TRANSACTIONS ON DATA PRIVACY 6 (2013) 107–126

Top Location Anonymization for Geosocial Network Datasets

Amirreza Masoumzadeh*, **James Joshi***

*School of Information Sciences, University of Pittsburgh, IS Building, 135 N. Bellefield Ave., Pittsburgh, PA 15260, USA. E-mail: amirreza@sis.pitt.edu, jjoshi@pitt.edu

Abstract. Geosocial networks such as Foursquare have access to users' location information, friendships, and other potentially privacy sensitive information. In this paper, we show that an attacker with access to a naively-anonymized geosocial network dataset can breach users' privacy by considering location patterns of the target users. We study the problem of anonymizing such a dataset in order to avoid re-identification of a user based on her or her friends' location information. We introduce *k*-anonymity-based properties for geosocial network datasets, propose appropriate data models and algorithms, and evaluate our approach on both synthetic and real-world datasets.

Keywords. Anonymization, Geosocial Networks, Location-Based Social Network

1 Introduction

Advances in positioning technologies and the proliferation of location-enabled mobile devices has recently given rise to Geosocial Networks (GSNs). These systems, which are also referred to as Location-Based Social Networks (LBSNs), are a type of social networking systems that primarily focus on the locations of users and the applications related to them. Users provide their location information to these systems, often using location-enabled mobile devices, and interactions between users and these systems and among users take place based on the location information they have provided. Foursquare, Facebook Places, and Yelp are examples of LBSNs.

Study of social networks is of significant interest of both academia and industry communities. And online social networking systems have made it possible to collect a huge volume of social network data. However, publishing such datasets has its complications regarding users' privacy. Recent research literature on publishing social network datasets has shown effective ways to re-identify nodes of naively anonymized social networks, where only user identifiers are removed. The attacks on naively anonymized datasets range from using nodes' degree in a network as identifying signature [9, 8], to actively implanting nodes in a network [1], to using other publicly available social networks for de-anonymization purposes [12]. A GSN dataset can be more vulnerable to privacy attacks as an adversary can also leverage users' location information for re-identification purposes. Use of location information in re-identification has been well investigated in the context of location-based services, and various privacy preserving protocols and anonymization techniques have been proposed as possible solutions [11, 7, 2, 5].

In this paper, we propose an anonymization approach for GSN datasets, that considers both location and social connections. In particular, we consider datasets collected by systems such as Foursquare where users can befriend other users in the system, and check into location venues. Such a GSN dataset is essentially a social network of users, i.e., users and their relations with each other, and a series of logged locations for each of the users in the network. The log may contain specific location information and the times at which a user has reported those locations. Various privacy attacks can be launched against a naively anonymized GSN dataset. We focus on the re-identification attack based on adversary's background knowledge about user locations. In a recent large-scale study of location data collected from cellphone users [16], Zang and Bolot report that a significant percentage of cellphone users are uniquely identifiable based on their top two or three locations. Moreover, Noulas et al. [13] report that home and corporate/office places are the top two locations from which people perform check-ins in Foursquare. Motivated by these studies, we argue that check-in locations in a GSN dataset can be used by attackers to re-identify a target user. Moreover, the location information of the target's connections may strengthen the possibility of re-identification. For instance, an attacker may know that the target frequently visits a certain coffee shop, and also knows about his/her workplace. In addition, the attacker may know the home address of a colleague of the target at work. Such background knowledge may easily enable re-identification of the target. We formulate the above problem and propose k-anonymization techniques to thwart such attacks.

In this work, unlike in the existing literature on social network anonymization, we do not consider friendship structure of a target node as a feasible background knowledge for adversaries. That is mainly because our observations on GSNs such as Foursquare show that users on average have much smaller number of friends in these systems than in general purpose social networking systems such as Facebook. Therefore, their social connections hardly represent their real friendship network. Instead, we focus on the location information revealed by social connections of a user that can assist in re-identification. Our contributions in this work can be summarized as follows:

- We formulate a simple and abstract model of GSNs, based on which we introduce two notions of anonymity for GSN datasets, i.e., $\mathcal{L}_k$-anonymity and $\mathcal{L}_k^2$-anonymity.
- We present a family of two location models for GSNs, called *top regions* and *top venues*, that we use as the underlying data models for our algorithms.
- We propose clustering algorithms for anonymizing a GSN dataset according to the notions of $\mathcal{L}_k$-anonymity and $\mathcal{L}_k^2$-anonymity, including a new approach to split oversize clusters.
- We present experimental results on a dataset collected from a real-world GSN, as well as a synthetic dataset generated based on previous studies.

To the best of our knowledge, this is the first work to study anonymization of GSN datasets. However, as mentioned earlier, related work has studied anonymization in the context of location-based services and social networks, separately. The rest of the paper is structured as follows. In Section 2, we introduce the notions of location equivalence and corresponding anonymity properties for GSN datasets. Section 3 presents appropriate location models for GSNs. In Section 4, we propose algorithms to anonymize a GSN dataset based on our proposed anonymity properties. We report results obtained by running the algorithms on real-world and synthetic datasets in Section 5. We briefly survey the related work in Section 6, and finally, conclude the paper in Section 7.

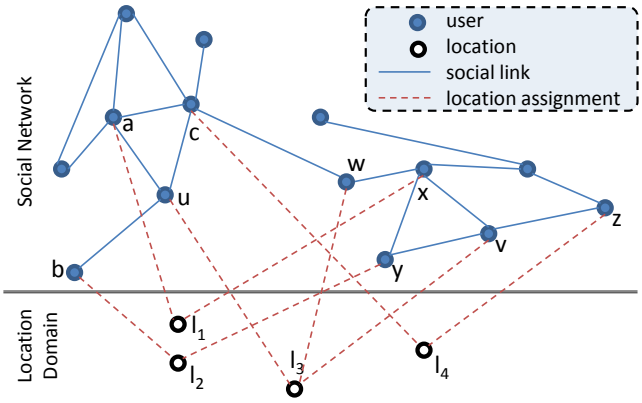


Figure 1: A small sample GSN

2 *k*-Anonymity for GSN Datasets

Location information contained about the users in GSN dataset may help an attacker to re-identify users. Such re-identification can lead to further privacy breaches such as disclosure of a user's complete location information and other privacy-sensitive attributes. We introduce two anonymity properties based on the concept of *k*-anonymity to prevent such a linking attack based on users' location information. While the first property protects inference based on the target user's location, the second property also considers location of her friends.

For our proposed framework, we define a GSN dataset as follows.

Definition 1. A GSN dataset is a 4-tuple $G = \langle V, E, L, \mathcal{L} \rangle$, where V is a set of users, $E \subseteq V \times V$ is a set of friendship links between users, L is a domain for location information, and $\mathcal{L} : V \to L$ is a function that assigns location information to users.

In the above definition, the location domain abstractly refers to the type of location information that can be used for a linking attack. The location values can be as simple as street addresses, or as complex as a spatio-temporal patterns of user movements. In this paper, we focus on a specific location model, top locations, which will be detailed in Section 3. Figure 1 illustrates a small GSN dataset based on the above definition. Users and the connections between them form a social network. Each user is also assigned a location value in the location domain.

Since we are interested in anonymity based on location information, we introduce the following notion of location equivalence for GSN users.

Definition 2. Given a GSN $\langle V, E, L, \mathcal{L} \rangle$, we say that users u and $v \in V$ are *$\mathcal{L}$-equivalent* ($u \equiv_{\mathcal{L}} v$) if they are assigned the same location information, i.e., $\mathcal{L}(u) = \mathcal{L}(v)$.

In Figure 1, users u, v, and w are $\mathcal{L}$-equivalent since all are assigned to location value l_3. Similarly, users a and x are $\mathcal{L}$-equivalent. Intuitively, anonymity can be provided by ensuring enough $\mathcal{L}$-equivalent users for every user in a GSN dataset. The following property captures the notion of *k*-anonymity based on $\mathcal{L}$-equivalence.

Definition 3. A GSN $\langle V, E, L, \mathcal{L} \rangle$ is *$\mathcal{L}_k$-anonymous* iff for every user $v \in V$, there are at least $k-1$ other users that are $\mathcal{L}$-equivalent to v. Formally, $\forall v \in V \exists v_1, v_2, \ldots, v_{k-1} \in V, v \equiv_{\mathcal{L}} v_1 \equiv_{\mathcal{L}} v_2 \ldots \equiv_{\mathcal{L}} v_{k-1}$.

Based on the above property, an attacker cannot re-identify a user with a certainty greater than $1/k$ even if he obtains the target's location information.

$\mathcal{L}_k$-anonymity considers only a target's location information as adversary's background knowledge and no further information about social relationships of a target. However, for a GSN dataset, an attacker may leverage knowledge about location information of a target's friends to perform more successful re-identification attacks. For instance, suppose an attacker knows that the target is working at the IS department in the University of Pittsburgh. Also, the attacker knows that she has a friend that works in the CS department and another friend that frequently visits a specific coffee shop. Although, $\mathcal{L}_k$-anonymity helps keep the user anonymous based on her location, her friends' location information can further help the attacker to reduce the anonymity set. The following location equivalence relation takes into account a user's friends' location information.

Definition 4. Given a GSN $\langle V, E, L, \mathcal{L}\rangle$, we say that users u and $v \in V$ are *$\mathcal{L}^2$-equivalent* ($u \equiv_{\mathcal{L}^2} v$) iff, in addition to themselves, their adjacent users in the social network are also $\mathcal{L}$-equivalent, i.e., $(\mathcal{L}(u) = \mathcal{L}(v)) \wedge (\{\mathcal{L}(u')|\langle u, u'\rangle \in E\} = \{\mathcal{L}(v')|\langle v, v'\rangle \in E\})$.

In Figure 1, users u and v are $\mathcal{L}^2$-equivalent; because they are $\mathcal{L}$-equivalent, and the set of u's friends' location information, i.e., $\{l_1, l_2, l_4\}$, is equal to those of v's. We define the *k*-anonymity property based on $\mathcal{L}^2$-equivalence as follows.

Definition 5. A GSN $\langle V, E, L, \mathcal{L}\rangle$ is *$\mathcal{L}^2_k$-anonymous* iff for every user $v \in V$, there are at least $k-1$ other users that are $\mathcal{L}^2$-equivalent to v. Formally, $\forall v \in V \exists v_1, v_2, \ldots, v_{k-1} \in V, v \equiv_{\mathcal{L}^2} v_1 \equiv_{\mathcal{L}^2} v_2 \ldots \equiv_{\mathcal{L}^2} v_{k-1}$.

$\mathcal{L}^2_k$-anonymity is obviously a stronger property than $\mathcal{L}_k$-anonymity, and consequently costlier to guarantee. Note that in Definition 4, the neighbors' locations are not required to map one-to-one to each other's. Only the collection of neighbors' locations needs to be the same. The latter is a more relaxed constraint than the former. We believe that it is more realistic to assume that the attacker may know about the location of a number of a target's friends rather than the exact locations of all her friends.

3 Top Locations Models

The anonymity properties defined in Section 2 are abstract with regards to the location model, i.e., no specific model is assumed. In this section, we introduce location models that describe top locations which are somehow revealing about a user, assisting an attacker to re-identify her. Top locations may be based on the frequency of a user's visit (e.g., workplace, a coffee shop on the way or close to work, etc.), or could be the ones that are more uniquely identified with the user (e.g., home location). Such models give us a realistic and reasonably powerful representation of an adversary's background knowledge about a target's location information in GSNs, and is also supported by previous studies in the literature. According to the data reported about cell sector location of cellular phone users [16], more than 80% of them are uniquely identifiable based on their top (most frequent) three locations. Based on the top two locations, about 45% are uniquely identifiable, and more than 85% have at most one other user with the same top location. Note that a cellular sector is of coarser granularity than specific venues that people report in GSNs such as Foursquare. Specific to GSNs, based on the data reported in [13], home and office locations are the two top venues people report on average in Foursquare, which seems good enough

for their re-identification. An advantage of considering a top locations model is that it is not very complex for performing *k*-anonymziation.

We consider two variations of top location models, *Top Regions* and *Top Venues*, each of which are suitable for different purposes and expected outcomes of anonymization. The *Top Regions* model considers a pure geospatial model of locations, while the *Top Venues* model considers named places with geospatial properties. For the *Top Regions* model, anonymizations can be achieved by reporting spatially cloaked regions instead of specific geographic coordinates. However, for the *Top Venues* model, we consider reporting a set of nearby venues instead of one specific venue. This is due to the prime importance of the concept of venue in such a dataset, and the fact that they are considered to be publicly known.

3.1 Top Regions Model TR_m

This model is useful for GSN datasets that contain self-reported geolocations of users, which might be at different levels of granularity. For simplicity, we assume that a location can fit in a rectangular region.

Definition 6. The TR_m location domain contains values as m-tuples such as $\langle r_1, r_2, \ldots, r_m \rangle$, where every r_i is a rectangular region.

In the above definition, a *rectangular region* is a geographic area surrounded by a rectangle. Rectangular region r is represented using a 4-tuple $\langle x, y, w, h \rangle$, where x and y are geographic coordinates of the top-left corner, and w and h are width and height of the area, respectively. Zero width and height determine an exact point as a region. We use dot notation to represent members of a tuple, e.g., $l.r_1$ represents region r_1 belonging to location l.

We consider spatial cloaking to anonymize a set of regions, and replace them with more coarse grain regions. For this purpose, k users that have close enough m top regions can be considered as one $\mathcal{L}$-equivalency class, and the same cloaked m top regions can be reported for all.

3.2 Top Venues Model TV_m

This model is suitable for GSN datasets taken from systems with pre-specified venues, e.g., check-in-based GSNs such as Foursquare.

Definition 7. The TV_m location domain contains values as m-tuples such as $\langle v_1, v_2, \ldots, v_m \rangle$, where every v_i is a venue with defined geographic coordinates.

We do not consider reporting a spatial cloaked region as a good option for anonymizing this type of GSN datasets. Location-equivalency classes should still be formed around users with geographically close top venues. However, as the anonymized location values, the sets of venues belonging to the class members and corresponding to each of a user's venues will be reported.

4 Top Locations Anonymization Algorithms

The first step in anonymizing a dataset using our model is to select representative top locations for each user as her location information. In this section, we review two intuitive

options for such a selection. We propose to use clustering to anonymize GSN datasets based on the top locations model. The goal is to form users into clusters of size at least k; then report the same location information for all the users in the same cluster. Our optimization goal is to minimize distortion of the location information in the anonymized dataset. However, the *Top Regions* and *Top Venues* models demand different strategies to achieve this goal. Therefore, we propose different notions of distance for each model. We propose generic clustering approaches that can work with either of the location models and corresponding distance measures.

4.1 Top Locations Selection

Both the studies on location patterns anonymity in GSNs [16, 13], that we referred to in Section 3, provide evidence on importance of places of frequent visit in identifying users. Therefore, our first heuristic for selecting top locations is to select the m most frequently reported locations for each user as her top locations information. Let $c(v, l)$ be the number of reports of user v in location l. A location l_i belongs to a user's m most frequent locations $L_{mf} = \{l_1, \ldots, l_m\}$ if and only if

$$\forall l_i \in L_{mf} \ \nexists l' \notin L_{mf}, c(v, l') > c(v, l_i).$$

We also mentioned briefly about the possibility of re-identification based on locations that are most unique to a user and not necessarily the most frequent. Intuitively, an attacker can filter out users better using such a background knowledge about its target. We employ the concept of *term frequency - inverse document frequency* in the information retrieval literature for selecting the most unique features to each user. A location l_i belongs to a user's m most unique locations $L_{mu} = \{l_1, \ldots, l_m\}$ if and only if

$$\forall l_i \in L_{mu} \ \nexists l' \notin L_{mu}, \frac{c(v, l')}{\sum_{u \in V} c(u, l')} > \frac{c(v, l_i)}{\sum_{u \in V} c(u, l_i)}.$$

Note that the type of the top locations information model (TR_m & TV_m) is independent of how those top locations are selected as described in this section.

4.2 Distance Measures for Top Locations Models

Central to our clustering algorithm is a distance measure that can determine which users should be clustered together. While anonymization involves introducing uncertainty to the dataset to achieve the desired privacy property, we need to do so by minimizing introduced inaccuracies. A suitable distance metric can ensure that the anonymized location values have minimum possible inaccuracy compared to the original values.

In the top regions model, regions should be replaced with their cloaked versions. Cloaking multiple location points is usually performed by finding a minimum bounding rectangle that includes all the locations [11, 5]. The cloaking process in TR_m is more complicated since each user's location consists of m regions. For example, consider users u and v's TR_2 locations depicted in Figure 2. Each user has r_1 and r_2 regions as a point. Combining the two location values results in a third location value with two regions. However, there are alternative ways for performing this. Each of the u's regions can be considered corresponding to any of v's regions for the cloaking purpose. This results in two different *region correspondences*: $\{\langle \mathcal{L}(u).r_1, \mathcal{L}(v).r_2 \rangle, \langle \mathcal{L}(u).r_2, \mathcal{L}(v).r_1 \rangle\}$ and $\{\langle \mathcal{L}(u).r_1, \mathcal{L}(v).r_1 \rangle, \langle \mathcal{L}(u).r_2, \mathcal{L}(v).r_2 \rangle\}$. As shown in Figure 2, the former correspondence results in smaller cloaked areas for the

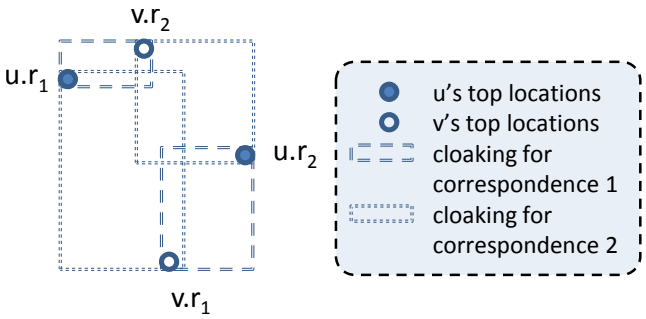


Figure 2: Example of Cloaking in the TR_m Model

results than the latter. Therefore, it is a better option in terms of preserving location accuracy. The correspondence that results in the smallest cloaking among the alternatives can be used as a natural distance measure for our clustering approach. Location information of two nodes can be combined to form larger cluster of nodes that meet the anonymity property, while minimum cloaking is applied to preserve the location information as much as possible.

We now define the distance measure for the top regions model. When two TR_m values are to be combined into one cluster, corresponding regions should be combined (cloaked) into one region. Since there are m different regions in each TR_m location tuple, there will be $m!$ different combinations for region correspondences. As mentioned earlier, we consider the minimum expansion in area that results from cloaking based on any of such correspondences as the distance measure between two TR_m values. The following definition formally captures the notion of distance in TR_m. Operator $\otimes$ in the following definition is an element-by-element binary operator for equally sized tuples, which outputs the set of 2-tuples from the elements of first and second operands. For instance, $\langle A, B, C\rangle \otimes \langle 1, 2, 3\rangle = \{\langle A, 1\rangle, \langle B, 2\rangle, \langle C, 3\rangle\}$. Also, function *Perm* calculates the set of all permutation tuples for a given set. For instance, $Perm(\{1, 2, 3\}) = \{\langle 1, 2, 3\rangle, \langle 1, 3, 2\rangle, \langle 2, 1, 3\rangle, \langle 2, 3, 1\rangle, \langle 3, 1, 2\rangle, \langle 3, 2, 1\rangle\}$.

Definition 8. Given TR_m values t and s, we define the distance between them, D_{TR_m}, as follows:

$$D_{TR_m}(t, s) = \min_{C \in Perm(\langle 1,\ldots,m\rangle)} \sum_{\langle i,j\rangle \in \langle 1,\ldots,m\rangle \otimes C} MBRA(t.r_i, s.r_j)$$

where *MBRA* calculates the area of *minimum bounding rectangle area* of two regions.

In the above definition, a correspondence is formed based on each permutation of numbers $1, \ldots, m$, and the sums of the minimum bounding rectangle areas of the corresponding regions are calculated. The minimum value among the calculated values for all correspondences is selected as the distance between the two locations. In the example depicted in Figure 2, *MBRA* calculates the area of each dashed rectangle. It is easy to see that the sum of areas of the bounding rectangles associated with correspondence 1 is smaller than that for correspondence 2. Therefore, the sum of the areas of the line-dashed rectangles is regarded as the distance between the locations of u and v.

The distance measure for the top venues model has a simpler formulation. Each user's location in the TV_m model includes m venues. Combining two such location values involves finding the appropriate correspondence between venues and reporting the set of corresponding venues in place of each of the original venues. Therefore, the goal is to minimize

the distance between venues in each set. In the following definition, we use the concepts of operator $\otimes$ and function *Perm* as defined above.

Definition 9. Given TV_m values t and s, we define the distance between them, D_{TV_m}, as follows:

$$D_{TV_m}(t,s) = \min_{C \in Perm(\langle 1,\ldots,m\rangle)} \sum_{\langle i,j\rangle \in \langle 1,\ldots,m\rangle \otimes C} ED(t.v_i, s.v_j)$$

where *ED* calculates the *euclidean distance* between geographic coordinates of two venues.

4.3 GSN $\mathcal{L}_k$-Anonymization

Using the distance measures defined in Section 4.2, we use a clustering approach similar to the Union-Split method [14], to create clusters with minimum k nodes, and produce an anonymized value for each cluster. However, we modify the algorithm to accommodate our data models, which are significantly different than the original numerical values it was proposed for. The input to our algorithm is a GSN dataset based on a top location model, for which location information is selected based on one of the criteria in Section 4.1. Note that according to our top location models, every user needs to be represented by m top locations. Therefore, any user with less than m unique locations should be either retracted from the dataset, or assigned to enough dummy location values to fulfill the constraint. Algorithm 1 shows the pseudo code for $\mathcal{L}_k$-anonymization. The algorithm is a special hierarchical agglomerative clustering algorithm that terminates when each cluster has at least k members. The same algorithm works for both top locations models by adjusting the distance and center computation corresponding to each model.

The algorithm starts with each user as a separate cluster, with the cluster center set as her location. In each iteration, the distance between every pair of clusters is computed. The pair of clusters with the minimum distance are selected to be merged. The cluster center of the merger is calculated according to the top locations model in use. As in Definitions 8 and 9, distance is calculated based on a correspondence with minimum value. In order to calculate cluster centers, it is important to know which top locations of each user in a cluster correspond to those of others. Therefore, we keep track of such correspondences and update them whenever users are assigned to new clusters. In order to calculate a cluster center, corresponding components of them members are combined. As a simple example, let have cluster $c_1 = \{v_1, v_2, v_3\}$, and user-to-cluster correspondences $P_1 = \{1, 3, 2\}$, $P_2 = \{2, 3, 1\}$, and $P_3 = \{3, 1, 2\}$. The cluster center's first top location is formed by combining $v_1.r_1$, $v_2.r_2$, and $v_3.r_3$. Its second top location is formed by combining $v_1.r_3$, $v_2.r_3$, and $v_3.r_1$. Finally, its third top location is formed by combining $v_1.r_2$, $v_2.r_1$, and $v_3.r_3$. The actual method to combine these components is dependent on the top locations model in use. In case of TR_m, locations are combined by calculating the minimum bounding rectangle of the corresponding locations. In case of TV_m, locations are combined by calculating the middle point of the corresponding locations. If the merger has more than two times the desired size of anonymity clusters, i.e., more than $2k$, the cluster needs to be broken into two (preferably equally-sized) clusters. We perform this to avoid ending up having clusters significantly larger than the required anonymity size (Step 11 in Algorithm 1). The larger the size of an anonymity set, the more location distortion. Instead of *special k-means clustering* proposed in [14] we take a much simpler approach to split an oversized cluster, which is outlined in Algorithm 2. In the last part of the algorithm (steps 15–23), location information of each user is anonymized according to its cluster. In case of TR_m, a cluster center represents cloaked top locations that can represent all its members. In case of TV_m,

Algorithm 1 $\mathcal{L}_k$*-Anonymize*

Input: GSN dataset $G = \langle V, E, L, \mathcal{L} \rangle$ and the anonymization parameter k.
Output: $\mathcal{L}_k$-anonymous dataset $G' = \langle V, E, L, \mathcal{L}' \rangle$.

Initialize clusters C as $\{c_j | v_j \in V, c_j \leftarrow \{v_j\}, c_j.center \leftarrow \mathcal{L}(v_j)\}$
Initialize user-to-cluster correspondences P for each v_i as $P_i \leftarrow \langle 1, \ldots, m \rangle$
while $\exists c \in C, |c| < k$ **do**
    for all $c_i, c_j \in C$ **do**
        Calculate distance $D_L(c_i.center, c_j.center)$ ▷ using Definition 8 or 9
    end for
    Merge c_x and c_y into c_m, where $D_L(c_x.center, c_y.center)$ is minimum
    Update P for c_m members according to the correspondence that resulted in the minimum distance above
    Calculate $c_m.center$ according to c_m members' correspondences
    if $|c_m| \geq 2k$ **then**
        Split c_m and replace it by c_{m1} and c_{m2} (s.t. $|c_{m1}| \geq k$ and $|c_{m2}| \geq k$) using Algorithm 2
    end if
end while
$G' \leftarrow G$
for all $c \in C$ **do**
    for all $u \in c$ **do**
        if $L = TR_m$ **then**
            $\mathcal{L}'(u) \leftarrow c.center$
        else if $L = TV_m$ **then**
            $\mathcal{L}'(u) \leftarrow \langle \{u_i.v_{P_i[1]} | \forall u_i \in c\}, \ldots, \{u_i.v_{P_i[m]} | \forall u_i \in c\} \rangle$
        end if
    end for
end for
return G'

each of a user's m top locations is replaced by a set that contains the corresponding top locations of all other members within the same cluster.

The *Split-Cluster* algorithm (Algorithm 2) divides one oversized cluster into two clusters with minimum size k. It starts by assigning two randomly chosen users as cluster centers. In each iteration, every user is assigned to its nearest cluster. Then if one of the clusters is undersized, it takes enough close users from the other cluster to reach size k. Finally, the center is recalculated for each cluster. The algorithm iterates until either both clusters have minimum size k in their initial member assignment or the loop reaches one of its stopping criteria. The first stopping criterion assesses whether further iterations improves the clusters or not. For this purpose, we calculate an error that measures changes in user-to-cluster distances due to changing in cluster assignment in Step 24. If we cannot fairly improve such an error compared to the last iteration (measured by small constant δ), we consider the formed clusters final and return them. The second criterion simply puts a limit on the number of iterations using constant *MI*.

Theorem 1. Algorithm 1 outputs an $\mathcal{L}_k$-anonymous dataset as per Definition 3.

Proof. All the users are members of clusters since the algorithm starts with every user as a single cluster and merges them iteratively. Also, the main loop in the algorithm (steps 3–13) does not terminate until all the clusters have at least k members, i.e., $\forall c \in C, |c| \geq k$. Therefore, assigning the same location information to all the members of the same cluster in steps 15–23 ensures that for every user v in a cluster c there are at least $k-1$ other users $v_1, v_2, \ldots, v_{k-1} \in c$ where $v \equiv_{\mathcal{L}} v_1 \equiv_{\mathcal{L}} v_2 \ldots \equiv_{\mathcal{L}} v_{k-1}$. □

The time complexity of an optimized implementation of Algorithm 1 is $O(m!n^2 \log n)$, where m is the parameter of the TR_m or TV_m model, and n is the number of users. Note that in practice, the $O(m!)$ will not be significant for small values of m and can be disregarded as we will explain later. With the assumption of rare need for split, the main loop of the algorithm (steps 3–13) iterates n times to merge the clusters, given that in each iteration two clusters are merged into one. An optimized implementation of distance calculation (steps 4–6) can be achieved by maintaining a sorted list of distances for each cluster, and updating only the entries related to the merger cluster at each iteration. Calculating the initial distances takes $O(m!n^2)$ (which should be performed before the main loop), and updating them takes $O(m!n \log n)$ in each iteration. In these complexity orders, the $O(m!)$ factor is due to each distance calculation according to Definition 8 or 9. However, we do not expect value m to be large (much more than 3) as a reasonable background knowledge for an attacker. Although, it would not break our algorithm, a large m value will result in a highly uncertain dataset which might not be useful any more. Such intuition is also based on our observations in the studies reported in the literature [16, 13]. Hence, the $O(m!)$ factor in our complexity will not be significant in practice. *Split-Cluster* (Algorithm 2) has also time complexity $O(m!n \log n)$. In each iteration of the main loop of Algorithm 2, we need to calculate the distances of the two cluster centers to each user and store them sorted. Given that the number of iterations is limited to a constant, the complexity is bounded by $O(m!n \log n)$. We want to note that in our experiments, Algorithm 2 quickly converges and terminates due to the error thresholding approach in less than 5 iterations, even for large inputs. The final loop in Algorithm 1 has complexity $O(n)$. Therefore, considering the number of iterations of the main loop, and the complexity of its distance calculations and the splitting approach as described above, the time complexity of Algorithm 1 is $O(m!n^2 \log n)$.

Algorithm 2 *Split-Cluster*

Input: A set of users V', their corresponding location function $\mathcal{L}$, and minimum cluster size k.

Output: Two clusters c_1 and c_2 that together include users V' and each one has at least k members, as well as the user-to-cluster correspondences P' for those users.

Initialize user-to-cluster correspondences P' for each $v_i \in V'$ as $P'_i \leftarrow \langle 1, \ldots, m \rangle$
$c_1.center \leftarrow \mathcal{L}(v_1 \in V')$
$c_2.center \leftarrow \mathcal{L}(v_2 \in V')$ ▷ v_1 and v_2 are selected at random.
error $\leftarrow +\infty$
iteration $\leftarrow 1$
repeat
    for all $v_j \in V'$ **do**
        Calculate distances $D_L(c_1.center, v_j)$ and $D_L(c_2.center, v_j)$ according to the corresponding measure to location model L (Definition 8 or 9)
        Assign v_j to the cluster with smaller distance
        Update P'_j according to the correspondence that resulted in the smaller distance above
    end for
    if $|c_1| < k$ **then**
        $x \leftarrow 1$ ▷ x is index for undersized cluster.
        $y \leftarrow 2$ ▷ y is index for oversized cluster.
        $t \leftarrow k - |c_1|$ ▷ t is the number of users to be added to c_x.
    else if $|c_2| < k$ **then**
        $x \leftarrow 2$
        $y \leftarrow 1$
        $t \leftarrow k - |c_2|$
    else
        break
    end if
    Let T be set of of first t users in c_y sorted in ascending order of their distance to c_x
    Assign T to c_x
    Update P' for users in T according to the correspondences that results in the smallest distance to c_x
    Calculate $c_x.center$ and $c_y.center$ based on P'
    last-error $\leftarrow$ *error*
    error $\leftarrow \frac{1}{t} \sum_{v_j \in T} \left(D_L\left(c_x.center, v_j\right) - D_L\left(c_y.center, v_j\right)\right)$
until $(|\textit{last-error} - \textit{error}| < \delta)$ or $(\textit{iteration} > MI)$
return c_1, c_2, and P'

4.4 GSN $\mathcal{L}_k^2$-Anonymization

We build on our proposed algorithm in Section 4.3 to make a GSN dataset $\mathcal{L}_k^2$-anonymous based on Definition 5. In fact, we rely on the clustering and cloaking performed by the $\mathcal{L}_k$-anonymization algorithm. Algorithm 1 clusters users and makes them $\mathcal{L}_k$-equivalent in each cluster. For $\mathcal{L}_k^2$-anonymization, we further modify edges in the social network to ensure the $\mathcal{L}_k$-equivalence of neighbors of users in each cluster. We consider two alternative approaches for performing this. In the first approach, the edges in the original network are preserved and only new edges are inserted. Therefore, no edge information in the social network is lost during anonymization. In the second approach, edges are both inserted and removed to assure the anonymity property. The rationale behind this approach is to avoid too much increase in the size of the social network due to anonymization.

Algorithm 3 shows the pseudo code for $\mathcal{L}_k^2$-anonymization. We first perform the steps in Algorithm 1 to achieve the resultant clusters and the $\mathcal{L}_k$-anonymous dataset. Next, for any two clusters c_i and c_j, we form set E_{ij} of inter-cluster edges between the clusters (edges connecting a member from one to a member from the other). If the number of the inter-cluster edges is less than a threshold θ, the algorithm removes those edges from the graph. Otherwise, the algorithm ensures that all the users in one cluster have at least a neighbor in the other cluster, by adding missing edges. If a user needs to have a neighbor from another cluster, one of the members of that cluster is randomly chosen and an edge is created between them. Note that c_i and c_j can refer to the same cluster in the special case. In that case, the edges E_{ij} are called intra-cluster rather than inter-cluster. However, the same procedure is applicable to guarantee $\mathcal{L}_k$-equivalent neighbors for the users.

The inter-cluster edge count threshold θ is significant in controlling the behavior of Algorithm 3. If we want to choose the approach of preserving all edges in the original social network θ should be set to zero. This ensures that Step 6 of the algorithm is never executed. Therefore, only edges will be inserted. In contrast, if we set θ to any non-zero value, it tends to keep the inter-cluster edges that seem to be important according to the threshold, and remove the other less essentials. For a non-zero threshold, we suggest setting θ to half the size of the smaller cluster between c_i and c_j. The rationale is to keep the change in the network minimal. If half of the members are already involved in inter-cluster relationships, it may be better to keep them and add the rest that satisfy our anonymization criteria rather than to remove them. Otherwise, removal of those sparse connections will introduce less changes in the social network.

In the following, we prove correctness of the proposed algorithm for $\mathcal{L}_k^2$-anonymization.

Theorem 2. Algorithm 3 outputs an $\mathcal{L}_k^2$-anonymous dataset as per Definition 5.

Proof. The first step of the algorithm generates an $\mathcal{L}_k$-anonymous dataset, and corresponding clusters with $\mathcal{L}$-equivalent members, according to Theorem 1. We need to show that every member of a cluster is also $\mathcal{L}^2$-equivalent to other members of the same cluster. Consider an arbitrary cluster c and one of its members $v \in c$. For any other cluster member $u \neq v \in c$, $u \equiv_{\mathcal{L}} v$ according to Theorem 1. For each edge adjacent to v, say $\langle v, v' \rangle \in E'$, there exists at least an adjacent edge to u, say $\langle u, u' \rangle \in E'$, where v' and u' are in the same cluster. This is assured by inserting the edges in the social network in steps 10 and 15. Therefore, set $\{\mathcal{L}(u') | \langle u, u' \rangle \in E'\}$ will be equal to set $\{\mathcal{L}(v') | \langle v, v' \rangle \in E'\}$, which completes the proof for $u \equiv_{\mathcal{L}^2} v$. □

The main part of Algorithm 3 (steps 3–19) has time complexity $O(n^2)$. Considering the minimum cluster size k, there will be at most $\lfloor n/k \rfloor$ clusters in C. Therefore the main loop

Algorithm 3 $\mathcal{L}_k^2$*Anonymize*

Input: GSN dataset $G = \langle V, E, L, \mathcal{L}\rangle$ and the anonymization parameter k.
Output: $\mathcal{L}_k^2$-anonymous dataset $G' = \langle V, E', L, \mathcal{L}'\rangle$.

$E' \leftarrow E$
$\mathcal{L}_k$*Anonymize*(G)
for all $c_i, c_j \in C$ **do**
  $E_{ij} \leftarrow \{\langle u, v\rangle \in E | \exists u \in c_i, v \in c_j\}$
  if $|E_{ij}| < \theta$ **then**
    $E' \leftarrow E' \setminus E_{ij}$
  else
    for all $u \in c_i$ **do**
      if $\nexists v \in c_j, \langle u, v\rangle \in E$ **then**
        $E' \leftarrow E' \cup \{\langle u, v\rangle\}$, where $v \in c_j$ is randomly chosen
      end if
    end for
    for all $v \in c_j$ **do**
      if $\nexists u \in c_i, \langle u, v\rangle \in E$ **then**
        $E' \leftarrow E' \cup \{\langle u, v\rangle\}$, where $u \in c_i$ is randomly chosen
      end if
    end for
  end if
end for
return G'

runs $O(n^2/k^2)$ times. Each cluster has less than $2k$ nodes, due to the splitting mechanism. So enumerating members of every pair of clusters for edge insertion purpose has complexity $O(k^2)$. Therefore, the overall complexity of the main part is $O(n^2/k^2) \times O(k^2) = O(n^2)$. Given that Algorithm 1 has higher time complexity than this, the time complexity of Algorithm 3 is bounded by Algorithm 1's complexity, i.e., $O(m!n^2 \log n)$.

5 Experimental Results

5.1 Datasets

We conduct our experiments on a real-world and a synthetic dataset. The former is a partial dataset collected from a check-in-based GSN, *Gowalla*. Gowalla was a service similar to Foursquare which has been acquired by Facebook recently. The original dataset [4] contains friendship links and public check-in data between Feb. 2009 and Oct. 2010. We extract a partial network, constituting a community of 1016 users and 2757 edges, detected by a label propagation scheme. This dataset is used for experimenting with the TV_m model and will be referred to as `GW` from here on. We extract a TV_3 GSN from `GW` dataset based on most frequent venues, in order to evaluate anonymization using the TV_m model. We note that the method of top location selection as described in Section 4.1, i.e., selecting the most frequent or most unique locations, does not affect the performance evaluation of our anonymization algorithms. Because the resulting top location dataset of using each method can be considered as an independent dataset. Therefore, without loss of generality, we only present our experimental results on the most frequent venues.

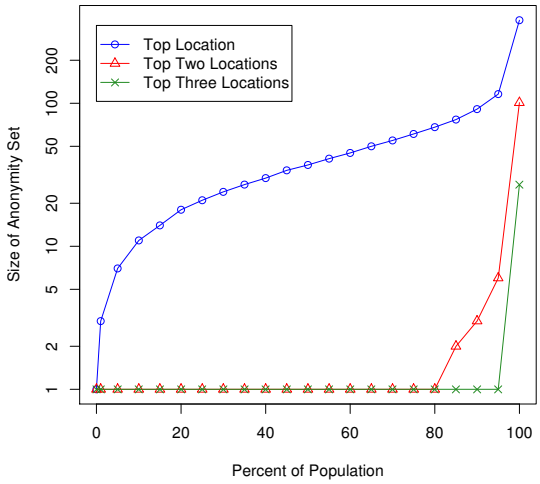
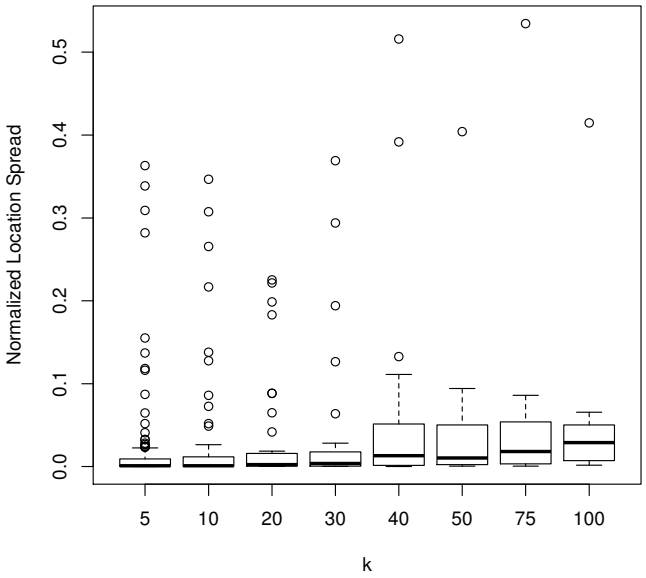


Figure 3: Size of the anonymity sets in dataset ST when top 1, 2, or 3 locations are revealed

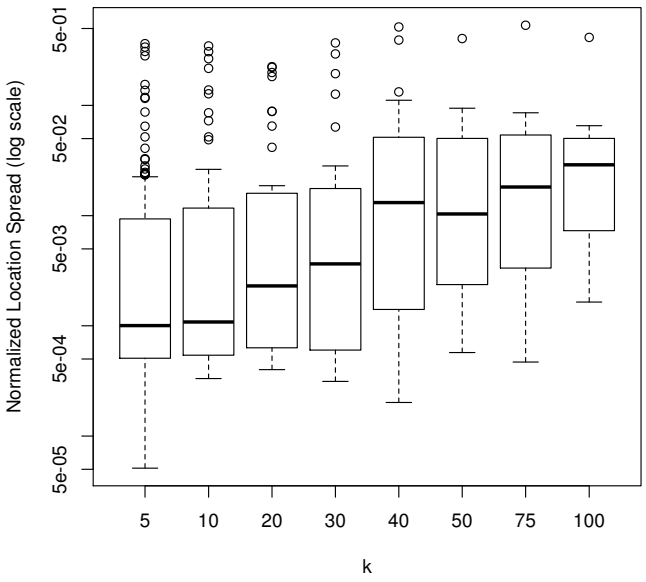
We generated the second dataset synthetically. In the generation process, we used the statistics reported in [16] on a nation-wide cellular network in order to have realistic distributions regarding identifiability of users with regards to their locations. We experiment on this dataset as a TR_3 model. More specifically, we leveraged the distribution of anonymity group sizes with regards to considering one, two, and three top locations per cellular sector. However, we scaled down the dataset size to 1500 users (the data reported in [16] is about 25 million users). Figure 3 shows the size of anonymity sets for different percentile of users in our dataset, depending on the condition of revealing one, two, or three top locations. In our synthetic dataset, each of the three user's regions is a point, randomly chosen on a 1000 by 1000 unit square-shaped area. These points are converted to cloaked regions as the result of the anonymization algorithm. We refer to this dataset as ST from here on.

5.2 $\mathcal{L}_k$-Anonymization

We ran the $\mathcal{L}_k$-anonymization algorithm using different k values, in the range of 5 to 100, on the TV_3 model of GW dataset. To evaluate our location anonymization performance in the context of the TV_3 model, we measure an error value that determines the spread of venues in each cluster. We first compute a center of the cluster corresponding to each of the three venues in each user's location. We calculate the error as the mean distance of users to the cluster center averaged across the three venue dimensions. We also normalize the error by the spread of all location values in the dataset. Note that a more accurate baseline for normalization of error values for our location model would need to consider grouping locations in m different dimensions, which would intuitively be a larger error value than our current baseline. All distances are measured on the geocoordinates of the venues based on Haversine formula. Lower location spread error ensures that the corresponding venues used for $\mathcal{L}_k$-anonymity are also physically close to each other. The box plots in Figure 4 show the location spread error calculated for clusters corresponding to the runs on different values of k. As suggested by the boxes in Figure 4a the $\mathcal{L}_k$-*Anonymize* algorithm has relatively low location spread error. However, there exist outliers that may significantly increase the average error. Our algorithm does not consider suppression of outlier values



(a) Location Spread Error (Normal Scale)



(b) Location Spread Error (Log Scale)

Figure 4: Location Preservation Performance in $\mathcal{L}_k$-Anonymization of TV_3-Based GW

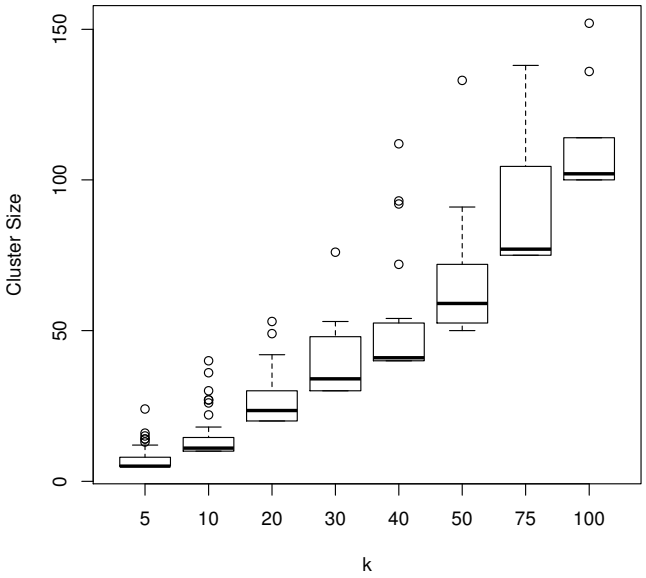


Figure 5: Clustering Performance in $\mathcal{L}_k$-Anonymization of TV_3-Based GW

in the dataset, in order to avoid removing possibly important information. A careful suppression approach can certainly improve the results. In the log scale plot of Figure 4b, it is more visible how location spread error lightly increases with the increase of k.

Figure 5 reports the distribution of cluster sizes for different values of k. The distributions verify that the algorithm produces clusters with minimum size k as required by the anonymity property. Because of the nature of clustering approach, we end up with clusters that are sightly larger than the minimum k value.

We ran Algorithm 1 on the TR_3 model of ST dataset, with values of k between 3 and 50. We calculate the average area of the regions in users' location information after anonymization, as a measure of performance of our algorithm in preserving location information. Figure 6 depicts this measure as a result of choosing different k values. As expected, the area has an increasing trend with the increase of k. However, we notice that the region areas do not change significantly after $k = 15$ in our dataset. We also consider the insignificant decrease in area size due to the greedy nature of our clustering approach, which does not guarantee to produce optimum results.

5.3 $\mathcal{L}_k^2$-Anonymization

Since our $\mathcal{L}_k^2$-anonymization algorithm is based on clusters created by the $\mathcal{L}_k$-anonymization algorithm, the analysis of location preservation performance in Section 5.2 directly applies to it as well. In order to evaluate the preservation/distortion of social network structure by the algorithm, we consider two simple measures. The *edge count ratio* calculates the proportion of edges in the resulting network to that of the original. The *edge overlap ratio* calculates the fraction of edges that have been preserved from the original network in the resulting network. These measures apply to both datasets as the choice of location model is not relevant. Figure 7 depicts these measures for the $\mathcal{L}_k^2$-anonymization result of GW dataset, based on different values of k. The measure is plotted for both suggested values

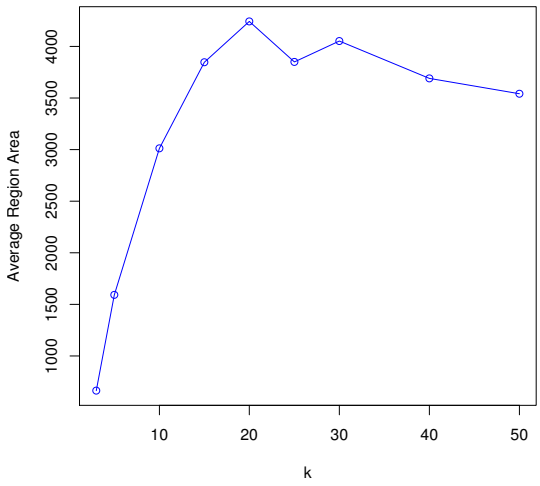
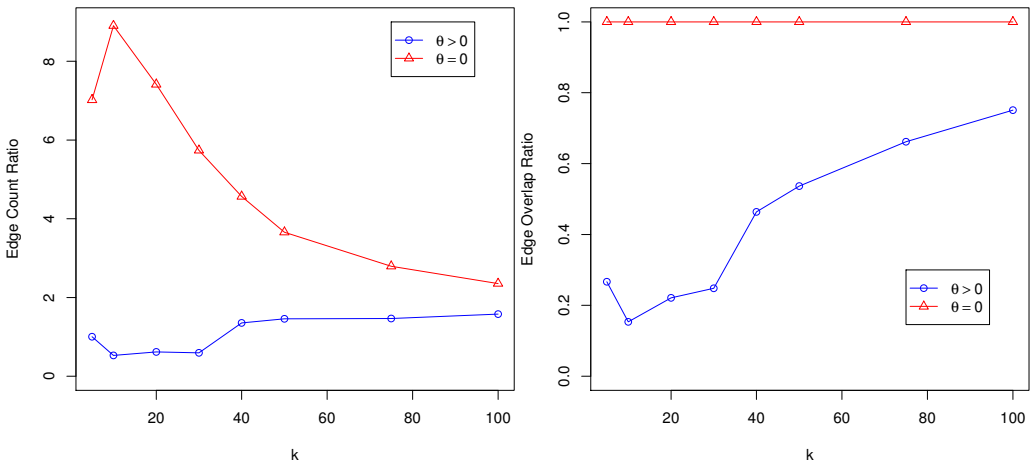
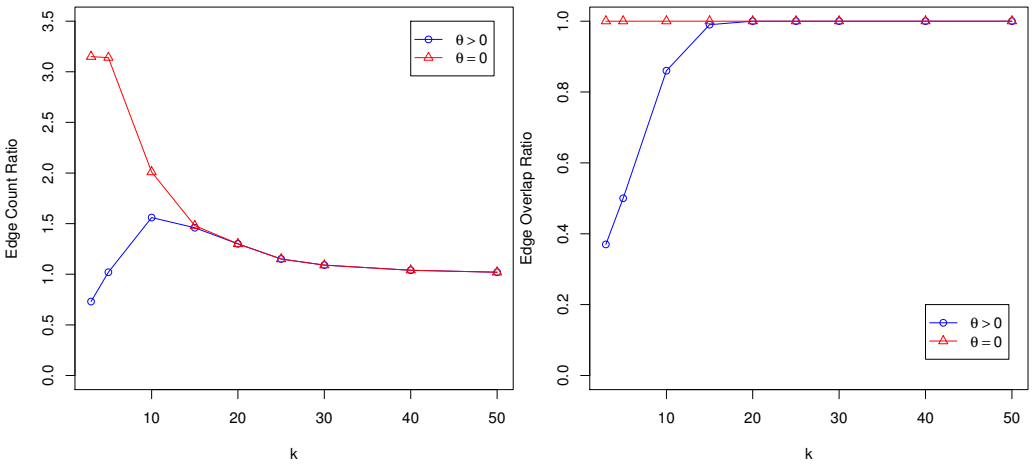


Figure 6: Location Preservation Performance in $\mathcal{L}_k$-anonymization of TR_3-based ST



(a) Edge Count Ratio (b) Edge Overlap Ratio

Figure 7: Network Preservation Performance in $\mathcal{L}_k^2$-Anonymization of TV_3-Based GW



(a) Edge Count Ratio (b) Edge Overlap Ratio

Figure 8: Network Preservation Performance in $\mathcal{L}_k^2$-Anonymization of TR_3-Based ST

for threshold δ. We ran Algorithm 3, i.e., zero and half of the size of the smaller cluster in a pair. A choice of $\delta = 0$ ensures preservation of the original edges, but also adds much more edges to the graph compared to the case of non-zero δ.

Figure 8 shows the edge count and overlap ratios for $\mathcal{L}_k^2$-anonymization of the TR_3-Based ST dataset. Interestingly, both approaches of δ threshold seem to perform similar at larger values of k, i.e., $k = 15$ or higher, in terms of both edge count ratio and overlap.

6 Related Work

Various anonymization techniques have been proposed in the literature based on location cloaking, i.e., reporting a larger area rather than a user's exact location, in order to provide *k*-anonymity for location-based service users. Approaches such as *New Casper* [11], *Privé* [7], and *PrivacyGrid* [2] cloak each query's location to include at least *k* other users. Therefore, an attacker's certainty about a specific user's issuance of a query is at most $1/k$. Other approaches such as *CliqueCloak* [5] collect and submit k queries with the same cloaked location at the same time to an LBS. The problem in LBS anonymization techniques is slightly different than the problem discussed in this paper as it deals with anonymizing queries one at a time. In contrast, anonymizing a dataset with location information involves more rigorous optimization as it needs to anonymize all the records at the same time. Moreover, our anonymization technique deals with a more complex location model, i.e., top m locations, rather than a single location for each user. Another obvious contrast is consideration of network connections in our approach. It is worth to mention that there exist other approaches to privacy for location-based services in the literature that avoid anonymization and use cryptography-based private information retrieval techniques instead [6].

Re-identification attacks on social network datasets and anonymization techniques to prevent them have been considered as very important emerging research problems recently. Backstrom et al. present a family of active/passive attacks that work based on uniqueness of some small random subgraphs embedded in a network [1]. Hay et al. show significantly

low *k*-anonymity in real, naively anonymized social networks when considering structural queries such as degree of a target node as adversarial background knowledge [9, 8]. The social network anonymization approaches that have been proposed in the literature can be categorized into two groups: graph generalization and graph perturbation. In *generalization techniques* [8, 17, 3], the network is first partitioned into subgraphs. Then each subgraph is replaced by a supernode, and only some structural properties of the subgraph alongside linkage between clusters are reported. In *perturbation techniques* , the network is modified to meet desired privacy requirements. This is usually carried out by adding and/or removing graph edges. The perturbation methods include randomly adding/removing edges [9, 15], and providing *k*-anonymity in terms of node degrees [10, 14] and node neighborhood [18]. Naturally, the focus of social network anonymization approaches is on anonymizing structural patterns such as node degree and neighborhood, and not on information associated with the nodes. However, some approaches such as [18] also consider anonymizing node labels based on generalization trees. Nevertheless, none of these methods deal with location data associated with nodes as needed for anonymization of GSN datasets.

7 Conclusions

Geosocial networks are growing fast and becoming popular social networking tools, which naturally brings up the privacy issues with regards to the huge amount of the location-rich data collected by these systems. Proper anonymization mechanisms are necessary to be developed specifically for GSN datasets since the location information is more complex and harder to anonymize and preserve at the same time, compared to other simpler data attributes.

In this work, we suggested that both users' and their friends' location information can be misused by an attacker for re-identification purpose. We formalized and proposed two notions of anonymity with regards to location information in GSN datasets, namely, $\mathcal{L}_k$-anonymity and $\mathcal{L}_k^2$-anonymity. We formalized a family of top location models, the top regions model and the top venues model, geared to different types of GSN datasets. Also, we proposed algorithms for anonymizing a GSN dataset based on these notions, and analyzed their performance in the context of anonymization and preservation of information, using two different datasets.

Acknowledgements

This research has been partially supported by the US National Science Foundation awards IIS-0545912 and DUE-0621274. We would like to thank our anonymous reviewers for their valuable comments and suggestions.

References

[1] L. Backstrom, C. Dwork, and J. Kleinberg. Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography. In *Proc. 16th Int'l conference on World Wide Web*, pages 181–190, Banff, Alberta, Canada, 2007. ACM.

[2] B. Bamba, L. Liu, P. Pesti, and T. Wang. Supporting anonymous location queries in mobile environments with PrivacyGrid. In *Proc. 17th Int'l Conference on World Wide Web*, pages 237–246. ACM, 2008.

[3] A. Campan and T. M. Truta. A clustering approach for data and structural anonymity in social networks. In *Proc. 2nd ACM SIGKDD Int'l Workshop on Privacy, Security, and Trust in KDD (PinKDD'08)*, 2008.

[4] E. Cho, S. A. Myers, and J. Leskovec. Friendship and Mobility : User Movement in Location-Based Social Networks. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1082–1090. ACM, 2011.

[5] B. Gedik and L. Liu. Protecting Location Privacy with Personalized k-Anonymity: Architecture and Algorithms. *IEEE Transactions on Mobile Computing*, 7(1):1–18, 2008.

[6] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-l. L. Tan. Private queries in location based services: anonymizers are not necessary. In *Proc. ACM SIGMOD Int'l Conference on Management of Data*, number ii, pages 121–132. ACM, 2008.

[7] G. Ghinita, P. Kalnis, and S. Skiadopoulos. {PRIVE}: anonymous location-based queries in distributed mobile systems. In *Proc. 16th Int'l Conference on World Wide Web*, pages 371–380, New York, NY, USA, 2007. ACM.

[8] M. Hay, G. Miklau, D. Jensen, D. Towsley, and P. Weis. Resisting structural re-identification in anonymized social networks. *Proc. VLDB Endow.*, 1(1):102–114, 2008.

[9] M. Hay, G. Miklau, D. Jensen, P. Weis, and S. Srivastava. Anonymizing Social Networks. Technical Report 07-19, University of Massachusetts Amherst, 2007.

[10] K. Liu and E. Terzi. Towards identity anonymization on graphs. In *Proc. ACM SIGMOD Int'l Conference on Management of Data*, pages 93–106, Vancouver, Canada, 2008. ACM.

[11] M. F. Mokbel, C.-y. Y. Chow, and W. G. Aref. The New Casper: Query Processing for Location Services without Compromising Privacy. In *Proc. 32nd Intl Conference on Very Large Data Bases*, number 1, pages 763–774, Seoul, Korea, 2006. ACM.

[12] A. Narayanan and V. Shmatikov. De-anonymizing Social Networks. In *Security and Privacy, IEEE Symposium on*, pages 173–187, Oakland, CA, USA, 2009. IEEE.

[13] A. Noulas, S. Scellato, C. Mascolo, and M. Pontil. An empirical study of geographic user activity patterns in foursquare. In *ICWSM 11*, pages 570–573, 2011.

[14] B. Thompson and D. Yao. The union-split algorithm and cluster-based anonymization of social networks. In *Proc. 4th Int'l Symposium on Information, Computer, and Communications Security (ASIACCS '09)*, pages 218–227, Sydney, Australia, 2009. ACM.

[15] X. Ying, K. Pan, X. Wu, and L. Guo. Comparisons of randomization and K-degree anonymization schemes for privacy preserving social network publishing. In *Proc. 3rd SIGKDD Workshop on Social Network Mining and Analysis (SNA-KDD 09)*, Paris, France, 2009. ACM.

[16] H. Zang and J. C. Bolot. Anonymization of Location Data Does Not Work: A Large-Scale Measurement Study. In *Proc. 17th annual int'l conference on mobile computing and networking (MobiCom' 11)*, pages 145–156, Las Vegas, NV, USA, Sept. 2011. ACM Press.

[17] E. Zheleva and L. Getoor. Preserving the Privacy of Sensitive Relationships in Graph Data. In F. Bonchi, E. Ferrari, B. Malin, and Y. Saygin, editors, *Proceedings of the International Workshop on Privacy, Security, and Trust in KDD (PinKDD'07)*, Lecture Notes in Computer Science, pages 153–171. Springer Berlin Heidelberg, 2008.

[18] B. Zhou and J. Pei. Preserving Privacy in Social Networks Against Neighborhood Attacks. In *IEEE 24th Int'l Conference on Data Engineering (ICDE '08)*, pages 506–515, Cancun, Mexico, 2008. IEEE.