8-11-2020

Privacy Preserving User Data Publication In Social Networks

Madhuri Siddula
Georgia State University

PRIVACY PRESERVING USER DATA
PUBLICATION IN SOCIAL NETWORKS

by

MADHURI SIDDULA

Under the Direction of Yingshu Li, Ph.D. and Zhipeng Cai, Ph.D.

ABSTRACT

Recent trends show that the popularity of Social Networks (SNs) has been increasing rapidly. From daily communication sites to online communities, an average person's daily life has become dependent on these online networks. Additionally, the number of people using at least one of the social networks have increased drastically over the years. It is estimated that by the end of the year 2020, one-third of the world's population will have social accounts. Hence, user privacy protection has gained wide acclaim in the research community. It has also become evident that protection should be provided to these networks from unwanted intruders. In this dissertation, we consider data privacy on online social networks at the network level and the user level.

The network-level privacy helps us to prevent information leakage to third-party users like advertisers. To achieve such privacy, we propose various schemes that combine the privacy of all the elements of a social network: node, edge, and attribute privacy by clustering

the users based on their attribute similarity. We combine the concepts of k-anonymity and l-diversity to achieve user privacy.

To provide user-level privacy, we consider the scenario of mobile social networks as the user location privacy is the much-compromised problem. We provide a distributed solution where users in an area come together to achieve their desired privacy constraints. We also consider the mobility of the user and the network to provide much better results.

INDEX WORDS: Data privacy, Social networks, User data, K-anonymity, Mobile social networks, Location based services, Dynamic clustering

PRIVACY PRESERVING USER DATA PUBLICATION IN SOCIAL NETWORKS

by

MADHURI SIDDULA

A Dissertation Submitted in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

in the College of Arts and Sciences

Georgia State University

2020

PRIVACY PRESERVING USER DATA
PUBLICATION IN SOCIAL NETWORKS

by

MADHURI SIDDULA

Committee Chair: Yingshu Li
Committee: Zhipeng Cai
Yanqing Zhang
Yubao Wu
Ruiyan Luo

Electronic Version Approved:

Office of Graduate Studies

College of Arts and Sciences

Georgia State University

August 2020

DEDICATION

To say I was lucky to have amazing parents and husband in my life is an understatement. The love and support I receive from them every day are immeasurable. So, I dedicate my dream, this dissertation, to them as a small way of saying how much I love and indebted to them.

My parents, Sree Girinadh and Satya Devi Siddula are my pillars of strength and support. They taught me to dream and never give up. They believed in my abilities even when times that I could not. They have taught me that a person's dreams should not be based on their gender but rather their will and abilities. It would not have been easier to have accepted their only child to move to a different country so that she can pursue her dream. But they made it seem so simple for me. They made me the person that I am today. And for this, I cannot thank them enough.

My husband, Venkatesh Chintapandu, is my inspiration. His constant support and guidance had got me through the toughest days of my life. He was with me through every step of my graduate school and life. He made sure that I never had to compromise for anything and supported me in my decisions. If it were not for his love and patience, I would not have successfully finished my Ph.D. I am truly thankful for having him in my life.

ACKNOWLEDGMENT

I want to thank my Ph.D. advisors, Dr. Yingshu Li, and Dr. Zhipeng Cai, for the continuous support of my Ph.D. study and research. Dr. Li is one of the smartest and compassionate people I have ever met. She has always provided me the freedom to pursue different projects and opportunities. Her insights and suggestions made me improve my overall growth as a graduate student and prepared me for future endeavors. I am also very grateful for Dr. Cai for his advice and insightful discussions. He always motivated me to reach my full potential and achieve success in every venture. I could not have asked for better advisors than them.

I would also like to take this opportunity to thank my committee members: Dr. Yanqing Zhang, Dr.Yubao Wu, and Dr. Ruiyan Luo. Their support and feedback has contributed toward my dissertation. I would like to thank their time and effort that they have shown me over these years.

Last but not least, it is my absolute pleasure to have been a graduate student in computer science department at Georgia State University. All the members in the department were always willing to help and only wished for my success.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

- OSN - Online Social Network
- MSN - Mobile Social Network
- LBS - Lcation-Based Service
- LPPMs - Location-Privacy Preserving Mechanisms
- DA - Degree of Anonymity
- EEC - Enhanced Equi-cardinal Clustering
- LEEC - L-diverse Enhanced Equi-cardinal Clustering
- IoT - Internet of Things
- AI - Artificial Intelligence
- DL - Deep Learning

CHAPTER 1

INTRODUCTION

1.1 Background

Social Networks (SNs) have attracted millions of people worldwide from the time they were introduced. In today's world, they have become an indispensable part of every human's life. There are many ways in which a social network is defined. According to Webster [1], a social network is nothing but a network of people. However, it is clear from our usage of social networks these days that it is so much beyond connecting people. Social networks are often used for online communication and hence, can be referred to as Online Social Networks (OSNs). They are now not just a means of communication between people. The social network industry has grown and spread its business into every part of ou daily life. More and more social networks are emerging every day for various requirements, and every social network is a booming business model. Some of the early social media sites include Friendster, Six Degrees, Orkut, and so on. Orkut became one of the premier OSN sites in Brazil and India during 2007 [2]. Another popular instant messaging system, QQ, was in widespread use in China is introduced in 1999 [3]. However, today's market is hugely dominated by Facebook and Instagram that combinedly has about 290 million users [4].

The main purpose of an OSN site is to provide web services that will allow users to perform the below actions:

1. Have a public profile to present to other users.

2. Have friends/connections to communicate with them

3. Search for new friends based on their various attributes like common friends, interests, and so on.

The above functionalities are the basic functionalities based on which a social network is initially developed. Hence, we categorize all the social networks into one/more of the following categories based on the works done by [5] [6] [7].

1. **Personal networks.** OSNs like Facebook, Friendster and MySpace are some examples of this category. This category of networks focuses on creating a detailed user's profile. Hence they allow many attributes for a single user.

2. **Status update networks.** Twitter is the best example of this category. These networks mainly focus on posting a status update. These updates might sometimes include a place or person.

3. **Shared-interest networks.** This category of networks focuses on bringing people with similar interests together. Dating apps like Tinder and professional apps like LinkedIn are examples of this category.

4. **Neighborhood Exploring networks.** These networks highly focus on user's current and exact location. Based on user's location, he will be able to share information, media files and interact with neighborhood people. Tinder is also an example of this category.

With more and more increase in the demand for various functionalities, social networks now do not belong to a single category. They are trying to expand and provide more features to please the users. For example, Facebook was initially a personal network, and now it can be considered as all the four categories mentioned above.

In today's world, social networks have developed into multi-goal applications. They are not just used to connect people but for business, finding local businesses, dating, job search, and many more. Also, as mobile phones boom started in the early 2000s, every major business website tried to enter the mobile application world. That formed the basis for the development of Mobile Social Networks (MSNs). With the introduction of 4G and LTE, internet speeds have skyrocketed, and the mobile applications started using real-time data of the mobile user. All the social network providers have their mobile application that

supports various functionalities. Hence, a social network is now divided into OSN and MSN applications. Although MSN is a part of OSN, there are more complications when privacy is to be provided for mobile users compared to a stationary user. In the following section, let us examine the privacy threats on OSN, and later in section 1.3, we will examine the privacy threats in MSN.

1.2 Privacy threats in OSN

Usage of Online social Networks is ever-increasing. In 2017, 71 percent of all internet users were using some kind of social networking site. It is expected that by the end of 2019, there will be 2.77 billion social network users. This constitutes 2/3rd of the world's population [8]. Hence, it is clear that the vast majority of the population uses OSN, and their profiles are online. As the initial OSN sites were introduced, connections in OSN were only based on real-life friends. As the market grew and people started to explore, OSNs now offers a wide range of friend-finding systems. We can now find friends based on the place we live, common interests, common friends, and so on. Hence, the public profile of a user has become an important part of one's profile. Without correct information about a user, it is difficult to find things or people that he would be interested in. Hence, users need to have a profile that contains personal information, including their date of birth to a recent place that they have visited. Also, this public profile contains a lot of sensitive information like age, gender, profession, current area, and so on, that includes his / her photo(s). As discussed before, public profiles are a common feature of any OSN. Due to this fact, a huge amount of personal data is vulnerable to attacks.

One of the biggest scandals of 2018 is the Cambridge Analytica. According to CNBC, The Guardian, The Observer, and the New York Times, it is suspected that two of the biggest social media sites Facebook and Analytica allegedly used user's personal information for the 2016 presidential election in the USA. It is claimed that 50 million user profiles have been mined to send them personalized messages that contributed to the election results.

Some of the common intentions of attacks that concentrate on the privacy violation of the users are:

1. **Behavioral advertising** is one of the most profitable businesses using social network profiles. Business models use a person's profile to personalize advertisements tailored to users. Products that the people interested in, if offered, have a high chance of purchase. Hence, this industry has become one of the main income of any social networking site. However, these business models use various user information such as the location of the user, relationship status, and so on. Google, Facebook, and Twitter use such models. We can see such advertisements on Facebook by changing a single attribute such as relationship status [9]. In cases like this, data is often leaked to a third party advertiser unintentionally. An incident like this happened with Facebook, where six million users' phone numbers and email addresses to unauthorized users for a year [10].

2. **Identity theft** uses an individual's personal information often for financial gain. The information posted on users' social network profiles is used to steal their identity. In 2009, researchers at Carnegie Mellon University had found that the data extracted from social networks and other online public databases can lead to the discovery of partial or full social security numbers [11].

3. **Stalking / Child Abuse** One of the early privacy cases in this regard occurred in 2010 on MySpace, where minors were bullied and led to the adoption of "age requirements and other safety measures". Another crime committed by Peter Chapman in 2009, who falsely obtained an identity of another person on Facebook and lured and raped a 17-year-old girl [12]. Events such as stalking and "catfishing" are frequent in present society and hence has become a prominent topic in social network security.

1.3 Privacy threats in MSN

The primary advantage of such MSN applications is that they provide various services like LBS, virtual reality, online dating, and so on. Among them, location-based queries are

the most widely used service. Online Social Network (OSN) has become a host for various business advertisements. These business models include hotels, restaurants, vacation spots, and sports. Users of OSN visiting these business models tend to review them, which helps other users visit them. Hence the business owners maintain very detailed information of their business, such as location, menu, phone numbers, and address. As the mobile internet has exploded with its increasing speed, humongous mobile applications have been developed in the past years. Similarly, all the OSN providers have their mobile-based applications designed to provide a more convenient and faster way to access their accounts on the go. MSN applications offer various advantages, including LBS, that help users to search for nearby business models.

LBSs have been gaining very considerable popularity by the rapid advances in positioning technologies, e.g., Global Position System (GPS), and the development of modern smart devices with data communication capabilities. LBS query is the most common usage of any mobile user. Users query for nearby restaurants, the distance between their position and another place, and so on. A typical example of such a service is: Alice would like to query for nearby restaurants. For such queries, all we need is the exact location but not the user profile information. If the user and his location are identified by the attacker who can snoop the network, then important information can be leaked. For example, Alice visits a cancer hospital regularly and goes to a pharmacy at a particular location every time she visits the hospital. Imagine she has enabled LBS to any of her online social networking applications. Then, the attacker can infer that Alice or a family member of hers has cancer, and also they live in the neighborhood of the pharmacy she visits. This is a piece of crucial information that should not be leaked.

While mobile users enjoy LBS, they have to provide their real locations to the LSP, which poses a severe threat to their privacy. It is to be observed that the users might not always want to reveal their location information to others. The key to addressing these concerns lies in preserving the users' privacy when efficiently providing correct query results.

However, some LBS does consider user profile. 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 history or interest. If the user preference is Asian cuisine, then the results are sorted accordingly, and so on. Hence, accurate results for LBS is only possible if we have exact location information and correct user profile. However, if we provide those to the LBs query, the user's privacy is not maintained.

1.4 Organization

The rest of the report is organized as follows: Chapter 2 provides the network model. Chapter 3 reviews the existing literature in the context of privacy in social networks. Chapter 4 investigates the problem of user privacy while a social network is published online. Chapter 5 discusses the problems and proposed solutions for user location privacy in mobile social networks. Chapter 6 introduces future research directions for this work.

CHAPTER 2

NETWORK MODEL

Let us describe the social network pictorially (Figure 2.1) to understand its properties and thereby formalizing its definition. We know that a social network is mainly comprised of people. As a manner of speaking, a user can be a person, business, or a group of people (Community). Let us assume everything is a user.

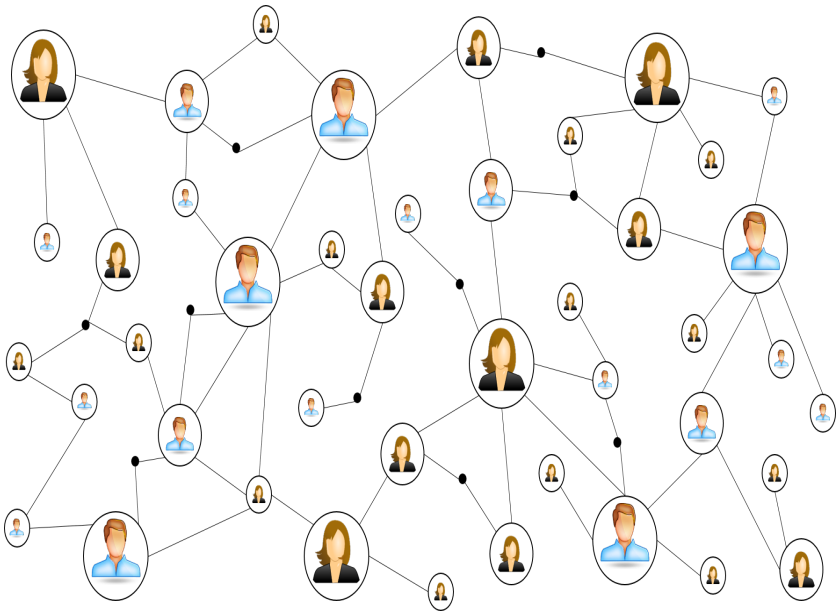

Figure 2.1. Social network with various users and shared attributes

Again, a profile can consist of various things according to the social network. For example, on Facebook, a profile of a user can contain the following things:

- Personal Information (DoB, Area of living, School/college attended, work info, and so on)
- Pictures

- Tags (Places visited, restaurants, and movies)

Another social network, like ResearchGate, contains completely different profile information. It might look like:

- Educational background
- Current position (Student, professor, and research candidate)
- His/her publications
- Contributions to the research community

All these profile information can be considered as "attributes". Many of these attributes are shared. For example, an attribute like "Georgia State University" does not belong to a single person. There will be a lot of students who might have attended this school, and hence, such an attribute becomes shared among all such users. In conclusion, there are two kinds of attributes:

1. Private
2. Shared

Finally, the main reason for the invention of a social network is to enable the connection between people. Irrespective of the type of social network, we always have people to connect to. These are called "Friendships" on Facebook, "Followers" on Instagram/Twitter, and "Matches" on Tinder. Names apart, the functionality of these connections connects two people and can be referred to as "links" in generic terms. However, these links are one dimensional in some social networks. For example, on Twitter, a celebrity os followed by two million people, but not all of them are followed back by the celebrity. In some networks like Facebook, if two people are connected, they are friends. Hence, the links in a social network are directed.

Additionally, some social networks provide different kinds of relations between people. For example, on Facebook, a user can have a mother, father, spouse, brother, sister, and cousin. The user is more closely related to these users than his colleagues or collegemates. Hence, it is also important to assign weights to these links to specify how close two persons are connected. This gives us a real-world replica of a social network.

To understand this better, let us reconstruct the social network problem as a graph-based problem. Any social network can be represented as a graph, where each user's profile is a node, and friendship between two users is an edge between those two nodes. Each user has attributes associated with him/her. Figure 2.2 shows a generalized social network graph with nodes (users), attributes, and links between them.

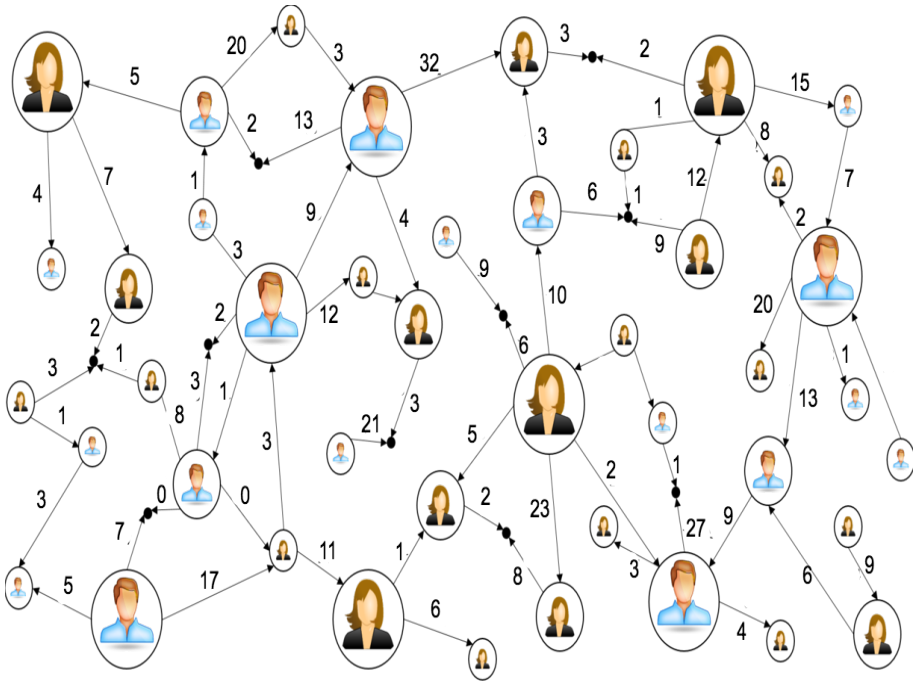

Figure 2.2. Social network considered for this research

Let us assume we have a social network graph $G = (V, E)$ where V represents the users and E represents the edges between them. $V = \{u_1, u_2, u_3, \ldots, u_n\}$ and $|V| = n$. Every user u_i has a attributes set A values associated with it $\{a_{i1}, a_{i2}, ..., a_{im}\}$. Edges are directed

and hence are represented as: $E = \{e_{12}, e_{21}, e_{13}, \ldots, e_{n,n-1}\}$ and $|E| = n * (n-1)$. Here, an edge e_{12} represents an edge from vertex 1 to vertex 2, which is not same as e_{21}, an edge from vertex 2 to vertex 1. Additionally, every edge has a corresponding weight.

CHAPTER 3

EMPERICAL ANALYSIS OF USER PRIVACY IN SOCIAL NETWORKS

3.1 Privacy conditions

To understand the definition of privacy, we need to analyze three popular anonymity definitions.

3.1.1 k-anonymity

K-anonymity was well studied by various authors in [13] [14] [15]. According to them, a graph is k-degree anonymous if, for every node v, there exist at least $k-1$ other nodes in the graph with the same degree as v. This helps to anonymize the graph while preserving its utility. In this method, anonymization is achieved by simply adding the edges to the existing graph. That is, k-anonymity consists of two steps: first constructing a new degree sequence, d', that is k-anonymous; secondly, construct a new graph with this new degree sequence. The running time of this method is $O(nk)$. It is also to be observed that no anonymized graph should be of the size larger than $2k-1$.

According to [13], k-anonymity method is implemented in the following way. First, we construct a new graph from the obtained new degree sequence by starting with the highest degree vertex. In the subsequent iterations, we always consider the highest remaining degree vertex available for that iteration.

K-anonymity is simple and efficient but suffers from its various drawbacks. One of the significant drawbacks is that it is vulnerable to attribute disclosure. Numerous researchers have worked on this drawback to improvise the algorithm, e.g., [16] [17]. Two other notable attacks were identified in [18]: homogeneity and the background knowledge attack.

3.1.2 l-diversity

According to [18], "An equivalence class is said to have l-diversity if there are at least l "well-represented" values for the sensitive attribute." In this method, the term "well-represented" can be interpreted in many ways. In this paper, authors have used entropy as the information-theoretic notion, and hence came the concept of Entropy l-diversity.

Entropy l-diversity. The entropy of an equivalence class E is defined to be

Entropy(E) = -$\sum_{s \in S} p(E, s) log p(E, s)$

Where S is the set of sensitive attributes, and $p(E, s)$ is the fraction of records in E that have sensitive value s. If the set S is divided into two sub-blocks S_a and S_b, then $Entropy(S) \geq min(Entropy(S_a), Entropy(S_b))$. To achieve l-diversity, we need to maintain an entropy of at least $log(l)$ for the entire table. One of the exceptions for this scenario is when the sensitive attribute is very common, and hence this method is restrictive.

While l-diversity is advantageous over k-anonymity, it still has drawbacks. Primarily, this method is complicated and may not be necessary to achieve [19]. Also, l-diversity is not sufficient to ensure the prevention of attribute disclosure. It is prone to skewness and similarity attacks. Authors of [17] have observed that the l-diversity, an improvement over K-anonymity, still cannot restrain attribute disclosure when the table contains multiple records of an individual. They proposed to have each specify privacy policies about his or her attributes.

3.1.3 t-closeness

Although k-anonymity and l-diversity define the necessary conditions for privacy preservation, they are insufficient. Both of these methods suffer from drawbacks. K-Anonymity is vulnerable to attribute disclosure; whereas, l-diversity needs at least 'l' well-represented values for each sensitive attribute. According to [19], "l-diversity is neither necessary nor sufficient to prevent attribute disclosure". Hence, a new technique is proposed in [19], called "T-closeness", to improve the performance by adding a constraint to l-diversity. According to this constraint, it is essential to maintain the distribution of a sensitive attribute comparable

to the distribution of its overall table. Specifically, the difference between the distributions in an equivalence class and the entire table for a given sensitive attribute should not be more than a given threshold. This additional constraint helps in preventing similarity attacks. To calculate the distance, authors of [19] have chosen to use the earth mover distance measure.

Privacy in online social networks has been a prominent research topic ever since they were introduced. However, the representation of OSNs has always been an issue. It was until [20] that social networks have not been represented in terms of graphs. As explained in the previous section, we consider a version of the social network, where there are three major components: Nodes, Edges, and Attributes. All the previous related work had concentrated on achieving privacy through one or more of these components. Hence, we are discussing some of how these individual components privacy is achieved and their problems. The survey about all the existing privacy-preserving mechanisms in an OSN has been discussed in detail in [21].

3.2 Node privacy

Nodes in a social graph represent the user. Additionally, if a user is considered a node, it means that none of his attributes defining the node are not considered. Hence, in this kind of privacy mechanism, we intend to preserve the privacy of just a user, not concentrating on his attributes. So, the only way a user's identity is preserved without considering his attributes is anonymization. Anonymization is a way of hiding information. One of the earliest techniques of preserving node privacy is the **Naive anonymization** [22]. In this technique, we try to replace each user with an id. That is, either a number or an alphabet [23]. For example, if there are three users in the network: Alice, Bob, and Cathy. Now, the naive anonymization technique replaces their name with some IDs. That is, Alice $\rightarrow$ "A", Bob $\rightarrow$ "B", and Cathy $\rightarrow$ "C". Hence, to identify them back, we need to maintain an ID table containing users and their corresponding ids.

However, it is clear from Figure 3.1 and Figure 3.2, that is the ID table is compromised; then, the privacy is lost. Similarly, by observing the original and anonymized social networks,

Name	ID
Alice	A
Bob	B
Cathy	C

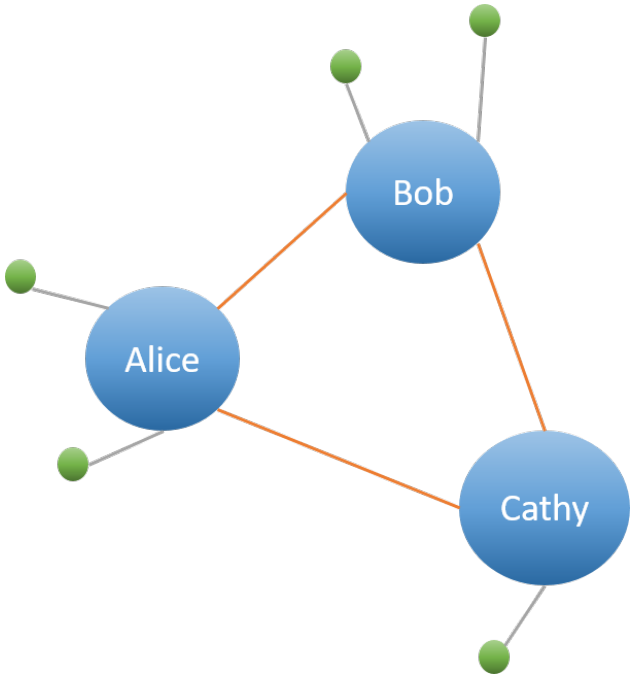

Figure 3.1. Naive node anonymization: Original social network

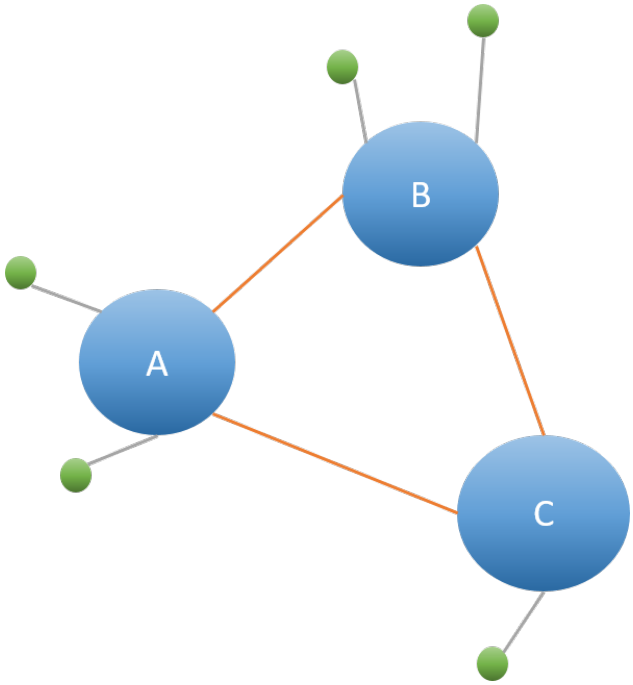

Figure 3.2. Naive node anonymization: Anonymized social network

the graph is the same. So, it is easy to identify the meta-information from the graph. For example, in networks like Facebook, it is easy to retrieve information of "Friends of Friends" if the profiles are made public. This type of crawling may lead to the discovery of a complete or partial network, leading to the development of the network structure locally [24]. Hence, we move on to the next anonymization technique called "Random Walk Anonymization". A similar technique has also been applied to edge anonymization. The idea behind this technique is to remove information randomly to preserve privacy randomly. In random walk node anonymization, we start with a random node and walk for random steps in a given direction. The node that the walk ends in will be removed. The process should be continued until the desired privacy is achieved. Figure 3.3 shows the original graph before the random

walk, and Figure 3.4 shows the anonymized graph after the random walk has been applied to the nodes.

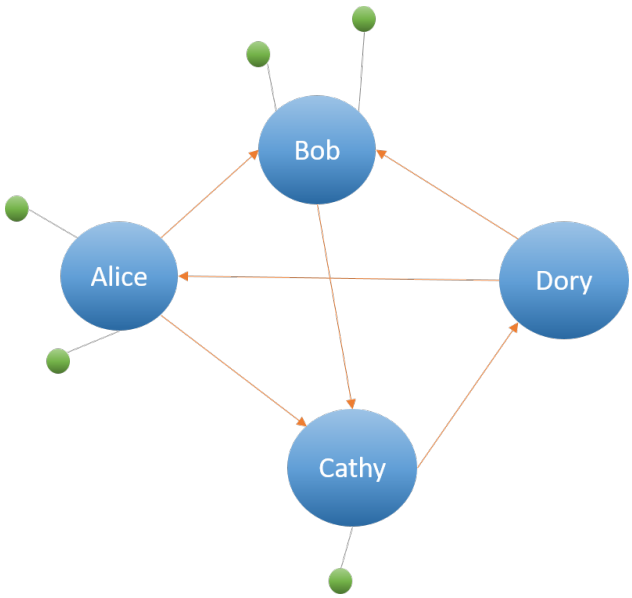

Figure 3.3. Random walk on nodes: Original social network

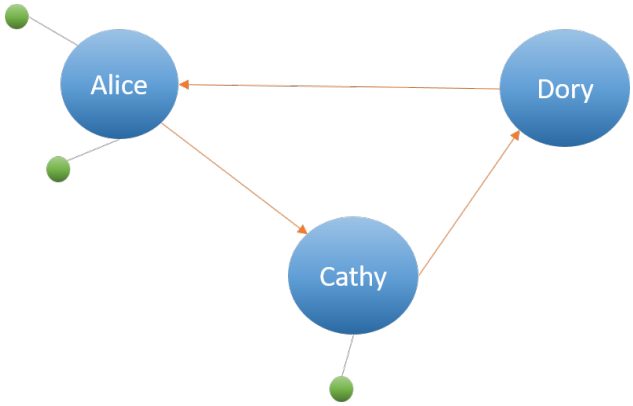

Figure 3.4. Random walk on nodes: Anonymized social network

Again, the subgraph information is evident in this kind of anonymization. Using attacks like "Infiltration", where an attacker is part of the network, and he knows who he is or his neighbors are. From the anonymized graph, it is easier to find out if his information is deleted. If it is deleted, he can reverse engineer the anonymization technique. If not, he can be able to retrieve information about more users in the network. Hence, this kind of anonymization also fails.

Another way of achieving anonymization is through "Perturbation". This is similar to a random walk technique, but instead of walking a distance, we find a random node and delete it. In this technique, we either delete a node or add a new dummy node, thereby increasing the randomness. However, according to [22], if the randomness is increased by more than 10%, it is hard to retrieve the information lost, and hence we have high information loss. Authors in [25] have enhanced the traditional k-anonymity and l-diversity methods to anonymize the social networks using both node and link perturbations. In the k-anonymity

method proposed by [25], we anonymize the vertices using a technique called "neighborhood component coding". In this method, vertices are greedily classified into 'n' similar groups, and the vertices in the same category are replaced with the same label. Following the power-law distribution, this method starts by finding the highest degree vertex. This technique is improvised in [26], which suggests that the graphs should be isomorphic. Similarly, authors in [25] have proposed that to achieve l-diversity, we need to divide all the nodes into equivalence groups. Previous studies have revealed that the background information of an adversary may also lead to the discovery of vertex degrees in a given network [22] [27] [28] [29], node pair similarities [30] [31], and link types [32] [33] [34] [35] [36]. An attack, called the walk-based attack, had been proposed in [37] to understand the connection between any two given nodes. In this method, an attacker creates k different accounts and links them randomly in the network. He then creates a specific link pattern with the nodes of interest. Once the adversary establishes these connections, it is easy to identify the sub-graph of the nodes, in the anonymized graph, that corresponds to his accounts with a high probability.

We now come to a better way of achieving anonymization, that is, grouping or clustering. In this technique, we try to group a certain number of users/nodes and form a single node. The most prominent information in a social network is the location. Hence, we group users based on their location information. For example, all the users in the same building can be grouped. All the users living in the same city can be grouped. However, there will be information loss if we group more people together. Hence, authors in [38] have adopted the k-anonymity principle while grouping people. Hence, we only need to group 'k' people in order to achieve k-anonymity. Various techniques talk about grouping users. In spatial cloaking mechanism [39], authors have proposed a method where every user searches for 'k' neighbors in a given radius 'r'. Once he identifies the circle in which the users exist, he then uses the circle's information rather than his own. Another way is grid-based cloaking [40], where the server divides the entire location into various grids such that every grid has at least 'k' users. In the prior technique, there is a computation overhead on the user while

in the latter, at the server. However, the essential information is that the metadata is still susceptible to attacks, and also there will be colossal information loss.

In conclusion, whenever an anonymization technique is proposed, we lose information and gain privacy. As both are inversely proportional to each other, we have to identify specific quantities that can determine a user's privacy level. According to [22], we have to measure the following quantities when a node anonymization technique is proposed.

- *Closeness Centrality:* This is the average shortest path from one node to all the nodes in the graph.
- *Betweenness Centrality:* This is the proportion of all shortest paths through a given node.
- *Path Length Distribution:* This is constructed from all the shortest paths between each pair of nodes.
- *Degree Distribution:* This is the distribution of all the vertices degrees
- *Diameter:* This is the maximum shortest path between any two given nodes.

All these quantities ensure that the properties of the graph are preserved while preserving user privacy.

3.3 Link privacy

The next component of privacy that we are discussing is the link/edge privacy. Edges represent the friendship between two users. According to various social networks, these edges can be assumed to be non-directional, one-directional, or bi-directional. However, the majority of the earlier works assumed these edges to be non-directional for simplicity purposes. Also, extending the techniques to bi-directional is complicated and time-consuming.

One of the earliest techniques proposed in edge privacy is "Perturbation" [41], [42]. This is similar to the perturbation technique in node privacy. We randomly select certain edges

and delete them. Additionally, we can choose two nodes randomly with no link between them and insert an edge. This technique aims to anonymize the edges by changing the graph structure. This method can be seen in figure 3.5 and Figure 3.6.

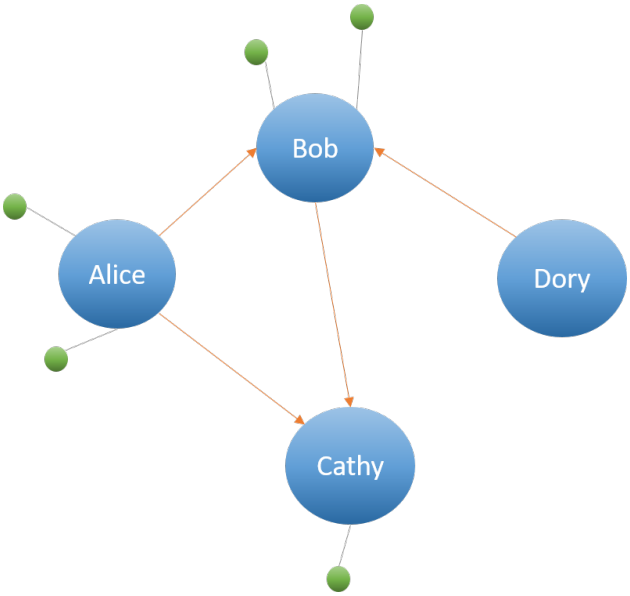

Figure 3.5. Edge perturbation: Original social network

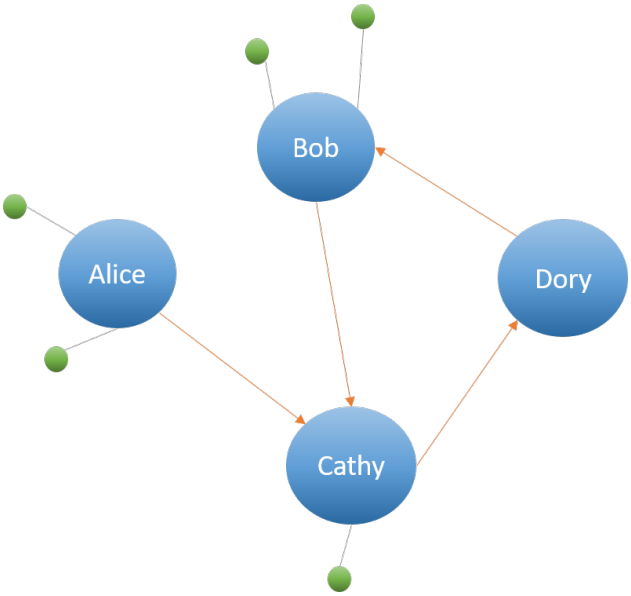

Figure 3.6. Edge perturbation: Anonymized social network

Similar to perturbation, a random walk can also be applied to edge privacy. In this technique, we start with a node and walk for a random number of steps in a direction. The final edge in this walk can be removed as a part of the privacy-preserving process. For example, let us assume that we have randomly selected vertex 'u' as the source node. Then, out of all the neighbors of vertex 'u', one neighbor is selected to continue the walk, say 'v', and parallelly we increment the path length by '1'. We continue the process of random walk till the desired length 'l' is obtained. However, this technique can be compromised by simple perturbation attacks. According to [43], a simple perturbation to a random walk technique is to introduce a new edge at the end of the existing path with a length '$l-1$'. Let us assume that there is a path '$a-b-c-d$', then the perturbation of node 'z' is done by adding it to the path '$a-b-c-d-z$'. In this case, we face two problems:

- Random walk path could be completed at node 'd' or
- Node 'z' is already on the path

To eliminate both the problems which might lead to self-loops or duplicate edges, we perform the random walk process until a suitable path is found. To maintain the utility of the graph, an application parameter l is maintained. This parameter can be used to control the random walk (l-hop random walk).

The major problem with perturbation is information loss. If the information loss exceeds more than a certain amount, then the anonymized graph is not useful. If it does not remove more information, it is easier for the background knowledge attacks to retrieve information. In this kind of attack, an attacker has certain background information. For example, he might be a part of the network ("infiltration") or know certain people in the network. This makes it easier for the attacker to identify patterns in the anonymized graph and retrieve information. Combining both node and edge anonymization through perturbation can also be compromised using background knowledge attack []. Thus, this kind of anonymization no longer provides the desired privacy.

Unlike node grouping, it is not simple to group edges as they do not have attributes. However, if the groups/clusters of users are formed, then the edges inside the cluster can be removed, and only edges between clusters can be maintained. We have shown how a simple edge grouping algorithm modifies the original network from Figure 3.7 into an anonymized (grouped) network in Figure 3.8.

In conclusion, we can anonymize the edges at various generalization levels [36]:

- *Intact edges:* This technique removes sensitive edges leaving all the other edges intact
- *Partial-edge removal:* This technique removes a certain percentage of observations based on a pre-specified criterion. For example, removing edges that connects high degree nodes at random.

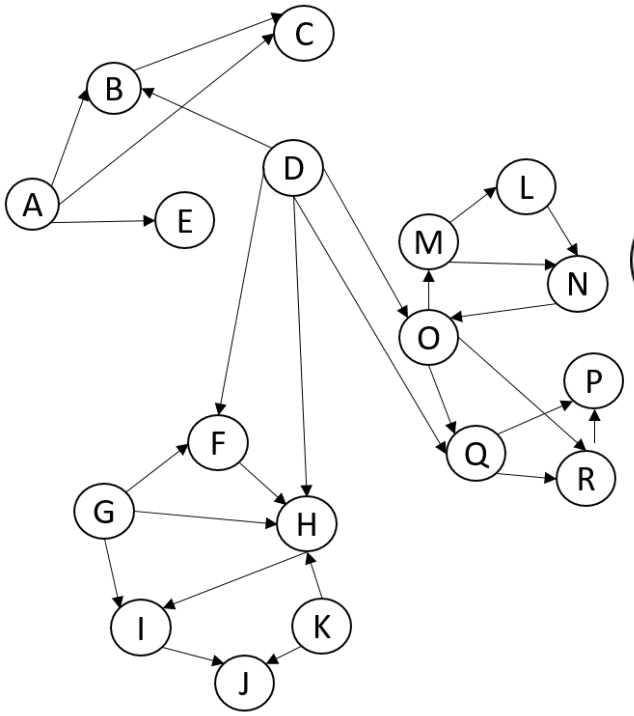

Figure 3.7. Original social network

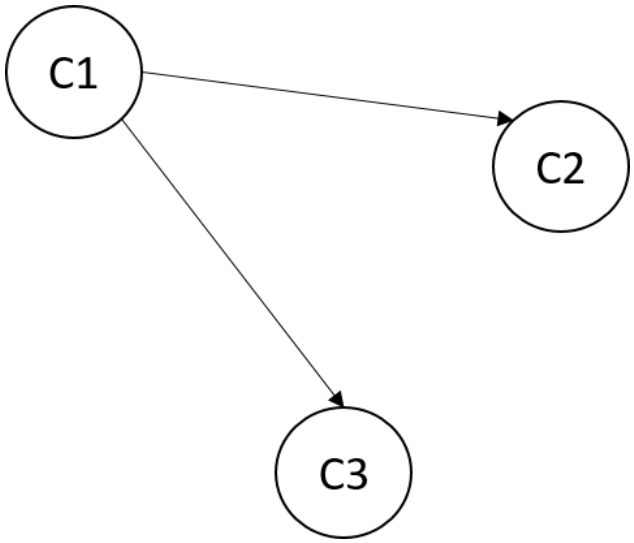

Figure 3.8. Anonymized social network

- *Cluster-edge anonymization:* In this method, we form clusters of different edge types and make sure the number of edges between these clusters remains the same, even after removing and adding random edges.

- *Cluster-edge anonymization with constraints:* This technique is an extension of the previous technique. Here, we assume additional constraints, like any two equivalence classes, should have some limitations to the corresponding nodes in the original graph.

- *Removed edges:* This technique removes all edges.

3.4 Attribute privacy

Most of the related work in this area focuses mainly on the location attribute of a user. Although a social network user has many attributes, one of his prime concerns will be to

hide his exact location details. Hence, preserving the user's location privacy has become a much-studied research area in this graph element privacy.

A user generally reveals his location information only when he searches for something that requires providing his location privacy. Hence, Location Based Services (LBS) is where user location privacy is much studied. Again, to achieve this, various researchers have proposed various techniques. One of the earliest techniques is to change the granularity of the location. For example, if a user is on the street ABC, Area A, City C. Then, instead of providing street ABC as the location, we provide Area A. Again, k-anonymity principle is widely used to determine the generalization of location granularity.

Another technique is to provide dummy locations and re-calculating the distance. Authors in [44] have utilized the spatial cloaking mechanism to change the location information. This protocol is based on the distributed model that computes a cloak area which covers all collaborative peers to satisfy the spatial k-anonymity. This approach is different from existing methods as it does not rely on any intermediate anonymizer; any collaborative user does not have to trust each other. It possesses stronger robustness in the scenarios with multiple initiators and the scenarios with a collision. First, a mobile user initiates the LBS, searching $k-1$ companions and collaborating to form a cloak area, which covers k peers according to a certain degree of k-anonymity and these peers exchange information through an ad-hoc network. The initiator randomly selects a peer in the group as an agent who sends query messages with location information on behalf of the initiator to the LBS server. Upon this request, the LBS server seeks the desired information in the database and returns appropriate answers to the initiator through the agent. Finally, the initiator selects a satisfactory solution.

Another algorithm is a grid-based cloaking mechanism proposed by [45]. This algorithm supports both k-anonymity and l-diversity. In this algorithm, a minimum grid area is obtained for every user. For every user, we start by expanding the area until we find 'k' users (k-anonymity) with 'l' different attributes (l-diversity). This helps in reducing any unnecessary computations. This cloaking algorithm creates a temporary cloaking area by

expanding its region. Starting 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. To efficiently perform this algorithm, a grid structure is used for storing the user location information. A post pruning technique is used to eliminate any unnecessary region expansion. Since the k-value is higher than l-value, there is a high probability that if we find k-users, we tend to obtain the l-value. Thus it helps to create the minimum cloaking region more efficiently. A grid structure is maintained to store buildings and users. All the anonymization techniques proposed previously have been summarized in [21] [46] [47].

This concludes the various privacy-preserving schemes for a graph-based social network. There are specific techniques, in which authors have suggested using completely different networks that are constructed based on privacy as the primary principle. These alternate systems are discussed in the following section.

3.5 Alternate systems

SPROUT Social Path ROUTing (SPROUT) is an alternate routing path proposed for communication between friends in a network [48]. It mainly concentrates on the reduction of communication costs between two active nodes in the system. In this method, the communication between nodes follows a peer-to-peer network, and hence there is no server. This technique adopts the CHORD protocol for the communication and thus introduces a Distributed Hash Table (DHT) to store all the friends' id values. According to SPROUT, when a user initially joins the network, it is assigned a random network identifier that lies in the range [0,1]. As the user forms friends and the node generates neighbors, they are assigned identifier numbers that are sequential to the user. This subsequent assignment of id values forms an id number ring. To reduce the communication delay between users, we also generate links halfway around the ring, quarter way, one-eighth way, and so on. Hence the

maximum communication delay between any two users in the network is $O(logn)$. The DHT stores the hash value of the neighbor's id keys, which is on the best route to reach every user. This helps us while communicating with online friends. The overall communication between two users 'u' and 'v' can be summarized into the following steps:

1. Find a neighbor of 'u' whose id value is close to 'v' but not greater than $logn$.
2. If a neighbor 'n' is found, the message is forwarded to 'n' and continue to find a neighbor of 'n' that is best reachable to 'v'.
3. If no such neighbor is found, then forward the message to the neighbor whose id value is the id value of 'n' + 1. Thus, the message goes to every node in the ring until the desired node 'v' is obtained.

Lockr is a system that can be integrated with any of the existing social networks [49]. It works well in both centralized and peer-to-peer network as the Lockr system is independent of the underlying network technology. The main aim of Lockr is to provide security to the users. It primarily concentrates on providing three security benefits: private communication between users, authenticating the users, and providing right access controls. To provide these benefits, Lockr has proposed the following techniques:

1. **Personal Identities and Address Books:** Each user in the network maintains a public/private key pair that can be used for communication. Each user is also entitled to manage the address book that contains information about friends and their public keys.
2. **Social Attestation:** This attestation acts like a certificate between two users confirming their friendship status. It includes information about the two users, type of friendship between them, and most importantly, a "friendship key". It is to be noted that two users can have more than one attestation/relation. For example, two people

who studied at the same university can now be co-workers. In this situation, we maintain two social attestations, but the attestation that has been issued for classmates has been expired.

3. **Social Access Control Lists (ACLs):** Every user also maintains an ACL that stores information about the content access to different friends. For example, a user can restrict personal information like movies he watched and places he visited from his employer.

Lockr provides the users with a complete choice of what technique to use and where to store the data. Since all the profile information is not public, it should be stored in a third-party database. Hence, users can choose which database to use. Once the information is decoupled from the social network, Lockr provides additional functionalities like creating the public/private key pair, ACLs, and so on. Additionally, Lockr also provides mechanisms to deal with expired attestations and revoking them if necessary. If attestation is expired, Lockr uses one-way hashing to protect them from being abused by non-authorized users.

Pisces is a system that has also been proposed for secure communication in the online social network [43]. It is a peer-to-peer network that uses onion routing for anonymous communication. Every node in the Pisces architecture generates a public/private key pair. They discover peers by using random walks. One of the major drawbacks of this technique is that it is vulnerable to route capture attacks. To address this issue, Pisces proposes "Reciprocal Neighbor Policy". According to this policy, we associate the routing table of each node with its neighbors. For example, if a Sybil node advertises itself as an honest node, all of its neighboring benign nodes exclude the Sybil node from their routing tables. To ensure the security of the proposed policy, there is a periodic check on every node's neighborhood list. Four other features are associated with this policy.

1. **Liveness Check:** Every node checks with its benign neighbor nodes if they are alive are reciprocating the trust.

2. **Degree Exchange:** Every node should broadcast the degree, the number of nodes in the neighborhood list, periodically to all the neighbors.

3. **Final List:** Upon receiving the degree information, each node finalizes its neighborhood list and encrypts with its private key.

4. **Local Integrity Check:** Every node in the network saves the signed versions of the neighborhood lists from all the neighboring nodes.

Safebook is a distributed online social network intended to preserve privacy based on real-life trust [50]. It is a de-centralized OSN (DOSN) developed to issue the problems of privacy, integrity, and availability. Safebook is a three-tier architecture. Every user in Safebook is considered a host node connected to all the peers through the P2P overlay. There are two types of overlays:

- A set of **matryoshkas** which are the structures in network-level that provides communication privacy.

- P2P substrate giving lookup services.

Matryoshkas are designed to provide trusted data storage and retrieval. It also uses indirection technique to communicate anonymously. The set of nodes in Matryoshkas are arranged in concentric rings, and all messages have to relay through all the rings. Since the immediate rings in the concentric circles are based on actual trust between the nodes, we assume that the information passage is secure and private. The innermost ring consists of various nodes that are in direct communication with the core. These nodes store all of the core's data in encrypted form and are mirrored among all the nodes. In contrast to the innermost ring, the outermost ring acts as a gateway to the core. The second overlay, which is P2P, is mainly used to provide location service for outside nodes to find the entry point of a user's matryoshkas. We also have a trusted identification server (TIS) that maintains a relation to node identifier and its pseudonym.

Supernova ia an alternative DOSN that was proposed to solve the issue of privacy and autonomy was proposed in [51] called "Supernova". It is a super-peer based architecture where incentives are provided to peers to help facilitate new node joins in the system. In this architecture, we save a node's data at multiple nodes, which are either friends or strangers suggested by super-peers. We have three types of nodes in this architecture: user profiles, friends list, storekeeper, and super-peers. Storekeepers of user 'n' are a list of nodes that have agreed to store the user's data' n' just to replicate the information. Super-peer is a special status given to any node if it provides its services to the system. These services include agreeing to store information of a new node who initially does not have many friends, maintains, and manages various services.

When we add a new node to the network, it has to go through several phases. The initial phase is when a user joins and store all his information. He will be informed of the amount of space given to him, how long he can save the information, what kind of data can be saved, and what kind of advertisements will be given to him. Once he agrees to all the terms and selects his super-peer, he enters into the next phase called "Take care phase". In this phase, the user tries to establish friendship in the system. Super-peer will keep track of new nodes up-down time and ask other stranger nodes to replicate the user data. Once this is finished, the user enters the final stage called "Settled phase" where he is settled and no longer requires a super-peer. Every update or change he does will be updated to all the storekeepers available at that time. If they are not available, they will be updated once they become available.

CHAPTER 4

USER PRIVACY IN ONLINE SOCIAL NETWORKS BASED ON ANONYMIZATION

4.1 Problem definition

We represent the social network in the form of a graph. The graph representation of the problem has already been discussed in Chapter 2. Let us assume a social network with 'n' users, and each user has 'a' attributes. The set $A = \{a_1, a_2, ...a_n\}$ are the set of all possible attributes that a user can have. If a user does not have attributes a_3 set, we assume it as NULL. Hence, all the attributes are initially set to NULL.

The friendships between the two users are directional. That is, if 'A' follows 'B', we represent an edge: $A \rightarrow B$. If 'A' is a friend of 'B', we represent an edge $A \leftrightarrow B$. Hence, all the edges are directional.

To define a social network formally, let us assume a graph G with 'n' vertices and edges between those vertices are represented by a set E. For example, if there is an edge between user 1 and user 2, then we form an edge e_{12}. Hence, $G = (V, E)$ is our social network where 'V' is the vertices/users and 'E' is the edges/links. As discussed previously, vertices represent the users and hence $V = u_1, u_2, ..., u_n$ is a set of users. Hence, $|V| = n$ and the maximum number of edges possible between 'n' users are $n \times (n-1)$. Therefore, $|E| = n \times (n-1)$ are the maximum number of edges. Also, each edge has a corresponding weight associated with it. This weight represents the closeness relation between two users. For example, if user 1 and user 2 are mother and daughter while user 2 and user 3 are colleagues. Then, there should be a way of representing that user 1 and user 2 are close. Hence, we chose a weight value of $w \in (0, 1)$, where 0 represents no relation at all, and hence 0 is not included. Similarly, 1 represents the self, and hence, 1 is also not included as there are no self-loops.

A weight for edge e_{12} is represented by w_{12}. Hence, maximum number of values possible for $W = \{w_{12}, w_{13}, ..., w_{n(n-1)}\}$ is also $n \times (n-1)$.

4.2 Privacy definition

4.2.1 Proposed privacy definition

This paper aims to anonymize graph G such that the anonymized graph $G' = (V', E')$ with anonymized vertex set V', anonymized edge set E', and anonymized attribute set A' following objectives:

- Given a constant k, for each node 'u', the probability that an attacker re-identifies u is at most $\frac{1}{k}$.
- Given a constant l, the probability that an attacker re-identifies u if one of the attributes is known is at most $\frac{1}{l}$.
- Minimize information loss while maximizing the degree of anonymity.

4.3 Privacy through anonymization

It is a common practice to publish data to a trusted third party, and due to such publication, often privacy leaks happen [52]. Researchers have identified two types of attacks: identity disclosure and attribute disclosure attacks [53] [54] [19]. Identity disclosure is a type of attack where the attacker tries to match an anonymized individual with a known user in the database. Attribute disclosure is a type of attack where the attacker tries to infer unknown or hidden attributes of the user based on the publicly available information. Although a complete identity disclosure is not possible, identifying hidden attributes is very high.

Based on the privacy laws, all the social network administrators need to sanitize the data before it is published. However, data sanitization or data anonymization is a complex problem and have no way of quantifying. Hence, we need to understand the usage of such data publication and anonymize it so that the user's privacy is maintained and the utility of

data. One of the naive methods to anonymize user data is to remove "Personally Identifiable Information", also known as the PII. This PII is a single attribute such as SSN or a group of attributes such as names, user name, age, and location information. This solution is far from sufficient in preserving privacy [37] [55]. We have discussed a variety of these anonymization techniques in chapter 3.

4.4 Clustering for anonymization

The Enhanced-Equicardinal clustering method to achieve user anonymization is proposed in [56]. A simple clustering method ensures that there is no more individual information that can be retrieved from the anonymized network. This mechanism ensures both vertex and edge anonymization without loss of much information. All similar users can be grouped into a single cluster. For example, all the Ph.D. students from the computer science department in a given city can be grouped. This ensures that the common (non-sensitive) attributes of all such users are now represented with a single cluster head. Assume a scenario where a user's profile is nothing but their location information that is their current address. For example, <Bob, Apt 1234, 123 Main St, Bay Area, California, USA>. If the user uses this information for any Location-Based Service (LBS) query, his address can be retrieved by an attacker. However, by using the proposed method, we find 'k users of the social network whose geographical locations are close. Assume all of them live in the Bay Area, but might be on different streets and different apartments. The generalized profile now would be <1, Bay Area, California, USA>. Using this generalized profile, the user asks for an LBS query. So, even if the attacker retrieves this information, there are 'k' other users living in the same area, and also the exact address is still protected for all the users. However, the results might vary, that is, the utility drops. Hence, we aim to preserve the utility and provide the highest obfuscation. However, it is to be noted that a simple clustering method will not ensure that all the clusters are of equal size. This is to ensure that there are no information leaks due to the difference in anonymization between users. Hence, we enhance a simple clustering algorithm like k-means to ensure that every cluster contains a minimum of 'k' users. This,

in turn, also ensures that each user cannot be distinguished from '$k-1$' similar users and hence guarantees k-anonymity.

As a method of clustering data, the k-means algorithm is widely used because of its simplicity and ability to converge extremely quickly in practice. In [27], Hay et al. applied structural generalization approaches that groups nodes into clusters, by which privacy details about individuals can be appropriately hidden.

The proposed method aims to cluster users whose profiles are similar. For example, two students studying at University A should be grouped than two students studying in different universities. k-means clustering is one such method that groups users based on their closeness. The main advantage of k-means is that it is fast and produces tighter clusters. k-medoids algorithm, a similar yet powerful technique, is an alternative to a k-means algorithm where we need to reduce the sum of squared distances (SSD) between cluster points and their corresponding cluster heads. Although k-medoids give us a smaller SSD value than k-means, it will not provide us better privacy in our scenario. This is because k-medoids is calculated based on the principle that a medoid is calculated for a cluster rather than mean. A medoid is one of the data points in a cluster. In our scenario, data points represent the user's profile. So, selecting a medoid as the cluster head indicates that we are choosing one of the user's profiles as the cluster head. Thus, we are exposing the exact user's attribute values. However, the main aim of this paper is precisely the opposite. Additionally, since the k-values are not large, a simple brute force attack will reveal the exact user to which this profile is linked.

Let us understand this in detail with the help of an example scenario. Assume that the user's profile only contains locations which are the exact addresses of where that user lives like Apt 1234, 123 Main street, Area, City, Country, Zip code. Now, if we select a medoid, this exact address would be chosen as a cluster head and all the users in that cluster has the same address. If users are searching for a Location Based Query (LBS), like nearby restaurants, they do not want to reveal a specific address. Instead, it would be much helpful if the query searches for restaurants near the area, which is achieved by k-means.

Also, if k-medoids is used, an attacker knows that out of 'k' users, one of them resides at that exact location. So, with the help of other profile information like gender and occupation, he can identify the person. This is possible because none of the profile attributes are anonymized. For example, there are four users in a cluster: Alice, Bob, Cathy, Danny. By using k-medoids, we generate a profile:

<Male, Ph.D. student, room no: 1234, Building 123, Main street, Atlanta>

Let us assume that the attacker has the background knowledge that Bob resides in Atlanta, and he is a Ph.D. student. Now, using this information, he can identify that the cluster head is indeed Bob's profile, and the attacker now knows Bob's exact address. To avoid such circumstances and knowledge attacks, we propose using k-means clustering where no cluster head's profile is the same as any user's profile. Instead, it generalizes, and the knowledge-based attacks are not possible. Another issue with k-means is its tolerance to outliers. We have used a k-means algorithm to consider outliers as a part of our system. In the problem statement, we have mentioned that the network we use contains user's profiles that have been scanned for any possible Sybil users. As there will not be any Sybil users in the system, all the users are genuine users of the social network. Hence, we need to provide the same privacy level to everyone. If we use a clustering technique that leaves out outliers, it means we are leaving out users just because they live in a remote area, or their likes and dislikes are different from other users. However, in fact, such users require more privacy than the rest, and the attacker who identifies that a user lives in a far area can be vulnerable to theft. Hence, using k-means avoid ignoring outliers and provide similar privacy for all of them.

Another advantage with k-means is its tolerance to outliers. We have used a k-means algorithm to consider outliers as a part of our system. In the problem statement, we have mentioned that the network we use contains user's profiles that have been scanned for any possible Sybil users. As there will not be any Sybil users in the system, all the users are genuine users of the social network. Hence, we need to provide the same privacy level to everyone. If we use a clustering technique that leaves out outliers, it means we are leaving

out users just because they live in a far area, or their likes and dislikes are different from other users. But in fact, such users require more privacy than the rest, and the attacker who identifies that a user lives in a far area can be vulnerable for theft. Hence, using k-means avoid ignoring outliers and provide similar privacy for all of them.

Hence, we use k-means clustering to cluster the users. We use the user's profile (attributes of the user) to calculate the distance between two users. Let us assume, user vector $\overrightarrow{|u_i|}$ represents all the attributes associated with user u_i and let 'R' is the number of attributes. Similarly, let $\overrightarrow{|u_i|}$ and $\overrightarrow{|c_j|}$ be user 'i' and centroid 'j'. Distance between them can be calculated in an R-dimensional space as:

$$D_{i,j}^2 = |P_i - C_j|^2 = \sum_{n=1}^{R} (p_{i,n} - c_{j,n})^2 \tag{4.1}$$

One problem by clustering the users using k-means clustering is that the resulting clusters can be of uneven size. This is a significant problem in our scenario as not all users are protected with the same level of privacy. Hence, we need clusters of equal size to ensure the even privacy distribution overall users. Hence, to achieve overall k-anonymity, let us construct a distance matrix in ascending order. Let D_A represents this ascending ordered distance array. The values in this array are described below:

$$\begin{pmatrix} D_{i_1,j_1} \\ D_{i_2,j_2} \\ D_{i_3,j_3} \\ . \\ . \\ . \\ D_{i_m,j_m} \end{pmatrix} \tag{4.2}$$

The array D_A is of size $nk \times 1$. This method is explained in detail in Algorithm 1.

Algorithm 1 Calculate Distance Matrix

Input: $N \rightarrow$ Set of users, $C \rightarrow$ Set of cluster centroids, $A \rightarrow$ Set of attributes **Output:** $D_A \rightarrow$ Ordered distance matrix

For $i = 1$ to n
    For $j = 1$ to k
        $D(u_i, c_j) = \frac{\Sigma_{a \in A} dist(u_{ia}, c_{ja})}{|A|}$
    endfor
endfor
$m \leftarrow 1$
For i = 1 to n
    For j = 1 to k
        $D_A(m, 1) \leftarrow D(u_i, c_j)$
        $D_A(m, 2) \leftarrow i$
        $D_A(m, 3) \leftarrow j$
        $m \leftarrow m + 1$
    endfor
endfor
$D_A \leftarrow$ MergeSort(D_A) based on first column

4.5 Termionology

4.5.1 Information loss

According to [57], information loss can be calculated as:

$$\text{Information Loss}(L) = \frac{SSE}{SST} \tag{4.3}$$

where SSE is the error sum of squares (information loss with in group), and SST is the total sum of squares (information loss between groups).

$$SSE = \sum_{i=1}^{k} \sum_{j=1}^{n_i} (x_{ij} - \overline{C_l})'(x_{ij} - \overline{C_l}) \tag{4.4}$$

where, k is the number of clusters, n_i is the user n that belongs to cluster C_i, and $\overline{C_i}$ is the average data vector over the whole set of users belonging to cluster C_i

$$\overline{C_i} = \sum_{i=1}^{k} \frac{t_i^2}{n_i} - \frac{t^2}{n} \tag{4.5}$$

where, t_i is the total sum of users in cluster C_i, T is the grand total of all clusters, n_i is the user n that belongs to cluster C_i, and n is the total number of users

$$SST = \sum_{i=1}^{n} \sum_{j=1}^{|C|} |x_{ij} - C_j| - \frac{\sum_{i=1}^{n} x_{ij} \times \sum_{j=1}^{|C|} C_j}{n \times |C|} \tag{4.6}$$

where, x_{ij} is the user x_i belonging cluster C_j, $|C|$ is the total number of clusters, and n is the total number of users

4.5.2 Degree of anonymity

According to k-anonymity [58], a graph is k-anonymous if for every node 'v', there exist at least $k-1$ other nodes in the graph with the same degree as 'v'. That is, user degree is the same as the degree of its assigned cluster. There are $\lfloor \frac{n}{k-1} \rfloor$ more users with the same degree. Hence, we say that by assigning the users to clusters with the algorithm mentioned Algorithm 2, we obtain an $\frac{n}{k}$ degree anonymous graph. We calculate Degree of Anonymization as,

$$\text{DA} = \text{Degree}(C_{u_i}) \times l \tag{4.7}$$

4.6 Enhanced equi-cardinal clustering

In the proposed method, the main aim is to anonymize all the users in the network in such a way that they cannot be distinguished among 'k' other users. To achieve this, we should make sure that each user's anonymity is greater than or equal to 'k'. In our method, we are achieving this anonymization using clustering. Also, 'k' is calculated based on the best utility. That is, the value of 'k' is determined based on the number of users, their attributes and the values of the attributes. The anonymization degree of user u_i is the number of users in the cluster that u_i belongs to. To maintain k-anonymity, we should make sure that each cluster has at least 'k' users. But let us assume that there are two clusters C_1 and C_2. C_1 has 'k users and C_2 has '$l = 10 \times k$' users. Then, all the users in C_1 are k-anonymous while users in C_2 are l-anonymous. This difference in the anonymization level

increases the probability of information leakage. Hence, to prevent any such information leaks, we maintain equal anonymity levels to all the users. Therefore, we create clusters that are of almost the same size that is equi-cardinal clustering.

To achieve equi-cardinal clustering, we rearrange the users into different clusters. The primary aim of this method is to rearrange users into similar size clusters with minimal increase in the information loss. Hence, we aim at removing users from the clusters which contain more than $\lceil \frac{n}{k} \rceil$ users and place them in the next best cluster. However, we also have to consider users who least belong in an existing cluster, thereby reducing the increase in information loss.

To achieve this, let us start from the first element in the ascending ordered distance array, D_{i_1,j_1}, and assign user u_{i_1} to cluster C_{j_1} if it satisfies the following two conditions:

1. Size of cluster C_{j_1} should be less than or equal to $\lceil \frac{n}{k} \rceil$;

2. User u_{i_1} is not assigned

Since all the clusters should be of equal size, the maximum number of users allowed per cluster is $\lceil \frac{n}{k} \rceil$. If any user cannot be assigned, we ignore and forward it with the next element in the array. This process is explained in the Algorithm 2.

Algorithm 2 Equi-cardinal k-means Clustering

Input: $N \rightarrow$ Set of users, $k \rightarrow$ number of clusters, $A \rightarrow$ Set of attributes
Output: $C \rightarrow$ Culsters with assigned users
$C \leftarrow k$-meansClustering(N,k)
$D_A \leftarrow$ CalculateDistance(N,C,A)
assigned $\leftarrow n \times 1$ zero matrix
$C \leftarrow k \times 1$ array of empty lists
For each element m in D_A:
    $u \leftarrow D_A(m, 2)$
    $c \leftarrow D_A(m, 3)$
    If assigned(u) $\neq 0$ AND $length(C_c) \leq \lceil \frac{n}{k} \rceil$
        $C_c \leftarrow$ append u
        assigned(u) $\leftarrow 1$
    endif
endfor

4.6.1 Achieving k-anonymity

Theorem 1: For each node u, the probability that an attacker re-identifies 'u' lies in the range $(\lfloor n/k^2 \rfloor, \lceil n/k \rceil)$.

Proof. Let us consider the initial graph or the un-anonymized graph $G = (V, E)$ where V is the user set, E is the edge set, and $|v| = n$. Let us assume the anonymized graph as $G' = (K, E)$ where K is the number of clusters and E' is the modified edge set. Also, according to the proposed algorithm, each cluster has at least $\lfloor n/k \rfloor$ users. Let us assume that the attacker has background knowledge of a user u_i which means that $\overline{|u_i|}$ is known. From the anonymized graph G', we know each cluster and its corresponding vector. Thus, an attacker can identify the cluster a user belongs to with the following probability:

$$p = \begin{cases} 1, & \text{if } u_i \in C_j \\ \frac{1}{k}, & \text{otherwise} \end{cases} \tag{4.8}$$

If an attacker correctly identifies the cluster, there are still $\lfloor n/k \rfloor$ or $\lceil n/k \rceil$ users in C_j and hence the probability that the attacker identifies the right user is:

$$p = \begin{cases} (\lfloor \frac{n}{k} \rfloor, \lceil \frac{n}{k} \rceil), & \text{if } u_i \in C_j \\ (\frac{1}{k}\lfloor \frac{n}{k} \rfloor, \frac{1}{k}\lceil \frac{n}{k} \rceil), & \text{otherwise} \end{cases} \tag{4.9}$$

Hence, the range can be guaranteed as $(\frac{1}{k}\lfloor \frac{n}{k} \rfloor, \lceil \frac{n}{k} \rceil) \approx (\lfloor \frac{n}{k^2} \rfloor, \lceil \frac{n}{k} \rceil)$.

□

Lemma 1: Objective 1 holds.

Proof. For a given constant 'c', choose the number of clusters 'k' in such a way that

$$\frac{1}{c} \geq \frac{n}{k^2} \tag{4.10}$$

This can be further reduced as:

$$k = \lceil \sqrt{cn} \rceil \tag{4.11}$$

□

Theorem 2: For a given set of users (n) and defined number of clusters (k), the minimum anonymity degree for any user u_i in the network is:

$$D(u_i) = n - (k-1) \times \lceil \frac{n}{k} \rceil \tag{4.12}$$

Proof. According to the algorithm proposed, the distance between user 'i' and cluster centroids are:

$$D(u_i, c_a) \leq D(u_i, c_b) \leq D(u_i, c_c) \leq ..... \leq D(u_i, c_k) \tag{4.13}$$

By this equation, it is clear that user u_i is the closest to cluster center c_a and hence it should be assigned to cluster c_a. But, let us assume cluster c_a already contains $\lceil \frac{n}{k} \rceil$ users. Then, u_i will be assigned to the next best option, that is cluster c_b. So,

$$\text{maximum number of users per cluster} = \lceil \frac{n}{k} \rceil \tag{4.14}$$

Let us assume the worst-case scenario of the ordered distance matrix D_A, that is all the users are nearest to the single cluster centroid. Also, assume $\gamma = \lceil \frac{n}{k} \rceil$. Then,

$$\begin{pmatrix} D_{i_1,j_1} \\ D_{i_2,j_1} \\ D_{i_3,j_1} \\ . \\ . \\ . \\ D_{i_n,j_1} \end{pmatrix} \begin{pmatrix} D_{i_1,j_2} \\ D_{i_2,j_2} \\ D_{i_3,j_2} \\ . \\ . \\ . \\ D_{i_n,j_2} \end{pmatrix} \quad ... \quad \begin{pmatrix} D_{i_1,j_k} \\ D_{i_2,j_k} \\ D_{i_3,j_k} \\ . \\ . \\ . \\ D_{i_n,j_k} \end{pmatrix} \tag{4.15}$$

According to the algorithm proposed,

users $u_1, u_2, u_3, \ldots, u_\gamma$ will be assigned to cluster c_1, users $u_{\gamma+1}, u_{\gamma+2}, u_{\gamma+3}, \ldots, u_{2\gamma}$ will be assigned to cluster c_2, ... , and users $u_{(n-1)(\gamma+1)}, u_{(n-1)(\gamma+2)}, u_{(n-1)(\gamma+3)}, \ldots, u_n$ will be assigned to cluster c_k.

So, clusters c_1 to c_{n-1} have $\lceil \frac{n}{k} \rceil$ users. Remaining users belong to cluster c_k. Total number of users in $c_1 = \lceil \frac{n}{k} \rceil$ Total number of users in c_1 through $c_{k-1} = (k-1)\lceil \frac{n}{k} \rceil$ Remaining users $= n - (k-1)\lceil \frac{n}{k} \rceil$ according to [2],

$$n - (k-1)\lceil \frac{n}{k} \rceil \leq k\lceil \frac{n}{k} \rceil \tag{4.16}$$

Hence, the lower bound on data anonymity is

$$D(u_i) \geq n - (k-1)\lceil \frac{n}{k} \rceil \tag{4.17}$$

□

4.6.2 Extending for l-diversity

The extension to the original algorithm is proposed in [59]. k-anonymity is efficient and straightforward but suffers from its drawbacks. One of the significant disadvantages is that it is vulnerable to attribute disclosure. Numerous researchers have worked on this drawback to improvise the algorithm, e.g., [16] [17]. Two other notable attacks were identified in [18]: homogeneity and the background knowledge attack.

According to [18], "An equivalence class is said to have l-diversity if there are at least l "well-represented" values for the sensitive attribute." In this method, the term "well-represented" can be interpreted in many ways. In this paper, we use entropy as the information-theoretic notion and employ the concept of Entropy l-diversity. We use this metric to measure the diversity of each cluster.

$$Entropy(E) = -\sum_{s \in S} p(E, s)\ \mathrm{logp}(E, s) \tag{4.18}$$

where S is the set of sensitive attributes, and $p(E, s)$ is the fraction of records in E that have sensitive value s. If the set S is divided into two sub-blocks S_a and S_b, then Entropy(S)≥min(Entropy(S_a), Entropy(S_b)). So, to achieve l-diversity, we need to maintain an entropy of at least $log(l)$ for the entire table. This process of this enhancement is explained in detail in Algorithm 3.

In this algorithm 3, we first compute the diversity of each cluster and save them in an array called 'D'. But any cluster which is at least l-diverse does not have to go through the post-processing step. Hence, we remove such clusters from our processing algorithm. The remaining clusters are then sent to the processing. Since this is a greedy algorithm, we combine the most, and the least diverse clusters to verify the combined cluster is l-diverse. Once we combine those two clusters, there are two outcomes: formed new cluster is 'l'-diverse or it is not. If it is l-diverse, it is removed from the processing step and continue with the remaining clusters. If the new cluster still is not 'l'-diverse, we now have new clusters in the processing algorithm and continue with our step. We keep doing this until all the clusters are at least 'l'-diverse. In all of this process, since we are not removing any elements from the cluster, our k-anonymity principle still holds for the newly formed clusters.

Algorithm 3 Post-process to ensure l-diversity

Input: $C \rightarrow$ Culsters ensuring k-anonymity, $l \rightarrow$ desired diversity
Output: $C \rightarrow$ Culsters ensuring k-anonymity and l-diversity
For each cluster c in C:
    $D[c] \leftarrow$ diversity of cluster 'c'
endfor
remove all elements from D whose value $\geq$ l
While any value in $D <$ l
    $H \leftarrow$ cluster with high diversity value
    $L \leftarrow$ cluster with least diversity value
    $M \leftarrow H + L$
    $C \leftarrow C - \{H, L\} + M$
endwhile

4.6.3 Adding directions to edges

It is also important to observe that the information leak can happen through edges. By observing how the nodes are connected in a graph, an attacker can gain insight on who that node might be if related background information is provided [32] [38] [34] [36]. Hence, it is imperative to conceal the link information between nodes. Our clustering technique will group users with similarity. Now, all the users in a single group are represented by a single cluster head. Hence, the edges between users belonging to two clusters can be modified into super edges. Let us say there are two users u_1 and u_2 in cluster c_1 and two users u_3 and u_4 in cluster c_2. Our initial OSN contains edges between users $u_1 \rightarrow u_2$, $u_1 \rightarrow u_3$, and $u_2 \rightarrow u_4$. So, there are two edges between clusters c_1 and c_2 and one edge inside cluster c_1. Since we are clustering all the similar users into a single cluster, we ignore all the edges between them. However, we have to focus on inter-cluster edges. Since there are two such edges, we now have a super edge with weight 2. Also, to impede more information from the attacker, we neutralize the weight by averaging with the number of users in both clusters. The final super edge between clusters c_m and c_n can be calculated as:

$$e_{c_m,c_n} = \frac{\sum e_{u_{c_m},u_{c_n}}}{|c_m| + |c_n|} \tag{4.19}$$

where, $e_{u_{c_m},u_{c_n}}$ is an edge where one node of the edge belongs to user in cluster c_m and the other belongs to user in cluster c_n. $|c_i|$ is the number of users in cluster c_i.

4.6.4 Adding weights to edges

A real-world social network has edges that are weighted and directed. Many social networks like Twitter, Instagram, and Reserchgate have a concept called followers and followed. Every user has some followers, and he might be following other people. This concept arises in need of direction for the edges. A celebrity with 1000 followers and following a single person is different from a regular person who follows 1000 celebrities and is followed by a single person. So, we give an outward-directed edge for the concept "follows" and an

inward-directed edge for the concept "followed by". In the previous example, the celebrity has 100 inward-directed edges and one outward-directed edge while the regular person has 100 outward-directed edges and one inward-directed edge.

For such social networks, ignoring the direction of edges results in huge information loss. Hence, we propose to maintain two different edges between clusters. There will be two edges between any two given clusters having both outward-directed edge and inward-directed edge. The edge weights of both the edges are calculated using the edge anonymization technique discussed in Section 4.6.3. This process is explained in detail in Algorithm 4.

Algorithm 4 Edge anonymization for a weighted directed network

Input: $C \rightarrow$ Culsters ensuring k-anonymity and l-diversity, $E \rightarrow$ original edge set of all users

Output: $C \rightarrow$ Culsters ensuring k-anonymity and l-diversity and are edge anonymized

For each cluster pair c, c' in C:
    $n \rightarrow$ number of users in cluster c
    $m \rightarrow$ number of users in cluster c'
    outward = 0
    inward = 0
    For each outward edge e_{out} from c to c':
        outward = outward + e_{out}
    endfor
    For each inward edge e_{in} in C from c to c':
        inward = inward + e_{in}
    endfor
    Outward edge weight from c to $c' \rightarrow \frac{outward}{|m+n|}$
    Inward edge weight from c to $c' \rightarrow \frac{inward}{|m+n|}$
endfor

4.7 Computational analysis

Theorem 3: Dividing the nodes into 'k' clusters is a subset of Edge partitioning problem for a number of users $n \geq 3$ which is NP-complete

Proof. Given a graph $G = (V; E)$, our problem is to determine whether the edge-set E can be partitioned into subsets E_1, E_2, ... 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 can deduce that several other edge-partition problems are NP-complete. To show that EP_n is NP-complete, we can reduce our problem to 3SAT problem, which is known to be NP-complete. A set of clauses $C = \{C_1, C_2, ..., C_r\}$ in variables u_1, u_2, ..., u_s is given, each clause C_i consisting 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. The problem is to determine whether C is satisfiable, that is, whether there is a truth assignment to the variables which simultaneously satisfies all the clauses in C. A clause is satisfied if exactly of its literals has value "true".

$$\sum_{j=1 \text{ to } k} l_{i,j} = 1 \tag{4.20}$$

Hence any final solution should contain exactly 'k' vertices and hence, it is an edge partition problem, which is NP-complete. □

Theorem 4: Computational complexity of the proposed algorithm for 'n' users with 'd' attributes divided into 'k clusters is: $O(n(d + k))$.

Proof. Equi cardinal k-means clustering, as proposed in Algorithm 2, combines two algorithms and computation of its own.

1. **k-means clustering:** We refer [60] to compute clusters. This algorithm has a running time of $O(nd)$ with approximation ratio of $(1+\epsilon)$.

2. **Calculation:** This is proposed in Algorithm 1 which runs in $O(nk)$ time.

3. **Re-arranging clusters to form equal sized clusters:** To re-arrange clusters, we compare with the array D_A which is of size $n \times k$. Hence, comparing and assigning each element in D_A takes $O(nk)$ time.

Thus the total time complexity of the algorithm is:

$$
\begin{aligned}
&= O(nd) + O(nk) + O(nk) \\
&= O(nd) + O(nk)(asO(2nk) = O(nk)) \\
&= O(nd + nk) \\
&= O(n(d + k))
\end{aligned}
$$

□

4.8 Experimental results

The effectiveness of the proposed algorithm has been verified by utilizing two different real-life datasets. The first dataset is the Yelp dataset [61]. The recent version of the dataset was released in January 2018 and has been online for the Yelp challenge. Yelp dataset is a customer review dataset where each user is connected with several other users and has the user's profile information. This dataset contains a total of 1.1 million users and their reviews. We have focused on two files in this dataset, users, and friends. These two files gave us information on user profiles/attributes and the connection/edge information between users. There are a total of 18 attributes that include information about the user like the number of reviews given, the number of user reviews, and average stars are given.

Another dataset that is used for our experiments is the Facebook dataset provided by Stanford [62]. There are a total of 1 million users in this dataset and 25 attributes for each user. User attributes include gender, country, language, and residence. Facebook is a social network data. But unlike real-world social networks, we do not have many attributes associated with each user. Hence, this dataset is also useful for understanding the behavior of the proposed method.

We compare our proposed algorithms with three other algorithms. The first one is the probability-based random obfuscation (PRO) introduced in [63]. In this method, random obfuscation is achieved by perturbing graph G, randomly removing edges. However, the

vertices are grouped if the edge has been removed. To generate a random number, the Bernoulli trial has been used. This trial generates random probability values. If the probability is greater than 0.5, the edge will not be removed. If the probability is less than 0.5, the edge is removed.

The second method is obfuscation by 2* neighborhood weighted grouping (NWG) as introduced in [64] [65]. In this method, we start by randomly choosing a node. We parse through all the 2-neighborhood nodes and obtain $(k-1)$ nodes whose weights are large. We can extend this algorithm to any neighborhood.

The third method is a simple k-means clustering (KMC). In this method, we cluster the users based on their profiles. These three methods are compared with our equi-cardinal clustering (ECC) algorithm proposed in Section 4.6.1 and our final l-diversity enhanced equi-cardinal clustering (LECC) for weighted and directed graphs proposed in Section 4.6.2, 4.6.3, and 4.6.4.

All the experiments were conducted on Windows 10 operating system with Intel Core (TM) Duo 2.66 GHz CPU,12 GB Memory, and Matlab 9.2 platform. Each observation has been averaged over 50 instances. We have devised three different experimental metrics to observe the performance of the proposed method. Each experiment considers the different settings of users and attributes. Evaluation metrics are discussed in Section IV as a part of the proposed method. Three metrics need to be observed in each experiment: Information Loss (IL), Degree of Anonymization (DA), and Running Time (RT).

4.8.1 Effect of the number of clusters

Our first experiment's goal is to observe the performance enhancement of equi-cardinal clustering and enhanced l-diversity clustering when compared with the traditional clustering algorithm. In this set of experiments, we ensure that both the datasets have an equal number of users, which is 10,000 users. Although the number of attributes in both datasets is different, we intend to observe the difference in the performance of all three algorithms.

Fig. 4.1 and Fig. 4.3 shows the experimental results for the Yelp dataset. Similarly, Fig. 4.2 and Fig. 4.4 shows the experimental results for the Facebook dataset.

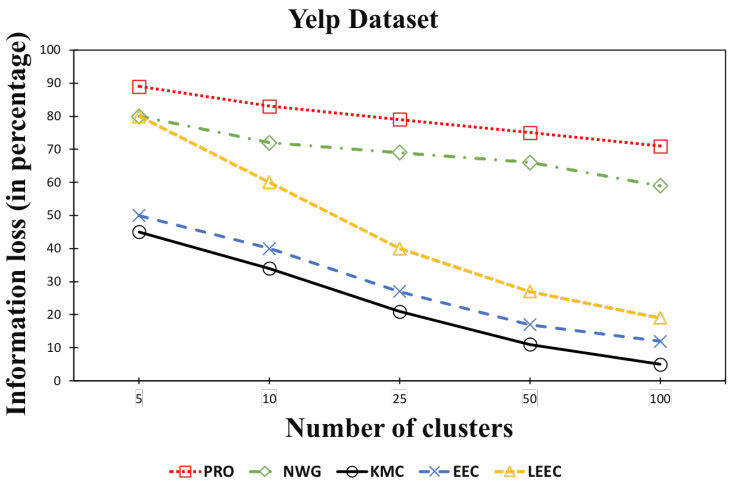

Figure 4.1. Information loss: Affect of anonymization while modifying number of clusters for Yelp dataset

We can observe from Fig. 4.3 that the DA has been increased at least ten times for equi-cardinal clustering and twenty times for l-diversity enhanced clustering. It can also be observed that this increment in anonymization is maintained by maintaining the difference in information loss to be less than 0.07% for equi-cardinal and 0.7% for enhanced l-diversity clustering. Also, from Fig. 4.4, we can observe that the degree of anonymization for equi-cardinal clustering has been increased by 50 times while the difference in information loss is less than 4%. Similarly, from Fig. 4.4, we can observe that the anonymization has increased at almost 100 times for enhanced l-diversity clustering. The difference between the achieved degree of anonymization between Yelp and Facebook datasets is due to the number of attributes in each dataset. Yelp dataset has 18 attributes, while the Facebook dataset has 25 attributes. As there is more information, clusters formed are more meaningful and closely associated. From the above observations, we can see that the proposed equi-cardinal clustering algorithm and enhanced l-diversity clustering performs better than traditional clustering when both information loss and degree of anonymization are considered.

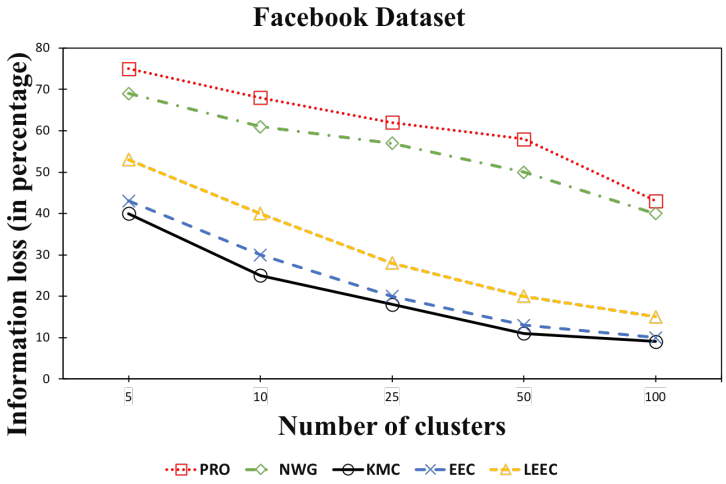

Figure 4.2. Information loss: Affect of anonymization while modifying number of clusters for Yelp datase

4.8.2 Effect of the number of users

Our second experiment's goal is to observe the performance comparison of equi-cardinal clustering, enhanced l-diversity clustering, and traditional clustering under the influence of the number of users. Both the datasets have a huge number of user data, and we have only considered 10,000 users for our previous experiment. Clusters formed can be more meaningful if we can provide a large number of points. Hence, in this set of experiments, we vary the users from 2,000 to 20,000 and observe the performance difference. Since we are focusing on a large dataset, we have considered a constant number of clusters, which is 100. Fig. 4.5 and Fig. 4.7 shows the experimental results for the Yelp dataset, while Fig. 4.6 and Fig. 4.8 shows the experimental results for the Facebook dataset.

From the experimental results of both the dataset results in Fig. 4.5, it is clear that information loss is decreased by increasing the number of users. This is a simple observation that due to the increase in data points, formed clusters are more meaningful, which means less information loss. But it is to be observed that the difference in information loss between traditional clustering and proposed clustering algorithms is also decreasing by increasing the number of clusters. Initially, let us compare the performance enhancement of equi-cardinal

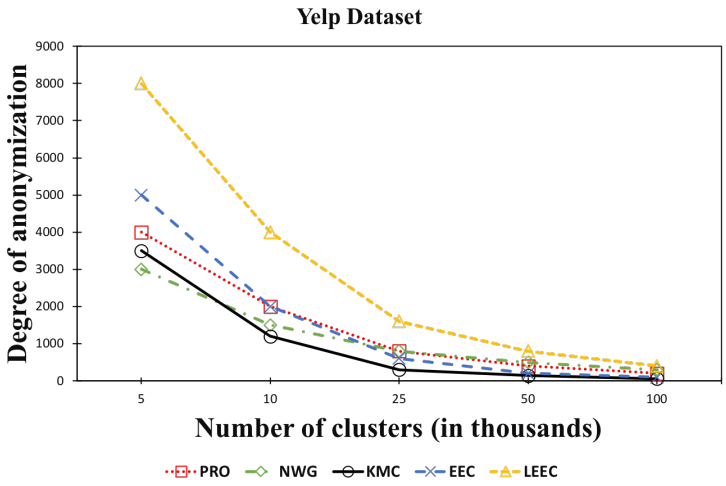

Figure 4.3. Degree of anonymization: Affect of anonymization while modifying number of clusters for Yelp dataset

clustering over traditional clustering. From Fig. 4.6, it can be observed that the degree of anonymization is increased with the increase in the number of users. It can also be observed that the increase in this value is almost exponential. With 2,000 users, the anonymization is 6% using the Yelp dataset and 5% using the Facebook dataset. While with 20,000 users, the value has increased to 30% using the Yelp dataset and 33% using the Facebook dataset.

Our next comparison is how enhanced l-diversity clustering further increases the performance. From Fig. 4.7, it can be observed that the information loss is 0.3% more than equi-cardinal and 0.5% more than traditional clustering at the maximum. But it can be observed from Fig. 4.8, that the degree of anonymization is at most two times greater than equi-cardinal clustering. As the number of users per cluster increases, there is a huge chance of their attribute similarity.

4.8.3 Effect of the removal of sensitive attributes

Our third experiment's goal is to observe the performance comparison of proposed algorithms compared to the traditional algorithm when sensitive attributes are removed. All the users in an OSN have sensitive attributes that they wish to keep private and not share

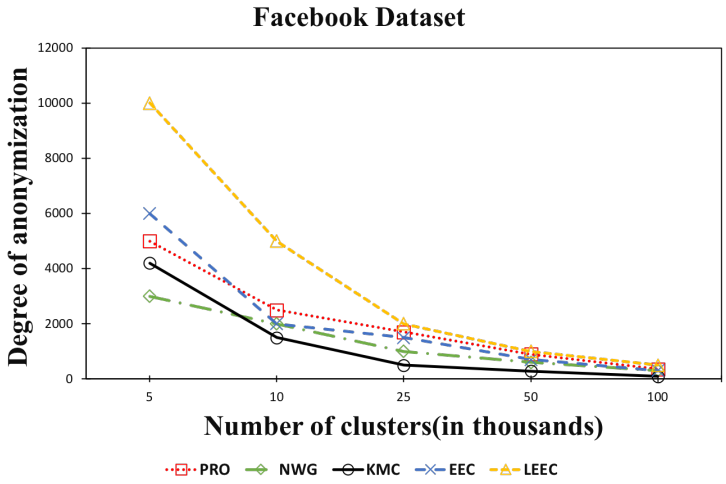

Figure 4.4. Degree of anonymization: Affect of anonymization while modifying number of clusters for Yelp datase

publicly with third-party advertisers. But it is up to the OSN administrator to decide which attributes are sensitive. In this experiment, we consider that each user provides a list of attributes that he deems as sensitive. When all such data is collected from the network, the OSN administrator has an aggregate list of which set of attributes are sensitive for the entire network. From this set, he can remove the top 'n' sensitive attributes. In this set of experiments, we have ranked all the attributes in the datasets according to their sensitivity towards users privacy. We then started removing the top 'n' to observe their effect on the anonymization process. Fig. 4.9 shows the experimental results of the Yelp dataset while Fig. 4.10 shows the results of the Facebook dataset.

From Fig. 4.10, we can observe that there is a linear increment in the information loss by removing the sensitive attributes. This observation can be seen in both the datasets. This observation confirms that the information loss is increased irrespective of the dataset. Also, the difference in information loss between naive clustering and the equi-cardinal clustering method has increased by 70% in the Yelp dataset and 50% in the Facebook dataset. Similarly, the difference in information loss between equi-cardinal clustering and the l-diverse enhanced clustering method has increased by 90% in the Yelp dataset and 60% in the Facebook dataset.

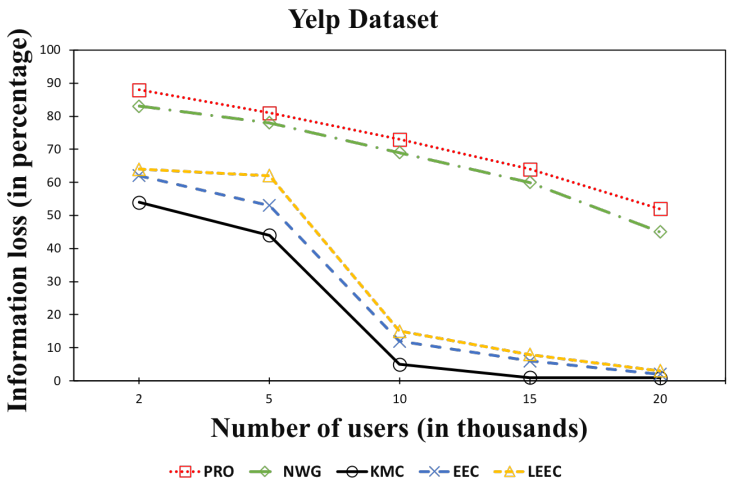

Figure 4.5. Information loss: Affect of anonymization while modifying number of users for Yelp dataset

This is because as the number of attributes decreased, the intra-cluster distance between various users increases. This leads to an increase in information loss in naive clustering. While the clusters formed are more sparse, by removing the users from the nearest cluster head and rearranging them into other clusters further increases the information loss.

It has to be noted that the degree of anonymization does not change as the sensitive attributes are removed. This is because the 'k value remains the same as there will be the same number of users in each cluster. Also, the l-value represents the average number of distinct values for each attribute. Sensitive attributes play a key role in clustering users. Hence, the values on those attributes always have minimum distinct values. By removing those sensitive attributes, we are not changing the average l-value, and hence our degree of anonymization remains the same.

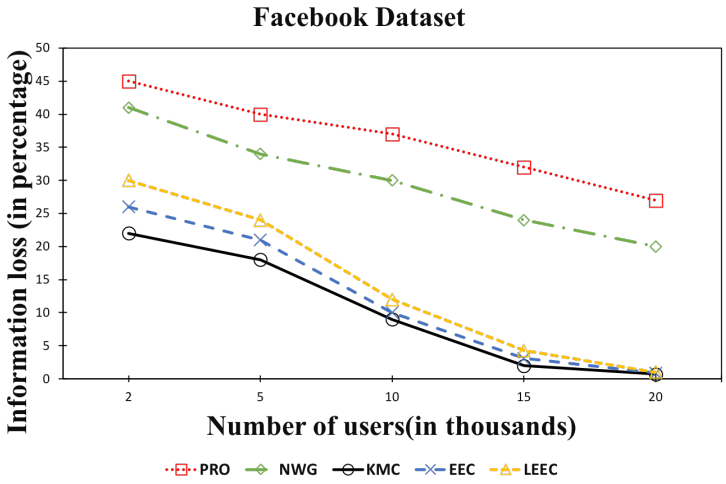

Figure 4.6. Information loss: Affect of anonymization while modifying number of users for Yelp datase

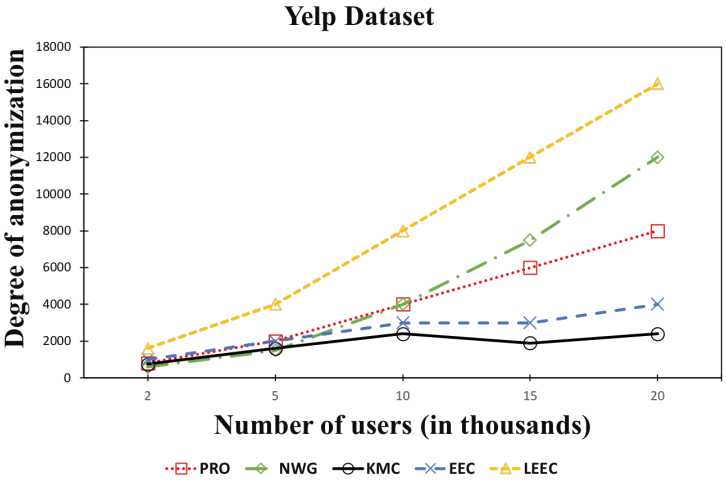

Figure 4.7. Degree of anonymization: Affect of anonymization while modifying number of users for Yelp dataset

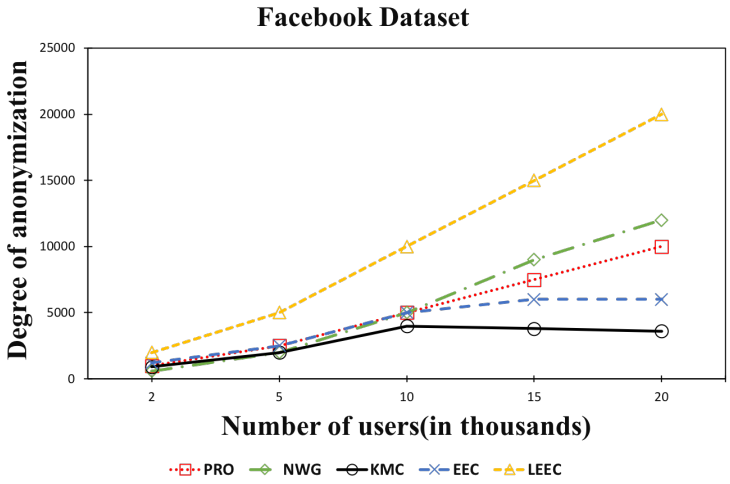

Figure 4.8. Degree of anonymization: Affect of anonymization while modifying number of users for Yelp datase

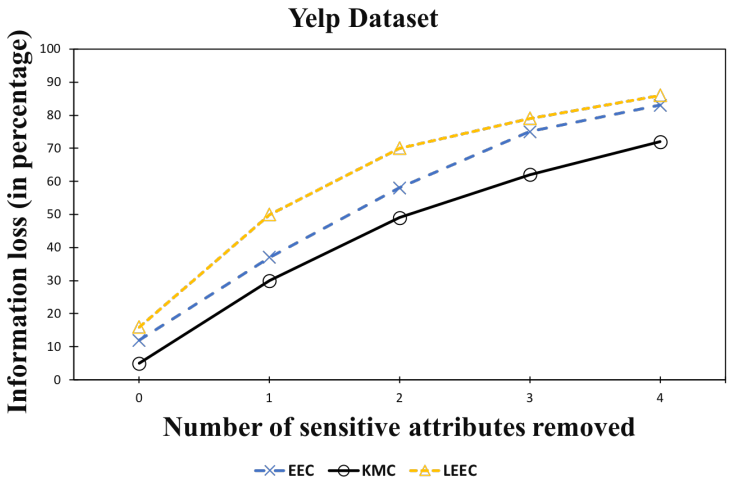

Figure 4.9. Information loss: Affect of anonymization while removing sensitive attributes for Yelp dataset

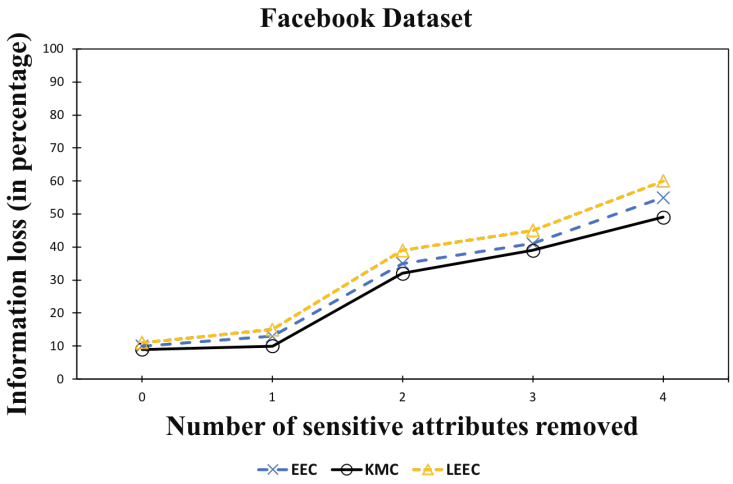

Figure 4.10. Information loss: Affect of anonymization while removing sensitive attributes for FB dataset

CHAPTER 5

PROTECTING PRIVACY OF MOBILE SOCIAL NETWORK USERS IN PREFERENTIAL LOCATION BASED SERVICES

5.1 Introduction

As mobile devices' usage has tremendously increased, Mobile social Networks (MSNs) have become an integral part of our lives. According to the recent survey by Statista [66], there are over 61% of users in North America use MSNs. It is expected that the number of users who use MSNs 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. By integrating various technologies and applications, social network providers market their products to reach a wide range of users. Their applications are compatible with PDAs, smartphones, and many more. The primary advantage of such MSN applications is that they provide various services like LBS, virtual reality, online dating, and so on. Among them, location-based queries are the most widely used service. The Online Social Network (OSN) has become a host for various business advertisements. These business models include hotels, restaurants, vacation spots, and sports. Users of OSN visiting these business models tend to review them, which in turn helps other users to visit them. Hence the business owners maintain very detailed information of their business, such as location, menu, phone numbers, and address. As the mobile internet has exploded with its increasing speed, humongous mobile applications have been developed in the past years. Similarly, all the OSN providers have their mobile-based applications designed to provide a more convenient and faster way to access their accounts on the go. MSN applications offer various advantages, including LBS, that help users to search for nearby business models.

LBSs have been gaining considerable popularity by the rapid advances in positioning technologies, e.g., Global Position System (GPS), and the development of modern smart devices with data communication capabilities. LBS query is the most common usage of any mobile user. Users query for nearby restaurants, the distance between their position and another place, and so on. A typical example of such a service is: Alice would like to query for nearby restaurants. For such queries, all we need is the exact location but not the user profile information. If the user and his location are identified by the attacker who can snoop the network, then important information can be leaked. For example, Alice visits a cancer hospital regularly and goes to a pharmacy at a particular location whenever she visits the hospital. Imagine she has enabled LBS to any of her online social networking applications. Then, the attacker can infer that Alice or a family member of hers has cancer, and also they live in the neighborhood of the pharmacy she visits. This is a piece of crucial information that should not be leaked. This attack can be seen in Fig. 5.1.

While mobile users enjoy LBS, they have to provide their real locations to the LSP, which poses a severe threat to their privacy. It is to be observed that the users might not always want to reveal their location information to others. The key to address these concerns lies in the preservation of the users' privacy when efficiently providing correct query results.

However, some LBS does consider user profile. 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 history or interest. If the user preference is Asian cuisine, then the results are sorted accordingly, and so on. Hence, accurate results for LBS is only possible if we have exact location information and correct user profile. However, if we provide those to the LBs query, then the privacy of the user is not maintained.

5.2 Problem definition

Let us denote a time series of social graphs as G_0,G_1,. . . .,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

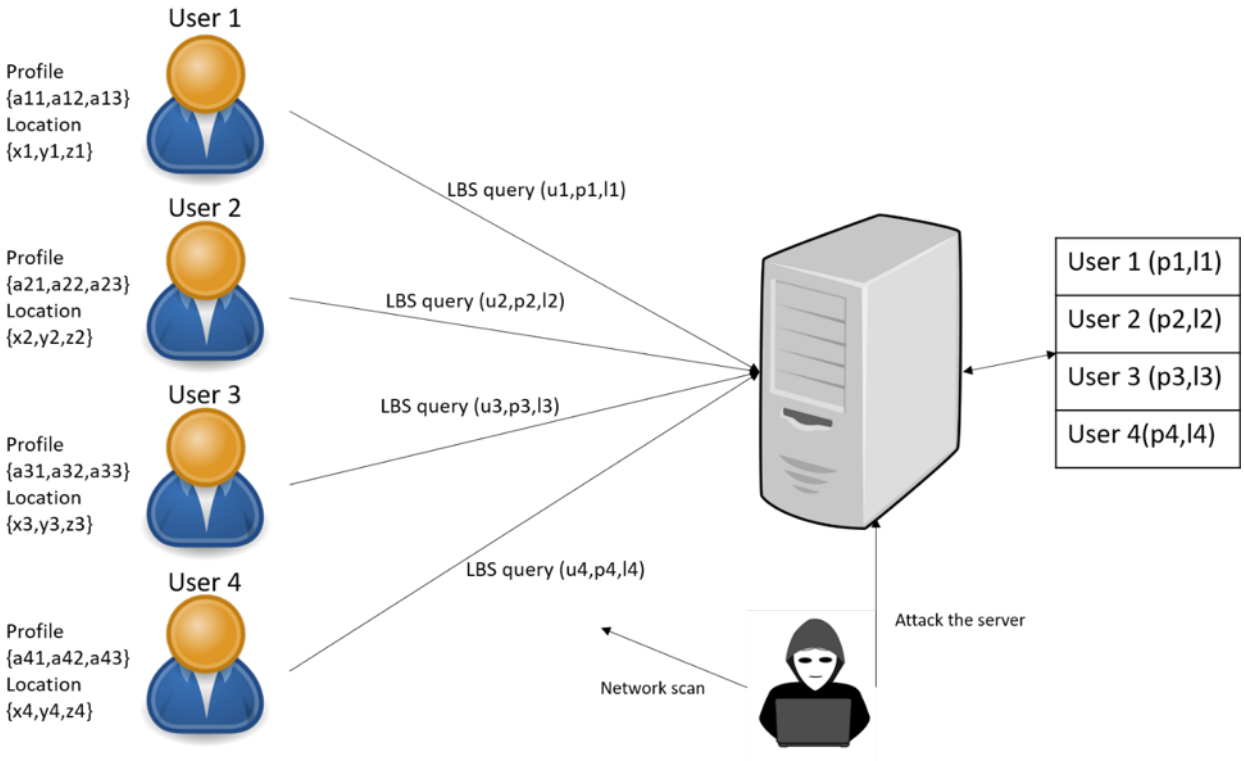

Figure 5.1. Problems of existing LBS query with user privacy

the users at 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 denote 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 equi-cardinal clustering and send out the obfuscated user details 'o' with its generalized attributes.

$$U = u_1, u_2, u_3, \ldots, u_n$$

$$u_i = a_1, a_2, a_3, \ldots, a_A$$

For a given 'k', this paper aims to obfuscate 'u' into 'o' in such a way that:

- There are at least 'k' users with the same attributes as 'o'
- Minimize information loss

5.3 Homomorphic encryption

Homomorphic encryption is a form of encryption method in which we can perform operations on the ciphertext, and the results are valid on the plain text [67]. One of the classic examples of homomorphic encryption is the RSA algorithm, in which multiplication of two ciphertexts is equal to the value of encryption of the multiplication of corresponding plain text messages. Hence, we can say that the RSA algorithm is homomorphic under multiplication. Such an encryption technique is highly useful when we have only the ciphertext, and we do not want to reveal our plain message but perform operations on the plaintext. Hence, homomorphic encryption is one of the best ways of implementing secure data mining algorithms. In this project, we consider the Paillier encryption technique. This encryption method is homomorphic under multiplication and addition, which are the only two operations required to implement any data mining algorithm.

5.3.1 Paillier encrytpion

Pascal Paillier introduced the Paillier cryptosystem in 1999 [68]. It is asymmetric encryption in which we generate a public and private key pair. We will see how to generate such a key pair in algorithm 5.

Algorithm 5 Paillier Key Generation

1: Choose two large primes p, q such that GCD(pq, (p-1,q-1)) = 1

2: Compute n=pq and λ = LCM(p-1,q-1)

3: Select a random integer g such that $g \in Z^*_{n^2}$

4: Compute $\mu = (L(g^\lambda \bmod n^2))^{-1} \bmod n$ where L(x) = $\frac{x-1}{n}$

5: Public Key (Encryption) is (n,g) and private key (Decryption) is(λ, μ)

Paillier Encryption has two distinctive properties.

Homomorphic under Addition:

The product of two ciphertexts results in the addition of plain text messages.

$$D(E(m_1, r_1) \times E(m_2, r_2) mod n^2) = m_1 + m_2 mod n$$

Homomorphic under Multiplication:

The product of two ciphertexts results in the addition of plain text messages.

$$D(E(m_1, r_1)^{m_2} mod n^2) = m_1 \times m_2 mod n$$

$$D(E(m_2, r_2)^{m_1} mod n^2) = m_1 \times m_2 mod n$$

5.4 Dynamic clustering based anonymization

This method is proposed in [69]. As the previous work suggests, by masking the user's actual location, we provide LBS services that cannot identify the user location. However, we can still unmask the user's private information such as name, gender, telephone number, and what LBS requires. Hence, such location masking methods prove to be worthless in an OSN application. Therefore, in this paper, we propose an approach where the user's information is masked rather than his location. By doing so, we cannot only provide accurate LBS services but also mask the user's details. To achieve this, we utilize the concept of profile generalization. For example, a user who works as a product developer wants to search for nearby restaurants. To answer this query, we do not require his occupation details. Hence, instead of sending the query with occupation as a computer scientist, it would not affect the results.

In the proposed method, we make such generalization over the entire profile to not reveal any personal attributes and not to differentiate between private and public attributes. To achieve this generalization, we utilize the clustering method. All the users in a given location radius can be clustered based on their profiles, and the profile of the cluster head is used as the profile for all the users in that cluster. By doing so, we are not randomly

generalizing the profile to the highest hierarchy but are only using the hierarchy required for that cluster.

We do not want to generalize every profile to a single master user profile. This is because some LBS services might require profile details to provide better and personalized results. For example, a query for nearby gaming areas might yield better results if it is provided with the details of the user's gaming history. It can sort the results based on his/her taste. Hence, we only generalize the user's profile to a certain extent so that the information loss of the profile is minimum, and yet identifying the user out of 'k' other users is not possible. So, our proposed method clusters users such that each cluster has 'k' users.

There are two ways to achieve the above-said results: pre-query clustering and post-query clustering. Pre-query clustering is where we perform the clustering algorithm and maintain the details of the cluster head's masked profile 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 service, there is no delay. On the other hand, post-query 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, pre-clustering has to handle mobility overhead that can be eliminated with post-clustering. However, we assume that a user once started querying for an LBS service might not stop with a single query but asks a series of queries. Hence, performing post-query 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 pre-query clustering method. Also, we take into account the user's mobility, and hence, the proposed method has to be dynamic to take into account the leaving, arriving, and returning users. The proposed algorithm is visually represented in Figure 5.2.

5.4.1 Dynamic clustering

We first assume that the entire coverage location of the OSN is divided into various circles/hexagons. This can be observed in Figure 5.3.

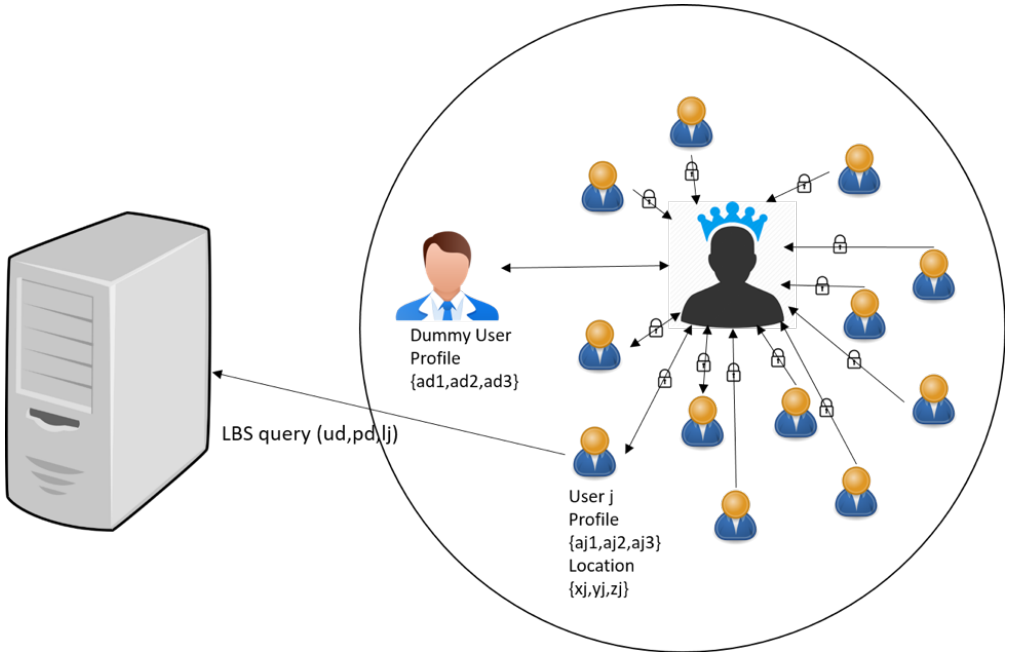

Figure 5.2. Proposed method

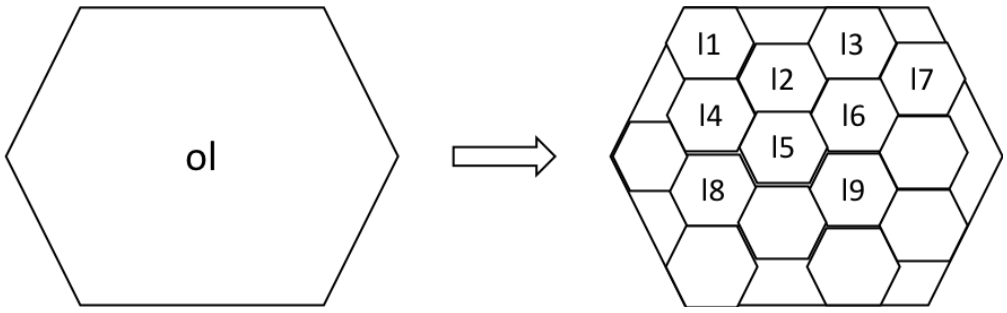

Figure 5.3. Dividing overall location into hexagons

In Figure 5.3, the overall location (o_l) is divided into various small locations ($l_1, l_2, \ldots . l_i$). Now, each location has a certain number of users. Every location is considered as a small scale online social network with users only in that hexagon. However, as the users are mobile, users tend to move from one location to another location or within the same location hexagon. We consider the user's speed and direction to estimate the users are leaving or arriving users 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 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 5.4.2. As we consider the users' 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' second, we have to redo the entire process. However, if the users' movement is slow, and there is no change in the network at location li, then we do not have to repeat the process. So, we redo the process only if the network has changed more than a certain percentage. Also, every 't' seconds is a vague number, and sometimes this might be a huge number, and sometimes it is small. Hence, we consider the speed of the user's to calculate this time. The average speed of all the users in the area is $\overline{S} = \frac{1}{n}\sum_{i=1}^{n} S_i$ and the standard deviation is $\sigma = \frac{1}{n}\sum_{i=1}^{n}(S_i - \overline{S})$.

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 $(\overline{S} - 3 \times \sigma, \overline{S} + 3 \times \sigma)$ with 99.73% probability. Hence, we assume that our $S_{min} = \overline{S} - 3 \times \sigma$ and $S_{max} = \overline{S} + 3 \times \sigma$. This gives us the information about the fastest moving user who leaves the area early, and the slowest moving user who leaves last. Hence, the average speed at a given location can also be written as the fastest-moving and slow-moving user's average speed.

$$S_{avg} = \frac{S_{max} + S_{min}}{2} \tag{5.1}$$

So, if we perform operations based on this average speed, we will capture the network change. As we know, $time = \frac{distance}{speed}$ and distance is the maximum distance user has to cover

in a location. For easier calculations, we assume hexagons to be circles, and hence maximum distance is the *diameter* $= 2 \times radius$. So, for every time $t = \frac{2 \times r}{s_{avg}}$ 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 redo the election process and clustering. This whole process is summarized in Algorithm 6.

Algorithm 6 Dynamic Clustering

Input: U $\rightarrow$ Users, UT $\rightarrow$ User trust factor, r $\rightarrow$ Range, c $\rightarrow$ Center, UP $\rightarrow$ User profiles, x $\rightarrow$ percentage change, and s $\rightarrow$ Speed
Output: UCH $\rightarrow$ User cluster head
While t$\%\frac{r \times x}{50 \times s} == 0$
    If Head==NULL $\|$ Head out of range == TRUE
        Head $\leftarrow$ Election(U_T, r, c)
        If u_i == Head
            $U_{CH} \leftarrow$ Cluster(U_P)
        endif
    endif
    If network change in percentage = x
        If u_i == Head
            Cluster(U_P)
        endif
    endif
endwhile

5.4.2 Trusted leader election

We have to perform the election to obtain a leader of the location. We utilize the network leader election protocol proposed by Zhou et al. [70]. However, we need to define a leadership score to find the leader with the maximum score.

$$LS = x \times CP + (1 - x) \times TS + \frac{d}{s} \tag{5.2}$$

Where, LS is the leadership score, CP is the computation power, TS is the Trust score provided by the server, d is the distance of the user from the hexagon center, and s is the speed of the user.

When we try to elect a leader in our scenario, there are three factors that we need to consider. Firstly, a leadership score that will be provided by the server based on user history, his connections, and many other factors. Many OSNs maintain such a score for validating whether a user is real or fake. We use such information to elect a local leader. Secondly, we need to identify a person who has better computational power. Since we need to cluster all the users at the leader, we need a more powerful device. Hence, we consider this factor. Lastly, we need to know the user's distance from the center and his speed to calculate the time he will be in the area. This is because if we elect a user who is on the edge of our hexagon/circle, he will most probably leave the area even before the clustering is finished or leave immediately afterward. We need such a user who stays in the local area for a maximum amount of time.

To securely compute the leader among all the users in the network, we utilize the Paillier encryption scheme provided in [68] and a secure minimum (SMIN) function proposed in the [71]. We use the Paillier encryption system as it is homomorphic encryption that can compute the difference between two numbers without finding their actual values. Algorithm 7 discusses the process in detail.

Algorithm 7 Trusted leader election

Input: global trust score values stored at the server, U $\leftarrow$ list of users in the area, $E_{pk} \leftarrow$ public encryption key generated using Paillier system.
Output: Leader
PE $\leftarrow E_{pk}(0)$
Leader $\leftarrow E_{pk}(0)$
For u_i in U
    Compute LS_i for user i based on its computation power and the trust score
    CE $\leftarrow E_{pk}(LS_i)$
    PE $\leftarrow$ SMIN(PE,CE)
    If PE == CE
        Leader $\leftarrow E_{pk}(i)$
    endif
endfor
At the server: Compute $D_{sk}(Leader)$ to get the leader
    Server informs all the users in the area about the Leader

---

We utilize the Enhanced equi-cardinal clustering algorithm explained in Chapter 4 to cluster the users with k-anonymity and l-diversity.

5.4.3 Anonymous LBS query

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 is shown in Fig. 5.4.

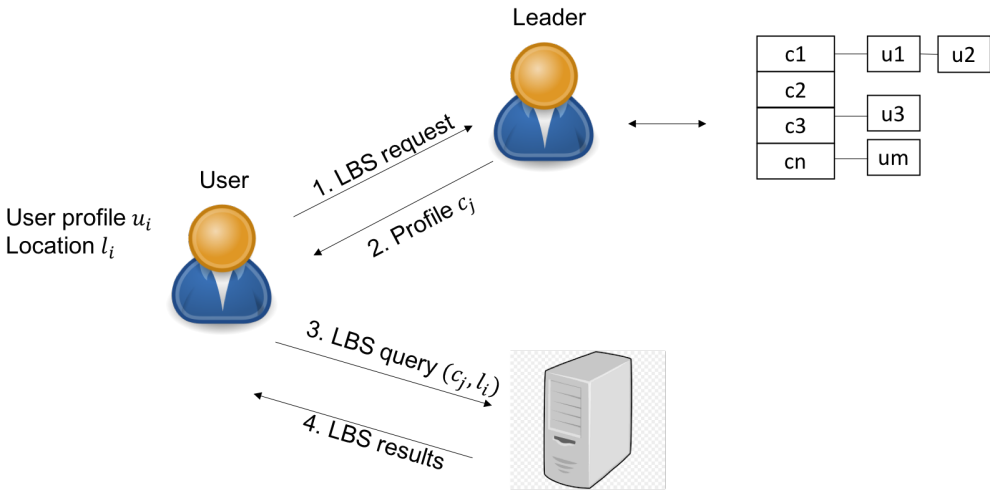

Figure 5.4. LBS query steps

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.

5.5 Analysis

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

$$p[u_{ij} = 1 | u^*_{ij}] \leq \frac{1}{k} \tag{5.3}$$

Theorem 1: 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 $(IL + \frac{1}{k})$

$$\frac{\sum_{i=1}^{k}\sum_{j=1}^{n_i}(x_{ij} - \overline{x_l})'(x_{ij} - \overline{x_l})}{\sum_{i=1}^{k}\sum_{j=1}^{n_i}(x_{ij} - \overline{x})'(x_{ij} - \overline{x})} + \frac{1}{k} \tag{5.4}$$

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.

$$\frac{\sum_{i=1}^{n/k}\sum_{j=1}^{n_i}(x_{ij} - \overline{x_l})'(x_{ij} - \overline{x_l})}{\sum_{i=1}^{n/k}\sum_{j=1}^{n_i}(x_{ij} - \overline{x})'(x_{ij} - \overline{x})} + \frac{1}{k} \tag{5.5}$$

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

$$k \times \frac{n}{k} \times n + \frac{1}{k} \tag{5.6}$$

$$n^2 + \frac{1}{k} \tag{5.7}$$

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

$$2 \times n - \frac{1}{k^2} = 0 \tag{5.8}$$

$$2n = \frac{1}{k^2} \tag{5.9}$$

$$k = logn \tag{5.10}$$

□

Theorem2: Achieving k-Anonymity is NP-Hard

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, ... in such a way that each E_i generates a subgraph of G isomorphic to the complete graph Kn 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 will exhibit a polynomial reduction from the known NP-complete problem 3SAT, which is defined as follows. A set of clauses $C = \{C_1, C_2, ..., C_r\}$ in variables u_1, u_2, ..., u_s is given, each clause C_i consisting 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. The problem is to determine whether C is satisfiable, that is, whether there is a truth assignment to the variables which simultaneously satisfy all the clauses in C. A clause is satisfied if exactly of its literals has value "true".

$$\sum_{j=1tok} l_{i,j} = 1 \tag{5.11}$$

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

Theorem 3: A network change of $x\%$ is equivalent to the time difference

$$t = \frac{rx}{100} \times (\frac{1}{s_{min}} + \frac{1}{s_{max}}) \tag{5.12}$$

Where, 'r' is the radius of the clustering area '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:

$$t = \frac{d}{s_i} \tag{5.13}$$

The maximum distance that the user has to travel out of the clustering area is the diameter of the circle: $2 \times r$. Hence, $t = \frac{2 \times r}{s_i}$.

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.

$$t = \frac{2 \times r}{s_{max}} \tag{5.14}$$

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

$$t = \frac{2 \times r}{s_{min}} \tag{5.15}$$

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

$$t = \frac{\frac{2\times r}{s_{min}} + \frac{2\times r}{s_{max}}}{2} = r \times \frac{1}{s_{min}} + \frac{1}{s_{max}} \tag{5.16}$$

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

$$t = r \times \frac{1}{s_{min}} + \frac{1}{s_{max}} \times \frac{x}{100} \tag{5.17}$$

$$t = \frac{rx}{100} \times (\frac{1}{s_{min}} + \frac{1}{s_{max}}) \tag{5.18}$$

□

5.6 Performance metrics

5.6.1 Profile generalization

Our aim in this proposed method is to generalize the user profile rather than generalize 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 is 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 that is the profile generalization is 0%, there is a chance that the user is identified by an attacker correctly. Hence, we need to have a mechanism to see how much the profile is generalized and how much privacy is maintained with that generalization. To do that, we provide a profile generalization calculation method.

$$profile\ generalization\ of\ user\ u_i = \frac{dist(u_i, C_{u_i})}{dist(u_i, GP)} \times 100 \tag{5.19}$$

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

5.6.2 Accuracy

It is essential to understand how accurate the results are with the profile generalization. Since we have LBS 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, we take into account the user's profile to sort the results according to the user's interest for a preferential search. This is how we ensure that the results are relevant to the user and not generalized results. We use an algorithm proposed by [72] 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:

$$Accuracy = \frac{number\ of\ different\ results}{total\ number\ of\ results} \times 100 \tag{5.20}$$

5.6.3 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 post-query method, we do update the clusters based on the user movement. So, it is possible that 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.

5.7 Experimental results

To measure the effectiveness of the proposed algorithm, we have considered a social network based synthetic dataset. We have used Mockaroo realistic data generation generator [73] to generate this synthetic dataset. This dataset contains 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 and the speed and direction he travels in.

To further analyze the effectiveness of our prosed algorithm, we have considered two other algorithms. The first one is a spatial cloaking mechanism [44]. This protocol is based on the distributed model that computes a cloak area which covers all collaborative peers to satisfy the spatial k-anonymity. This approach is different from existing methods as it does not rely on any intermediate anonymizer; any collaborative user does not have to trust each other. It possesses stronger robustness in the scenarios with multiple initiators and the scenarios with a collision. First, a mobile user initiates the LBS, searching k-1 companions and collaborating to form a cloak area, which covers k peers according to a certain degree of k-anonymity, and these peers exchange information through an ad-hoc network. The initiator randomly selects a peer in the group as an agent who sends query messages with location information on behalf of the initiator to the LBS server. Upon this request, the LBS server seeks the desired information in the database and returns appropriate answers to the initiator through the agent. Finally, the initiator selects a satisfactory solution.

Another algorithm is a grid-based cloaking mechanism proposed by [45]. This algorithm supports both k-anonymity and l-diversity. In this algorithm, a minimum grid area is obtained for every user. For every user, we start by expanding the area until we find 'k' users (k-anonymity) with 'l' different attributes (l-diversity). This helps in reducing any unnecessary computations. This cloaking algorithm creates a temporary cloaking area by expanding its region. Starting 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. To efficiently perform this algorithm, a grid structure is used for storing the user location information. A post pruning technique is used to eliminate any unnecessary region expansion. Since the k-value is higher than l-value, there is a high probability that if we find k-users, we tend to obtain the l-value. Thus it helps to create the minimum cloaking region more efficient grid structure is maintained to store buildings and users.

Also, we have utilized the Yelp Fusion API for generating the LBS results. Yelp Fusion API provides a mechanism where we can search for a business by providing the location information. However, results are only sorted according to the distance with the actual location we sent in. As we are not modifying the location information and also focusing on the profile based LBS search, we are utilizing an algorithm proposed by [72]. This algorithm lets us filter the results further based on the user profile.

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 three different experimental metrics to observe the performance of the proposed method. Each experiment considers the various settings of users and attributes. Evaluation metrics are discussed in section IV as a part of the proposed method. Three metrics need to be observed in each experiment: Information Loss in percentage (IL), Execution Time (ET), and Accuracy of the LBS results.

5.7.1 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. We could consider an example like this. Since the dataset is over the entire USA, it is a concern if we want to divide areas that represent 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.

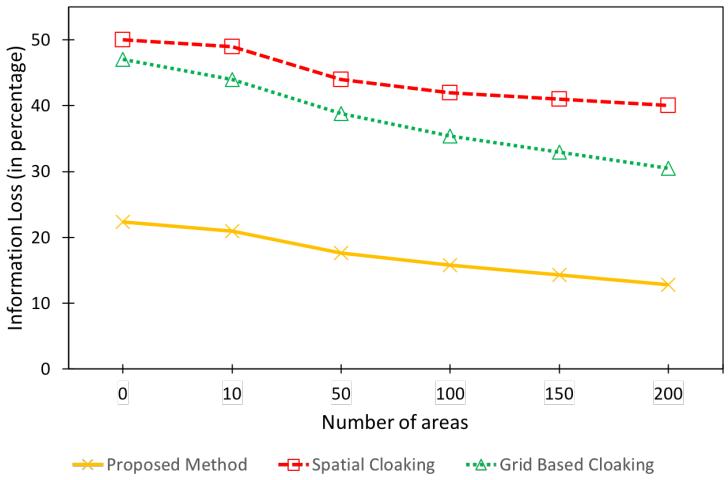

Figure 5.5. Effect of information loss while modifying number of areas

Information loss is calculated based on the amount of generalization that the user profile has undergone. As our intention in the proposed method is to reduce that information loss, you can observe that the IL is maintained at 15-20%. Although, from Figure 5.5, it can be seen that information loss has gone up to 35%. This is because, as the number of areas less, the number of users per cluster is more. As the users are more and their profiles are entirely different, information loss keeps increasing. However, other methods like spatial cloaking and grid-based cloaking achieve anonymity by just performing the location-based clustering and not based on the profile. That is why their information loss is double that of our proposed method as the profile is nowhere considered for clustering.

Execution times are shown in Figure 5.6. Execution time depends on the k-means convergence. If there are few users, then the k-means algorithm converges faster and faster. Also, it 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

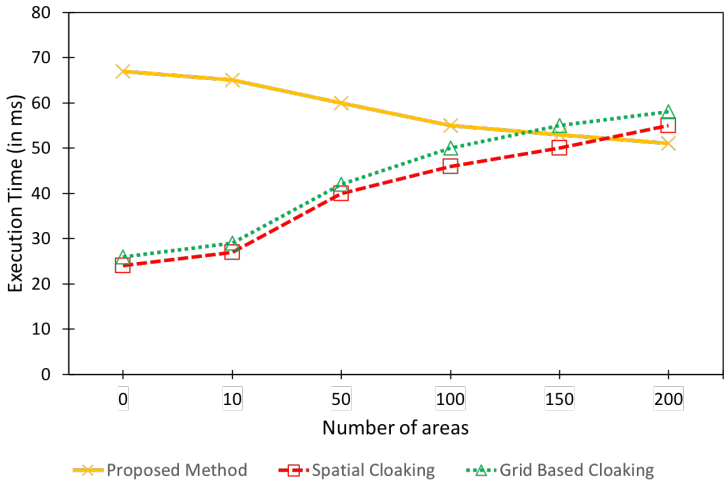

Figure 5.6. Effect of execution time while modifying number of areas

into different areas based on their location, profiles might be completely different, and hence, convergence takes longer.

Table 5.1. Accuracy while modifying number of areas

No. of areas	**Proposed Method**	**Spatial Cloaking**	**Grid-Based Cloaking**
5	81.2	53.9	54.2
10	82.3	55.17	56.2
15	84.5	57.9	60.3
20	85.9	59.3	65.4
30	87.9	60.2	68.8
50	90.4	62.4	70.9

Accuracy and information loss can be related. Results for the accuracies are shown in Table 5.1. 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 greatly 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.

5.7.2 The effect of change in the number of clusters

The goal of this experiment is to observe the performance enhancement of the proposed algorithm. We observe the effect of varying the number of clusters. By increasing the number of clusters, we decrease the number of 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. Although this might increase the generalization, it is also possible that the clusters are more meaningful and are more related. We observe the effect of this change in the following Figure 5.7 and Figure 5.8.

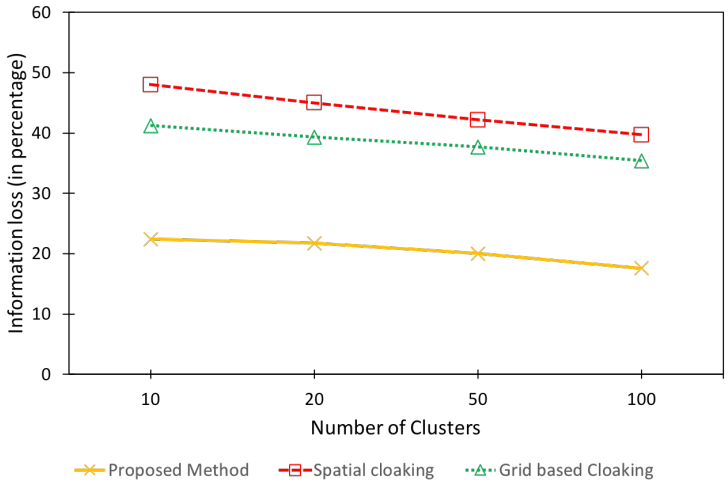

Figure 5.7. Effect of information loss while modifying number of clusters

As the k value increases, the number of users per cluster decreases. That means we have more opportunities to cluster very tight clustering. Hence, information loss can be greatly reduced by increasing the 'k' value. However, we are also compromising on the level of accuracy 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, we are not providing anonymity to the user. Hence, we should choose a 'k' such that it provides a good anonymity level and lesser information loss. By this experiment, we found k=20 provides us with 20% information loss. Hence, we maintained that value for other experiments. Additionally, it can also be observed that information loss for spatial cloaking is almost 2.5 times our proposed method information loss. This is again because the clustering happens based on the user location information.

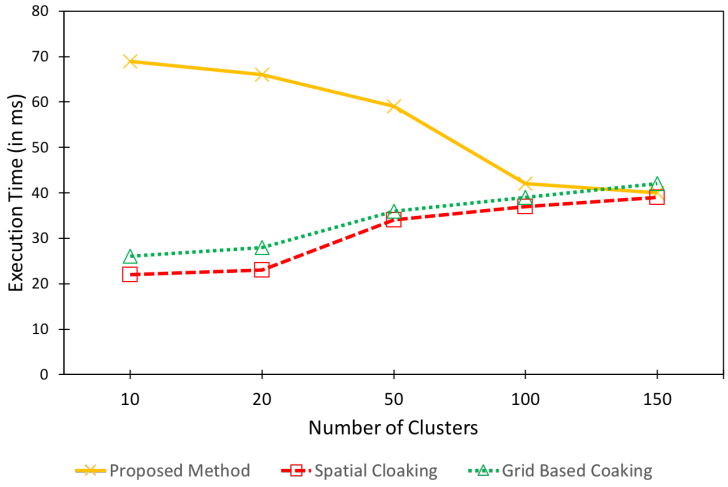

Figure 5.8. Effect of execution time while modifying number of clusters

Accuracies are shown in Table 5.2. With the same explanation as in Experiment 1, as the number of users per cluster decreases, clustering based on profiles converges faster, and clustering based on the location takes more time. However, as k=150, the execution time for all the three methods are almost the same. This is because the number of users per area is approximately 500, and users per cluster are approximately 3. Hence, there is not much difference in clustering by either location or profile as the number of users is minimal.

As the information loss is at least two times lesser than other methods, our accuracy is also almost two times higher than other methods.

Table 5.2. Accuracy while modifying number of clusters

k	**Proposed Method**	**Spatial Cloaking**	**Grid-Based Cloaking**
10	79.8	51.4	65.7
20	81.2	53.2	66.9
50	83.9	55.7	70.3
100	86.7	59.6	71.4
150	89.1	62.5	73.5

5.7.3 Mobility

Our third experiment's goal is to observe the performance of the proposed algorithm when the users are mobile. The first two experiments focus on the initial clustering and generalization of users's profile even before the user starts moving. However, when the user moves and requests real-time LBS queries, it is essential to maintain the profile generalization along with k-anonymity at the current location. To analyze this, we are considering the algorithm performance at various time instances. Results are shown in Figure 5.9. It is to be noted that the user requests continuous LBS queries. However, we only perform re-clustering when the clustered areas are modified by more than 40%. These experiments will give us an idea that if we take the snapshot of our algorithm performance at random times, how the information loss and execution time changes.

Also, we tried to observe how the accuracy changes if the re-clustering happens when there is a percentage change of existing clusters less than 50%.

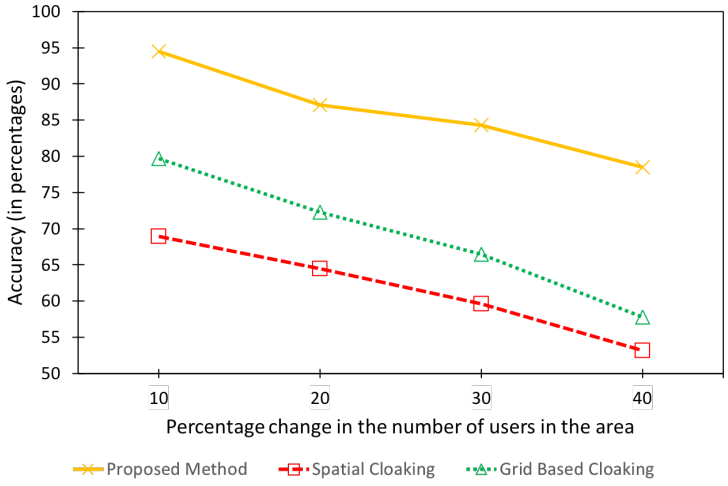

Figure 5.9. Effect of accuracy while the users are mobile and re-clustering at x% change

CHAPTER 6

FUTURE RESEARCH DIRECTIONS

6.1 Preserving privacy in continuous LBS queries

As explained in chapter 5, LBS has gained massive popularity in recent times. Almost all the MSNs now provide their own LBS services in different ways. As customers, we utilize those features even without noticing them. For example, Facebook offers various services that include finding nearby sellers; that is, we can find nearby people selling their items on Facebook. Also, we can find nearby restaurants and filter them based on the user ratings. Similarly, Instagram provides "Geotag" option to mobilize and engage more local followers. Dating networks like Tinder allows users to find nearby users with similar interests/profile. All such scenarios increase the privacy concerns of social network users.

6.1.1 Introduction

All the mobile devices in recent times are enabled with GPS location tracking. By utilizing such technology, users can query for many places in their vicinity, like restaurants and shopping, without providing their location every time manually. With the introduction of local businesses over social networks, and reviews of nearby places, users of the social network are using these location-based services with their social profiles. This geotagging is also used to find friends in a given area and notify them like Facebook's Places, Google Plus, and Loopt.

These location sharing applications have many uses. For example, the functionality "Find Friends" on an iPhone lets users know where their loved ones are. An application like "Find my Phone" helps people track their mobile phones if they are lost. This allows users to save a lot of time, effort, and money. This location sharing feature is also helpful in smartwatch applications. On a smartwatch, like FitBit, and apple watch, it allows users

to track their steps and walking distance. Hence, location-sharing is a means of identifying user's health. On a kid's smartwatch, location sharing can be life-saving. It helps parents find their kids' location and intimate them if they are in any danger area. Also, using real-time location can identify crimes going on in the area, new, and weather alerts, and many more. Hence, continuous location sharing has many advantages, and people are using these applications without realizing their privacy concerns.

The current technology for location-based social networking systems depends on a central server. This server keeps tracks of user movements and location trajectories that the user took over a period of time. By storing such information, we violate the privacy concerns of the users [74] [75] [76] [77]. Previous research on such privacy concerns aims at disguising the location information before sending it to the central server [78] [39] [79] [80].This location granularity generalization technique might be a good idea. However, if the server collects data over time, we reveal the user location trajectory information. And this still is not solved. Other methods focusing on protecting the privacy of user location is through secure message exchange between the user and his friends [81] [82]. These methods, although, are secure and private, comes at the cost of communication overhead. Since we deal with mobile devices, the availability of power is often a huge problem, and we can only obtain approximate results [83].

The primary concern in LBS queries is that these queries are not independent or isolated queries. When users enable location services for their social network, their precise location information is shared with the network servers for providing better and accurate results. As the name suggests, continuous LBS send precise location information at all times. Hence, an attacker who seeks to find a user's trajectory finds it easier to identify a future location if the current location is compromised. Therefore, the problem of continuous LBS arises and is more threatening to the user privacy than an isolated LBS query.

6.1.2 Problem statement

Let us assume a user u_i sends his LBS query $q_i = \{(x_i, y_i), s_i, d_i, k_i, u_{Amin}\}$, where

(x_i, y_i) is the precise location of user u_i , s_i is the speed of user u_i, d_i is the direction of travel for user u_i, k_i is the anonymity level requested by user u_i, and u_{Amin} is the privacy area constraint by user u_i.

A_{min}, a minimum privacy area, is to determine the smallest area where a random location and the user's exact location has more probability of colliding. Hence, every user has a desired "minimum privacy area" such that the anonymity area is always higher than A_{min}.

6.1.3 Proposed method

Privacy through k-anonymity: Various research methods [84] [85] [86] have adopted the concept of trusted third party (TTP) for providing user location privacy. The primary aim of such methods is to reduce the risk of privacy disclosure in an LBS system. This TTP that acts as an anonymizer is a middle-tier between a user and the LBS server, Location Service Provider (LSP). An anonymizer takes in the original user query with the actual precise location and sends the query with the anonymized location to the server. The functionality of an anonymizer is illustrated in Figure 6.1. k-anonymity is provided at the anonymizer. When the anonymizer generalizes the actual location, it makes sure that there are at least $k-1$ other users in the given location. To provide such a functionality, an anonymizer utilizes various methods proposed in [78] [86] [82]. When such a query is sent to the LSP, it generates the Points Of Interest (POIs) as a result of the anonymized location and sends it back to the TTP. It then refines the results based on the actual location and sends accurate results back to the user.

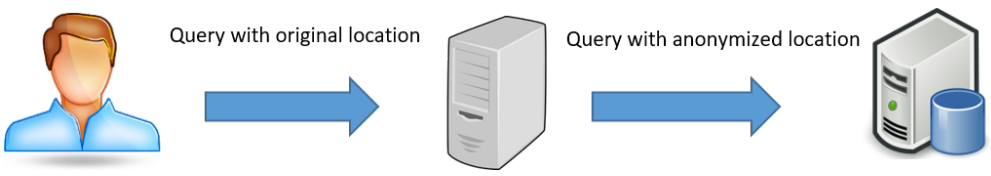

Figure 6.1. Functionality of an anonymier

An anonymizer collects a user's location information and analyzes their desired anonymity level. Based on those levels, it forms clusters and generalizes the location accordingly. There are different methods proposed on how an anonymizer changes the location information to provide the desired k-anonymity to the users. Authors in [87] have proposed a technique where 'k' dummy locations are generated by considering that the side information is easily accessible by the adversaries. Another work by [88] proposed a method called "ICliqueCloak". This method aims to maintain maximum cliques incrementally to provide k-anonymity at all times.

However, specific methods have analyzed the use of k-anonymity and cloaking mechanisms. In [89], the authors have developed a mechanism to generate dummy locations based on a virtual grid. They have analyzed that k-anonymity alone does not guarantee privacy as the user distribution, and density affects the probability. Also, k-anonymity based cloaking methods are better for a snapshot of time as they cannot prevent the attacks that focus on user trajectory rather than individual location.

Privacy through deep learning: This research aims to ensure that the attackers cannot learn the trajectory that a given user follows. For this, we need to understand how such trajectories are predicted. Neural networks and deep learning methods are playing a significant role in understanding these paths. For example, authors in [90] [91] have proposed methods based on neural networks to predict user trajectories given their history. In social networks, this history, along with user profile, plays a major role. To explain a scenario, let us assume a user who travels from point x to point y every day during weekdays at 9 AM and back from point y to point x at 5 PM. This clearly indicates that point x is the user's home, and point y is the user's office/school. However, if the user has been looking for places to visit in another city and have booked a flight ticket, it is most likely that the user will not follow the trajectory x to y during those vacation days. This gives the advertisers to advertise certain restaurants or places in that new location during those vacation days. This learning is all possible using deep learning methods.

6.1.4 Challenges/Requirements

1. **Shot-term linkability:** If a user sends two or more messages in a time frame, δ_t, an attacker should not identify that they originate from the same source. This ensures that Sybil attacks have no effect over user privacy as they tend to generate a huge number of summy queries over a short period.

2. **Long-term linkability:** Two or more messages from the same user should also be unlinkable if the time frame is large. This is because as the user tends to move, his location area is revealed. That is, any privacy-preserving mechanism can only increase the location granularity by a certain amount as the accuracy of LBS queries should not affect. Hence, if the attacker knows that the user is in area 'a_1' at time 't_1', and area 'a_2' at time 't_2' and over time, the attacker might also learn that the area 'a_1' is repeating multiple times in a day. Then, the attacker can deduce that the area 'a_1' is either the user's home or work location with a high probability.

3. **Accuracy:** With any anonymity implemented, we should guarantee that the results if the LBS query by the user are accurate. As the market is highly competitive, if the results are not accurate, the user might not use the social network, and that affects the business.

4. **Acheiveing k-anonymity:** The aim of k-anonymity should always be that the probability of finding a user in the anonymized are always greater than or equal to $\frac{1}{k}$. This should take into account multiple criteria:

   (a) Density of the area

   (b) User distribution in the area

   (c) User's profiles

   (d) User's LBS query history

6.2 Privacy preserving data aggregation in Medical IoT network

Internet of Things (IoT) has gained massive popularity in recent times due to the development of many smart devices. Smart homes are connected with voice-controlled intelligent personal assistants, and smart cities are the way of achieving solutions to many problems, including traffic control, accident detection through v2v communication, and many more. Utilizing the services of such devices has made life simpler and provided ease of access. Hence, the utilization of such smart devices has become an integral part of many people's lives.

The ultimate goal for any technology is not just to make life easier but to improve the quality of life. Focus on health improvement has never been more important than in times of COVID-19. The only current solution to such a pandemic is to maintain a healthier life. According to [92], the global healthcare sector will invest nearly $400 billion in IoT devices, software, and services. Scarpato et al. [93] provide us details about how IoT networks of heterogeneous sensors will augment the medical system. It is crucial to notice that the e-healthcare IoT system, called the Internet of Medical Things (IoMT), contains various sensors that collect data over time and across various locations. These sensors include wearable sensors like smartphones and smartwatches, ECG sensors, diabetes sensors, and pulmonary disease monitoring sensors that provide heterogeneous data collection. With an increase in several sensors, the medical field is benefitting by improving and providing many essential services. Telemedicine, telesurgery, and automated tracking of patients have never been more critical than in current times. However, the availability of such personal and detailed data might reveal sensitive information if fallen into the wrong hands. Hence, privacy in IoMT devices is vital.

Data mined from the data collected from these sensors is essential to identify problems with the patients. Imagining these sensors as basic building blocks, communication between these blocks form a network. Many problems, including routing, topology control, are still an issue in IoT networks. However, one of the significant concerns in IoT networks is the data aggregation problem. The problem is due to the lack of processing capacity and low

power availability at individual sensors. Hence, raw data is collected and aggregated at a central node. This problem has been studied in detail in some of our previous works [94] [95]. We have focused on creating an energy-efficient solution for data aggregation models in IoT networks. Nevertheless, the problem of raw data being sent over to these central nodes still poses privacy issues. Hence, it is important to provide a privacy-preserving data aggregation scheme.

6.2.1 Problem statement

Sensor devices: Let us consider an IoMt network with 'n' sensor nodes $\{S_1, S_2, ..., S_n\}$. S_{s_i} is the sensor value at node i, S_s is the aggregated sensor value. S_{s_i} values are periodically reported to a trusted third party (TTP) device like a mobile edge computing (MEC) device, or Fog device. Hence, it is important to note that:

- $|S_{s_i}| = \text{n}$
- $|S_{s_i}| \cap |S_{s_j}| = \phi$
- $\bigcup_{i=1}^{n} S_{s_i} = S_s$

Every sensor node S_i, has a storage capacity S_{c_i}, computation power S_{p_i}, and energy S_{e_i}. These three components determine what algorithms can be executed at these individual devices. In general, we assume that the power is minimal, the storage capacity of each sensor is for a limited time, and the energy is sufficient to transfer data to a TTP device.

TTP device: In any hybrid IoT network, we have a local TTP device that collects data from the sensors in its area and performs a local analysis before forwarding it to the central server. Hence, if 'n' sensors are divided into 'k' area, we have 'k' TTP devices collecting data from these S_k subset of sensor nodes. Let us assume T_i is the TTP device and S_{t_i} is the subset of sensor nodes communicating with T_i.

6.2.2 Proposed method

The overall architecture of an IoMT network can be observed in Figure 6.2. Since the sensor nodes do not have enough storage and processing power, they often communicate with an intermediate device. These devices regularly collect data from the sensors, process, and forward the information deduced to the central server. We consider two important such devices: Mobile Edge Computing (MEC) device, and Fog device.

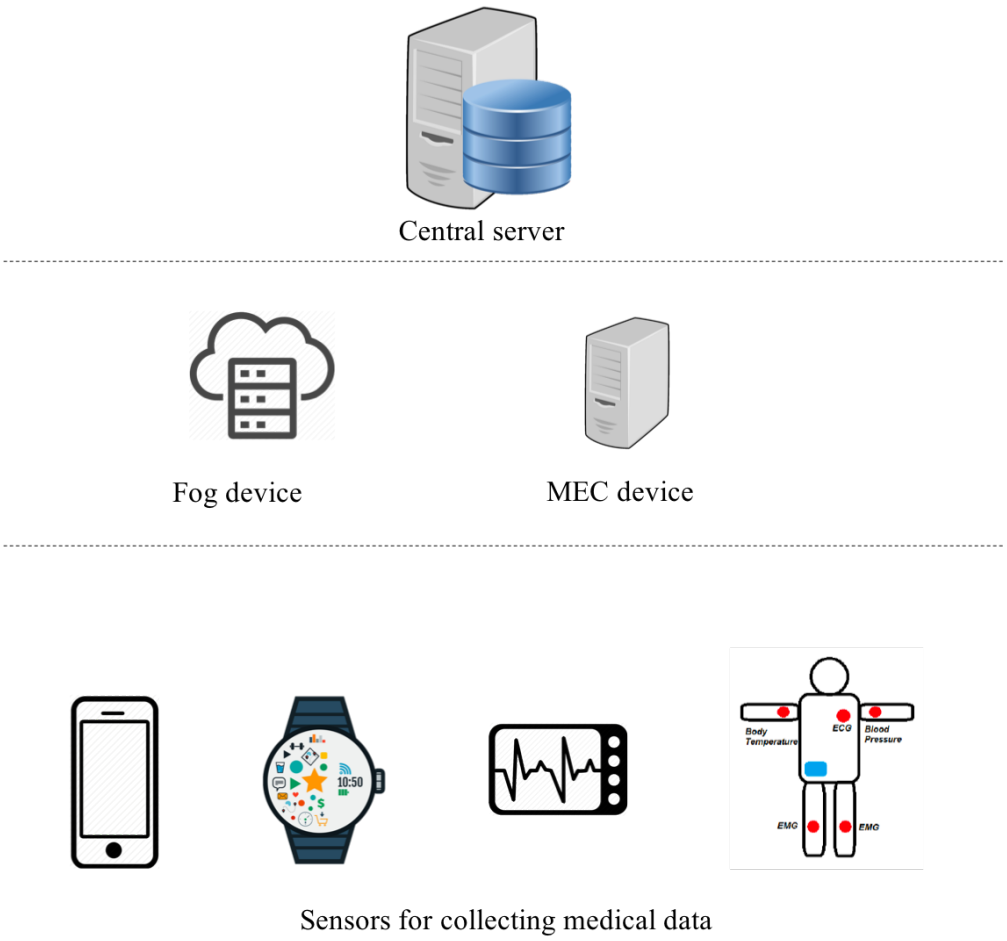

Figure 6.2. Overall architecture oif an IoMT network

MEC devices: MEC is a key technology in 5G communications. It has been proposed to address various issues, including storage, processing, and network resources. The main idea of a MEC device is to deploy it in a highly connected IoT network. This way, a MEC

device collects data from all the sensors in its area with the least latency and high reliability. Researchers in this area have focused on developing algorithms to solve the privacy-preserving health monitoring system [96], integrating healthcare to the monitoring system [97], and data mining of distributed health data [98].

Fog devices: Cisco introduced the concept of fog computing in 2012, aiming to pre-process parts of the problem and act as an edge device. The primary idea behind introducing fog computing is to reduce communication costs in a high volume environment. These devices also act as an edge device to collect information. However, fog devices have become a source of security vulnerabilities in the recent past. Several privacy-preserving data aggregation models have been proposed using fog-enhanced IoT network [97] [98] [99]. These methods only focus on two types of aggregation problems: sum and mean. However, we intend to focus on proposing a privacy-preserving data mining mechanism that aims to store the information at this fog device, rather than collect individual sensor data.

6.2.3 Challenges/Requirements

Analysis of aggregated data will result in a better understanding of the patient and thus aid the doctor in treating accordingly. However, we have to address specific problems, including:

1. **Data Confidentiality:** Only patient and authorized users should be able to see the data. Methods like Homomorphic encryption [99] and [100] have been proposed to solve this issue. However, homomorphic encryptions often involve heavy communication costs, and the power in these IoMT sensors is limited.

2. **Data Privacy:** Previous solutions to privacy challenges can be broadly classified into two categories. The first category is where we introduce noise into the system such that the utility loss is minimum [101] [102]. The second category includes designing lightweight cryptosystems that address both security and privacy challenges [103] [104]. Both these solutions do not provide realistic solutions to the challenge of untrusted

Fog/MEC servers exist. Also, utility loss sometimes leads to quandary as we deal with sensitive health data on which doctors depend on the patient's treatment.

3. **Fault Tolerance:** In a real-world scenario, we have to take into account that there might be a failed data transmission from the sensor to the cloud/ local server. This loss of data might affect the analysis that is done in the cloud. However, this failed transmission could be due to the man-in-the-middle attack that could read partial data. Hence, the developed mechanism should consider the fault tolerance that the fog/MEC based cloud system can handle [105] and how to preserve privacy while transmitting data.

6.3 Privacy preserving friend discovery in MSN

6.3.1 Introduction

One of the functionalities of a mobile social network is to connect to people in the current area. Dating social networks like Tinder, Match, and Bumble find people with similar interests in the user's area. These features also help us find groups with similar interests near the user's location. This kind of business model attracts many youngsters who love to meet new people. Hence, this can also be called as Proximity-based Mobile Social Network (PMSN). As with any social network, it is always hard to find out who is a real user and who is an attacker. Hence, the privacy of a user is compromised when such a public friend finder profile is leaked. As this model is most cost-effective, many social network providers often ignore the user's privacy details. Although the communication is secure, we might want to consider that an attacker is posing as a legitimate user and hence can gain information on users in his area. Some works have proposed privacy-preserving schemes in profile matching [106] [107] [108] [109] [110] [111]. However, these methods rely heavily on encryption methods and hence can have a high computational overhead when the number of users increases. Hence, it is not practical to apply for a mobile network user as these devices are not meant to process such heavy computations.

PMSN is a network of finding many friends with the least connections and thereby reducing the information exchange. Therefore, identifying the user nodes with the highest impact is a crucial first step. However, the problem arises when the data exchange occurs between a user and an untrusted stranger. The stranger might have a valid public key, but the intention of the stranger with user data is unknown. Hence, a viable solution is to provide a profile matching algorithm that does not involve in information exchange between two random users. This profile matching could be based on public attributes or private. By only using public attributes, we might often not generate a good matching profile. Hence, we do consider some sensitive attributes such as location, health conditions, and movie interests. Some of the recent research on private matching for PMSN has addressed this issue [106] [109] [112] [113] [114]. Researchers have considered matching based on a given sensitive attribute, such as interests [106], friends [109], and disease symptoms [112]. The idea behind these techniques is to find the intersection of profiles without disclosing actual information. This is a well-known problem, and homomorphic encryption addresses it. However, as we discussed previously, communication costs increase dramatically using encryption techniques. Therefore, a light-weight privacy-preserving profile intersection algorithm should be proposed.

6.3.2 Problem statement

Consider two user, Alice with profile $u =< u_1, \ldots, u_d >$ and Bob with profile $v =< v_1, \ldots, v_d >$. Let us assume a intersection function, f, calculated over two profiles $f(u, v)$. We also define various privacy levels and based on which the calculation of $f(u, v)$ depends.

6.3.3 Related work

PMSN can be categorized into two categories:

- FNP approach
- Cryptographic based approach

FNP approach: In this scheme [115], we have a client and server communication, as shown in Fig. 6.3. Users behave like clients, and the server takes care of matching profiles. At the server, an intersection set is calculated such that the client gets the result while the server learns nothing. In another method, Kissner et al. [116] implemented profile matching with more operations, including set intersection, union, cardinality, and over-threshold operations [117].

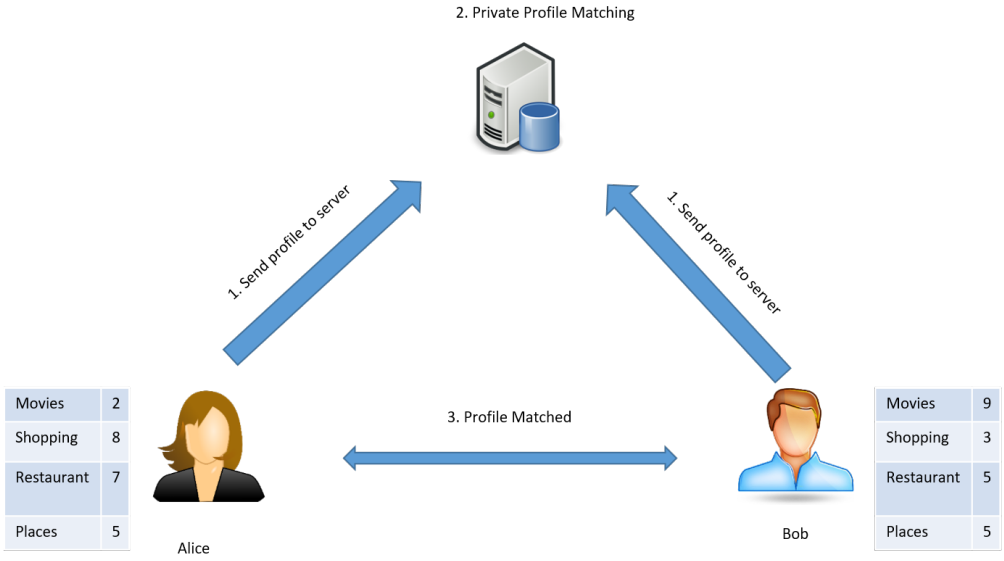

Figure 6.3. Example of a PMSN application

Another work proposed by Ye et al. [118] extends the before mentioned FNP scheme for a distributed environment. However, the complexity of this algorithm is too large. Therefore, another method has been proposed to reduce the complexity by Dachman-Soled et al. [119]. Other works like [120] [121] [122] built oblivious pseudorandom functions for matching profiles of two users. By implementing such functions, communicational and computational efficiency has been improved. Li et al. [106] implemented profile matching according to three increasing privacy levels: i) revealing the common attribute set of the two users; ii) revealing the size of the common attribute set; and iii) revealing the size rank of the common attribute sets between a user and its neighbors. They considered an honest-but-curious (HBC) adversary

model, which assumes that users try to learn more information than allowed by inferring from the profile matching results, but honestly following the protocol. They applied secure multi-party computation, the Shamir secret sharing scheme, and the homomorphic encryption scheme to ensure user profiles' confidentiality.

Cryptographic based approach: In this category, each user is represented in terms of a vector [123] [124] [125]. Each vector value is nothing but the user's attribute values. Hence, a simple naive approach is to compute two vectors' product to see the matching of profiles. However, as to maintain the privacy of each individual attribute value, authors in [126] [127] have adopted secure two-party computations. Two of the most recent works in this area include [124] [128] . Authors in [124] utilizes the concept of the dot product. Let the user 1 has an attribute vector 'u' and user 2 has an attribute vector 'v'. By computing the DotProduct(u, v), the authors measure the proximity of two attribute vectors. Another work by [128] improves on the previous solution by enabling verifiable, secure computation. The improved protocol only reveals whether the dot product is above or below a given threshold. The threshold value is selected by the user who initiates the profile matching. They pointed out the potential anonymity risk of their protocols; an adversary may adaptively adjust the threshold value to narrow down the value range of the victim profile quickly. Thus, it is required that the threshold value must be larger than a predefined lower bound (a system parameter) to guarantee user anonymity.

6.3.4 Challenges/Requirements

1. **Definition 1: Level-I privacy**

   When the protocol ends, Alice only learns $f(u, v)$, and Bob only learns f.

2. **Definition 2: Level-II privacy**

   When the protocol ends, Alice only learns $f(u, v)$, and Bob learns nothing.

3. **Definition 3: Level-III privacy**

   When the protocol ends, Alice learns if $f(u, v) < \tau_A$ holds for her personal threshold τ_A without learning $f(u, v)$, and Bob learns nothing.

If Alice and Bob both faithfully follow the protocol execution, which corresponds to the honest-but-curious (HBC) model [8], neither of them can learn the other's personal profile for all three privacy levels. In addition, with level-I privacy, although Bob cannot learn $f(u, v)$, he learns the matching metric f chosen by Alice. In contrast to level-I privacy, level-II privacy additionally requires that Bob learn nothing other than $f \in F$. Finally, level-III privacy discloses the least amount of information by also hiding $f(u, v)$ from Alice.

There might also be some external attackers (other than Alice and Bob) trying to infer users profile or disrupt PMSN operations. For example, an external attacker may eavesdrop on the messages between Alice and Bob. All our protocols can ensure that the eavesdroppers are completely blind to Alice and Bob's profiles and the matching metric(s) chosen by them, which will all be encrypted during the protocol execution.

ACKNOWLEDGMENT

This dissertation is partly supported by the National Science Foundation (NSF) under grant NOs. 1252292, 1741277, 1704287, 1829674, 1704274, 1704397, 2011845 and 1912753.

REFERENCES

[1] "Merriam-webster." https://www.merriam-webster.com/dictionary/. Accessed: 2019-06-01.

[2] "Hello, it's orkut again!." https://timesofindia.indiatimes.com/business/india-business/hello-its-orkut-again/articleshow/63716611.cms. Accessed: 2018-04-11.

[3] "Tencent qq." https://economictimes.indiatimes.com/topic/Tencent-QQ/. Accessed: 2019-08-09.

[4] "Most popular mobile social networking apps in the united states as of march 2019, by monthly users (in millions)." https://www.statista.com/statistics/248074/most-popular-us-social-networking-apps-ranked-by-audience/. Accessed: 2019-06-01.

[5] "Social networking privacy: How to be safe, secure and social." https://www.privacyrights.org/consumer-guides/social-networking-privacy-how-be-safe-secure-and-social. Accessed: 2010-06-01.

[6] Z. He, Z. Cai, and J. Yu, "Latent-data privacy preserving with customized data utility for social network data," *IEEE Transactions on Vehicular Technology*, vol. 67, pp. 665–673, Jan 2018.

[7] Z. He, Z. Cai, and X. Wang, "Modeling propagation dynamics and developing optimized countermeasures for rumor spreading in online social networks," in *2015 IEEE 35th International Conference on Distributed Computing Systems*, pp. 205–214, June 2015.

[8] "Number of social media users worldwide from 2010 to 2021 (in billions)." https://www.statista.com/statistics/278414/number-of-worldwide-social-network-users/. Accessed: 2019-06-01.

[9] "How facebook uses your data to target ads, even offline." https://lifehacker.com/how-facebook-uses-your-data-to-target-ads-even-offline-5994380/. Accessed: 2013-11-04.

[10] "Facebook was repeatedly warned of security flaw that led to biggest data breach in its history." https://www.telegraph.co.uk/technology/2020/02/09/facebook-repeatedly-warned-security-flaw-led-biggest-data-breach/. Accessed: 2020-02-09.

[11] "The ethics of social media." https://rampages.us/ashleykucuk/2015/04/27/the-ethics-of-social-media/. Accessed: 2015-04-27.

[12] "Serial sex offender admits using facebook to rape and murder teen." http://www.huffingtonpost.com/2010/03/08/peter-chapman-admits-usin-n-489674.html/. Accessed: 2015-04-27.

[13] P. Samarati and L. Sweeney, "Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression," tech. rep., 1998.

[14] P. Samarati, "Protecting respondents identities in microdata release," *IEEE Transactions on Knowledge and Data Engineering*, vol. 13, pp. 1010–1027, Nov 2001.

[15] L. Sweeney, "k-anonymity: A model for protecting privacy," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 05, pp. 557–570, 2002.

[16] T. M. Truta and B. Vinay, "Privacy protection: p-sensitive k-anonymity property," in *22nd International Conference on Data Engineering Workshops (ICDEW'06)*, pp. 94–94, April 2006.

[17] X. Xiao and Y. Tao, "Personalized privacy preservation," in *Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data*, SIGMOD '06, (New York, NY, USA), pp. 229–240, ACM, 2006.

[18] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam, "L-diversity: privacy beyond k-anonymity," in *22nd International Conference on Data Engineering (ICDE'06)*, pp. 24–24, April 2006.

[19] N. Li, T. Li, and S. Venkatasubramanian, "t-closeness: Privacy beyond k-anonymity and l-diversity," in *2007 IEEE 23rd International Conference on Data Engineering*, pp. 106–115, April 2007.

[20] M. E. J. Newman, D. J. Watts, and S. H. Strogatz, "Random graph models of social networks," *Proceedings of the National Academy of Sciences*, vol. 99, no. suppl 1, pp. 2566–2572, 2002.

[21] 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.

[22] M. Hay, G. Miklau, D. Jensen, P. Weis, and S. Srivastava, "Anonymizing social networks," Tech. Rep. 19, University of Massachusetts, Amherst, 2007.

[23] J. Qian, X.-Y. Li, C. Zhang, and L. Chen, "De-anonymizing social networks and inferring private attributes using knowledge graphs," in *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*, pp. 1–9, IEEE, 2016.

[24] A. Korolova, R. Motwani, S. U. Nabar, and Y. Xu, "Link privacy in social networks," in *Proceedings of the 17th ACM conference on Information and knowledge management*, pp. 289–298, ACM, 2008.

[25] B. Zhou and J. Pei, "The k-anonymity and l-diversity approaches for privacy preservation in social networks against neighborhood attacks," *Knowledge and Information Systems*, vol. 28, no. 1, pp. 47–77, 2011.

[26] Xifeng Yan and Jiawei Han, "gspan: graph-based substructure pattern mining," in *2002 IEEE International Conference on Data Mining, 2002. Proceedings.*, pp. 721–724, Dec 2002.

[27] M. Hay, G. Miklau, D. Jensen, D. Towsley, and P. Weis, "Resisting structural re-identification in anonymized social networks," *Proc. VLDB Endow.*, vol. 1, pp. 102–114, Aug. 2008.

[28] K. Liu and E. Terzi, "Towards identity anonymization on graphs," in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pp. 93–106, ACM, 2008.

[29] X. Ying and X. Wu, "Randomizing social networks: a spectrum preserving approach," in *SDM*, 2008.

[30] X. w. Ying and X. Wu, "On link privacy in randomizing social networks," *Knowledge and Information Systems*, vol. 28, pp. 645–663, Sep 2011.

[31] X. Ying and X. Wu, "On randomness measures for social networks," in *Proceedings of the 2009 SIAM International Conference on Data Mining*, pp. 709–720, SIAM, 2009.

[32] S. Bhagat, G. Cormode, B. Krishnamurthy, and D. Srivastava, "Class-based graph anonymization for social network data," *Proceedings of the VLDB Endowment*, vol. 2, no. 1, pp. 766–777, 2009.

[33] A. Campan and T. M. Truta, "A clustering approach for data and structural anonymity in social networks," 2008.

[34] G. Cormode, D. Srivastava, T. Yu, and Q. Zhang, "Anonymizing bipartite graph data using safe groupings," *Proceedings of the VLDB Endowment*, vol. 1, no. 1, pp. 833–844, 2008.

[35] Z. Cai and X. Zheng, "A private and efficient mechanism for data uploading in smart cyber-physical systems," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 2, pp. 766–775, 2020.

[36] E. Zheleva and L. Getoor, "Preserving the privacy of sensitive relationships in graph data," in *Privacy, Security, and Trust in KDD* (F. Bonchi, E. Ferrari, B. Malin, and Y. Saygin, eds.), (Berlin, Heidelberg), pp. 153–171, Springer Berlin Heidelberg, 2008.

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

[38] A. Campan and T. M. Truta, "Privacy, security, and trust in kdd," ch. Data and Structural k-Anonymity in Social Networks, pp. 33–54, Berlin, Heidelberg: Springer-Verlag, 2009.

[39] M. Gruteser and D. Grunwald, "Anonymous usage of location-based services through spatial and temporal cloaking," in *Proceedings of the 1st International Conference on Mobile Systems, Applications and Services*, MobiSys '03, (New York, NY, USA), pp. 31–42, ACM, 2003.

[40] L. Šikšnys, J. R. Thomsen, S. Šaltenis, and M. L. Yiu, "Private and flexible proximity detection in mobile social networks," in *2010 Eleventh International Conference on Mobile Data Management*, pp. 75–84, May 2010.

[41] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of Cryptography* (S. Halevi and T. Rabin, eds.), (Berlin, Heidelberg), pp. 265–284, Springer, 2006.

[42] C. Dwork, "Differential privacy," *Encyclopedia of Cryptography and Security*, pp. 338–340, 2011.

[43] P. Mittal, C. Papamanthou, and D. Song, "Preserving link privacy in social network based systems," *CoRR*, vol. abs/1208.6189, 2012.

[44] H. Zhangwei and X. Mingjun, "A distributed spatial cloaking protocol for location privacy," in *2010 Second International Conference on Networks Security, Wireless Communications and Trusted Computing*, vol. 2, pp. 468–471, April 2010.

[45] J. Um, H. Kim, Y. Choi, and J. Chang, "A new grid-based cloaking algorithm for privacy protection in location-based services," in *2009 11th IEEE International Conference on High Performance Computing and Communications*, pp. 362–368, June 2009.

[46] B. Zhou, J. Pei, and W. Luk, "A brief survey on anonymization techniques for privacy preserving publishing of social network data," *ACM Sigkdd Explorations Newsletter*, vol. 10, no. 2, pp. 12–22, 2008.

[47] X. Wu, X. Ying, K. Liu, and L. Chen, *A Survey of Privacy-Preservation of Graphs and Social Networks*, pp. 421–453. Boston, MA: Springer US, 2010.

[48] S. Marti, P. Ganesan, and H. Garcia-Molina, "Sprout: P2p routing with social networks," in *Current Trends in Database Technology - EDBT 2004 Workshops* (W. Lindner, M. Mesiti, C. Türker, Y. Tzitzikas, and A. I. Vakali, eds.), (Berlin, Heidelberg), pp. 425–435, Springer Berlin Heidelberg, 2005.

[49] A. Tootoonchian, S. Saroiu, Y. Ganjali, and A. Wolman, "Lockr: better privacy for social networks," in *Proceedings of the 5th international conference on Emerging networking experiments and technologies*, pp. 169–180, ACM, 2009.

[50] K. Chard, K. Bubendorfer, S. Caton, and O. F. Rana, "Social cloud computing: A vision for socially motivated resource sharing," *IEEE Transactions on Services Computing*, vol. 5, pp. 551–563, Fourth 2012.

[51] R. Sharma and A. Datta, "Supernova: Super-peers based architecture for decentralized online social networks," in *2012 Fourth International Conference on Communication Systems and Networks (COMSNETS 2012)*, pp. 1–10, Jan 2012.

[52] G. Beigi and H. Liu, "Privacy in social media: Identification, mitigation and applications," *CoRR*, vol. abs/1808.02191, 2018.

[53] G. T. Duncan and D. Lambert, "Disclosure-limited data dissemination," *Journal of the American Statistical Association*, vol. 81, no. 393, pp. 10–18, 1986.

[54] D. Lambert, "Measures of disclosure risk and harm," *JOURNAL OF OFFICIAL STATISTICS-STOCKHOLM-*, vol. 9, pp. 313–313, 1993.

[55] A. Narayanan and V. Shmatikov, "Robust de-anonymization of large sparse datasets," in *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pp. 111–125, May 2008.

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

[57] J. Domingo-Ferrer and J. M. Mateo-Sanz, "Practical data-oriented microaggregation for statistical disclosure control," *IEEE Transactions on Knowledge and data Engineering*, vol. 14, no. 1, pp. 189–201, 2002.

[58] P. Samarati, "Protecting respondents privacy in microdata release," *IEEE TransacY tions on Knowledge and Data Engineering*, vol. 13, no. 6, 2001.

[59] M. Siddula, Y. Li, X. Cheng, Z. Tian, and Z. Cai, "Anonymization in online social networks based on enhanced equi-cardinal clustering," *IEEE Transactions on Computational Social Systems*, vol. 6, pp. 809–820, Aug 2019.

[60] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 28, no. 1, pp. 100–108, 1979.

[61] "Yelp dataset challenge." `https://www.yelp.com/dataset/challenge`. Accessed: 2019-07-06.

[62] "Stanford large network dataset collection." `http://snap.stanford.edu/data/`. Accessed: 2019-07-01.

[63] F. Bonchi, A. Gionis, and T. Tassa, "Identity obfuscation in graphs through the information theoretic lens," *Information Sciences*, vol. 275, pp. 232–256, 2014.

[64] Q. Liu, G. Wang, F. Li, S. Yang, and J. Wu, "Preserving privacy with probabilistic indistinguishability in weighted social networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 5, pp. 1417–1429, 2016.

[65] M. E. Skarkala, M. Maragoudakis, S. Gritzalis, L. Mitrou, H. Toivonen, and P. Moen, "Privacy preservation by k-anonymization of weighted social networks," in *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pp. 423–428, Aug 2012.

[66] "Mobile social media - statistics and facts." `https://www.statista.com/topics/2478/mobile-social-networks/`. Accessed: 2018-07-17.

[67] C. Gentry *et al.*, "Fully homomorphic encryption using ideal lattices.," in *Stoc*, vol. 9, pp. 169–178, 2009.

[68] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Advances in Cryptology — EUROCRYPT '99* (J. Stern, ed.), (Berlin, Heidelberg), pp. 223–238, Springer Berlin Heidelberg, 1999.

[69] M. Siddula, Y. Li, X. Cheng, Z. Tian, and Z. Cai, "Privacy-enhancing preferential lbs query for mobile social network users," *Hindwai Wireless Communications and Mobile Computing*, 2020 Uner review.

[70] R. Zhou and K. Hwang, "Powertrust: A robust and scalable reputation system for trusted peer-to-peer computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, pp. 460–473, April 2007.

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

[72] A. Jaszkiewicz and R. Słowiński, "The lbs-discrete interactive procedure for multiple-criteria analysis of decision problems," in *Multicriteria Analysis* (J. Clímaco, ed.), (Berlin, Heidelberg), pp. 320–330, Springer Berlin Heidelberg, 1997.

[73] "Mockaroo." `Mockaroo`. Accessed: 2019-04-11.

[74] S. Consolvo, I. E. Smith, T. Matthews, A. LaMarca, J. Tabert, and P. Powledge, "Location disclosure to social relations: Why, when, & what people want to share," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '05, (New York, NY, USA), pp. 81–90, ACM, 2005.

[75] L. Barkhuus, B. Brown, M. Bell, S. Sherwood, M. Hall, and M. Chalmers, "From awareness to repartee: Sharing location within social groups," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, (New York, NY, USA), pp. 497–506, ACM, 2008.

[76] E. Toch, J. Cranshaw, P. H. Drielsma, J. Y. Tsai, P. G. Kelley, J. Springfield, L. Cranor, J. Hong, and N. Sadeh, "Empirical models of privacy in location sharing," in *Proceedings of the 12th ACM International Conference on Ubiquitous Computing*, UbiComp '10, (New York, NY, USA), pp. 129–138, ACM, 2010.

[77] Y. Lin, Z. Cai, X. Wang, and F. Hao, "Incentive mechanisms for crowdblocking rumors in mobile social networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 9, pp. 9220–9232, 2019.

[78] C.-Y. Chow, M. F. Mokbel, and W. G. Aref, "Casper*: Query processing for location services without compromising privacy," *ACM Trans. Database Syst.*, vol. 34, pp. 24:1–24:48, Dec. 2009.

[79] M. F. Mokbel, C.-Y. Chow, and W. G. Aref, "The new casper: Query processing for location services without compromising privacy," in *Proceedings of the 32Nd International Conference on Very Large Data Bases*, VLDB '06, pp. 763–774, VLDB Endowment, 2006.

[80] T. Wang and L. Liu, "Privacy-aware mobile services over road networks," *Proc. VLDB Endow.*, vol. 2, pp. 1042–1053, Aug. 2009.

[81] L. Šikšnys, J. R. Thomsen, S. Šaltenis, and M. L. Yiu, "Private and flexible proximity detection in mobile social networks," in *2010 Eleventh International Conference on Mobile Data Management*, pp. 75–84, May 2010.

[82] L. Šikšnys, J. R. Thomsen, S. Šaltenis, M. L. Yiu, and O. Andersen, "A location privacy aware friend locator," in *Proceedings of the 11th International Symposium on Advances in Spatial and Temporal Databases*, SSTD '09, (Berlin, Heidelberg), pp. 405–410, Springer-Verlag, 2009.

[83] S. Mascetti, C. Bettini, and D. Freni, "Longitude: Centralized privacy-preserving computation of users' proximity," in *Secure Data Management* (W. Jonker and M. Petković, eds.), (Berlin, Heidelberg), pp. 142–157, Springer Berlin Heidelberg, 2009.

[84] R. Schlegel, C. Chow, Q. Huang, and D. S. Wong, "User-defined privacy grid system for continuous location-based services," *IEEE Transactions on Mobile Computing*, vol. 14, pp. 2158–2172, Oct 2015.

[85] S. Zhang, Q. Liu, and Y. Lin, "Anonymizing popularity in online social networks with full utility," *Future Generation Computer Systems*, vol. 72, pp. 227 – 238, 2017.

[86] H. Zhu, F. Liu, and H. Li, "Efficient and privacy-preserving polygons spatial query framework for location-based services," *IEEE Internet of Things Journal*, vol. 4, pp. 536–545, April 2017.

[87] B. Niu, S. Gao, F. Li, H. Li, and Z. Lu, "Protection of location privacy in continuous lbss against adversaries with background information," in *2016 International Conference on Computing, Networking and Communications (ICNC)*, pp. 1–6, Feb 2016.

[88] X. Pan, J. Xu, and X. Meng, "Protecting location privacy against location-dependent attacks in mobile services," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, pp. 1506–1519, Aug 2012.

[89] H. Lu, C. S. Jensen, and M. L. Yiu, "Pad: Privacy-area aware, dummy-based location privacy in mobile services," in *Proceedings of the Seventh ACM International Workshop on Data Engineering for Wireless and Mobile Access*, MobiDE '08, (New York, NY, USA), pp. 16–23, ACM, 2008.

[90] Y. Xu, Z. Piao, and S. Gao, "Encoding crowd interaction with deep neural network for pedestrian trajectory prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5275–5284, 2018.

[91] R. Khodayi-mehr and M. M. Zavlanos, "Deep learning for robotic mass transport cloaking," *IEEE Transactions on Robotics*, 2020.

[92] "The global market for iot healthcare tech will top $400 billion in 2022." `https://www.businessinsider.com/`

the-global-market-for-iot-healthcare-tech-will-top-400-billion-in-2022-2016-5. Accessed: 2016-05-25.

[93] N. Scarpato, A. Pieroni, L. Di Nunzio, and F. Fallucchi, "E-health-iot universe: A review," *management*, vol. 21, no. 44, p. 46, 2017.

[94] J. Li, M. Siddula, X. Cheng, W. Cheng, Z. Tian, and Y. Li, "Sampling based δ-approximate data aggregation in sensor equipped iot networks," in *International Conference on Wireless Algorithms, Systems, and Applications*, pp. 249–260, Springer, 2018.

[95] J. Li, M. Siddula, X. Cheng, W. Cheng, Z. Tian, and Y. Li, "Approximate data aggregation in sensor equipped iot networks," *Tsinghua Science and Technology*, vol. 25, no. 1, pp. 44–55, 2019.

[96] H. Liu, X. Yao, T. Yang, and H. Ning, "Cooperative privacy preservation for wearable devices in hybrid computing-based smart health," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1352–1362, 2018.

[97] L. Gu, D. Zeng, S. Guo, A. Barnawi, and Y. Xiang, "Cost efficient resource management in fog computing supported medical cyber-physical system," *IEEE Transactions on Emerging Topics in Computing*, vol. 5, no. 1, pp. 108–119, 2015.

[98] P. Verma and S. K. Sood, "Fog assisted-iot enabled patient health monitoring in smart homes," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1789–1796, 2018.

[99] R. Lu, X. Liang, X. Li, X. Lin, and X. Shen, "Eppa: An efficient and privacy-preserving aggregation scheme for secure smart grid communications," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 9, pp. 1621–1631, 2012.

[100] A. Abdallah and X. S. Shen, "A lightweight lattice-based homomorphic privacy-preserving data aggregation scheme for smart grid," *IEEE Transactions on Smart Grid*, vol. 9, no. 1, pp. 396–405, 2016.

[101] M. Yang, T. Zhu, B. Liu, Y. Xiang, and W. Zhou, "Machine learning differential privacy with multifunctional aggregation in a fog computing architecture," *IEEE Access*, vol. 6, pp. 17119–17129, 2018.

[102] V. Bindschaedler, S. Rane, A. E. Brito, V. Rao, and E. Uzun, "Achieving differential privacy in secure multiparty data aggregation protocols on star networks," in *Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy*, pp. 115–125, 2017.

[103] A. Ara, M. Al-Rodhaan, Y. Tian, and A. Al-Dhelaan, "A secure privacy-preserving data aggregation scheme based on bilinear elgamal cryptosystem for remote health monitoring systems," *IEEE Access*, vol. 5, pp. 12601–12617, 2017.

[104] D. He, N. Kumar, S. Zeadally, A. Vinel, and L. T. Yang, "Efficient and privacy-preserving data aggregation scheme for smart grid against internal adversaries," *IEEE Transactions on Smart Grid*, vol. 8, no. 5, pp. 2411–2419, 2017.

[105] C. Guo, P. Tian, and K.-K. R. Choo, "Enabling privacy-assured fog-based data aggregation in e-healthcare systems," *IEEE Transactions on Industrial Informatics*, 2020.

[106] M. Li, N. Cao, S. Yu, and W. Lou, "Findu: Privacy-preserving personal profile matching in mobile social networks," in *2011 Proceedings IEEE INFOCOM*, pp. 2435–2443, April 2011.

[107] L. Zhang, X. Li, K. Liu, T. Jung, and Y. Liu, "Message in a sealed bottle: Privacy preserving friending in mobile social networks," *IEEE Transactions on Mobile Computing*, vol. 14, pp. 1888–1902, Sep. 2015.

[108] M. Li, S. Yu, N. Cao, and W. Lou, "Privacy-preserving distributed profile matching in proximity-based mobile social networks," *IEEE Transactions on Wireless Communications*, vol. 12, pp. 2024–2033, May 2013.

[109] M. von Arb, M. Bader, M. Kuhn, and R. Wattenhofer, "Veneta: Serverless friend-of-friend detection in mobile social networking," in *2008 IEEE International Conference on Wireless and Mobile Computing, Networking and Communications*, pp. 184–189, Oct 2008.

[110] Z. He, Z. Cai, J. Yu, X. Wang, Y. Sun, and Y. Li, "Cost-efficient strategies for restraining rumor spreading in mobile social networks," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 3, pp. 2789–2800, 2017.

[111] A. M. V. V. Sai and Y. Li, "A survey on privacy issues in mobile social networks," *IEEE Access*, pp. 1–1, 2020.

[112] R. Lu, X. Lin, X. Liang, and X. Shen, "A secure handshake scheme with symptoms-matching for mhealthcare social network," *Mob. Netw. Appl.*, vol. 16, pp. 683–694, Dec. 2011.

[113] Z. Cai and Z. He, "Trading private range counting over big iot data," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pp. 144–153, 2019.

[114] X. Zheng, Z. Cai, G. Luo, L. Tian, and X. Bai, "Privacy-preserved community discovery in online social networks," *Future Generation Computer Systems*, vol. 93, pp. 1002 – 1009, 2019.

[115] M. J. Freedman, K. Nissim, and B. Pinkas, "Efficient private matching and set intersection," in *Advances in Cryptology - EUROCRYPT 2004* (C. Cachin and J. L. Camenisch, eds.), (Berlin, Heidelberg), pp. 1–19, Springer Berlin Heidelberg, 2004.

[116] L. Kissner and D. Song, "Privacy-preserving set operations," in *Advances in Cryptology – CRYPTO 2005* (V. Shoup, ed.), (Berlin, Heidelberg), pp. 241–257, Springer Berlin Heidelberg, 2005.

[117] G. Li, Z. Cai, G. Yin, Z. He, and M. Siddula, "Differentially private recommendation system based on community detection in social network applications," *Security and Communication Networks*, vol. 2018, 2018.

[118] Q. Ye, H. Wang, and J. Pieprzyk, "Distributed private matching and set operations," in *Information Security Practice and Experience* (L. Chen, Y. Mu, and W. Susilo, eds.), (Berlin, Heidelberg), pp. 347–360, Springer Berlin Heidelberg, 2008.

[119] D. Dachman-Soled, T. Malkin, M. Raykova, and M. Yung, "Efficient robust private set intersection," in *Applied Cryptography and Network Security* (M. Abdalla, D. Pointcheval, P.-A. Fouque, and D. Vergnaud, eds.), (Berlin, Heidelberg), pp. 125–142, Springer Berlin Heidelberg, 2009.

[120] S. Jarecki and X. Liu, "Efficient oblivious pseudorandom function with applications to adaptive ot and secure computation of set intersection," in *Theory of Cryptography* (O. Reingold, ed.), (Berlin, Heidelberg), pp. 577–594, Springer Berlin Heidelberg, 2009.

[121] Z. He, Z. Cai, Q. Han, W. Tong, L. Sun, and Y. Li, "An energy efficient privacy-preserving content sharing scheme in mobile social networks," *Personal and Ubiquitous Computing*, vol. 20, no. 5, pp. 833–846, 2016.

[122] C. Hazay and Y. Lindell, "Efficient protocols for set intersection and pattern matching with security against malicious and covert adversaries," in *Theory of Cryptography* (R. Canetti, ed.), (Berlin, Heidelberg), pp. 155–175, Springer Berlin Heidelberg, 2008.

[123] Rui Zhang, Y. Zhang, Jinyuan Sun, and Guanhua Yan, "Fine-grained private matching for proximity-based mobile social networking," in *2012 Proceedings IEEE INFOCOM*, pp. 1969–1977, March 2012.

[124] W. Dong, V. Dave, L. Qiu, and Y. Zhang, "Secure friend discovery in mobile social networks," in *2011 Proceedings IEEE INFOCOM*, pp. 1647–1655, April 2011.

[125] B. Goethals, S. Laur, H. Lipmaa, and T. Mielikäinen, "On private scalar product computation for privacy-preserving data mining," in *Information Security and Cryptology – ICISC 2004* (C.-s. Park and S. Chee, eds.), (Berlin, Heidelberg), pp. 104–120, Springer Berlin Heidelberg, 2005.

[126] A. C. Yao, "Protocols for secure computations," in *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*, SFCS '82, (Washington, DC, USA), pp. 160–164, IEEE Computer Society, 1982.

[127] O. Goldreich, S. Micali, and A. Wigderson, "How to play any mental game," in *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, STOC '87, (New York, NY, USA), pp. 218–229, ACM, 1987.

[128] I. Ioannidis, A. Grama, and M. Atallah, "A secure protocol for computing dot-products in clustered and distributed environments," in *Proceedings International Conference on Parallel Processing*, pp. 379–384, Aug 2002.