Hindawi
Wireless Communications and Mobile Computing
Volume 2020, Article ID 8892321, 13 pages
https://doi.org/10.1155/2020/8892321

Research Article

Privacy-Enhancing Preferential LBS Query for Mobile Social Network Users

Madhuri Siddula,[1] Yingshu Li,[1] Xiuzhen Cheng,[2] Zhi Tian,[3] and Zhipeng Cai[1]

[1]*Computer Science, Georgia State University, Atlanta, Georgia 30302, USA*
[2]*Computer Science, George Washington University, Washington, DC 20052, USA*
[3]*Computer Science, George Mason University, Fairfax, VA 22030, USA*

Correspondence should be addressed to Zhipeng Cai; zcai@gsu.edu

Received 10 June 2020; Revised 30 July 2020; Accepted 13 August 2020; Published 1 September 2020

Academic Editor: Kim-Kwang Raymond Choo

While social networking sites gain massive popularity for their friendship networks, user privacy issues arise due to the incorporation of location-based services (LBS) into the system. Preferential LBS takes a user's social profile along with their location to generate personalized recommender systems. With the availability of the user's profile and location history, we often reveal sensitive information to unwanted parties. Hence, providing location privacy to such preferential LBS requests has become crucial. However, the current technologies focus on anonymizing the location through granularity generalization. Such systems, although provides the required privacy, come at the cost of losing accurate recommendations. Hence, in this paper, we propose a novel location privacy-preserving mechanism that provides location privacy through k-anonymity and provides the most accurate results. Experimental results that focus on mobile users and context-aware LBS requests prove that the proposed method performs superior to the existing methods.

1. Introduction

Online Social Network (OSN) has become an inevitable part of our lives. We use them to connect with people and find new places, things, news, games, and many more. As OSN has become a one-stop-shop for any information, they have found their spot on our mobile phones to ease the access. Such applications are called mobile social networks (MSN), and they have been a huge hit ever since they were introduced. According to the recent survey by Statista [1], there are over 61% of users of MSNs in North America. It is expected that the number of users who use MSN reaches around 2.46 billion in 2017 and 3.02 billion in 2021. This number is almost one-third of the current earth's population. One of the significant advantages of MSN is its access to precise location information. This information is used for various purposes like geotagging, augmented reality, and location-based services.

LBS have further eased the access of information. Users, querying for nearby restaurants, and geotagging friends, all use LBS in one way or another. When a user requests an LBS, the request is sent to the location service provider (LSP). This acts like a central server that gathers all the information. However, we also require all the business providers to upload their accurate location information so the LBS can give exact results. All the business owners in the example provided in Figure 1 are restaurants, which are returned for the user. However, every restaurant has a profile, i.e., cuisine, timings, takeout or dine-in, and reviews.

The concept of preferential or context-aware LBS came into light in the recent past with the introduction of local businesses into many social networks. To enhance the user experience of LBS, personalized recommendations are generated. However, these recommendations can be best produced by considering the user's social profile and history. For example, if a user visits Starbucks every day at 8 AM, it is most likely that he likes coffee at a specific location. In this situation, Google can suggest travel time to nearby Starbucks at 7:30 AM for the user's convenience and favorite drink on

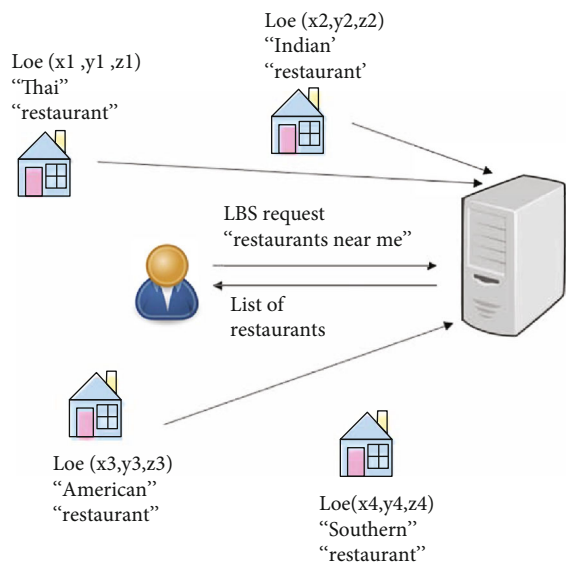

Figure 1: An example LBS query with the user requesting nearby restaurants.

the Starbucks app to order. This scenario, although it provides useful information to the user, has many privacy concerns, including the data leak to untrusted third parties.

Preferential LBS uses machine learning techniques to identify which business listings are best suited to the user's liking. This requires not only the profile of the business listing but also the user's profile and history. For example, if a person searches for a nearby restaurant, the result contains all the restaurants in the current area but sorted according to the user's history or interest. If the user preference is Asian cuisine, then the results are sorted accordingly, and so on. Hence, accurate results for LBS are only possible if we have exact location information and correct user profile. However, providing such information might lead to other privacy leaks. Let us consider a scenario of the man-in-the-middle attack or a server attack, as shown in Figure 2. In both these scenarios, the attacker finds out not only user profile but also their precise location. This leads to both security and privacy leaks in the system. In this paper, we focus on the privacy leak that is the user location leak in such attack scenarios. Additionally, if a series of locations are revealed to the attacker, the trajectory of the user path is disclosed.

To evade the above-discussed problems, we have introduced a novel privacy-preserving algorithm that anonymizes LBS request in a mobile environment. Our contributions can be summarized as follows:

(1) To the best of our knowledge, we are the first to consider profile generalization instead of location granularity for providing privacy to the MSN user

(2) Privacy through attribute clustering to preserve k -anonymity

(3) Local clustering to avoid complexity at the server

(4) Dynamic clustering to include mobility of the users

(5) This work also addresses the issue of knowledge graph attack

The rest of the paper is organized as follows. Section 2 introduces the basic definitions and nomenclature used in this paper. Section 3 provides existing methods that are proposed for protecting the privacy of an MSN user. The problem statement is defined in Section 4. The proposed novel dynamic clustering is explained in Section 5. Analysis of the proposed methods is discussed in Section 6. In Section 7, we evaluate the algorithm by comparing it with existing technologies. Finally, conclusions and future work are discussed in Section 8.

2. Definitions and Nomenclature

2.1. Definitions

2.1.1. Profile Generalization. Our aim in this proposed method is to generalize the profile of the user rather than generalizing the location information. Hence, we need to know how much of the profile has been generalized. It is a common understanding that if we generalize all the user's profiles to a single profile that profile is generalized 100%; then, the privacy maintained is high. However, the LBS results will be far from accurate. Also, if we reveal the profile is not changed at all, then the profile generalization is 0%; there is a chance that the user is identified by an attacker correctly. Hence, we need a mechanism to quantify the profile generalization and how much privacy is maintained with that generalization. To do that, we provide a profile generalization calculation method.

$$\text{Profile generalization of user } u_i\left(g_{u_i}\right) = \frac{\text{dist}\left(u_i, C_{u_i}\right)}{\text{dist}\left(u_i, \text{GP}\right)} * 100, \tag{1}$$

where u_i is the user profile, C_{u_i} is the cluster that the user belongs to, and GP is the general profile.

2.1.2. Information Loss. The information loss of a user u_i, IL_i, is measured using the profile generalization done for that user. The more profile generalization happened for a user by the proposed methods, the more information loss occurred, and hence, it reduces the accuracy of the LBS results. To calculate information loss, we use the distance between the user's original profile and the generalized profile.

$$\text{Information Loss of } u_i(\text{IL}_i) = \text{dist}\left(u_i, g_{u_i}\right). \tag{2}$$

The total information loss over all the users can be computed as follows:

$$\text{Information Loss (IL)} = \sum_{i=1}^{n} \text{IL}_i. \tag{3}$$

2.1.3. Accuracy. It is essential to understand how accurate the results are with the profile generalization. Since we have LBS

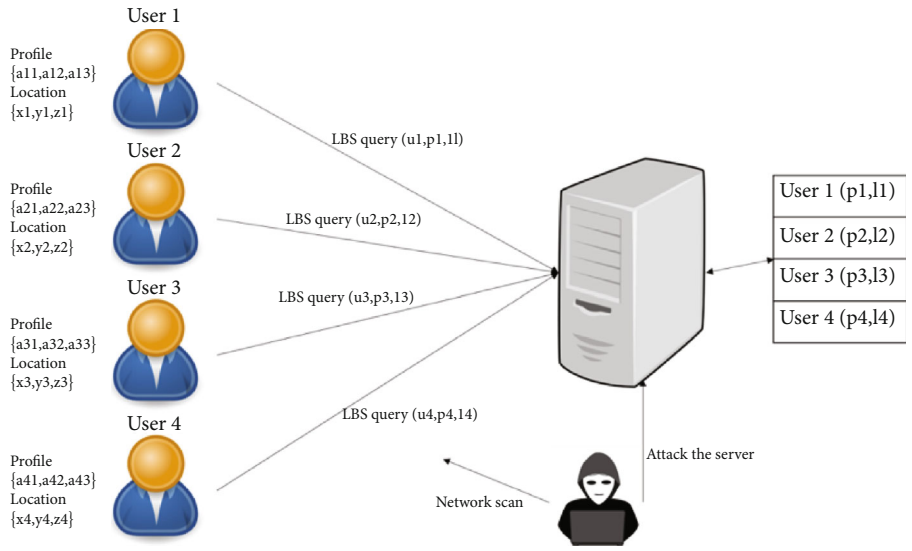

Figure 2: An example LBS query with the user requesting nearby restaurants.

queries, a query that takes location information will give us accurate results. This is because we are sending exact location information in our LBS query. However, for a preferential search, we consider the user's profile to sort the results according to the user's interest. This is how we ensure that the results are relevant to the user and not generalized results. We use an algorithm proposed by [2] to perform the preferential search. This algorithm lets us filter the results further based on the user profile. We measure the accuracy of the preferential search results by using the below method:

$$\text{Accuracy} = \frac{\text{number of different results}}{\text{total number of results}} * 100. \quad (4)$$

2.1.4. Execution Time. It is essential to understand how much computation time is required for the algorithm to run. This is because LBS queries are real-time, and the user can request a query at any point in time. Although our algorithm is not a postquery method, we do update the clusters based on the user movement. So, the user might request for a query at the exact moment that the election and clustering method starts executing. Hence, we need to understand what is the execution time for these methods.

2.2. Nomenclature. Table 1 describes all the notations used in the paper and their description.

Table 1: Symbols and notations.

Notation	Description
G_t	Temporal social graph at time "t"
V	Vertices set
E	Edge set
A	Attribute set
a_i	i^{th} attribute
L_t	Location set (locations of all the users) at time "t"
E_t	Edge set at time "t"
u_i	User i
o_i	Obfuscated user i
l	Location
n	Number of users
R	Radius considered for local group formation
k	Desired anonymity level
C_{u_i}	Cluster that the user belongs to
g_{u_i}	Profile generalization of the user u_i
IL_i	Information loss of the user u_i

3. Related Works

Privacy-preserving mechanisms in the mobile environment have been summarized in [3, 4]. These surveys point out that the obfuscation is considered heavily on user location [5]. Researchers in [6, 7] have further classified these obfuscation techniques for our understanding.

(1) Differential Privacy (DP). These methods focus on preserving the privacy of the user by generating noise. However, when such noise is added, while aggregating data from two users, the probability of whether the user's data is included has a bound. Initial methods using DP included a centralized database where trusted devices collect data [8]

(a) Local Differential Privacy (LDP). Recently, more research has been done in terms of LDP. Some of these research focus on utility aware privacy-preserving data collection [9], and geoindistinguishability [10]. Although these methods seem to cloak the user's location, it has some problems, including random cloaking and user identification via knowledge graph attack

(b) Distributed Differential Privacy (DDP). These methods focus on generating a distributed noise [11, 12]. The assumption in these settings is that a user has access to a group of users. DDP aims at aggregating the user's data with the data from the set of users using the generated noise. These methods lack the assumption that the attacker poses as a naive user and gains access to this subset of users

(2) Location-Based Grouping. One of the simplest ways to group the users is based on their exact location. However, this grouping heavily depends on calculating the distance between two users. Following are some of the categories in which these grouping techniques are based on:

(a) Relative Distance Only. These methods aim to provide a relative distance between two users or a user and a place. That is, instead of revealing their exact location on the LBS, these methods calculate a secure relative distance method [9, 13, 14]. The access permission for the location can be set to public or controlled access

(b) Setting the Minimum Accuracy Limit. In this method proposed by [15], authors set a limit on the location accuracy. That is, we ask the users if they are okay with getting skewed results, and if so, what percent of skewness is acceptable. Then, we decide the location accuracy based on the user response. This type of location obfuscation is used in skout where the localization accuracy is set to 1 mile, and in WeChat, Momo uses around 100 m to 10 m

(c) Setting the Localization Coverage Limits. This is another technique to obtain location obfuscation. In this technique proposed by [16], authors have provided a localization coverage limit. This ensures that the location is obfuscated to a certain level defined by the user

(3) Cryptography-Based Approaches. These methods are based on the principle that to find the distance between two entities, we use encryption techniques. This helps us in generating a distance value without revealing the actual location information [17–19]. This kind of approach can achieve higher computational accuracy and provide a reliable privacy guarantee for users but exists the problem of sizeable computational cost and communication overheads

As can be seen above, the obfuscation techniques provided are based on the user preference in geolocations. However, none of the above methods consider combining user locations for cloaking their locations. Hence, researchers have considered some privacy-aware proximity detection approaches, such as spatial generalization-based methods. In spatial generalization methods like [20, 21], we divide the user space into different levels or different grids. Thus, there are multiple users per grid, and the locations are generalized over the grid. We use this grid location information for the LBS rather than the precise location of the user. However, dividing the user space into such multilevel partitions takes huge processing time and requires robust computation capable devices. Hence, such methods are not practical on a mobile device.

Recent papers in mobile social network privacy also talk about the existing attacks and solutions provided to address them [22]. Authors in [22] have proposed a method to calculate the similarity score between shared locations and real-world locations based on the datasets that they have collected. Authors in [23] review the literature on privacy-preserving ad hoc mobile social networks. Authors in [24, 25] talk about identity and location privacy with a technique called multiple pseudonyms. This technique focuses on generating a pseudo id for patients to hide their original identities. However, all these methods focus on a trusted authority and the exchange of information with that authority to generate these pseudo ids. To address this issue, authors in [26, 27] have developed a group signature protocol. Authors in [28] have discussed about inference attacks and how to prevent data leaks; they have provided data sanitization techniques.

Based on the above discussion, it is evident that the previous methods have not considered user profile information as a metric for obfuscation techniques. Also, the primary focus was on the location of the user rather than combining users. Hence, in this paper, we provide a method where users are clustered based on their profiles. Also, there is minimal work that is done for mobile users. Hence, we also add mobility to our model and propose a dynamic clustering technique for providing privacy to the users in MSN.

4. Problem Definition

Let us denote a time series of social graphs as $G_0, G_1, \cdots, G_T$. For each temporal graph, $G_t = (V, E, L_t)$, the set of vertices is V, and the set of edges is E. L_t is the location set of all the users at the time "t." For our theoretical analysis, we focus on undirected graphs where all the $|E_t|$ edges are symmetric, i.e., $(i, j) \in E$ if and only if $(j, i) \in E$.

In each temporal graph G, vertices denote the users, and edges indicate the connection between them. Given a user "u" with location, "l" wants to search for LBS with users of similar interests. Let us consider that there are "n" users in the location radius "r" each with "A" attributes. We obfuscate user "u" based on the equicardinal clustering and send out the obfuscated user details "o" with its generalized attributes.

$$U = \{u_1, u_2, u_3, \cdots, u_n,$$
$$u_i = \{a_1, a_2, a_3, \cdots, a_A. \tag{5}$$

For a given "k," the aim of this paper is to obfuscate "u" into "o" in such a way that

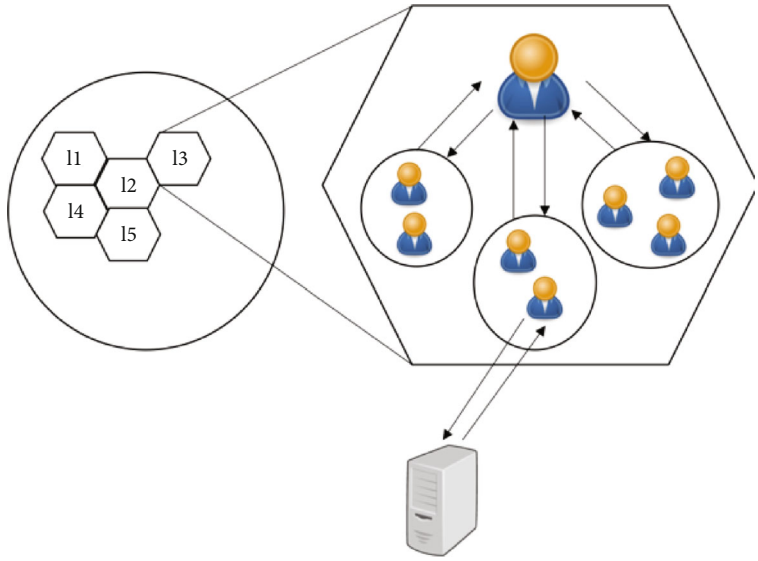

Figure 3: Proposed method.

(1) There are at least "k" users with the same attributes as "o"

(2) Minimize information loss

5. Proposed Method

5.1. Overall Architecture. The main idea behind the proposed method is to anonymize the user rather than the location. This is due to the drawbacks of the previous approaches discussed above and also to provide accurate results. To achieve user anonymization, we follow the attribute-based equicardinal clustering proposed in [29]. However, it is easy to observe that global clustering has many disadvantages. As the users are mobile, if we perform a global clustering and generate a generalized profile, it is still possible that there are only one or a few users with that profile in the attacker's neighborhood. This makes the user vulnerable to the knowledge-based attacks. Also, the desired k-anonymity is not achieved for the user. Therefore, we need to perform a local clustering that combines users based on their profiles but also deliver k -anonymity.

To perform such a clustering algorithm, we need to choose a neighborhood such that we can achieve the desired anonymity level for all the users. We will initially divide the entire region into small neighborhoods. Now, each neighborhood is checked for the minimum number of users. That is, there are at least k users in the neighborhood. Otherwise, we keep on expanding the neighborhood until k users are included. Once a neighborhood is finalized, we move on to elect a leader to perform clustering. Leader election is based on four factors: trust score, computation power, speed, and distance from the edge. Once a leader is elected, he will be responsible for a secure equicardinal k-means clustering algorithm and generate generalized profiles. These profiles are used by the user to send an LBS request.

There are two ways to achieve the above-said results: prequery clustering and postquery clustering. Prequery clustering is where we perform the clustering algorithm and maintain the details of the masked profile of the cluster head at every user. These masked details are used at the time of the query. This will make sure when the user asks for an LBS, there is no delay. On the other hand, postquery clustering performs clustering when the users ask for an LBS query. This will remove the overhead of clustering beforehand but also increases the response time. Also, with mobile users, preclustering has to handle mobility overhead that can be eliminated with postclustering. However, we assume that a user once started querying for an LBS might not stop with a single query but asks a series of queries. Hence, performing postquery clustering affects the response time for every query, and we might end up with a frustrated user resulting in losing the customer of the OSN. Hence, in this paper, we propose a prequery clustering method. Also, we consider the user's mobility, and therefore, the proposed method has to be dynamic to consider the leaving, arriving, and returning users. The proposed algorithm is visually represented in Figure 3.

5.2. Trusted Leader Election. In this module of the proposed algorithm, we utilize a network leader election algorithm proposed by Zhou and Kai [30]. As mentioned earlier, we need to incorporate the four factors into the election scheme. Hence, we propose a leadership score (LS) calculated as follows:

$$\mathrm{LS} = x * \mathrm{CP} + (1 - x) * \mathrm{TS} + \frac{d}{s}, \tag{6}$$

where LS is the leadership score, CP is the computation power, TS is the trust score provided by the server, d is the user's distance from the hexagon center, and s is the user's speed.

Input: global trust score values stored at the server, U ⟵ list of users in the area, $E_pk \longleftarrow$ public encryption key generated using the Paillier system;
Output: Leader;
1: PE ⟵ $E_{pk}(0)$;
2: Leader⟵$E_{pk}(0)$;
3: **for**$u_i \in U$**do**
4: Compute LS_i for user 'i' based on its computation power and the trust score;
5: $CE \longleftarrow E_{pk}(LS_i)$;
6: PE $leftarrow$ SMIN(PE,CE)
7: **if** PE == CE **then**
8: Leader⟵$E_{pk}(i)$;
9: **end if**
10: **end for**
11: **At the server:** : Compute D_{sk} (Leader) to get the leader;
12: Server informs all the users in the area about the leader'
13: **return**;

Algorithm 1: Trusted leader election.

Let us understand each factor in detail and how it provides value to the calculation. The first factor, computation power, is to estimate whether a device can perform the clustering and maintain the generalized profiles. Although the area and the number of users are small, we still need to ensure that the selected device is capable of performing the computation. The second factor, trust score, makes sure that we are not electing an attacker in this process and leak sensitive data to him. Hence, we use Google's method of maintaining trust scores for every user, as provided by Dhillon et al. [31]. The higher the TS value is, the better chance that the user is not an attacker. Finally, speed and distance ensure that the leader stays in the neighborhood for a maximum amount of time. If the leader is on the verge of leaving the area, then we need to perform the election and clustering algorithms again. Hence, we search for a more stable user as the leader.

To securely compute the leader among all the users in the network, we utilize the Paillier encryption scheme provided in [32] and a secure minimum (SMIN) function proposed in [18]. We use the Paillier encryption system as it is homomorphic encryption that can compute the difference between two numbers without having to find their actual values. Algorithm 1 discusses the process in detail.

5.3. Dynamic Clustering. As shown in Figure 3, the entire region is divided into small neighborhoods such that each neighborhood contains at least "k" users. However, the users may not stay in the same neighborhood for long as they are mobile. Hence, we need to identify when the neighborhood changes by a certain threshold, we need to perform reclustering, if the leader is still in the neighborhood. Otherwise, we have to reelect the leader and then perform clustering. Since checking for the neighborhood change is a computation overhead, we have to identify a regular interval in which this change might occur.

We consider the speed and direction of the users to estimate the users are leaving or arriving in the given location hexagon. As a first step, every location hexagon has to elect a leader to perform any future algorithms. Once a leader is elected, he then has to gain access to the user's profiles to perform clustering. Hence, a trusted leader is required. This election algorithm is discussed in detail in Section 5.2. As we consider the user's mobility, it is to be noted that users are always changing. It is also possible that the leader might leave the location. Hence, for every "t" seconds, we have to redo the entire process. However, if the movement of the users is slow, and there is no change in the network at location l_i, then we do not have to repeat the process. So, we redo the procedure only if the network has changed more than a certain percentage. Hence, we consider the speed of the users to calculate this time. The average speed $\bar{S}$ and the variance σ are as follows:

$$\bar{S} = \frac{1}{n}\sum_{i=1}^{n} S_i,$$
$$\sigma = \frac{1}{n}, \tag{7}$$
$$\sum_{i=1}^{n}\left(S_i - \bar{S}\right).$$

To further simplify, we assume that all the user speeds follow a normal distribution. It is a common assumption when n is large ($n \geq 30$). Hence, in a normal distribution, all the values fall in the range $(\bar{S} - 3 * \sigma, \bar{S} + 3 * \sigma)$ with a 99.73% probability. Hence, we assume that our $S_{\min} = \bar{S} - 3 * \sigma$ and $S_{\min} = \bar{S} + 3 * \sigma$. This gives us the information about the fastest moving user and the slowest moving user. Hence, the average speed at a given location can also be written as the average speed of the fastest moving user and slowest moving user.

$$S_{\text{avg}} = \frac{(S_{\max} + S_{\min})}{2}. \tag{8}$$

So, if we perform operations based on this average speed, we will capture the network change. As we know, time = distance/speed and distance is the maximum distance user

Input: U ⟵ Users, UT⟵ User trust factor, r ⟵ Range, c ⟵Center, UP ⟵User profiles, x ⟵ percentage change, and s ⟵ Speed;
Output: UCH ⟵ User cluster head;
1: **while**t%$(r * x/50 * s) == 0$**do**
2: **if** Head==NULL ‖ Head out of range == TRUE **then**
3: Head $leftarrow$ Election(U_T,r,c);
4: **if**u_i == Head **then**
5: $U_{CH} leftarrow$ Cluster(U_P);
6: **end if**
7: **end if**
8: **if** network change in percentage == x **then**
9: **if**u_i == Head **then**
10: Cluster(U_P);
11: **end if**
12: **end if**
13: **end while**
14: **return**;

Algorithm 2: Dynamic clustering.

1: **At the user**u_i:
2: [Message 1:User $u_i \longrightarrow$ Leader l]
3: $<t_s, u_i>$
4: **At the Leader l:**
Input: N⟵ Set of users, k ⟵ anonymity level
5: **while** true **do**
6: **if** Message 1 is received from the user u_i**then**
7: $C_i \longleftarrow$ C.Find(u_i);
8: [Message2: Leader l $\longrightarrow$ User u_i];
9: $<t_{sl}, t_{si}, C_i > 10$: **end if**
11: **end while**
12: **return**;

Algorithm 3: Anonymous LBS query request.

has to cover in a location. For more straightforward calculations, we assume hexagons to be circles, and hence, maximum distance is the diameter = 2 * radius. So, for every time $t = (2 * r)/s_{\text{avg}}$ seconds, the network will scan for any changes. If the network has changed more than x%, then we redo the clustering. If the leader has left the location, then we have to repeat the election process and clustering.

Once a leader has been elected, he will be responsible for forming an equicardinal clustering for the users in the area. This is based on the algorithms proposed in [29].

5.4. Anonymous LBS Query. The final step of the algorithm is where the user receives a generalized profile. When a user u_i has to send an LBS query, it first sends a request to the location leader. The leader then identifies the cluster that u_i belongs to cluster c_j. Now, the leader responds by sending the c_j profile. When the user u_i receives c_j profile, it then sends the query by providing c_j profile and l_i location. This can be further explained in Algorithm 3.

By performing such an anonymous query, we do not mask the location, and hence, the results are accurate. Also, if the attacker tries to sniff, he gets hold of a location where there are "K" users with the same profile, and hence, the user is K-anonymous.

6. Analysis

Definition 1. We say that $G*$ is k-anonymous if

$$p\left[u_i j = 1 \mid u_{ij}^*\right] \leq \frac{1}{k}. \tag{9}$$

Theorem 2. *To minimize information loss at a given time and with the given number of users, "k" should be chosen in a way such that* $k = \sqrt{n}$.

Proof. Let us assume that all the users in a given cluster are equidistant from its cluster center.

According to the objective, we have to minimize IL and maximize k, i.e., minimize (IL + $1/k$).

$$\frac{\sum_{i=1}^{k}\sum_{j=1}^{n_i}\left(x_{ij}-\bar{x}_l\right)'\left(x_{ij}-\bar{x}_l\right)}{\sum_{i=1}^{k}\sum_{j=1}^{n_i}\left(x_{ij}-\bar{x}\right)'\left(x_{ij}-\bar{x}\right)} + \frac{1}{k}. \tag{10}$$

We are also assuming that each cluster has an equal number of users. Hence, the number of users in each cluster is k, and the number of clusters is n/k.

$$\frac{\sum_{i=1}^{n/k}\sum_{j=1}^{n_i}\left(x_{ij}-\bar{x}_l\right)'\left(x_{ij}-\bar{x}_l\right)}{\sum_{i=1}^{n/k}\sum_{j=1}^{n_i}\left(x_{ij}-\bar{x}\right)'\left(x_{ij}-\bar{x}\right)} + \frac{1}{k}. \tag{11}$$

By substituting the constant distance values and summing over, we get

$$\begin{aligned} k * \frac{n}{k} n + \frac{1}{k}, \\ n^2 + \frac{1}{k}. \end{aligned} \tag{12}$$

To minimize this function, we take a single derivative and equate it to 0.

$$\begin{aligned} 2 * n - \frac{1}{k^2} &= 0, \\ 2n &= \frac{1}{k^2}, \\ k &= \log n, \end{aligned} \tag{13}$$

Theorem 3. *Achieving K-anonymity is NP-hard [33–35].*

Proof. Given a graph $G = (V ; E)$, the problem is to determine whether the edge set E can be partitioned into subsets E_1, $E_2, \cdots$ in such a way that each E_i generates a subgraph of G isomorphic to the complete graph K_n on "n" vertices. Our main result is that the problem EP_n is NP-complete for each $n \geq 3$. From this, we deduce that several other edge-partition problems are NP-complete. To show that EP_n is NP-complete, we reduce our problem to the well-known 3SAT problem. We know that 3SAT is an NP-complete problem. A set of clauses $C = \{C_1, C_2, \cdots, C_r\}$ in variables $u_1, u_2, \cdots, u_s$ is given, each clause C_i consists of three literals $l_{i,1}, l_{i,2}, l_{i,3}$ where a literal $l_{i,j}$ is either a variable u_k or its negation u_k. Now, the problem is to identify whether C is satisfactory. That is if we can satisfy all the conditions that are defined in C. A clause is satisfied if exactly of its literals has value "true."

$$\sum_{j=1\text{tok}} l_{i,j} = 1. \tag{14}$$

Hence, any final solution should contain exactly "k" vertices and therefore is an edge partition problem, which is NP-complete.

Theorem 4. *A network change of x% is equivalent to the time difference*

$$t = \frac{rx}{100} * \left(\frac{1}{s_{\min}} + \frac{1}{s_{\max}}\right), \tag{15}$$

where "r" is the radius of the clustering area and "s" is the average speed of the max speed and min speed users.

Proof. If a user u_i travels at a speed of s_i, the maximum time taken for the user to cover distance "d" is

$$t = \frac{d}{s_i}. \tag{16}$$

The maximum distance that the user has to travel out of the clustering area is the diameter of the circle: $2 * r$. Hence, $t = (2 * r)/s_i$. Out of all the users in the given area, the minimum time taken to cross the entire clustering area is by the user whose speed is maximum.

$$t = \frac{2 * r}{s_{\max}}. \tag{17}$$

Similarly, the maximum time taken would be by the slowest traveling user.

$$t = \frac{2 * r}{s_{\min}}. \tag{18}$$

Hence, the average time for all the users to travel out of the clustering area is

$$t = \frac{(2 * r)/s_{\max} + (2 * r)/s_{\min}}{2} = r * \frac{1}{s_{\min}} + \frac{1}{s_{\max}}. \tag{19}$$

However, this time holds for 100% of the users to travel out of the clustering area. But we need to find time for x% of the users to move out of the area.

$$\begin{aligned} t &= r * \left(\frac{1}{s_{\min}} + \frac{1}{s_{\max}}\right) * \frac{x}{100}, \\ t &= \frac{r * x}{100} * \left(\frac{1}{s_{\min}} + \frac{1}{s_{\max}}\right). \end{aligned} \tag{20}$$

7. Experimental Results and Discussion

We have implemented the proposed method on a synthetic dataset generated using a Mockaroo realistic data generation generator [36]. Mockaroo gives us an imitation of real-world social networks with specified user attributes and creates random and meaningful friendships between users. Hence, this is an apt tool for the experimental results. Using this tool, we have generated 25 attributes for each user. These attributes include the occupation, highest level of education, university/school attended, places visited, and the city he/she lives in currently. We have considered only users from the USA, and hence, all the cities and universities belong to the USA. This dataset also has a user's current latitude and longitude information along with the speed and direction of travel.

We compare our proposed method with the four other algorithms that also focus on providing location anonymization in a mobile environment. First, we consider location-based generalization methods, including spatial cloaking (SC) and grid-based cloaking (GBC). Secondly, we consider profile-based generalization to achieve k-anonymity, including the top-k algorithm and k-means clustering (KMC). The proposed method is referred to as enhanced equicardinal clustering (EEC) for convenience.

Spatial cloaking [20] focuses on a distributed model where collaborative peers form a cloak area to satisfy the spatial k-anonymity principle. The use of collaborative peers is mainly due to the fact of unreliable users in a mobile environment. As it is difficult to judge an authentic user from an attacker, they consider an intermediate anonymizer that acts as a trusted third party. In this method, when a user initiates an LBS request, it starts by searching k-1 companions and securing ad hoc information exchange between them to form a k-peer cloak area. The user then randomly selects a peer in the group, and that chosen peer sends the request to the LBS server. Upon this request, the LBS server seeks the desired

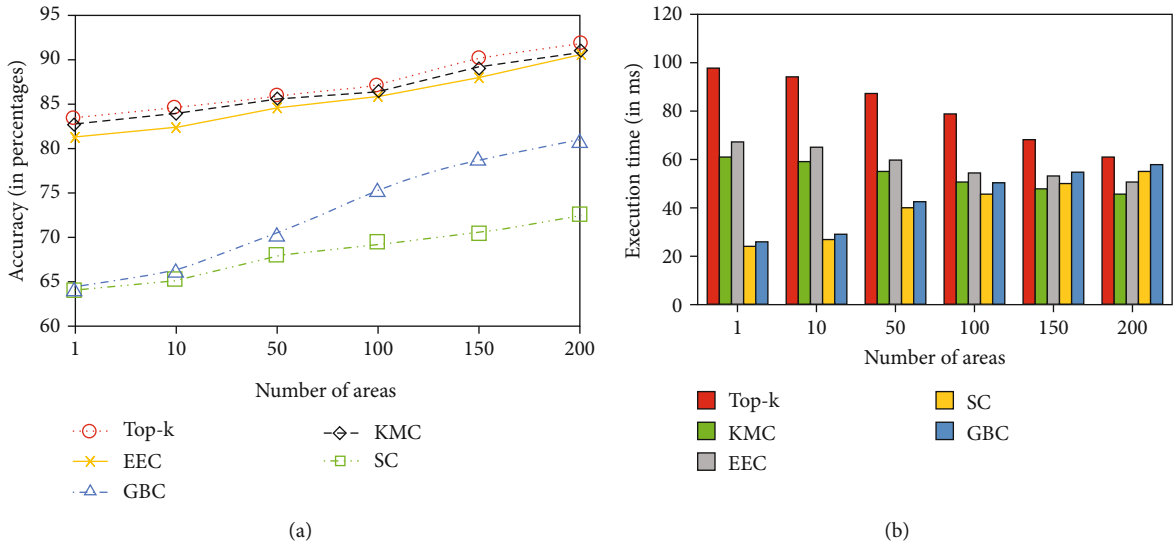

Figure 4: Effect of change in the number of areas.

information in the database and returns appropriate answers to the initiator through the agent. Finally, the initiator selects a satisfactory solution.

The second method is the grid-based cloaking mechanism proposed by [21]. This method focuses on finding a minimum grid area for every individual user who wishes to send an LBS query. Every user starts with himself and expands the grid until the desired number of users, achieving k-anonymity, with different attributes and achieving l -diversity, is found. Beginning at a two-dimensional coordinate system where the user is currently residing, it expands in the shape of a hexagon by one unit at a time. Once a step of expansion is made, the algorithm compares the "k" and "l" values. If anyone of the values does not match, then the expansion step continues. The algorithm comes to a termination point, once it finds the desired area. The top-k method uses the K-nearest neighbors (KNN) algorithm to find the best user profiles to match with every user. When a user wishes to send an LBS query, it will identify the k best profiles in the temporal graph area. Therefore, the algorithm is robust and does not consider any user movements as the algorithm is performed every time the user wished to query. The final method is the k-means clustering (KMC) method. We use this method as it is the baseline algorithm for the proposed method. Therefore, identifying the amount of information loss and decrease in accuracy will be easier. Although the accuracy for KMC will be higher than EEC and execution time will be lower, readers should note that the k-anonymity will not be maintained in the KMC algorithm.

We have utilized the Yelp Fusion API to generate the LBS results to mimic the context-aware preferential LBS query system. This is an excellent tool in searching for nearby restaurants, given the user's location. Once these results are obtained, we use an algorithm proposed by [2] to sort the results based on the user's preference and history. Hence, the results will now be sorted accordingly. As discussed earlier, preferential LBS provides excellent user experience and boosts the service.

All the experiments were conducted on Windows 10 operating system with Intel Core (TM) Duo 2.66 GHz CPU, 12 GB memory, and Java platform. Each observation has been averaged over 50 instances. We have devised four different experimental settings to observe the performance of the proposed method. Each experiment considers the various settings of users and attributes. Evaluation metrics are discussed in Section 2 as a part of the definitions. Two metrics need to be observed in each experiment: accuracy of the LBS results and execution time (ET).

7.1. The Effect of Change in the Number of Areas. Our first experiment's goal is to observe how the initial partition of areas affects our proposed algorithm. The following example illustrates the importance of this experiment. The dataset generated is distributed over the USA. Therefore, it is a concern to decide on the area division that represents cities or states. If we divide the whole area into states, our number of areas would be less. However, this might not produce good results, as we are trying to combine people from various cities. However, if we consider the entire area to be divided into cities, this might result in more processing time.

The second observation is made on the information loss as it plays a crucial role in determining our preferential LBS results. The goal is to reduce the information loss as much as we can. From the experimental results, the proposed method has performed much better than the location generalization methods. It can be observed from Figure 4(a) that the accuracy of EEC is 15% to 25% more than the SC and GBC. The reason behind being the reduction in the number of areas indicates that the number of users per cluster is more. The probability that more users mean profile differences can also be huge. However, other methods like SC and GBC achieve anonymity by just performing the location-based clustering and not based on the profile. Hence, the proposed method beats other methods in such vast differential profiles as the clusters ensure that only the users who closely relate are clustered together. Similarly, the

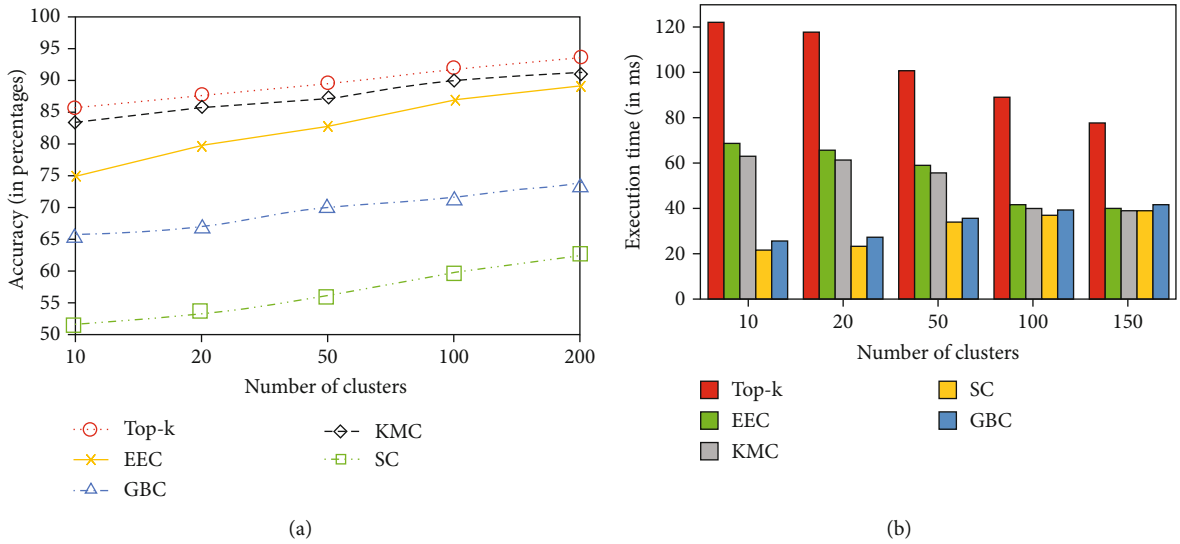

Figure 5: Effect of change in the number of clusters.

accuracy achieved by the profile-based generalization methods is not significantly higher than EEC due to user distribution. The number of users moved from the original cluster to the next best cluster is not high. Therefore, the difference in accuracy is not much. However, k-anonymity is maintained for all the users, unlike KMC.

Execution time depends on the k-means convergence. If there are few users, then the k-means algorithm converges faster and faster. Additionally, the convergence also depends on user profiles. If there are 100 users and we want 10 clusters, then running k-means on their profiles is much easier as there will be at least ten users with a similar profile. However, it is not the same as the location. As users should be clustered into different areas based on their location, profiles might be completely different, and hence, convergence takes longer. The top-k algorithm takes the maximum execution time as the user has to run the KNN algorithm every time he has to send a query. Although this algorithm removes the dependency on the leader, the execution time increases as the user's LBS requests are more. Another important observation made through this experiment is that the execution time for a profile-based generalization decreases as the number of areas increases, whereas the time increases in location-based methods. This is because the number of users per area decreases when we increase the number of areas. Therefore, we have mobile users in a tiny area who continually moves in and out of the area and thereby increasing the computations for location-based methods.

Accuracy and information loss can be related. With minimum information loss, a profile-based LBS query gives better results. Our proposed method reaches a maximum of 92% accuracy. However, grid-based clustering also performs well in this scenario. This is because the LBS query dramatically depends on location information, and grid-based generalization of location is much more efficient than spatial clustering. However, it is not on par with our method as the location is still generalized and not accurate location. Even with the profile-based sorting, our algorithm provides very high accuracy.

7.2. The Effect of Change in the Number of Clusters. The second experiment focuses on observing the proposed method's performance under the change in the number of clusters. The key here is that the increase in the number of clusters means fewer users per cluster. Additionally, by decreasing the number of clusters, we increase the number of users per cluster. By increasing the number of clusters per user, we are also increasing the chance of having more related users in the cluster. While increasing the generalization, it is also possible that the clusters formed are more meaningful and are more related. We observe the effect of this change in the following Figure 5(a).

As the "k" increases, the number of users per cluster decreases. That means we have more opportunities to generate very tight clusters. Hence, information loss can be significantly reduced and thereby increasing accuracy while increasing the "k" value. However, we are also compromising on the anonymity provided to the user. If there are 100 users and we want to achieve 100-anonymity, then each user is a cluster by itself. In this scenario, although information loss is 0, and accuracy is 100%, we are not providing anonymity to the user. Therefore, we should choose a "k" such that it offers the right anonymity level with lesser information loss and higher accuracy. By this experiment, we found $k = 50$ provides us with 83% accuracy. Hence, we maintained that value for other experiments. The accuracy difference between KMC and EEC is initially high because when there are a smaller number of clusters, we have more users per cluster. EEC moves the users to their next best cluster until all the clusters achieve k-anonymity. This results in information loss and hence decreasing the accuracy. In general, profile-based generalization gives better accuracy as the LBS query depends on user preferences that are well maintained by the EEC and KMC algorithms. However, as the cluster size increases, the users per cluster decreases. While EEC forms tighter clusters, thereby increasing accuracy, accuracies of GBC and SC also increase due to lesser profile generalization as the number of users in the area is small.

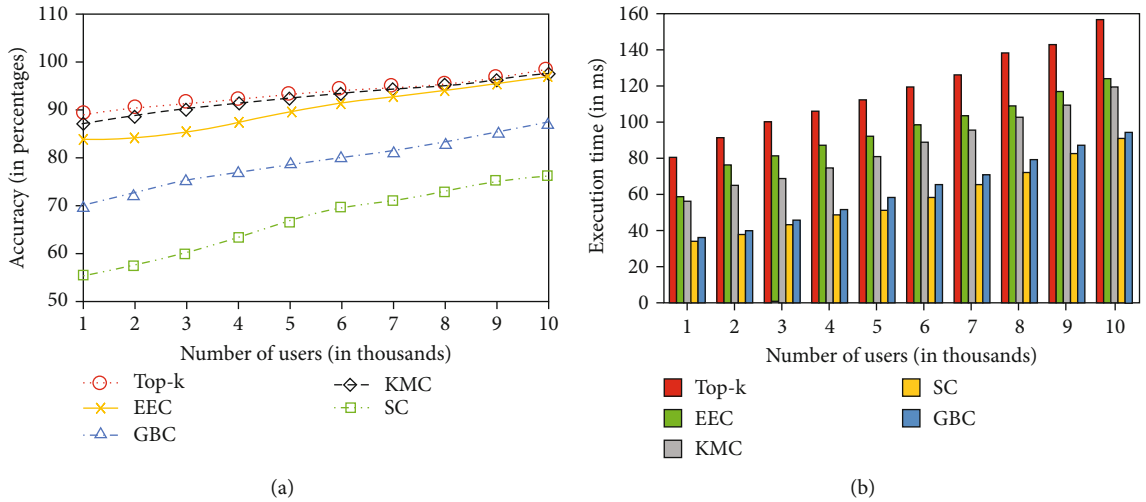

Figure 6: Effect of change in the density of users.

Figure 5(b) shows the execution times for all the methods. While the cluster size increases, the number of users per cluster decreases and thereby decreases the computations in KMC and EEC. However, due to the small size, which is a smaller area selected for location-based methods like GBC and SC, user movement gets high. For example, if the cluster size is 2, then SC and GBC might choose users who are next to each other. So even if one person moves out of the location, they have to rerun the algorithm. Therefore, the execution time increases when we decrease the users per cluster. This, however, will not be the case in profile-based methods, as the area is constant. We change the clusters but maintain the same area.

7.3. The Effect of Change in the Number of Users. The third experiment focuses on observing the proposed method's performance under the change in the number of users. These experiments are designed to identify how the algorithm behaves in a crowded environment compared to a sparse environment. Hence, we maintain the number of areas, the number of clusters, and the average user speed to be constant. All the experiments shown here have considered a single area, the number of clusters is 50, and the average speed is 30 miles/hr. From the previous observation, we have observed that when the number of clusters is 50, we get an acceptable accuracy level and good user anonymity. Therefore, by increasing the number of users, we increase the user density in the area.

Figure 6(a) shows the average accuracy achieved by the users, while Figure 6(b) shows the execution time. We can see that the accuracy achieved by the proposed algorithm, EEC, is significantly higher than location generalization algorithms like SC and GBC. We have previously mentioned that top-k is the highest achievable accuracy with the k-anonymity-based profile generalization, and the EEC algorithm has achieved comparable results to the top-k and KMC. This is because the accuracy of the results is heavily dependent on the user's precise location. By generalizing the location, we lose vital information. In a profile-based generalization method, we maintain the exact location, and hence, accuracy levels are higher. However, user preferences also affect the accuracy, and the proposed method aims at attaining least IL and thus higher accuracies.

In sparse environments, the accuracy achieved by EEC is lower than that of top-k. This is due to the lack of similar user profiles. Through this experimental procedure, we have learned that the probability that users with similar profiles end up in the same area is much less. Hence, the formed clusters have higher information loss. However, the execution time of top-k is significantly higher than the EEC algorithm as every user calculates the top-k best-matched user profiles in the entire area. Additionally, moving users to other clusters to achieve$k$-anonymity has further increased the IL, and thus, accuracy levels compared to KMC are also less.

In dense environments, the accuracies achieved by the EEC, top-k, and KMC algorithms are similar. This is because as the number of users is more, we can produce more meaningful clusters and thereby reducing the IL and increasing the accuracies. However, the accuracies achieved by GBC and SC are significantly lower compared to EEC as the clusters are formed only based on their location. The accuracy for SC and GBC has increased due to the increase in the number of users. As the number of users increased, the locations of users are now much closer, and hence, location generalization is reduced and thereby decreasing the IL.

7.4. Mobility. The goal of our final experiment is to observe the performance of the proposed algorithm when the user's mobility increases. When the user is moving and requests real-time LBS queries, it is essential to maintain the profile generalization along with k-anonymity at the current location he is in. To analyze this, we are considering the algorithm performance by increasing the average speed of the users from 10 miles/hour to 80 miles/hour. It is to be noted that the user requests continuous LBS queries. However, we only perform reclustering when the clustered areas are modified by more than 40%. These experiments will give us an understanding that if we take the snapshot of our algorithm

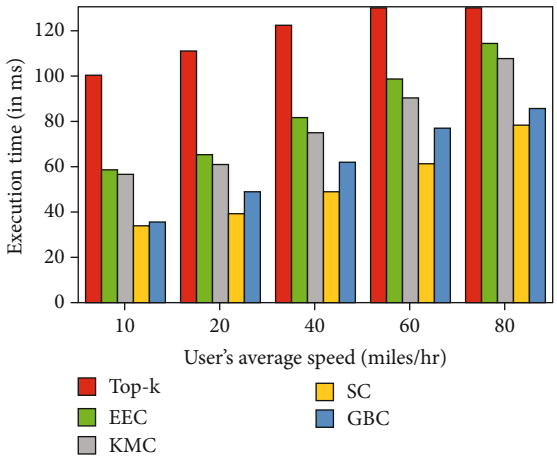

Figure 7: Effect of user mobility on execution time.

performance at random times, how the information loss and execution time changes.

Figure 7 demonstrates the performance of the proposed algorithm compared to other methods when we increase the speed of the users. It can be observed that the execution time for both profile-based and location-based methods had increased when we increased the speed of the users. Execution time for top-k has drastically increased due to constant recomputations of every user. The execution time for the proposed algorithm had increased twice when the speed of the users changed from 10 miles/hour to 80 miles/hour. This setup also gives us an understanding of heavy movement and low movement areas. For example, users are more mobile in the morning rather than at night. Also, users on an interstate move faster compared to the users on a small street. The performance of the proposed algorithm had not deteriorated when we increased the speeds. It is still in the tolerable time limit, given the fact that it preserves the privacy of the user.

8. Conclusion and Future Work

As the location-based searches are moving towards context awareness to provide the user with a better experience, we tend to lose the user's personal information. Providing the user with a privacy-enhanced search not only improves the trust but also attracts more customers. Hence, in this paper, we have proposed a privacy-enhancing preferential LBS search algorithm in a mobile user environment. Previous techniques like spatial-based and grid-based achieve user anonymity by clustering nearby users to achieve k-anonymity. However, the users are clustered based on their locations. This type of method reveals vital information, which is the user's profile. If these clusters contain users with completely different profiles, it is easy to deanonymize the users based on their profiles. A simple knowledge-based attack can identify the exact user from the cluster. Also, since the location is generalized, LBS query results do not give us accurate results. Hence, our proposed method anonymizes users based on their profile and sends the exact location for the LBS query. As the user's mobility is considered, the reclustering of users should occur when the initial clusters are no longer valid. Experimental results show that our proposed method provides at least two times lesser information loss and three times better accuracies than the previous anonymization techniques.

Although the proposed method performs much better than existing techniques, it can be further improved in specific ways. We should implement a clustering mechanism where it might exclude outliers, promote the cluster togetherness, and not lose the privacy for outlier users. This might include a mechanism for outliers separately. Also, this method assumes that the election algorithm is secure. As one of the devices in the area has to perform clustering, we have to select a safe and computationally capable device. This might be difficult if we are integrating multiple social networking LBS queries. Hence, a better method should be proposed as to who performs clustering when users with various social networking applications try to query.

Data Availability

The data is generated using http://mockaroo.com.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the NSF under Grants 1704287, 1741277, 1912753, 2011845, 1829674, 1704274, and 1704397.

References

[1] https://www.statista.com/topics/2478/mobile-social-networks/.

[2] A. Jaszkiewicz and R. Słowiński, "The LBS-discrete interactive procedure for multiple-criteria analysis of decision problems," in *Multicriteria Analysis*, J. Clã-Maco, Ed., pp. 320–330, Springer, Berlin Heidelberg, 1997.

[3] B. Claudio, "Privacy protection in location-based services: a survey," in *Handbook of Mobile Data Privacy*, pp. 73–96, Springer, Cham, 2018.

[4] R. Gupta and U. P. Rao, "An exploration to location based service and its privacy preserving techniques: a survey," *Wireless Personal Communications*, vol. 96, no. 2, pp. 1973–2007, 2017.

[5] M. Siddula, L. Li, and Y. Li, "An empirical study on the privacy preservation of online social networks," *IEEE Access*, vol. 6, pp. 19912–19922, 2018.

[6] P. Wang and Q. Ma, "Issues of privacy policy conflict in mobile social network," *International Journal of Distributed Sensor Networks*, vol. 16, no. 3, 2020.

[7] M. Li, H. Zhu, Z. Gao et al., "All your location are belong to us: breaking mobile social networks for automated user location tracking," in *Proceedings of the 15th ACM international symposium on Mobile ad hoc networking and computing*, pp. 43–52, Philadelphia Pennsylvania USA, 2014.

[8] A. Inan, M. E. Gursoy, and Y. Saygin, "Sensitivity analysis for non-interactive differential privacy: bounds and efficient algorithms," *IEEE Transactions on Dependable and Secure Computing*, vol. 17, pp. 194–207, 2020.

[9] M. E. Gursoy, A. Tamersoy, S. Truex, W. Wei, and L. Liu, "Secure and utility-aware data collection with condensed local differential privacy," *IEEE Transactions on Dependable and Secure Computing*.

[10] T. Wang, M. Xu, B. Ding et al., "MURS: practical and robust privacy amplification with multi-party differential privacy," *Annual Computer Security Applications Conference*, 2019, https://arxiv.org/abs/1908.11515.

[11] X. He, A. Machanavajjhala, C. Flynn, and D. Srivastava, "Composing differential privacy and secure computation: a case study on scaling private record linkage," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1389–1406, Philadelphia Pennsylvania USA, 2017.

[12] F. Y. Rao, J. Cao, E. Bertino, and M. Kantarcioglu, "Hybrid private record Linkage," *ACM Transactions on Privacy and Security (TOPS)*, vol. 22, no. 3, pp. 1–36, 2019.

[13] A. Thapa, W. Liao, M. Li, L. Pan, and J. Sun, "SPA: a secure and private auction framework for decentralized online social networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 8, pp. 2394–2407, 2016.

[14] X. Zheng, Z. Cai, J. Li, and H. Gao, "Location-privacy-aware review publication mechanism for local business service systems," in *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, pp. 1–9, Atlanta, GA, 2017.

[15] T. Ji, C. Luo, Y. Guo, Q. Wang, L. Yu, and P. Li, "Community detection in online social networks: a differentially private and parsimonious approach," *IEEE Transactions on Computational Social Systems*, vol. 7, no. 1, pp. 151–163, 2020.

[16] P. Asuquo, H. Cruickshank, J. Morley et al., "Security and privacy in location-based services for vehicular and mobile communications: an overview, challenges, and countermeasures," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 4778–4802, 2018.

[17] L. Li, R. Lu, and C. Huang, "EPLQ: efficient privacy-preserving location-based query over outsourced encrypted data," *IEEE Internet of Things Journal*, vol. 3, no. 2, pp. 206–218, 2016.

[18] Y. Elmehdwi, B. K. Samanthula, and W. Jiang, "Secure k-nearest neighbor query over encrypted data in outsourced environments," in *IEEE 30th International Conference on Data Engineering*, pp. 664–675, Chicago, IL, 2014.

[19] S. Zhang, G. Wang, M. Z. A. Bhuiyan, and Q. Liu, "A dual privacy preserving scheme in continuous location-based services," *IEEE Internet of Things Journal*, vol. 5, no. 5, pp. 4191–4200, 2018.

[20] Z. Huang and M. Xin, "A distributed spatial cloaking protocol for location privacy," in *International Conference on Networks Security, Wireless Communications and Trusted Computing*, pp. 468–471, Wuhan, Hubei, China, 2010.

[21] S. Zhang, K. K. R. Choo, Q. Liu, and G. Wang, "Enhancing privacy through uniform grid and caching in location-based services," *Future Generation Computer Systems*, vol. 86, pp. 881–892, 2018.

[22] H. Li, H. Zhu, S. Du, X. Liang, and X. S. Shen, "Privacy leakage of location sharing in mobile social networks: attacks and defense," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 4, pp. 646–660, 2016.

[23] M. A. Ferrag and A. Ahmim, "ESSPR: an efficient secure routing scheme based on searchable encryption with vehicle proxy re-encryption for vehicular peer-to-peer social network," *Telecommunication Systems*, vol. 66, no. 3, pp. 481–503, 2017.

[24] R. Lu, X. Lin, X. Liang, and X. Shen, "A secure handshake scheme with symptoms-matching for mHealthcare social network," *Mobile Networks and Applications*, vol. 16, no. 6, pp. 683–694, 2011.

[25] X. Liang, M. Barua, R. Lu, X. Lin, and X. S. Shen, "HealthShare: achieving secure and privacy-preserving health information sharing through health social networks," *Computer Communications*, vol. 35, no. 15, pp. 1910–1920, 2012.

[26] K. Zhang, X. Liang, R. Lu, and X. Shen, "PIF: a personalized finegrained spam filtering scheme with privacy preservation in mobile social networks," *IEEE Transactions on Computational Social Systems*, vol. 2, no. 3, pp. 41–52, 2015.

[27] X. Liang, Z. Cao, J. Shao, and H. Lin, "Short group signature without random oracles," in *International Conference on Information and Communications Security*, pp. 69–82, Philadelphia Pennsylvania USA, 2007.

[28] Z. Cai, Z. He, X. Guan, and Y. Li, "Collective data-sanitization for preventing sensitive information inference attacks in social networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 4, pp. 577–590, 2016.

[29] M. Siddula, Z. Cai, and D. Miao, "Privacy preserving online social networks using enhanced equicardinal clustering," in *IEEE 37th International Performance Computing and Communications Conference (IPCCC)*, pp. 1–8, Orlando, FL, USA, 2018.

[30] R. Zhou and H. Kai, "Powertrust: a robust and scalable reputation system for trusted peer-to-peer computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, no. 4, pp. 460–473, 2007.

[31] M. Dhillon, R. Tut, K. Snyder et al., "Dynamic trust score for evaluating ongoing online relationships," 2016, US Patent 9-390243.

[32] P. Paillier and Pascal, "Public-key cryptosystems based on composite degree residuosity classes," in *Advances in Cryptology — EUROCRYPT '99*, pp. 223–238, Springer, 1999.

[33] R. J. Bayardo and R. Agrawal, "Data privacy through optimal k-anonymization," in *21st International Conference on Data Engineering (ICDE'05)*, pp. 217–228, Tokyo, Japan, 2005.

[34] A. Meyerson and R. Williams, "On the complexity of optimal K-anonymity," in *Proceedings of the twenty-third ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pp. 223–228, Philadelphia Pennsylvania USA, 2004.

[35] B. Kenig and T. Tassa, "A practical approximation algorithm for optimal k-anonymity," *Data Mining and Knowledge Discovery*, vol. 25, no. 1, pp. 134–168, 2012.

[36] "Mockaroo," https://mockaroo.com/.