re-identification of users when the noise is less than 5%. However, the difference between the performance of two algorithms greatly increases when the noise is above 15% and 20%. Specifically, for 15% of noise and 16 seeds, $A(G)$ is 0.2 and 0.84 (or the anonymity is 3.29 and 13.82 bits) when using community-aware and community-blind algorithms, respectively. In other words, the community-aware algorithm *reduces the*

anonymity by 10 additional bits compared to the community-blind algorithm. In this case the success rate of the community-aware algorithm ($\sim$65%) is almost twice as much as that of the community-blind algorithm ($\sim$ 33%). Note that the maximum anonymity for this network is 16.46.

The results on the Twitter network with 90,332 nodes also shows community-aware algorithm is more robust to the noise where even with 20% of noise, it is able to correctly map about 40% of users while the community-blind algorithm can barely re-identify any user. It also reduces the degree of anonymity about 60% (from 1 to 0.41). In other words, the anonymity is reduced by about 10 bits from 16.46 to 6.75. For 30% and 40% of noise, the community-aware algorithm also re-identifies about 3.5% and 0.04% of the users and reduces the degree of anonymity to 0.86 and 0.92, respectively.

7.1.2 *Impact of number of seeds*

Figure 3(b) shows the impact of the number of seeds on $A(G)$, P_s, and P_e. The noise level is set to 10%, while the number of seeds changes from 4 to 128. Both algorithms are more successful when more seeds are provided to them. However, our community-aware approach is more robust to a smaller number of initial seeds. You can see that in all networks, over different numbers of initial seeds, the community-aware algorithm always reduces the anonymity (and re-identifies users) more than the community-blind algorithm. For example, in the collaboration network (Figure 3(b), left column), when the number of seeds is 32, $A(G)$ is 0.38 and 0.74 (or the anonymity is 5.76 and 11.21 bits) when using community-aware and community-blind algorithms, respectively.

In the Twitter network with 90,332 nodes (Figure 3(b), right column), a smaller number of seeds significantly affects the performance of the community-blind algorithm. However, the number of seeds only slightly affects the performance of the community-aware algorithm. For example, when the number of seeds is only four, the community-aware algorithm successfully re-identifies 77% of users while the community-blind algorithm only re-identifies about 7% of the users. Similarly, the degree of anonymity is about 0.13 and 0.97 (and anonymity is 2.14 and 15.97) when using community-aware and community-blind algorithms, respectively. In other words, the community-aware algorithm *decreases the anonymity by 13.83 additional bits* compared to the community-blind algorithm.

7.1.3 *Impact of network size*

Comparing the results for the Twitter network with 90,332 nodes and the Twitter network with 9,745 nodes (Figure 3(b), right and middle columns, respectively) illustrates the impact of size on the performance of both algorithms. Having a smaller network, both algorithms perform better in re-identifying users and tolerating noise. For example, both algorithms are successful at re-identifying users even with 40% noise. However, the performance difference between the community-aware and community-blind algorithms is more obvious when the network is bigger. For example, with 20% noise, $A(G)$ in the larger Twitter network is 0.41 and 1 when using the community-aware and community-blind algorithms. Thus, the difference is about 0.60. However, having 20% noise, $A(G)$ in the smaller Twitter network is 0.17 and 0.32 when using community-aware and community-blind algorithms, respectively, and the difference is only about 0.15. Thus, our community-aware approach is more robust to the size of network.

The error rates of both algorithms show similar trends. Our approach exhibits slightly higher error rate in some cases but most of them occur when the community-blind approach completely fails, and ours correctly identifies many more users.

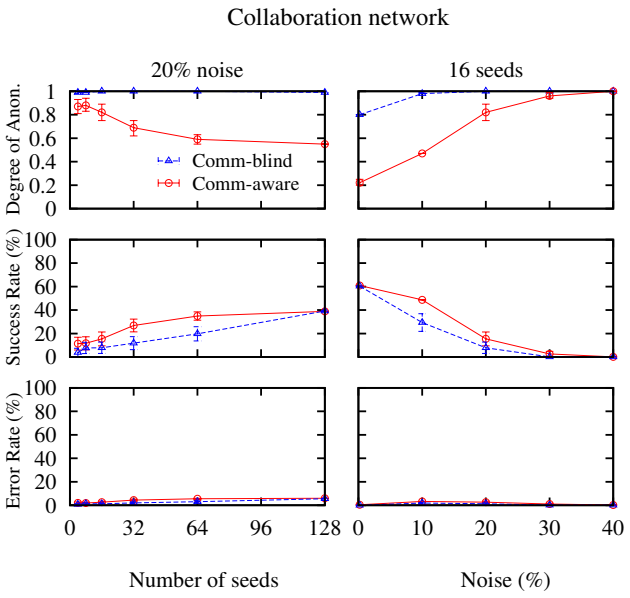


Figure 4: Performance on overlapped data sets

In summary, if one aims to map two networks that are not identical to each other, using our community-based mapping algorithm is almost always guaranteed to reduce the anonymity more and find more successful mappings than the community-blind, global mapping algorithm. In addition, we expect a larger boost as the prior information (seeds) decreases.

7.2 Results for overlapping data sets

Figure 4 demonstrates that the community-aware algorithm also outperforms the community-blind algorithm when the original and the noisy networks do not have the exact same user base. We add noise in two steps: first, we rewire edges as usual to create G'. Then, we remove the same number of randomly selected nodes from both G and G'. For instance, 20% of noise in Figure 4 means that we first rewire 10% of G's edges to generate G' and then remove 10% of nodes (and connected edges) independently from both G and G', making $V \neq V'$. Since $V \neq V'$, the success rate is normalized by $|V \cap V'|$.

We used the formula for the degree of anonymity laid out in Section 5 without any modification for the sake of simplicity. Although one can make the formula more precise by considering cases for the nodes that belong to only V or V', we adopt a simpler approach as our main objective is comparing our method with an existing one.

Figure 4 (left column) shows that the community-aware algorithm reduces the degree of anonymity while the community-blind algorithm fails regardless of the number of seeds. With 20% noise, the community-aware algorithm reduces the degree of anonymity from 0.87 to 0.55 (and anonymity is reduced by 2 and 6.7 bits) with 4 and 128 seeds respectively.

Looking at the right column of Figure 4 (16 seeds with varying noise), we see that the community-blind algorithm fails completely when the noise level is more than 10%, whereas the community-aware algorithm fails when the noise level is more than 30%. The

community-aware algorithm re-identifies 20% more users than the community-blind algorithm when the noise is 10% and it identifies about twice as much when the noise is 20% and 30%. Specifically, when the noise is 10%, and the number of seeds is 16, the success rate is about 49% and 29% for community-aware and community-blind algorithms, respectively. In addition, the community-aware approach reduces the degree of anonymity almost twice as much as the community-blind algorithm from 0.98 to 0.47. In other words, the anonymity is reduced by more than 7 bits from 14.77 to 7.08 bits. Note that the maximum entropy for this network is 15.08.

7.3 Time complexity

The community detection step is not a bottleneck as Infomap's time complexity is estimated to be O(m). For instance, the Louvain method, another popular community detection method has been applied to large graphs with more than 20M nodes and 100M edges. As community mapping and local propagation use the same de-anonymization algorithm (on a much smaller graph), the time complexity of these steps is smaller than the final global propagation step. Therefore, the time-limiting step in the whole process is the final, global propagation step, which is determined by the algorithm being boosted. If the original algorithm can be applied to a given graph, our approach can also be applied to the graph without increasing the overall time complexity.

8. RELATED-WORK

In this section we discuss relevant research about graph anonymization techniques, de-anonymization attacks based on structure alone, and those based on attributes as well. Finally, we discuss applications of network alignment to other fields.

8.1 Graph anonymization

Anonymization techniques can be classified into following four approaches [48]: 1) clustering, 2) clustering with constraints, 3) modification of graph, and 4) hybrid. The 'clustering-based method' applies generalization techniques [9, 60] to aggregate edges or node information so that there are many possible mappings from the clustering back to the graphs, which will always include the original [11]. 'Clustering with constraints' [56] merges all nodes of each cluster to a single node and then decides which edges to include in the anonymized graph so that equivalence class nodes to have some constraints as any two nodes in the original data. The 'modification of graph' approach [54, 55] aims to defeat attacks that exploit known structures in the graph, by adding, removing, and/or swapping some nodes and edges in a social network. In this paper, we focused on this approach and show that applying our community-enhanced de-anonymization approach, sometimes the attacker can de-anonymize the anonymized graph, even when 20% of edges of a graph are randomly added/deleted. The 'hybrid approach' includes combinations of any of the above [51, 58, 60].

A recent approach by Mittal *et al*. [34] perturbs the structure of the graph to provide *link privacy*, where relationships are sensitive while node identities may be known. Their approach preserves community structure to maintain the utility of certain applications (e.g., anonymous routing leveraging trusted links) while hiding individual relationships. While their approach is not focused on *vertex privacy*, we caution the application of their technique to graph de-anonymization as it explicitly retains community structure.

8.2 De-anonymization attacks based on structure

Structural de-anonymization attacks leverage patterns of connectivity in the social network. They can be classified into either 'active' or 'passive.' In active attacks, such as the method suggested by Backstrom *et al*. [5], the adversary chooses its victims to de-anonymize prior to the release of the network. Then, they create a small number of new user accounts('Sybils') and try to form connections to the victims. Because the attacker can impose a unique structure on the subgraph of Sybil nodes, they can be identified from the whole anonymized graph. It has been shown that it is possible to re-identify both Sybil accounts and the victims when the anonymized network is released. However, as Narayanan and Shmatikov [37] point out, active attacks are not scalable because creating thousands of fake user accounts is expensive and this attack may not be as effective in directed graphs when legitimate users do not link back to the sybil nodes.

On the other hand, passive attacks do not actively modify the network. Backstrom *et al*. [5] have also suggested a passive attack where a small coalition of attackers identifies its location in the released network, and tries to discover the existence of edges among users to whom they are linked. This attack is less effective than the active one and works only at a small-scale because the attackers do not choose any user as a victim and they can compromise the privacy of nodes only in their proximities in the network. Narayanan and Shmatikov [37] proposed a large-scale, passive de-anonymization attack technique. This attack, as explained in Section 3, exploits the network structure more extensively than previous attacks. They show that about 30% of the verifiable members of Twitter and Flickr could be recognized with 12% error rate. Their work demonstrated the feasibility of successful re-identification solely based on the network topology. Our results show that our community-based approach can boost the performance of their, and in principle *any*, algorithm under *higher levels of noise*, *larger number of nodes*, or *fewer known seeds*.

8.3 De-anonymization attacks based on other attributes

Attacks that leverage additional information beyond the structure of the networks also have been proposed. For example, it has been shown that one can reveal private information of users by using their public and non-sensitive data [3, 6, 19, 21, 57]. Wondracek *et al*. [53] introduced a technique that narrow down user identity by examining social-network group membership stolen from browsing history. Users who are members of multiple social networks may have a public appearance in one website and be more cautious about their information in another one. Identifying users from different websites and aggregating their information may reveal sensitive information. However, matching users even across two public networks is not a trivial problem. Some of the existing approaches exploit users' activity patterns (Korayem and Crandall [29]), tagging behavior (Lofciu *et al*. [23]), item preferences (Narayanan and Shmatikov [36]), and communication patterns (Diaz et al. [13]).

8.4 Network alignment

Network alignment has been of interest in other fields including Biology [30, 49]. In biological contexts, this technique is used to map two protein interaction networks to infer the function of unknown proteins in each species. The problem is formulated as a quadratic program and solving it is NP-hard. Different approaches [7, 26, 28] have been proposed to relax the constraints or find proper heuristic functions. These approaches have been applied successfully in some applications such as finding common path-ways in biological networks [49, 50] and ontology alignment between Citeseer papers and DBLP papers [22]. However, these networks are of smaller scale compared to social networks and more investigation is needed for large-scale social networks.

9. CONCLUSION

We show how 'mesoscopic' properties of a social network can be leveraged to improve the degree of de-anonymization of anonymized social network datasets. In particular, decomposing the network into 'communities' allows for de-anonymization at a coarser granularity first, and then at the node level. This approach is more robust against added noise to the anonymized data set, and can perform well with fewer known seeds as well as larger networks.

Our work demonstrates the utility of community detection to de-anonymization, thus exposing the importance of structural properties of networks. We would like to highlight that our approach is, in principle, not tied to any specific algorithm; we anticipate other community detection methods and community-blind network alignment algorithms could be 'plugged in' to our framework. Future work could explore ways to apply community detection to other attributes (such as location, language, time, and messages/posts) as well. These attributes can be studied at the community level before drilling down into individual communities, potentially providing even more powerful de-anonymization.

10. ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Award CNS-1115693 and in part supported by Microsoft. We thank Lilian Weng for providing us with network datasets; Nikita Borisov and the anonymous reviewers for their valuable comments; and John McCurley for his editorial feedback.

11. REFERENCES

[1] Infomap clustering tool. `http://www.tp.umu.se/~rosvall/code.html`.

[2] A. Abou-Rjeili and G. Karypis. Multilevel algorithms for partitioning power-law graphs. In *Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International*, pages 10–pp. IEEE, 2006.

[3] A. Acquisti and R. Gross. Predicting social security numbers from public data. *PNAS*, 106(27):10975–10980, 2009.

[4] Y.-Y. Ahn, J. P. Bagrow, and S. Lehmann. Link communities reveal multiscale complexity in networks. *Nature*, 466:761–764, 2010.

[5] L. Backstrom, C. Dwork, and J. Kleinberg. Wherefore art thou r3579x?: Anonymized social networks, hidden patterns, and structural steganography. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 181–190, New York, NY, USA, 2007. ACM.

[6] M. Balduzzi, C. Platzer, T. Holz, E. Kirda, D. Balzarotti, and C. Kruegel. Abusing social networks for automated user profiling. In *Recent Advances in Intrusion Detection*, pages 422–441. Springer, 2010.

[7] M. Bayati, M. Gerritsen, D. F. Gleich, A. Saberi, and Y. Wang. Algorithms for large, sparse network alignment problems. In *Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on*, pages 705–710. IEEE, 2009.

[8] O. Berthold, A. Pfitzmann, and R. Standtke. The disadvantages of free mix routes and how to overcome them. In *Designing Privacy Enhancing Technologies*, pages 30–45. Springer, 2001.

[9] A. Campan and T. Truta. Data and structural k-anonymity in social networks. *Privacy, Security, and Trust in KDD*, pages 33–54, 2009.

[10] D. Chaum. The dining cryptographers problem: Unconditional sender and recipient untraceability. *Journal of cryptology*, 1(1):65–75, 1988.

[11] G. Cormode, D. Srivastava, T. Yu, and Q. Zhang. Anonymizing bipartite graph data using safe groupings. *Proceedings of the VLDB Endowment*, 1(1):833–844, 2008.

[12] C. Diaz, S. Seys, J. Claessens, and B. Preneel. Towards measuring anonymity. In *Privacy Enhancing Technologies*, pages 54–68. Springer, 2003.

[13] C. Diaz, C. Troncoso, and A. Serjantov. On the impact of social network profiling on anonymity. In *Privacy Enhancing Technologies*, pages 44–62. Springer, 2008.

[14] Facebook First Quarter 2013 Results, May 1, 2013. `http://investor.fb.com/releasedetail.cfm?ReleaseID=761090`.

[15] S. Fortunato. Community detection in graphs. *Physics Reports*, 486(3):75–174, 2010.

[16] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.

[17] M. Girvan and M. Newman. Community structure in social and biological networks. *PNAS*, 99(12):7821–7826, 2002.

[18] Google+ Bigger than Twitter with 359 Million Active Users (IGN Report), May 3, 2013. `http://www.ign.com/articles/2013/05/02/report-google-bigger-than-twitter-with-359-million-active-users`.

[19] V. Griffith and M. Jakobsson. Messin' with texas deriving mother's maiden names using public records. In *Applied Cryptography and Network Security*, pages 91–103. Springer, 2005.

[20] R. Gross and A. Acquisti. Information revelation and privacy in online social networks. In *Proceedings of the 2005 ACM Workshop on Privacy in the Electronic Society*, WPES '05, pages 71–80, New York, NY, USA, 2005. ACM.

[21] R. Heatherly, M. Kantarcioglu, and B. Thuraisingham. Preventing private information inference attacks on social networks. 2009.

[22] W. Hu, Y. Qu, and G. Cheng. Matching large ontologies: A divide-and-conquer approach. *Data & Knowledge Engineering*, 67(1):140–160, 2008.

[23] T. Iofciu, P. Fankhauser, F. Abel, and K. Bischoff. Identifying users across social tagging systems. In *Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media (ICWSM'11)*, Barcelona, Spain, July 2011.

[24] B. W. Kernighan and S. Lin. An Efficient Heuristic Procedure for Partitioning Graphs. *The Bell system technical journal*, 49(1):291–307, 1970.

[25] D. Kesdogan, J. Egner, and R. Büschkes. Stop-and-go-mixes providing probabilistic anonymity in an open system. In *Information Hiding*, pages 83–98. Springer, 1998.

[26] G. W. Klau. A new graph-based method for pairwise global network alignment. *BMC bioinformatics*, 10(Suppl 1):S59, 2009.

[27] J. Kleinberg. Anonymized social networks, hidden patterns, and privacy breaches. In *International Workshop and Conference on Network Science (NetSci07)*, May 2007.

[28] G. Kollias, S. Mohammadi, and A. Grama. Network similarity decomposition (nsd): A fast and scalable approach to network alignment. *IEEE Trans. on Knowl. and Data Eng.*, 24(12):2232–2243, Dec. 2012.

[29] M. Korayem and D. J. Crandall. De-anonymizing users across heterogeneous social computing platforms. In *ICWSM*, 2013.

[30] O. Kuchaiev, T. Milenković, V. Memišević, W. Hayes, and N. Pržulj. Topological network alignment uncovers biological function and phylogeny. *Journal of the Royal Society Interface*, 7(50):1341–1354, 2010.

[31] A. Lancichinetti and S. Fortunato. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Physical Review E*, 80(1):016118, 2009.

[32] A. Lancichinetti and S. Fortunato. Community detection algorithms: a comparative analysis. *Physical Review E*, 80(5):056117, 2009.

[33] D. Lazer, A. Pentland, L. Adamic, S. Aral, A.-L. Barabási, D. Brewer, N. Christakis, N. Contractor, J. Fowler, M. Gutmann, T. Jebara, G. King, M. Macy, D. Roy, and M. Van Alstyne. Computational social science. *Science*, 323(5915):721–723, 2009.

[34] P. Mittal, C. Papamanthou, and D. Song. Preserving link privacy in social network based systems. 2013.

[35] A. Narayanan, E. Shi, and B. I. P. Rubinstein. Link prediction by de-anonymization: How we won the kaggle social network challenge. In *IJCNN*, pages 1825–1834. IEEE, 2011.

[36] A. Narayanan and V. Shmatikov. Robust de-anonymization of large sparse datasets. In *Proceedings of the 2008 IEEE Symposium on Security and Privacy*, SP'08, pages 111–125, 2008.

[37] A. Narayanan and V. Shmatikov. De-anonymizing social networks. In *Proceedings of the 2009 30th IEEE Symposium on Security and Privacy*, SP'09, pages 173–187. IEEE Computer Society, 2009.

[38] M. E. Newman. The structure of scientific collaboration networks. *PNAS*, 98(2):404–409, 2001.

[39] S. Nilizadeh, N. Alam, N. Husted, and A. Kapadia. Pythia: A privacy aware, peer-to-peer network for social search. In *Proceedings of the 10th Annual ACM Workshop on Privacy in the Electronic Society*, WPES '11. ACM, 2011.

[40] G. Palla, I. Derényi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–818, June 2005.

[41] A. Pfitzmann and M. Köhntopp. Anonymity, unobservability, and pseudonymity — a proposal for terminology. In *Designing privacy enhancing technologies*, pages 1–9. Springer, 2001.

[42] A. Pothen. Graph partitioning algorithms with applications to scientific computing. In *Parallel Numerical Algorithms*, pages 323–368. Springer, 1997.

[43] D. Rosenblum. What anyone can know: The privacy risks of social networking sites. *Security & Privacy, IEEE*, 5(3):40–49, 2007.

[44] M. Rosvall and C. Bergstrom. Mapping change in large networks. *PLOS One*, 5(1):e8694, 2010.

[45] M. Rosvall and C. T. Bergstrom. Maps of random walks on complex networks reveal community structure. *PNAS*, 105(4):1118–1123, 2008.

[46] K. Schloegel, G. Karypis, and V. Kumar. *Graph partitioning for high performance scientific simulations*. Army High Performance Computing Research Center, 2000.

[47] A. Serjantov and G. Danezis. Towards an information theoretic metric for anonymity. In *Privacy Enhancing Technologies*, pages 41–53. Springer, 2003.

[48] S. Sharma, P. Gupta, and V. Bhatnagar. Anonymisation in social network: A literature survey and classification. *International Journal of Social Network Mining*, 1(1):51–66, 2012.

[49] R. Singh, J. Xu, and B. Berger. Pairwise global alignment of protein interaction networks by matching neighborhood topology. In *Research in computational molecular biology*, pages 16–31. Springer, 2007.

[50] R. Singh, J. Xu, and B. Berger. Global alignment of multiple protein interaction networks with application to functional orthology detection. *Proceedings of the National Academy of Sciences*, 105(35):12763–12768, 2008.

[51] B. Tripathy and G. Panda. A new approach to manage security against neighborhood attacks in social networks. In *Advances in Social Networks Analysis and Mining (ASONAM), 2010 International Conference on*, pages 264–269. IEEE, 2010.

[52] L. Weng, F. Menczer, and Y.-Y. Ahn. Virality Prediction and Community Structure in Social Networks. *Scientific Reports*, 3, Aug. 2013.

[53] G. Wondracek, T. Holz, E. Kirda, and C. Kruegel. A practical attack to de-anonymize social network users. In *Proceedings of the 2010 IEEE Symposium on Security and Privacy*, SP'10, pages 223–238. IEEE Computer Society, 2010.

[54] X. Ying and X. Wu. Randomizing social networks: a spectrum preserving approach. In *SDM*, volume 8, pages 739–750. SIAM, 2008.

[55] X. Ying and X. Wu. On link privacy in randomizing social networks. *Knowledge and information systems*, 28(3):645–663, 2011.

[56] E. Zheleva and L. Getoor. Preserving the privacy of sensitive relationships in graph data. In *Privacy, security, and trust in KDD*, pages 153–171. Springer, 2008.

[57] E. Zheleva and L. Getoor. To join or not to join: the illusion of privacy in social networks with mixed public and private user profiles. In *Proceedings of the 18th international conference on World wide web*, pages 531–540. ACM, 2009.

[58] 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):47–77, 2011.

[59] B. Zhou, J. Pei, and W. Luk. A brief survey on anonymization techniques for privacy preserving publishing of social network data. *ACM SIGKDD Explorations Newsletter*, 10(2):12–22, 2008.

[60] L. Zou, L. Chen, and M. T. Özsu. K-automorphism: A general framework for privacy preserving network publication. *Proceedings of the VLDB Endowment*, 2(1):946–957, 2009.