Bangalore University

University Visvesvaraya College of Engineering

Department of Computer Science and Engineering

K R Circle Bengaluru-560001

A project report on

"PRESERVE-WHILE-SHARING: EFFICIENT TECHNIQUES FOR PRIVACY PRESERVING IN ONLINE SOCIAL NETWORK DATA SHARING"

submitted in partial fulfillment for the project work of 8th semester, B. Tech(CSE),

Project Associates:

K Thanujashree	18GAEC9025
Lakshmi Priya B	18GAEC9030
Nimitha J	18GAEC9042

VIII Semester B.Tech (CSE)

Under the guidance of:

Dr Venkatesh

Associate Professor, Dept. of CSE, UVCE

Year 2021-22

BANGALORE UNIVERSITY

UNIVERSITY VISVESVARAYA COLLEGE OF ENGINEERING

Department of Computer Science and Engineering

K R Circle , Bangalore- 560001

CERTIFICATE

This is to certify that the project entitled **"Preserve-While-Sharing : Efficient Techniques for Privacy Preserving in Online Social Network Data Sharing"** is a bonafide work carried out by **K Thanujashree(18GAEC9025), Lakshmi Priya B(18GAEC9030), Nimitha J (18GAEC9042)** bonafide students of University Visvesvaraya College of Engineering in partial fulfillment for the project in 8th semester, Computer Science & Engineering, Bangalore University, Bengaluru during the year 2021-22. It is certified that all the corrections/suggestions indicated for Internal Assessment have been incorporated in the Report. The project report has been approved as it satisfies the academic requirements in respect of project work prescribed for the said degree.

Guide:

Dr. Venkatesh
Associate Professor
Department of CSE
UVCE

Chairperson:

Dr. H S Vimala
Professor & Chairperson
Department of CSE
UVCE

Examiner 1: _____

Examiner 2: _____

ACKNOWLEDGEMENT

We take this opportunity to thank our institution University Visvesvaraya College of Engineering for having given us an opportunity to carry out this project.

We would like to thank Dr. Ramesh H N, Principal, UVCE, for providing us all the facilities to work on this project. We are indebted to him for being our pillar of strength and inspiration.

We wish to place our gratitude to Dr. H. S. Vimala, Professor and Chairperson, Department of Computer Science and Engineering, UVCE, who helped us to make our project a great success.

We are grateful to Dr. Venkatesh, Department of Computer Science and Engineering, Assistant Professor UVCE, for his valuable suggestions and support, which has sustained us throughout the course of the project.

We express our sincere thanks to all teaching and non-teaching staff, Department of Computer Science and Engineering, UVCE for all the facilities that they provided us to successfully complete this project.

We also thank our parents and friends for their continuous support and encouragement.

- K THANUJASHREE (18GAEC9025)

- LAKSHMI PRIYA B (18GAEC9030)

- NIMITHA J (18GAEC9042)

ABSTRACT

Online Social Networks (OSNs) have become one of the major platforms for social interactions, such as building up relationships, sharing personal experiences, and providing other services. Rapid growth in Social Network has attracted various groups like the scientific community and business enterprise to use these huge social network data to serve their various purposes. The process of publishing these huge online social network data for various trend analysis raises privacy concerns due to personal data shared online. Privacy control mechanisms have been deployed in popular OSNs for users to determine who can view their personal information. However, user's sensitive information could still be leaked even when privacy rules are properly configured due to inference attacks. Even if Online Social Network owners allow their users to set customizable privacy, attackers can still find out users' private information by finding the relationships between public and private information with some background knowledge and this is termed as inference attack. In order to defend against these inference attacks this project could completely anonymize the user identity. Though complete anonymization provides 100 percent privacy, it reduces the utility of data exponentially.

Project goal is to design an optimization algorithm that achieves a trade-off between self-disclosure utility and their privacy. We propose two privacy preserving algorithms to defend against an inference attack that is privacy preserving algorithm (PPA) targeting high utility and social relation based multidimensional knapsack problems with greedy heuristics with low computational complexity.

TABLE OF CONTENT

Title	Page No
1. Introduction 1.1 History of Social Network 1.2 Challenges in social networks 1.3 Objective of the project 1.4 Scope of the project 1.5 Motivation	
2. Literature Survey 2.1 Introduction 2.2 Summary of papers	
3. System Design 3.1 Problem Statement 3.2 Proposed Model 3.3 Privacy Preserving Algorithm (PPA) 3.4 Social relations based dkp Algorithm (S-dkp) 3.5 Inference attack techniques	
4. Implementation 4.1 Dataset 4.2 System Defined Functions 4.3 User Defined Functions	
5. Results 5.1 Requirements Specifications 5.2 Performance evaluation metrics 5.3 Experiment Results	
6. References	

LIST OF FIGURES

Title	Page No
Figure 1.1.1 Timeline of the launch dates of many major SNS	
Figure 1.2.1 Social Network Graph	
Figure 3.2.1 Flow Chart of the Project	
Figure 3.3.1 Flowchart of PPA	
Figure 3.4.1 Flowchart of S-dkp	
Figure 4.3.1 Function to get Adjacency matrix of the OSN	
Figure 4.3.2 Function to calculate Privacy	
Figure 4.3.3 Function of PPA Algorithm	
Figure 4.3.4 Function of S-dkp Algorithm	
Figure 4..35 Function of Triadic Closure	
Figure 4.3.6 Function of Jaccard	
Figure 4.3.7 Function of Resource Allocation	
Figure 4.3.8 Function of Adamic Adar	
Figure 4.3.9 Function of Preferential Attachment	
Figure 5.3.1 Performance of Inference attacks before PPA	
Figure 5.3.2 Performance of Inference attacks after PPA	
Figure 5.3.3 F-score Vs local classifiers after using PPA	
Figure 5.3.4 Performance of Inference attacks before Sd-kp	
Figure 5.3.5 Performance of Inference attacks after Sd-kp	
Figure 5.3.6 F-score Vs local classifiers after using S-kdp	
Figure 5.3.7 Masked Attribute - δ (100 nodes)	

Figure 5.3.8 Uniqueness Score - δ (100 nodes)	
Figure 5.3.9 Commonness Score - δ (100 nodes)	
Figure 5.3.10 Masked Relation - δ (100 nodes)	
Figure 5.3.11 Jaccard Score - δ (100 nodes)	
Figure 5.3.12 Adamic/Adar Score - δ (100 nodes)	
Figure 5.3.13 Masked Attribute - δ (500 nodes)	
Figure 5.3.14 Uniqueness Score - δ (500 nodes)	
Figure 5.3.15 Commonness Score - δ (500 nodes)	
Figure 5.3.16 Masked Relation - δ (500 nodes)	
Figure 5.3.17 Jaccard Score - δ (500 nodes)	
Figure 5.3.18 Adamic/Adar Score - δ (500 nodes)	

Chapter 1
INTRODUCTION

We describe social network sites as web-based services that allow users to create a public or semi-public profile within a constrained system, articulate a list of other users with whom they share a connection, and browse and navigate their list of connections as well as those made by others. The nature and terminology of these linkages may differ from one site to site.

Social networking sites are defined as "A website that provides a virtual community for those interested in a certain subject or just to 'hang out' together,".

Online social networks refer to online social groups where individuals mingle or exchange information and opinions, such as blogs, social networking web sites, or even virtual worlds.

1.1 History of OSN:

The first recognisable social networking site was founded in 1997. SixDegrees.com allowed users to establish profiles, list their Friends, and browse the Friends lists starting in 1998. Where, each of these characteristics existed in some manner prior to SixDegrees.

SixDegrees advertised itself as a platform for connecting with others and sending messages. While SixDegrees gained millions of users, it was unable to sustain itself and was shut down in 2000. It had a drawback where early adopters complained that there was little to do after accepting Friend requests, and that most users didn't want to meet strangers.

Several community tools began allowing various combinations of profiles and publicly stated Friends between 1997 and 2001. Users could construct personal, professional, and dating accounts on AsianAvenue, BlackPlanet, and MiGente, and Friends may be identified on their personal profiles without requiring approval.

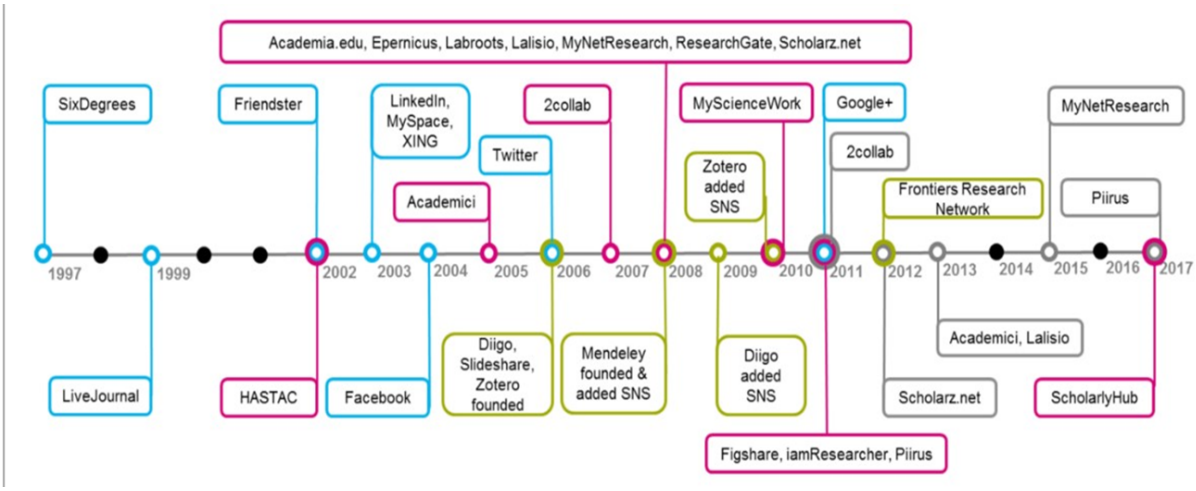


Figure 1.1.1 Timeline of major OSN

1.2 Privacy Security issues in OSN

We divide online social network privacy and security concerns into the following groups:

1. **Information leakages and content linkages**: these concerns are related to information disclosure dangers. A number of entities are identified as being associated with user information, content leakage, and linkage.

    A. **Leakages into other users**: Interacting with other users can put users in danger, especially if some of them are strangers or not a close friend. Furthermore, some of these individuals may not be human (for example, social robots), or they may be crowdsourcing workers stalking and interacting with users for malicious intentions.

    B. **Leakages into social applications**: Users can interact with numerous third-party social applications linked to their profiles for better functionality. The OSN provides application developers with an interface to access user information in order to facilitate interaction between OSN users and these other applications. Unfortunately, OSNs put users at risk by giving these apps more information than they need.

C. **Leakages into the OSN**: The OSN services support users' interactions with other users and social apps in exchange for full control over the information they publish on the OSN. While this exchange is expressly specified in Terms of Service contracts the user must agree with, few users understand the scope of the exchange and most users do not have a genuine choice if they do not agree with it. As a result, the OSN's usage of users' personal data is perceived as a breach of trust, and several methods have been offered to hide personal data from the service that holds it.

D. **Aggregators' linkages**: Professional data aggregators use the OSN APIs or scrape publicly available profile pages to build databases from user-profiles and social links. Insurance businesses, background-check agencies, credit-rating agencies, and others buy databases from professional data aggregators. Users' data from numerous sites and domains is gathered and linked together to improve profiling accuracy. This profiling could evolve to "public surveillance," in which an overly interested organization (such as the government) monitors people in public using a range of media.

2. **OSN Assaults**: these attacks target the service provider directly, posing a threat to its main business.

A. **Sybil Attacks**: Sybil attacks are defined as users establishing several identities in order to affect the outcome of a service. Sybil attacks, which are not unique to OSNs, have been used to influence the outcome of electronic voting, artificially raise the popularity of some media, and alter social search results. However, Sybil assaults have made OSNs vulnerable: Sybil users are illegitimately increasing their influence and authority in OSNs by controlling several accounts.

B. **Attacks from compromised accounts**: Compromised accounts are valid user accounts that have been compromised by attackers . They were created and used by their rightful owners. These accounts, unlike Sybil accounts, already

have established social connections and a history of normal social network usage.

C. **Social spam and malware**: Social spam is content or profiles that "legitimate" users of an OSN do not want to receive. Spam contributes to phishing assaults, unwanted commercial communications, and website promotion, undermining resource sharing and participation among users. Due to the established trust ties among online friends, social spam spreads quickly via OSNs, motivating a user to read messages or even click on links provided by her peers. Malware refers to programmes that obtain access to a computer, disrupt its operation, acquire sensitive information, or damage it without the owner's knowledge. ONSs are being used to spread malware, with social spam being a common tactic.

D. **Distributed Denial-of-service attacks (DDoS)**: Distributed Denial-of-Service (DDoS) attacks (DDoS). DDoS assaults are popular types of attacks in which a service is bombarded with a high number of relatively harmless service requests that overload the service and prevent access. OSNs, like many other popular services, are exposed to coordinated, distributed attacks.

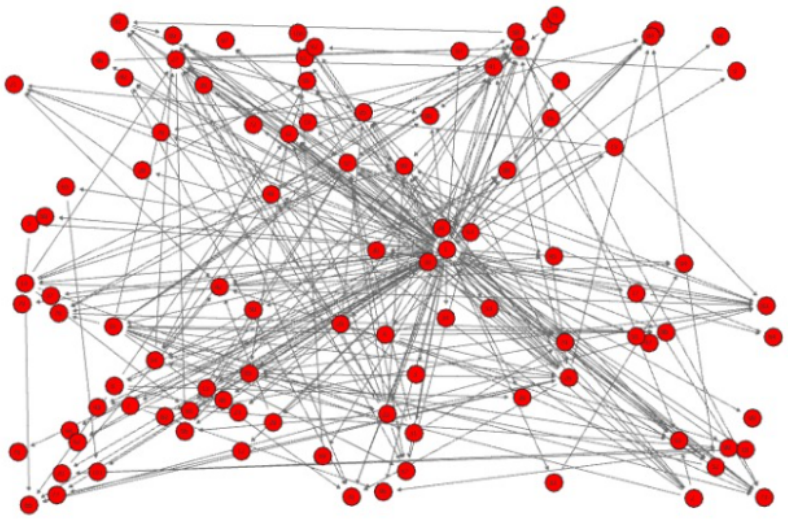


Fig.1.2.1 Social Network Graph

In our model we have designed an optimization algorithm that achieves a trade-off between self-disclosure utility and their privacy.

1.3 Objectives of the project

- To design a Social-attribute network model, we describe original social network data and attacker's knowledge.
- A self-disclosure rate is defined to measure the privacy loss of user secrets in a published network regardless of background knowledge of the attacker.
- To design a novel privacy-preserving social network data sharing model that maximizes the user self-disclosure utility with privacy guarantees in order to defend against the inference attack.
- To enable a flexible self-disclosure evaluation in order to satisfy different user demands and scenarios that takes different user concerns into account and enables an optimization problem is formulated.
- To propose social network data sharing methods designed for self disclosure utility while preserving privacy.
- To design an Privacy-Preserving Algorithm(PPA) algorithm which helps to achieve high utility by allowing users to share their data with utmost privacy.
- To design a Social relation based Multidimensional Knapsack Problem disclosure algorithm(Sd-kp) that deals with social relation disclosure problems with low computational complexity.

1.4 Challenges in online social networks

To achieve the goal, there are three main challenges.

- **Privacy Concerns**: Different users will have different privacy concerns. A certain profile attribute may be a secret to one person while it may not be sensitive to another.
- **Attacker's Ability:** Usually background knowledge of an attacker is unknown. So we have to design a general privacy protection model by assuming the attacker knows almost all the details about the target user.

- **Privacy and Utility Evaluation**: We need to quantify, evaluate and make a trade-off between the extent of self-disclosure and privacy leakage in order to establish the privacy protection model.

1.5 Scope of the project

- To design a novel privacy-preserving social network data sharing model that maximizes the user self-disclosure utility with privacy guarantees in order to defend against the inference attack.
- To design privacy models that preserve the content and the context of the information being exchanged in a network or uploaded to the server and other third-party applications.
- To design frameworks that provide user privacy-preservation solutions that preserve both other users(Node) and link information connected to the user.
- To develop techniques for various attacks such as identity-based, location-based,eavesdropping manipulation based and device based attacks.

1.6 Motivation

- It is critical to ensure that users enjoy the benefits of the services offered on Online Social Network and free from the worries about privacy leakage.
- It is very essential to defend sensitive information of individuals on OSNs that are being leveraged/disclosed based on the information extracted from the published dataset and background knowledge of the attackers.
- Balance trade-off between excessive concealment of sensitive information and the self-disclosure utility of OSN by user.
- Releasing the data collected from social networks would directly compromise users' privacy. It is an indispensable task for network data publishers to preserve data privacy.

Chapter - 2
LITERATURE SURVEY

2.1 Introduction

Literature survey defined as to identify the appropriate methodology, design of the study, methods of measuring concepts and techniques of analysis.

- **Assessment of the current state of research on a topic**: Once a researcher has determined an area to work with for a research project, a search of relevant information sources will help determine what is already known about the topic and how extensively the topic has already been researched.
- **Identification of the experts on a particular topic:** One of the additional benefits derived from doing the literature review is that it will quickly reveal which researchers have written the most on a particular topic and are, therefore, probably the experts on the topic.
- **Identification of key questions about a topic that need further research**: In many cases a researcher may discover new angles that need further exploration by reviewing what has already been written on a topic.
- **Determination of methodologies used in past studies of the same or similar topics**: It is often useful to review the types of studies that previous researchers have launched as a means of determining what approaches might be of most benefit in further developing a topic.

2.2 Summary of Papers

Authors in [1] have proposed a local recording-driven mechanism to preserve privacy in social networks. The proposed methodology adopts differential privacy to construct the framework. The proposed framework protects user privacy. However, the proposed framework does not guarantee reliable user privacy

The various identity anonymization techniques are used to preserve the user's data privacy. However, the identity of a user is revealed using available public information [2]. Author in [2] designed an anonymized dataset that satisfies l-diversity anonymity. To achieve l-diversity the author designed MaxSub, MinSub graph manipulation algorithms, MaxSub algorithm deletes only edges which disclose more privacy while MinSub performs insertion of edges or vertices to achieve l-diversity anonymity.

The anonymisation models used for protecting the private information of users in social networks will result in loss of information. So in [3] the author has used centrality measures to identify the importance of nodes in the social network and then anonymization by preserving important nodes. In [3] two different models of anonymization methods namely the k-degree anonymity achieved through the Fast K-Degree Anonymization algorithm (i.e., graph-modification anonymization approach) and the k-anonymity for social networks model achieved through the Sangreea algorithm (i.e., clustering-based anonymization approach) are proposed.

The location privacy of users is disclosed in many mobile social networks. In [4] the author proposes a radius-constrained dummy trajectory algorithm for privacy preservation scheme in MSN (Mobile Social Networks). The proposed scheme generates a dummy location set for the user's real location by constraining the radius where a user can send the LBS(Location based service) request.

Though there are various algorithms for privacy preservation which cannot be directly applied to a social network as there are structural properties along with labels for nodes. In [5] a method named GASNA (Greedy Algorithm for Social Network Anonymization) with three phases namely the clustering phase, adjustment phase, and anonymization phase is proposed. The clustering phase involves the process of gathering the nodes with a similar structure to form clusters. The adjustment phase involves the process of moving the nodes from a cluster that has less than k-nodes into a cluster with similar properties. Anonymisation phase involves either addition of edges by adding fake nodes, fake edges or deleting edges of user's behavior and the relationship between the users.

Many popular OSNs use centralized architecture where a single service provider develops and deploys the OSN system. Though it eases the job of updating, an extension of the network, and manipulating architecture. However it enables attackers to build social network graphs. In [6] the author proposes three classes of servers namely the first class server, second class server and third class server. First class server includes Diaspora and OneSocialWeb which uses federated architecture for independent servers, second class aims to protect data from storage providers using end to end encryption and third class uses Distributed Hash Table for server architecture.

Even though we anonymize the user's identity it's still possible for an attacker to find the user from the published anonymized records of a user on a social network. In [7] a k-couplet anonymity method is introduced to protect privacy under attribute couplet attacks. In this method if the dataset contains at least k-1 couplets having the same attributes then the dataset is said to satisfy k-couplet anonymity. In order to promote k-couplet anonymity the author has proposed 3 algorithms namely, Attribute Generalization (AG) , Attribute Cluster Anonymisation (ACA) , and Approximate Multiple-Attribute Generalization (AMAG).

The three potential privacy leakage problems such as Edge weight disclosure, Link Disclosure and Identity disclosure are solved in [8]. A greedy algorithm by name MinSwap which uses weight unlinkability knowledge is designed in order to protect edge weight disclosure problems. Further, delta-MinsawapX is an improvised version of MinSwap which solves all three issues: identity disclosure,link disclosure and edge weight disclosure problem.

In order to use various data running and ML techniques used to process the huge data generated by the OSN, the social network graph should be converted into a low-dimensional vector which is prone to privacy leakages. In [9] a model named LPPG (Link-Privacy Preserving Graph) is proposed in order to preserve privacy along with achieving privacy utility tradeoff between utility and privacy.

There are various techniques based on graph generation and differential privacy to protect users' identity in online social networks. However these techniques do not provide optimal data utility. In [10] the author proposes an approach based on differential privacy and field

theory which involves 2 steps. The approach includes two steps for publishing a social network. In step-1 the degrees of the nodes are first perturbed with the differential privacy by adding noise that follows a Laplacian distribution. In step-2 the edges of the social network are synthesized with field theory. A field theory model for social networks is proposed by simulating the law of gravity in physics and establishing the correspondence of the gravitational field in physics to the field theory model. When an edge is formed, the starting node is preferentially chosen with high probability from the nodes with high degrees, and then the ending node is selected with high probability when the interaction force between the starting node and the ending node is large. This approach can preserve more real social ties compared with previous approaches and will not incur a loss of structure features over the datasets, such as the degree distribution and clustering coefficients. However this method will preserve only the structure information of social networks. But in the case of real social networks that contain a substantial amount of attribute information, this model fails to maintain the topological characteristics and the correlations between attributes and edges.

Due to the large volume of data and high sensitivity it is challenging for privacy protection schemes to allocate a reasonable amount of noise, while preserving the desirable data and executing data utility services efficiently. In [11] PBCN (Privacy Preserving Approach Based on Clustering and Noise), a privacy protection approach based on clustering. This proposal is composed of five algorithms including random disturbance based on clustering, graph reconstruction after disturbing degree sequence and noise nodes generation, etc. Furthermore, a privacy measure algorithm based on adjacency degree is put forward in order to objectively evaluate the privacy-preserving strength of various schemes against graph structure and degree attacks. Simulation experiments are conducted to achieve performance comparisons between PBCN, Spctr Add/Del, Spctr Switch, DER and HPDP. The experimental results show that PBCN realizes more satisfactory data availability and execution efficiency. Finally, parameters utility analysis demonstrates PBCN can achieve a “trade-off” between data availability and privacy protection level. However PBCN is more complex if we try to reduce complexity then there is a chance of losing data availability.

The traditional Deep Packet Inspection (DPI) Mechanisms like NIDS and NIPS, due to granularity limitation and poor performance cannot efficiently adapt to privacy preservation and privacy detection in OSN (Online Social Network). In [12] A privacy preserving system based on domain gateway called Shutter Roller is used. Existence of the social features such as user generated contents and user behaviors, that causes privacy leakage is examined by Shutter Roller by the detection of OSN traffic through the gateway.

In case of weighted neighborhood attack, the attacker is considered to have information on the target's 1-neighborhood graph as well as degrees and edge weights. Using this data, an attacker can find the identity of a target given that any node’s 1-neighborhood is isomorphic with (k-1) other nodes. In [13] the author introduces a heuristic indistinguishable group anonymization (HIGA) scheme to build an anonymized social network which includes four steps namely Node grouping, Approximate Matching Test, Group Anonymization and randomization.

In cyber physical social networks (CPSN), an enormous amount of data is frequently shared between users. In order to provide privacy in CPSN a privacy protection scheme based on differential privacy is used which introduces noises into the social network. However this creates an unexpected relationship between noises and social actors that eases the process of identifying secrets. The two major attacks on customisable privacy protection are background attacks and collusion attacks. In [14] the author aims to offer customisable protection to every user and ensures users are attack resistant using a model called customizable reliable differential privacy model (CRDP).

In order to provide quality services for users Various Data mining applications use huge crowd sourced data of mobile devices available on social networks. Though users avail quality service , exposing these data to the public leads to privacy leakage of mobile users. In [15] the author introduces a scheme that generates groups from regions with minima statistics based on similarity of data change and further in order to reduce perturbation error , laplace noise is added to group instead of region. This scheme is called REal-time Spatiotemporal Crowd-soUrcEd Data Publishing with Differential Privacy.

Chapter -3

SYSTEM DESIGN

3.1 Problem Statement

The major problem in online social networks is how to preserve user's privacy. Generally, online social networks provide a platform to publish the data in such a way that users' privacy is protected and allow the maximum utilization of the data (i.e, the published data is capable of predicting new important decisions from the ML model). A design greedy based method that preserves privacy of a data and allows data analyser or data analytics to utilize data at maximum level to perform knowledge discovery. For example, the data utility feature of instagram provides recommendations of ads, reels etc to the user. At the same time, the user's privacy should also be protected.

This problem can be solved in two categories, one is by considering characteristics of the social actor while the other one is by considering social links of the actor.

In the first category we assume that the characteristics of each social actor is independent i.e; the change of characteristics of one social actor will not affect the other social actor.

  - Here for every public attribute of the user there will be a utility value, self-privacy disclosure value and the threshold value.
  - The total utility of a social actor is calculated by the sum of utility values of his published attributes.
  - We will select the attributes of the user in such a way that it has maximum utility and its self privacy disclosure value will not exceed the privacy protection threshold.

- Each edge involves two social actors. Unlike the first category, which considers social actors independent of each other in the second category, publishing the connection of

- a social actor will also affect the other social actor involved in that particular connection.
    - Here we are considering all the protection constraints including all social actors and their secrets.
    - Here the attributes and social connections are selected in such a way that maximum utility is achieved while self privacy disclosure value will not exceed the privacy protection threshold.

3.1.2 Terminologies

- **Inference Attack :** Inference attack occurs when a user is able to infer from trivial information more robust information about a database without directly accessing it.
- **Social Network Graph**: A social network graph is a graph where the nodes represent people and the lines between nodes, called edges, represent social connections between them, such as friendship or working together on a project.
- **Background knowledge :** Some prior knowledge or auxiliary information about the target of attack. e.g. some public demographic information about a specific group.
- **Types of Attributes:**
    - **Categorical** : An attribute where the values correspond to discrete categories.
    - **Numerical** : An attribute where the values are continues or real values
- **Attack Graph** : The privacy attack graph seeks to access sensitive information concealed in social networks, such as identification, friendship, and other personal information.
- **Adversary** : A person or a group who aims to attack a social network graph using some background knowledge.
- **Utility:** Information that are useful to attackers. There are two types of utility
    - Attribute Utility: Utility by disclosure of attributes of a social actor.
    - Social Relation Utility: Utility by disclosure of relation of a social actor.
- **Self Privacy Disclosure:** Self-disclosure involves sharing personal information – such as your thoughts, dreams, fears, goals, preferences, and experiences. It's an important way to strengthen relationships and build trust.

- **Degree Centrality :** Degrees the number of edges a node has. Higher the degree more central the node is.
- **Submodular set function :** A function f on finite set Ω is submodular if it satisfies one of these conditions
    - For every X,Y ⊂ Ω such that X⊆Y and every x ∈ Ω\Y
    - we have that f(X ∪ $\{x\}$) - f(X) ≥ f(Y ∪ $\{x\}$) - f(Y)
    - For every X,Y ⊂ Ω
        - we have that f(X) + f(Y) ≥ f(X∪Y)+f(X∩Y)
    - For every X⊂Ω such that x_1,x_2 ∈ Ω\X
        - we have that f(X ∪ {x_1})+f(X ∪ {x_2}) ≥ f(X ∪ {x_1,x_2}) + f(X)
- **Monotone set function** : A submodular function f is monotone if for every T⊆S f(T) ≤ $f(S)$

3.2 Proposed Method

In our project, the privacy-preservation in online social network data sharing is formulated as knapsack problem. We propose two social network data disclosure methods to solve this problem. Following are the two methods :

1. Privacy preserving algorithm.
2. Social relation based multidimensional knapsack problem algorithm. (Sd-kp)

3.2.1 Preliminaries

Here is some preliminary information before formulating the privacy-preserving data sharing problem in OSN.

TABLE 1

Notation

Symbol	Definition
V_N	Vertex set of social actors
V_A	Vertex set of attributes
E_N	Edge set of social relations
E_A	Edge set of attribute links
N_u / N_a	Social actors connected with actor u / attribute a
A_u	Attribute set of actor u
S_u	S_u Secret attributes of actor u
P_u	P_u Public attributes of actor u
G_A	G_A Attack graph
G_p	G_p Published social network graph
$t_u(s)$	Indicating whether actor u has secret s
$\Phi(u,s,G_p)$	Self privacy disclosure of actor u's secret s in G_p
p_i	Value of the i^{th} attribute or social relation
ϵ , $\mathbb{C}$	Privacy budget (set)
δ , Δ	Relaxation variable (set)
θ , Θ	Privacy threshold (set) determined by ϵ and δ

Table 1 shows the basic notations commonly used in our work.

3.2.2 Project Flow Chart

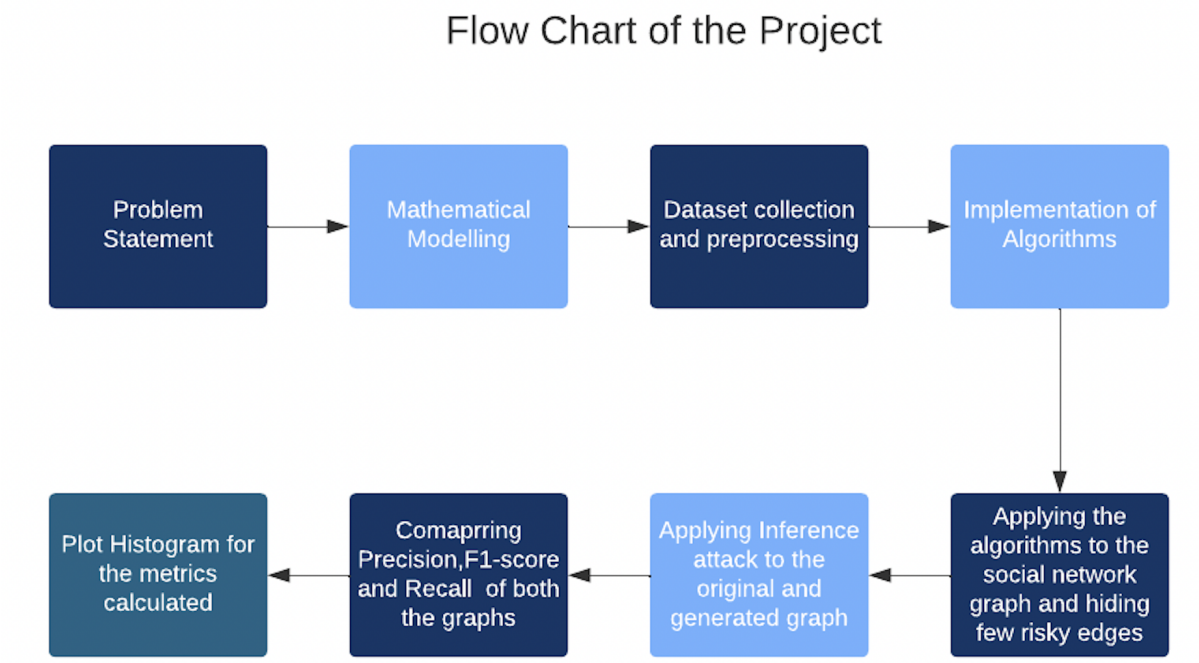


Figure 3.2.1 Flow Chart of the Project

Step-1: First the problem statement is formulated, by considering all the necessary requirements.

Step-2: After formulating the problem all the mathematical models or equations are designed.

Step-3: In this step the data is collected and pre-processed into a proper format so that the algorithm which we design can be applied.

Step-4: In this step the algorithms are designed

Step-5: In this step the designed algorithms are applied to the pre-processed data set and few risky edges from the social network are removed.

Step-6: In order to check the efficiency of our algorithm inference attack is done to original and processed social network and F-score, Precision is calculated.

Step-7: In this step we compare F-Score, Precision and Recall of original and processed graphs.

Step-8: The Histogram is plotted for the metrics calculated in the previous step.

3.2.3 Mathematical Model

- The **Uniqueness** of the attribute a, is given by the reciprocal of log of the number of social actors having that particular attribute.

$$p_u(a) = \frac{1}{log(|Na|)+1}$$

- The **Commness** of the attribute a and social actor u is given by the number of social actors who are friends of actor u and has the attribute a divided by the number of friends of u.

$$p_c(a,u) = \frac{|Nu \cap Na|}{|Nu|}$$

We can use either uniqueness or commonness score to find the utility of the attribute a.

- The **Jaccard Coefficient** of a relation associated with user u and v is given by the number of common friends of u and v divided by the friends in either u or v.

$$p_c(a,u) = \frac{|Nu \cap Nv|}{|Nu \cup Nv|}$$

- The **Adamic Score or Adar Score** is the measure of similarity between two actors involved in a social link. It is given by

$$p_A(e_{u,v}) = \sum_{k \in fu \cap fv} \frac{1}{log|Nk|}$$

We can use either Jaccard Score or Adamic Score to find the utility of the edge $e_{u,v}$.

- In any social network we measure the probability of identifying the private attributes based on the attacker's background knowledge as disclosure risk or self privacy disclosure.
- Given the background knowledge of a particular attacker the probability of disclosing a secret s of actor u in social network is given by:

$$Pr\{t_u(s)=1|G_{a,}G_p\}=Pr\{t_u^A(s)=1|g(A_u^P)$$

where ,

- ➢ s is secret and 1 indicates the user has secret s and 0 indicates user does not hold the secret s
- ➢ G_a is the background knowledge graph
- ➢ G_p is the social network graph for published data
- ➢ $g(A_u^P)$ maps the common attributes of published network graph over attack graphs.

- Different attackers may have different background knowledge so is the attack graphs. So we define a general form to find disclosure risk considering strong attacker having almost similar attack graph as published graph and is given by

$$\phi_a(u,s,G_p) \triangleq Pr\{t_u(s=1) \mid A_u \cap A_u^P)$$

where,

- ➢ A_u is the attribute set of actor u in original network graph
- ➢ A_u^p is the attribute set of actor u in published social network graph

- Similarly the probability of actor u having secret s based on social actors connected to u is given by

$$\phi_n(u,s,G_p) \triangleq Pr\{t_u(s=1) \mid N_u \cap N_u^P)$$

where,

- ➢ N_u is the set of actors connected to actor u in original graph
- ➢ N_u^p is the set of actors connected to actor u in published graph
- ➢ ϕ_n is the self privacy disclosure for actor u having secret s in G_p

- Self Privacy disclosure ranges between 0 and 1.

The published graph is considered privacy preserving iff the attackers knowledge about the user's secret before publishing the graph and after publishing the graph are almost same or the difference is minimal.

Therefore we define a threshold and we consider our published graph privacy preserving iff for every node the self privacy disclose measure does not exceed this threshold and the condition is given by

$$\phi(u,s,G_p) \leq \exp(\epsilon)\Pr\{tu(s=1)\} + \delta$$

where,

- ➢ ϵ and δ are the two criterion that manages the self privacy disclose.
- ➢ ϵ is the privacy budget that reflects how well the self-privacy disclosure rate matches the prior beliefs of attacker
- ➢ δ sets the limit on how much personal information can be revealed and its always less than $1-\exp(\epsilon)\Pr\{tu(s=1)\}$

- Every secret s is represented as a triplet (s,ϵ,δ) where ϵ and δ are the privacy parameters for the secret s.
- The union of all users privacy constraints over each and every secrets is given by aggregation of all the tuples i.e, super set $C=(S,\epsilon,\Delta)$.
- Considering all users privacy concerns , a published graph G_p is privacy preserving iff

$$\phi(u,S_u,G_p) \leq \Theta_u \text{ , for every actor u in social network nodes set } V_N$$

- ➢ Θ_u is a vector of privacy thresholds $\theta_{u,i}$ of i^{th} secret s of actor u and it is given by

$$\theta_{u,i} = \exp(\epsilon_{u,i})\ \Pr\{t_u(S_{u,i})=1\} + \delta_{u,i} \text{ ,}$$

where,

$i=1,2,3,..|S_u|$

$\epsilon_{u,i} \in \epsilon$

$\delta_{u,i} \in \Delta$

- The aim is to obtain a disclosed social network Gp using the gised edge set

  $T = E'_N \cup E'_A$ such that $E'_N \subset E_N$ and $E'_A \subset E_A$ which satisfies the privacy constraints specified in $C=(S,\epsilon,\Delta)$ while providing the highest utility.

- Highest utility is given by :

$$y = \max \{p(T) : \phi(u,S_u,G_p) \leq \Theta_u \text{ , for every actor u in } V_N \}$$

$$T \subseteq E_N \cap E_A$$

- The self privacy disclosure function p(T) is **Non Submodular**.This can be proved with an example as follows.Given a social network graph with
  - $V_N=\{N_1,N_2,N_3,N_4,N_5,N_6\}$
  - $V_A=\{S,a_1,a_2\}$
  - $E_A=\{(N_1,S),(N_1,a_1),(N_1,a_2),(N_2,S),(N_2,a_1),(N_3,S),(N_3,a_1),(N_3,a_2),(N_4,a_1),(N_4,a_2),(N_5,a_1),(N_6,a_2)\}$
  - $e=(N_1,a_2)$
  - $T_1= E_A \setminus \{ (N_1, S),(N_1,a_1),(N_1,a_2)\}$
  - $T_2= E_A \setminus \{ (N_1, S),(N_1,a_1)\}$
  - $T_1 \subseteq T_2$, such that

$$\mathbf{p(T_1 \cup \{e\}) - p(T_1) < p(T_2 \cup \{e\}) - p(T_2)}$$

  which clearly says the function p(T) does not satisfy submodularity.

- Similarly based on monotonic function definition , the self disclosure function p(T) is non-monotonic. This can be proved with an example as follows. Considering same

parameters as previous example with $e_1=(N_1,a_1)$, $e_2=(N_1,a_2)$ and $T=E_A\setminus \{ (N_1, S),(N_1,a_1),(N_1,a_2)\}$ such that,

$$p(T)<p(T \cup \{e_1\}) \text{ and } p(T \cup e_1) > p(T \cup \{e_1,e_2\})$$

which clearly dissatisfied the monotonicity conditions.

- Since self privacy disclosure function does not follow either Submodulairty or monotonicity, we can conclude that privacy disclosure problem for a online social network cannot be solved as a single problem directly instead it should be solved two separate problem considering attributes and social connections independently.
- The utility of user u having user profile with n attributes is given by

$$Utility = \sum_{i=1}^{|P_u|} p_i x_i$$

- The condition to be satisfied in order to obey all the necessary privacy constraints in attribute disclosure problem is

$$\Phi_A(u, s_j, \mathbf{x}) \leq \theta_j$$

$$\text{where } j=1, \ldots., |S_u|$$

$$\mathbf{x} = \{x_i \in \{0,1\}, i=1, \ldots, |E_N|\}$$

where each element x_i in vector **x** denotes whether or not to reveal the corresponding public attribute a_i with utility value p_i, which can be defined as the attribute's value for varied purposes.

- The relation disclosure problem in undirected social networks cannot be handled for each social actor because the removal of one undirected edge will affect two social actors. As a result, we should address the relation disclosure problem as a whole, as outlined below.

$$max(x)= \sum_{i=1}^{|E_n|} p_i x_i$$

$$\text{s.t} \quad \Phi_N(u_k, s_{kj}, x) \leq \theta_{k,j},$$

$$j=1, \ldots., |S_{u_k}|, k = 1, \ldots., |V_N|,$$

$$x_i \in \{0,1\}, i=1, \ldots., |E_N|$$

where $\theta_{k,j} = \exp(\epsilon) \Pr\{t_{uk}(s_{k,j}) = 1\} + \delta_{k,j}$ and all security limitations are examined collectively, including all social actors and their secrets.

3.3 Privacy Preserving Algorithm (PPA)

- In this algorithm we are co-relating the social network privacy preserving problem to the Knapsack problem and then solving it.
- We are considering every edge of the Social network graph as an item in a knapsack.
- The total contribution of the selected items to social actor n's secret as the self privacy disclosure rate is considered as the weight of the edge/item.
- The utility of the selected items/nodes is considered as profit gained.
- The aim is to find the maximum utility possible with the minimum self privacy disclosure rate. i,e

$$max(x)= \sum_{i=1}^{|E_n|} p_i x_i$$

$$\text{s.t} \quad \Phi_N(u_k, s_{kj}, x) \leq \theta_{k,j},$$

$$j=1, \ldots., |S_{u_k}|, k = 1, \ldots., |V_N|,$$

$$x_i \in \{0,1\}, i=1, \ldots., |E_N|$$

where $\theta_{k,j} = \exp(\epsilon) \Pr\{t_{uk}(s_{k,j}) = 1\} + \delta_{k,j}$

3.3.1 Algorithm steps of PPA

Step-1: Initialize

- $\vec{S}$= {S_1, S_2, S_3…..S_n} (Secrets of a social actor)
- $\vec{N}$ = {N_1, N_2, N_3…..N_n} (Neighbor of a social actor)
- θ = {θ_1,θ_2,θ_3…..θ_n} (Secret threshold list)

Step-2: Calculate Privacy using Jaccard Coefficient $\vec{p}$ = {p_1,p_2,p_3…..p_n}

$$p_j(e_{u,v}) = \frac{|Nu \cap Nv|}{|Nu \cup Nv|}$$

For every social relation existing between social actor u and social actor v :

P(Social actor u, Social actor v) = $\frac{Total\ Common\ neighbors\ of\ u\ and\ v}{Total\ neighbors\ of\ u\ or\ v}$

Step-3: Initialize

- C = V_n (Vertex set of social actors)
- Sel = Ø
- V_{max} = 0
- $\vec{l}$ = [1, 2, 3,....,n]

Do Step-4 to Step-10 until $\vec{l}$ not equal to Ø

Step-4 : Initialize

- ρ_{max}= -1
- S = -1
- W_{sel} = Ø

Do Step- 5 to Step-8 for every i in $\vec{l}$

Step-5: J = [1, 2, 3,....,n]

Do Step-6 for every j in J

Step-6: Calculate w_j using the below formula

$$W_j \leftarrow \frac{|C \cap N_i \cap S_j|}{|C \cap N_i|}$$

Step 7 : Calculate ρ using below formula

$$\rho \leftarrow \frac{P_i}{\sum_{k=1}^{m} w_j / \theta_j}$$

Do Step- 8 if $\rho > \rho_{max}$

Step 8 :

- $\rho_{max} = \rho$
- s=i
- $W_{sel} = W_j$

Do Step 9 if $w_j <= \theta_j$ for every j in {1,2,3,...m}

Step-9:

- Sel = Sel or {s}
- C = C and N_s
- $p_{max} = p_{max} + p_s$

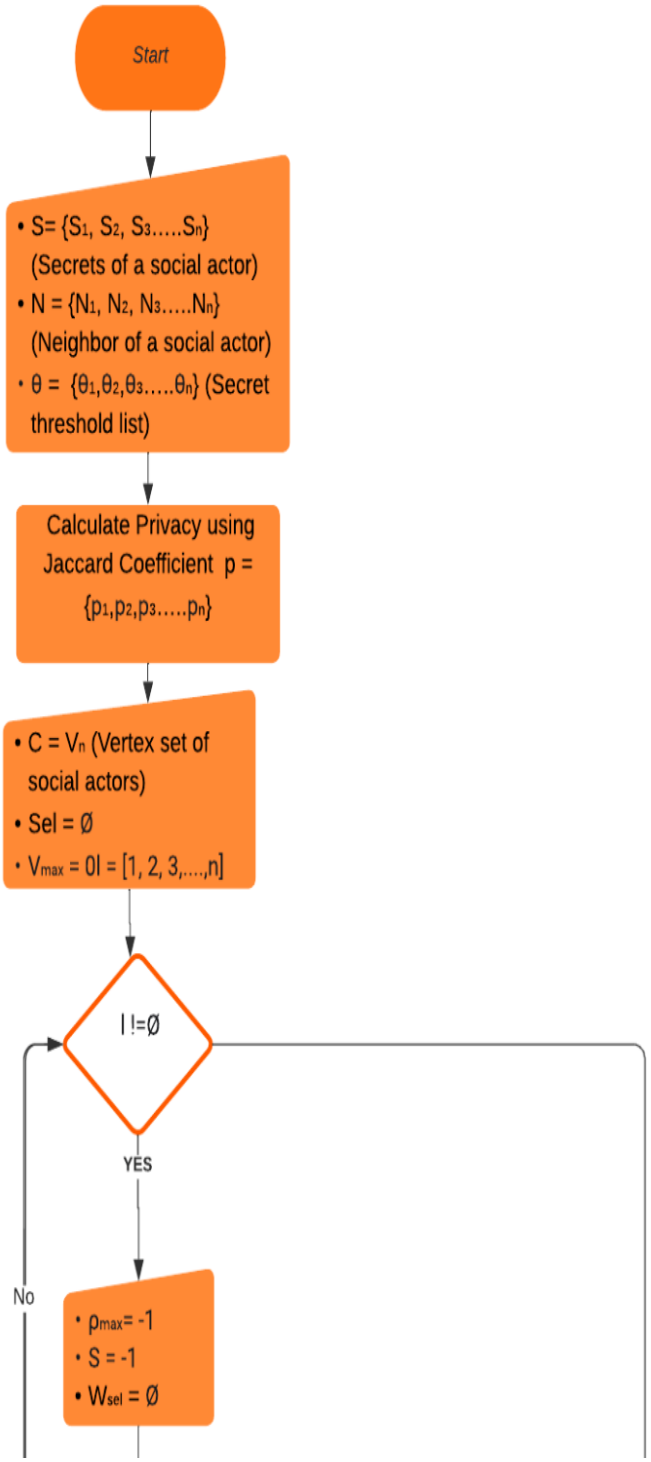
Step-10: remove s from $\vec{l}$

Step-11 : Return p_{max} , Sel

3.3.2 Explanation of Privacy Preserving Algorithm

- We have used the greedy approach to solve this knapsack edge masking problem.
- The main idea is to select the edges of the nodes which have high utility and with minimal leakage of information about private attributes while satisfying all the privacy requirements.
- Initially the utility of all the edges is pre-computed.
- In every iteration weights of all secrets is computed and for every edge we will find ρ, which is the ratio of edge utility to the total sum of ratios of weights of secrets to its threshold value. The edge with maximum ρ and satisfying all privacy constraints is selected.

3.3.3 Flowchart of PPA



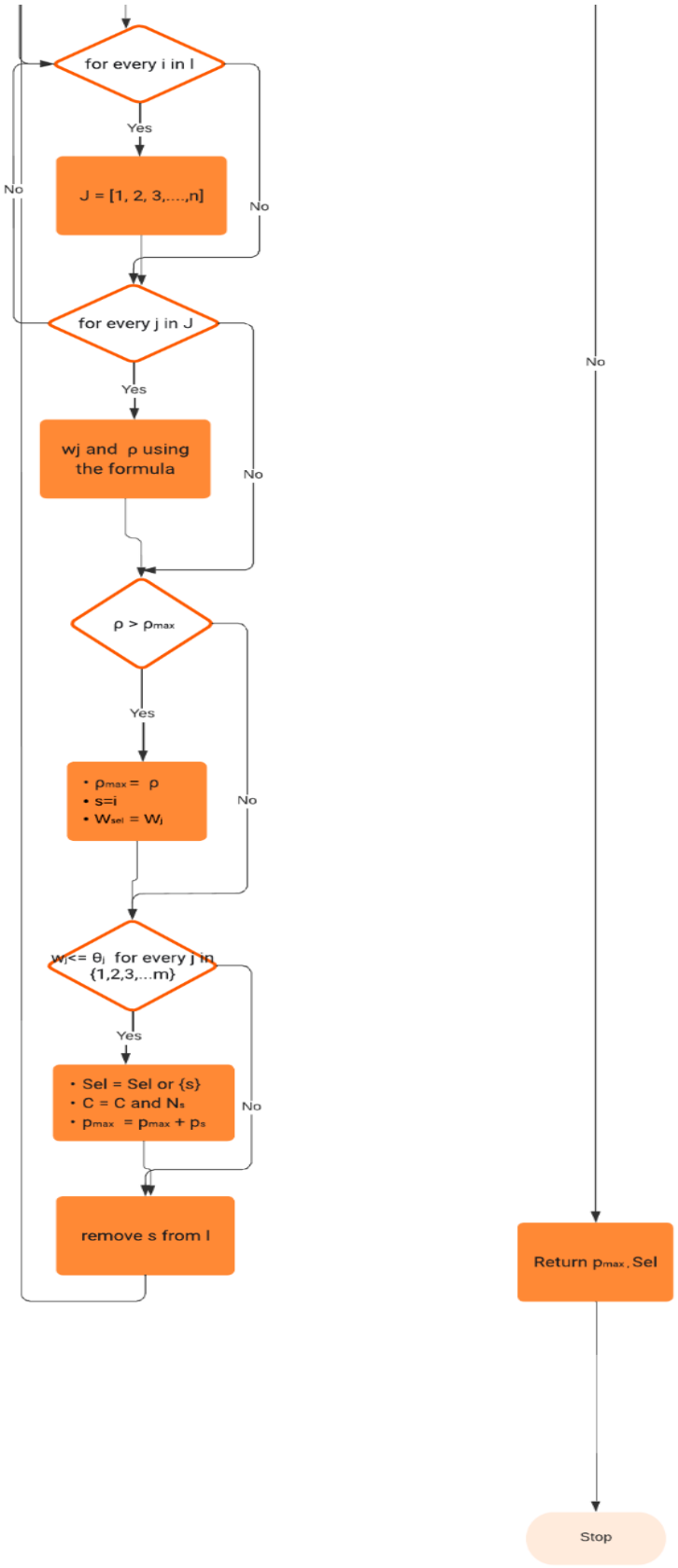


Figure 3.3.1 Flowchart of PPA

3.4 Social relation based d-kp (S-dkp)

- The fundamental cause of the high complexity is because self-privacy disclosure takes into account all of the relationships between qualities and social relationships. On the one hand, omitting the correlations and treating all public information as (conditionally) independent may reduce privacy protections because two public attributes/social relations may give more information than the sum of the information presented individually. However, it will also simplify the problem by determining the weight of each public attribute or social relationship.
- The aim is to find the maximum utility possible with the minimum self privacy disclosure rate. i,e

$$max(x) = \sum_{i=1}^{|E_n|} p_i x_i$$

$$\text{s.t} \quad \Phi_N(u_k, s_{kj}, x) \leq \theta_{k,j},$$

$$j = 1, \ldots., |S_{u_k}|, \; k = 1, \ldots., |V_N|,$$

$$x_i \in \{0,1\}, \; i = 1, \ldots., |E_N|$$

$$\text{where } \theta_{k,j} = \exp(\epsilon) \Pr\{t_{uk}(s_{k,j}) = 1\} + \delta_{k,j}$$

3.4.1 Algorithm steps of s-dkp

Step-1: Initialize

- secret neighborhood setlist $\vec{S} = \{S_1, S_2, S_3 \ldots .. S_n\}$,
- item neighborhood setlist $\vec{N} = \{N_1, N_2, N_3 \ldots .. N_n\}$, and
- secret threshold list $\theta = \{\theta_1, \theta_2, \theta_3 \ldots .. \theta_n\}$

Step-2: Calculate Privacy using Jaccard Coefficient $\vec{p} = \{p_1, p_2, p_3 \ldots .. p_n\}$

$$p_j(e_{u,v}) = \frac{|Nu \cap Nv|}{|Nu \cup Nv|}$$

For every social relation existing between social actor u and social actor v :

P(Social actor u, Social actor v) = $\frac{Total\ Common\ neighbors\ of\ u\ and\ v}{Total\ neighbors\ of\ u\ or\ v}$

Step-3: Initialize

- C = V_n (Vertex set of social actors)
- Sel = Ø
- p_{max} = 0
- $\vec{l}$ = [1, 2, 3,....,n]

Do Step-4 to Step-13 until $\vec{l}$ not equal to Ø

Step-4 : Initialize

- ρ_{max}= MIN_INT
- S = -1
- I_{sel} = Ø

Do Step- 5 to Step-13 for every i in $\vec{l}$

Step-5:

- J = N[i] (friends of i)

Do Step-6 to Step 8 for j in J

Step-6 : Calculate Information gain for node i and node j using below formula

- Ii = 0
- Ij =0
- Ii = log(1/(nCr(n-1, len(N[i]))))
- Ij = log(1/(nCr(n-1, len(N[N[i][j]]))))

Step-7 : Calculate ρ using below formula

- ρ = P[i][j]/(Ii+Ij)

Do step 8 if ρ> ρ_{max}

Step-8 :

- s = N[i][j]
- ρ = ρ_{max}
- Isel = Ij

Step-9 :

- flag = True
- J = θ[s]

Do step-10 for every j in J , if I_{sel} > log(θ[s][j])

Step-10 :

- flag=False

Do step 11 if flag=True

Step-11:

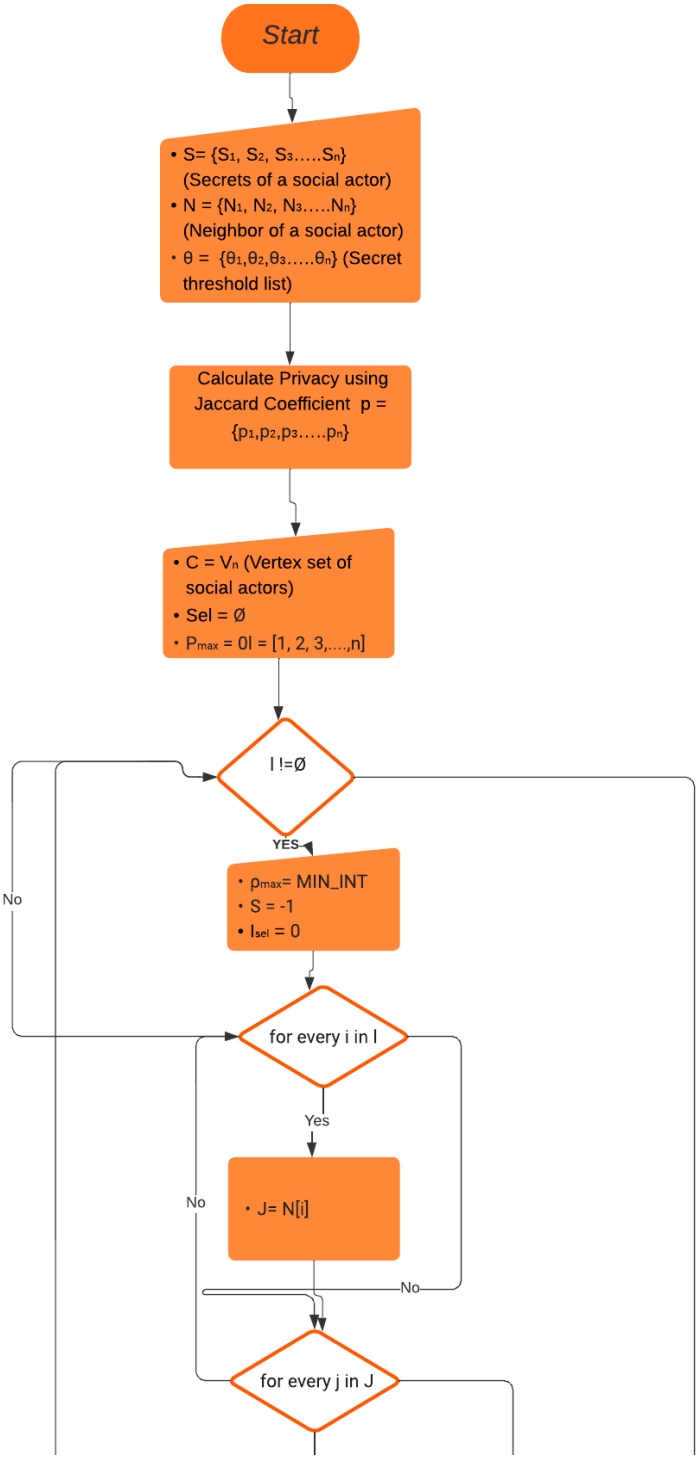
- Sel = Sel or {s}
- p_{max}= p_{max} + sum(p_s)

Step 13: return Sel, p_{max}

3.4.2 Explanation of S-dkp

- We have used the greedy approach to solve this multidimensional knapsack edge masking problem.
- The main idea is to select the edges of the nodes which have high utility and with minimal leakage of information about social relations while satisfying all the privacy requirements.
- Initially the utility of all the edges is pre-computed.
- In every iteration mutual information disclosed of all the edges is computed and for every edge we will find ρ, which is the ratio of edge utility to the total sum of information disclosure of both the nodes involved in the social relation. The edge with maximum ρ and satisfying all privacy constraints is selected.

3.4.3 Flowchart of sd-kp



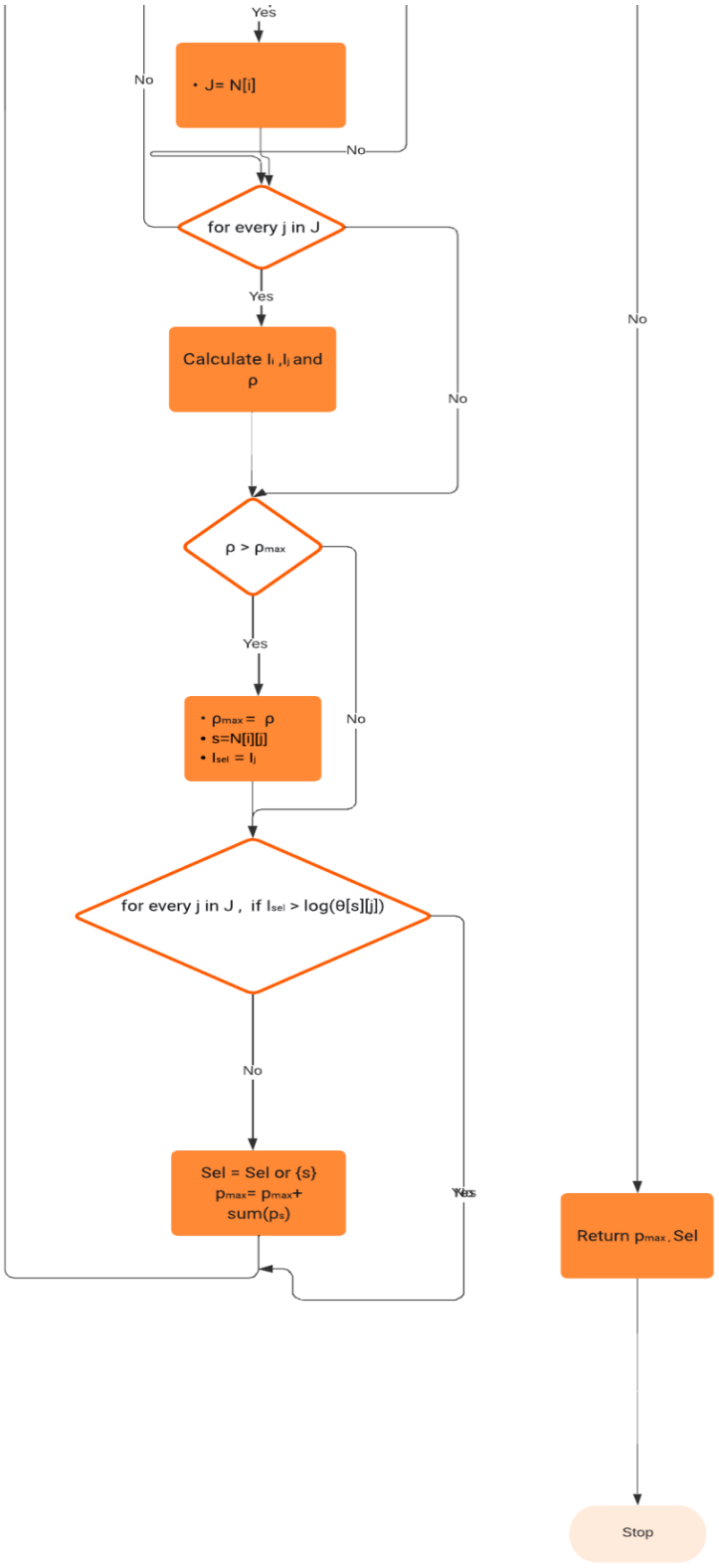


Figure 3.4.1 Flowchart of S-dkp

3.5 Inference attack techniques

3.5.1 Triadic Closure

Triadic Closure occurs when two vertices are connected to the identical third vertices.

$comm\_neighb(X, Y) = |N(X) \backslash cap\ N(Y)|,\ where\ N(X)\ is\ the\ set\ of\ all\ neighbors\ of\ X$

3.5.2 Jaccard

It is calculated by dividing the total number of neighbors by the number of common neighbors. It is defined as the size of the intersection divided by the size of the union of two finite sample sets, and it is used to quantify the similarity between two finite sample sets.

$Jaccard\ Coefficient(X, Y) = |N(X) \backslash cap\ N(Y)|/|N(X) \backslash cup\ N(Y)|$

3.5.3 Resource Allocation

Research Allocation Index outperforms a number of similarity-based approaches for predicting missing links in a complex network with less time complexity. It's a fraction of a resource that a node can send to another node via their shared neighbors.

$Research\ Allocation\ Index(X, Y) = \Sigma_{u \backslash epsilon\ N(X) \backslash cap\ N(Y)}\ 1/|N(u)|$

3.5.4 Adamic Adar Index

In 2003, this metric was established to anticipate missing links in a network based on the number of common links between two nodes. It is calculated as follows:

$Adamic\ Adar\ Index(X, Y) = \Sigma_{u \backslash epsilon\ N(X) \backslash cap\ N(Y)}\ 1/log(|N(u)|)$

3.5.5 Preferential Attachment

The more linked a node is, the more likely it is to obtain new linkages, which is referred to as preferential attachment. More neighbors are attracted to nodes with a higher degree.

$Preferential\ Attachment(X, Y) = |N(X)|.|N(Y)$

Chapter-4

IMPLEMENTATION

4.1 Dataset

We apply our algorithm on a social media dataset which has 500 social actors and 124440 social relations.

Attributes	Number of social actors
First Name	247
Last Name	259
Date of birth	244
Home-Town	258
School	248
Degree	254
Education	260
Graduation	225
Gender	248
Languages	249
Location	244
Zip Code	252
Phone Number	232
Company	254

Work Location	257
School	248
Degree	254
Work Position	259
Marital Status	259

4.2 System defined functions

List Functions:

Lists are one of 4 built-in data types in Python used to store collections of data,Lists are used to store multiple items in a single variable.Lists are created using square brackets.

Example: Create a List: thislist = ["apple", "banana", "cherry"].

Few functions of lists are

- **Range:** The range() function returns a sequence of numbers, starting from 0 by default, and increments by 1 (by default), and stops before a specified number.
- **Append:** A call to .append() will place new items in the available space.
- **Remove:** The remove() method removes the first matching element (which is passed as an argument) from the list.

Files :

Files are named locations on disk to store related information. They are used to permanently store data in a non-volatile memory (e.g. hard disk).

Since Random Access Memory (RAM) is volatile (which loses its data when the computer is turned off), we use files for future use of the data by permanently storing them.

When we want to read from or write to a file, we need to open it first. When we are done, it needs to be closed so that the resources that are tied with the file are freed.

Hence, in Python, a file operation takes place in the following order:

- Open a file
- Read or write (perform operation)
- Close the file

Open : To open a file, use Python's built-in open() method. This function returns a file object, often known as a handle, which can be used to read or change a file.

When we open a file, we can choose the mode. We choose whether to read r, write w, or add a to the file in mode.

Write : In order to write into a file in Python, we need to open it in write w or append a mode.

Read: To read a file in Python, we must open the file in reading r mode.

Close : Closing a file will free up the resources that were tied with the file. It is done using the close() method available in Python.

NetworkX Functions:

NetworkX is a Python-based software package for creating, manipulating, and studying complex networks' structure, dynamics, and function. It's used to investigate big, complicated networks that are represented as graphs with nodes and edges. We can load and store complex networks with networkx. We may create a variety of random and traditional networks, study their structure, establish network models, create new network algorithms, and draw them.

nx.Graph(edgeList) : A Graph is a collection of nodes (vertices) along with identified pairs of nodes (called edges, links, etc). The function nx.Graph() generates a graph for given edge list.

The following Networkx function are used to predict edges in a network:

- Triadic closure
- Jaccard Coefficient
- Resource Allocation Index
- Adamic Adar Index
- Preferential Attachment

Triadic closure

If two vertices are connected to the same third vertices, the tendency for them to share a connection is Triadic Closure

comm_neighb(X, Y) = |N(X) ∩N(Y)|, where N(X) is the set of all neighbors of X.

Jaccard Coefficient:

It's calculated by dividing the total number of neighbors by the number of common neighbors. It is defined as the size of the intersection divided by the size of the union of two finite sample sets and is used to determine how similar two finite sample sets are.

The Networkx built-in function jaccard coefficient always returns a list of three tuples (u, v, p), where u, v is the new edge that will be added next with a probability measure of p. (p is the Jaccard Coefficient of nodes u and v).

nx.jaccard_coefficient(G)

Resource Allocation Index:

The Research Allocation Index outperforms a number of similarity-based approaches for predicting missing links in a complicated network. It is a fraction of a resource that a node can send to another via shared neighbors.

nx.resource_allocation_index(G)

Adamic Adar Index

This measure was introduced in 2003 to predict missing links in a Network, according to the amount of shared links between two nodes.

The networkx package offers an in-built function of adamic_adar_index which offers a list of 3 tuples (u, v, p) where u, v is the new edge and p is the adamic adar index of the new edge u, v.

nx.adamic_adar_index(G)

Preferential Attachment

Preferential attachment means that the more connected a node is, the more likely it is to receive new links . Nodes with higher degrees get more neighbors.The networkx package offers an in-built function of preferential_attachment which offers a list of 3 tuples (u, v, p) where u, v is the new edge and p is the preferential attachment score of the new edge u, v.

nx.preferential_attachment(G)

Metrics

Accuracy score: This function calculates subset accuracy in data set: the set of labels predicted for a sample must exactly match the corresponding set of labels in y true.

sklearn.metrics.accuracy_score (y_true, y_pred)

Precision score:

The precision is the ratio tp / (tp + fp) where tp is the number of true positives and fp the number of false positives. The precision is intuitively the ability of the classifier not to label as positive a sample that is negative.

The best value is 1 and the worst value is 0.

sklearn.metric.precision_score(y_true, y_pred)

Recall:

The recall is the ratio tp / (tp + fn) where tp is the number of true positives and fn the number of false negatives. The recall is intuitively the ability of the classifier to find all the positive samples.

The best value is 1 and the worst value is 0.

sklearn.metrics.recall_score(*y_true*, *y_pred)*

F1 score:

The F1 score can be thought of as a harmonic mean of precision and recall, with the best value being 1 and the poorest being 0. Precision and recall both make an equal proportion to the F1 score. The F1 score is calculated as follows:

F1 = 2 * (precision * recall) / (precision + recall)

sklearn.metrics.f1_score(y_true, y_pred)

4.3 User defined function

genAdjecencyMatrix: This method takes filename as input reads the file which has list of social actors with corresponding friends list and generates adjacency matrix

```
def genAdjecencyMatrix(filename):
        file1= open(filename)
        n = 500
        m = 500
        a =[[0 for i in range(n)] for j in range(n)]

        edges =[]
        for i in range(m):
                edges.append(file1.readline().split(" "))
                edges[i].remove('\n')
                edges[i] = list(map(int, edges[i]))

        for i in range(len(edges)):
                a[edges[i][0]][edges[i][1]]=1
                a[edges[i][1]][edges[i][0]]=1
        return a

file1 = open("fbsecrets.txt", "w")

for i in range(len(a)):
        for j in range(len(a[i])):
                file1.write(str(a[i][j])+" ")
        file1.write("\n")
file1.close()
```

Figure 4.3.1 Function to get Adjacency matrix of the OSN

- **calculatePrivacy(edges,n,m):** This method calculates the privacy disclosure of each edge in a list of n edges.

```
def calculatePrivacy(edges, n):
    privacy  =[[0 for i in range(n)] for j in range(n)]

    for i in range(n):
        for j in range(n):
                x = edges[i][j]
                y=0
                z=0
                if x==1:
                    for k in range(n):
                            y += edges[i][k] and edges[j][k]
                            z += edges[i][k] or edges[j][k]
                if(z):
                    privacy.append((n/z))
                else:
                    privacy.append(0)
    return privacy
```

Figure 4.3.2 Function to calculate Privacy

- **PPA(N,S,P,Q):** This method returns the edgelist of privacy preserving graph obtained by applying the ppa algorithm. pmax is the maximum utility achieved**.**

```
def ppa(N, P, Q, S):
        C = [i for i in range(n)]
        Sel =set()
        Pmax = 0
        l = [i for i in range(n)]
        while(len(l)):
                Rmax = -1
                s=-1
                Wsel={}
                w=[0 for i in range(m)]
                for i in l:
                        for j in range(m):
                                X = [val for val in C if(val in N[i])]
                                if(len(X)):
                                        sec=S[X[0]]
                                else:
                                        sec = []
                                for val  in X:
                                        sec=[v for v in sec if v in S[val]]
                                numerator=len(sec)
                                if len(X):
                                        w[j]=numerator/len(X)
                                else:
                                        w[j]=0
                        Sl = sum([w[j]/Q[i][j] for j in range(len(Q[i]))])
                        if Sl ==0:
                                R=0
                        else:
                                R=sum(P[i])/Sl
                        if R>Rmax:
                                s=i
                                Rmax=R
                                Wsel=w
                        flag=True
                        for j in range(len(Q[s])):
                                if Wsel[j]>Q[s][j]:
                                        flag=False
                                        break
                        if flag:
                                Sel.add(s)
                                C=[val for val in C if val in N[s]]
                                Pmax+=sum(P[s])
                        if(s in l):
                                l.remove(s)
        return Pmax, Sel
```

Figure 4.3.3 Function of PPA Algorithm

- **sd-kp()**: This method returns the edgelist of privacy preserving graph obtained by applying the sd-kp algorithm. pmax is the maximum utility achieved**.**

```
def sdkp(N, P, Q, S):
    Sel = set()
    Pmax = 0
    l = [i for i in range(n)]
    for xx in range(n):
        Rmax = -1
        s = -1
        Isel = 0  # to be checked
        for i in l:
            for j in range(0, len(N[i])):
                Ii = 0
                Ij = 0
                Ii = log(1/(nCr(n-1, len(N[i]))))
                Ij = log(1/(nCr(n-1, len(N[N[i][j]]))))
                R = P[i][j]/(Ii+Ij)
                if R > Rmax:
                    s = N[i][j]
                    R = Rmax
                    Isel = Ij

            flag = True
            for j in range(0, len(Q[s])):
                if Isel > log(Q[s][j]):
                    flag = False
            if flag:
                Sel.add(s)
                Pmax += sum(P[s])
    return Pmax, Sel
```

Figure 4.3.4 Function of S-dkp Algorithm

- Triadic(e) : Apply Triadic Closure to predict hidden edges from published graph

```
def triadic(e):
    new_edges = []

    for i in e:
        a, b = i

        for j in e:
            x, y = j

            if i != j:
                if a == x and (b, y) not in e and (y, b) not in e:
                    new_edges.append((b, y))
                if a == y and (b, x) not in e and (x, b) not in e:
                    new_edges.append((b, x))
                if b == x and (a, y) not in e and (y, a) not in e:
                    new_edges.append((a, y))
                if b == y and (a, x) not in e and (x, a) not in e:
                    new_edges.append((a, x))

    return new_edges
```

Figure 4.3.5 Function of Triadic Closure

- **jaccard(G):** Apply Jaccard similarity method to find masked edges from published graph.

```
def Jaccard(G):
    Pred = list(nx.jaccard_coefficient(G))
    Pred = [(Pred[i][0], Pred[i][1])for i in range(len(Pred))]
    Pred = set(Pred)
    Pred = list(Pred)
    sum = 0.0
    for i in Pred:
        if(i in I):
            sum = sum + 1.0
    falsenegetive = 0.0
    for i in I:
        if(i not in Pred):
            falsenegetive += 1.0


    Prednegetive = ((n*(n-1))) - len(Pred)
    print(len(Pred))
    truepositive = sum
    falsepositive = len(Pred)-sum
    Fscore = (truepositive/(truepositive+(0.5*(falsepositive+falsenegetive))))
    Precision = (truepositive/(truepositive+falsepositive))
    Recall = (truepositive/(truepositive+falsenegetive))
    return [Fscore, Precision, Recall]
```

Figure 4.3.6 Function of Jaccard

- **ResourceAllocation(G):** Apply **ResourceAllocation** method to find masked edges from published graphs.

```
def ResourceAllocation(G):
    Pred = list(nx.resource_allocation_index(G))
    Pred = [(Pred[i][0], Pred[i][1])for i in range(len(Pred))]

    Pred = set(Pred)
    Pred = list(Pred)
    sum = 0.0
    for i in Pred:
        if(i in I):
            sum = sum + 1.0
    falsenegetive = 0.0
    for i in I:
        if(i not in Pred):
            falsenegetive += 1.0
    Prednegetive = ((n*(n-1))) - len(Pred)
    truepositive = sum
    falsepositive = len(Pred)-sum
    Fscore = (truepositive/(truepositive+(0.5*(falsepositive+falsenegetive))))
    Precision = (truepositive/(truepositive+falsepositive))
    Recall = (truepositive/(truepositive+falsenegetive))
    return [Fscore, Precision, Recall]
```

Figure 4.3.7 Function of Resource Allocation

- **AdamicAdar(G):** Apply **AdamicAdar** method to find masked edges from published graph.

```
def AdamicAdar(G):
    Pred = list(nx.adamic_adar_index(G))
    Pred = [(Pred[i][0], Pred[i][1])for i in range(len(Pred))]

    Pred = set(Pred)
    Pred = list(Pred)
    sum = 0.0
    for i in Pred:
        if(i in I):
            sum = sum + 1.0
    falsenegetive = 0.0
    for i in I:
        if(i not in Pred):
            falsenegetive += 1.0


    Prednegetive = ((n*(n-1))) - len(Pred)
    truenegetive = Prednegetive - falsenegetive
    truepositive = sum
    falsepositive = len(Pred)-sum
    Fscore = (truepositive/(truepositive+(0.5*(falsepositive+falsenegetive))))
    Precision = (truepositive/(truepositive+falsepositive))
    Recall = (truepositive/(truepositive+falsenegetive))
    return [Fscore, Precision, Recall]
```

Figure 4.3.8 Function of Adamic Adar

- **preferentialAttachment(G):** Apply method to find masked edges from published graph.

```
def preferentialAttachment(G):
    Pred = list(nx.preferential_attachment(G))
    Pred = [(Pred[i][0], Pred[i][1])for i in range(len(Pred))]
    Pred = set(Pred)
    Pred = list(Pred)
    sum = 0.0
    for i in Pred:
        if(i in I):
            sum = sum + 1.0
    falsenegetive = 0.0
    for i in I:
        if(i not in Pred):
            falsenegetive += 1.0
    Prednegetive = ((n*(n-1))) - len(Pred)
    print(len(Pred))
    truepositive = sum
    falsepositive = len(Pred)-sum
    Fscore = (truepositive/(truepositive+(0.5*(falsepositive+falsenegetive))))
    Precision = (truepositive/(truepositive+falsepositive))
    Recall = (truepositive/(truepositive+falsenegetive))
    return [Fscore, Precision, Recall]
```

Figure 4.3.9 Function of Preferential Attachment

Chapter 5

RESULTS

5.1 Hardware and Software Requirements

For the successful, efficient and problem free designing of any project or program, the system should meet some requirements. The hardware requirements of a system are Intel core i3, 4GB RAM , monitor, keyboard and a mouse. This project is platform independent, we have used windows. The software used are VScode, Juypter notebook and Anaconda. The project is implemented using Python language. Pandas, Matplotlib, Numpy, and Scikit learn are some of the modules of python used in this project.

5.2 Performance evaluation metrics

The performance of the classifiers are evaluated using different metrics called performance evaluation metrics. Different performance evaluation metrics used in this study are accuracy, sensitivity, and specificity. Confusion matrix is used to calculate values for these metrics. Mathematically it is calculated as:

- Accuracy = (TP + TN) /(TP + TN + FP + FN) × 100
- Precision = TP/ (TP + FP)
- Fscore = (2*Precision*Recall)/(Precision+Recall)

T_p= Number of true edges prediction.

T_N= Number of true non-edges predictions.

F_P= Number of false edges prediction

F_N = Number of false non-edges predictions

5.3 Experiment Results

5.3.1 Performance of Inference attacks before PPA

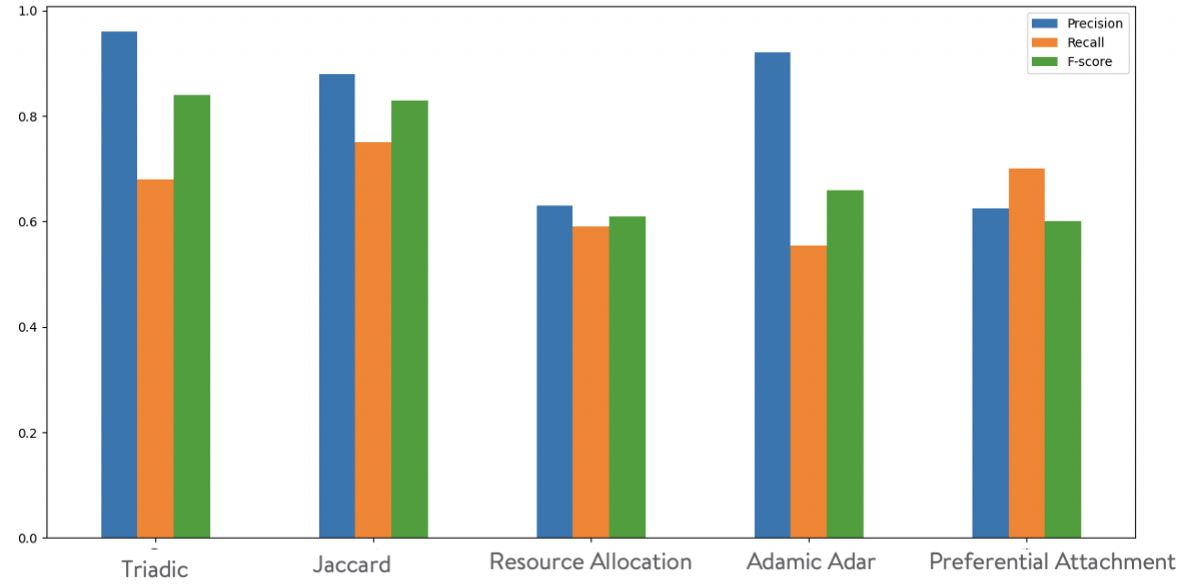


Figure 5.3.1 Performance of Inference attacks before PPA

5.3.2 Performance of Inference attacks after PPA

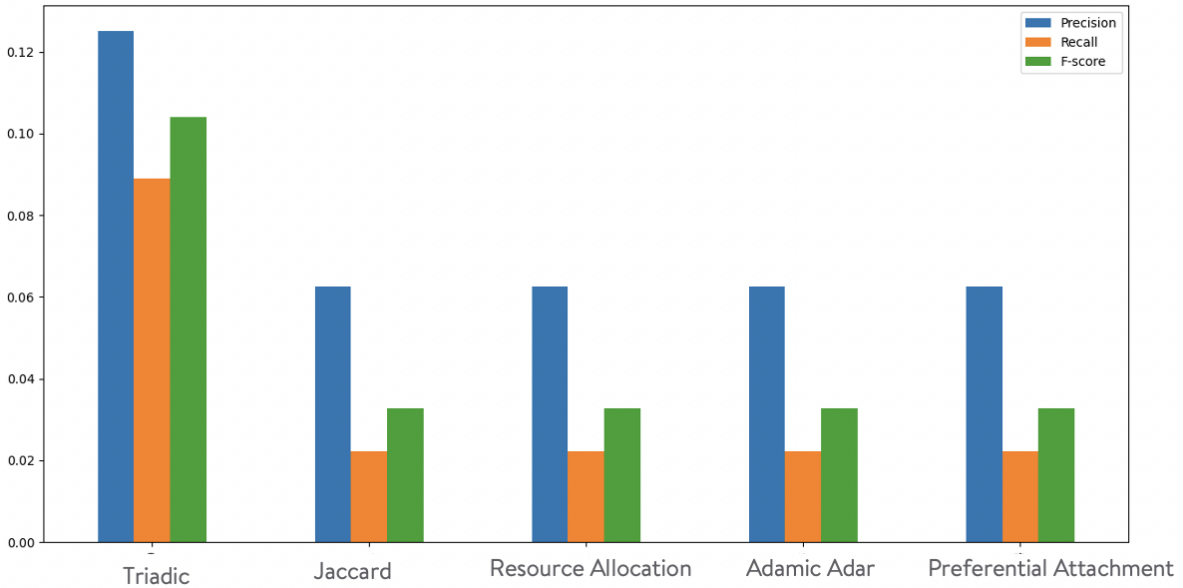


Figure 5.3.2 Performance of Inference attacks after PPA

From the above results we can see that after applying the PPA algorithm the F Score, Recall and Precision of inference attacks has reduced immensely.

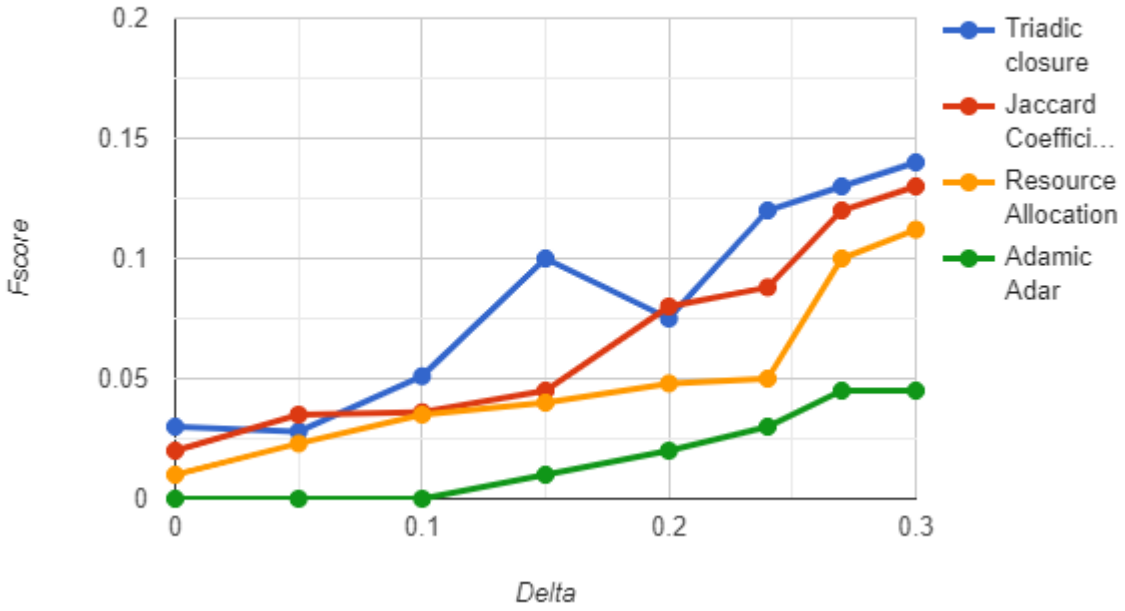


Figure 5.3.3 F-score Vs local classifiers after using PPA

5.3.3 Performance of Inference attacks before Sd-kp

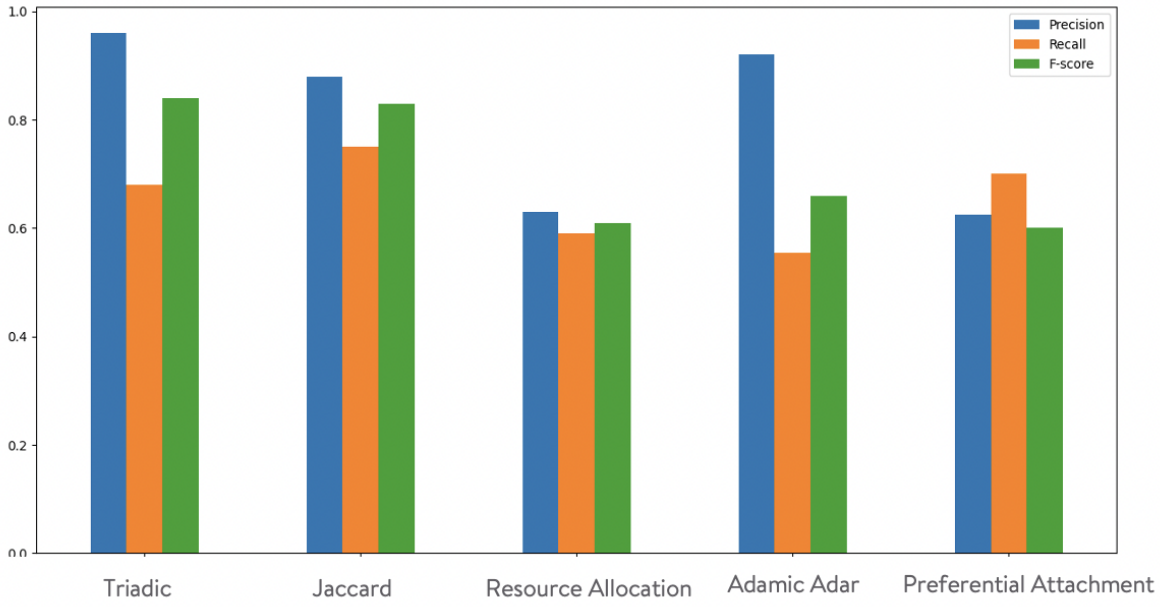


Figure 5.3.4 Performance of Inference attacks before Sd-kp

5.3.4 Performance of Inference attacks after Sd-kp

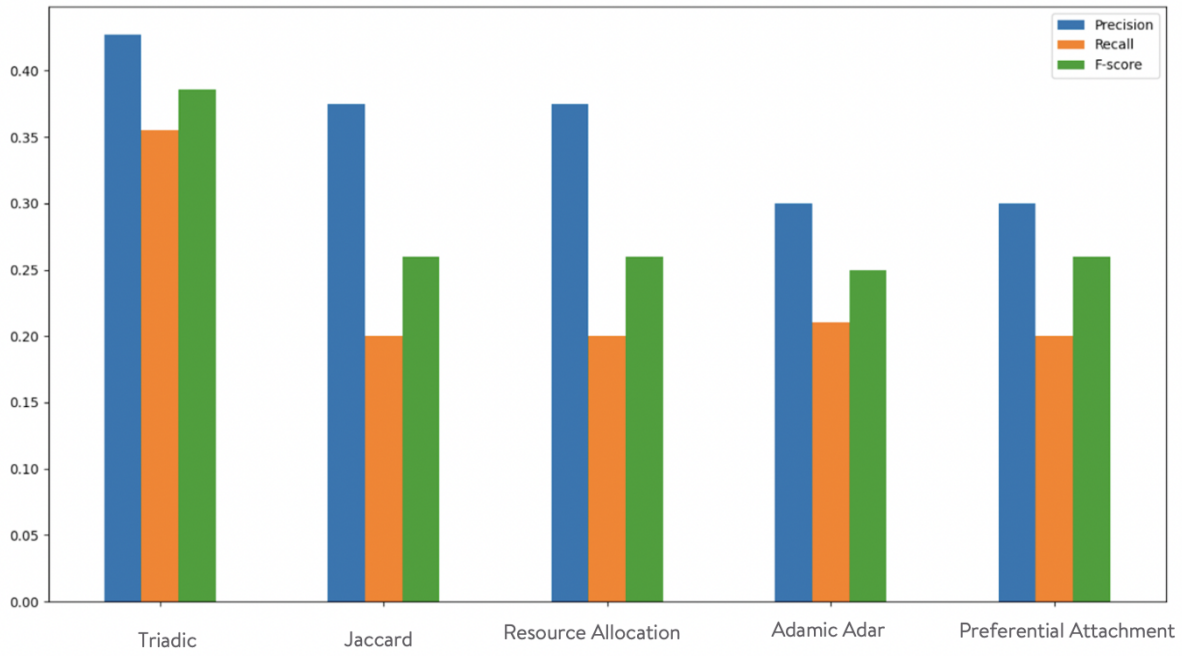


Figure 5.3.5 Performance of Inference attacks after Sd-kp

From the above results we can see that after applying the sdkp algorithm the F Score, Recall and Precision of inference attacks has reduced immensely.

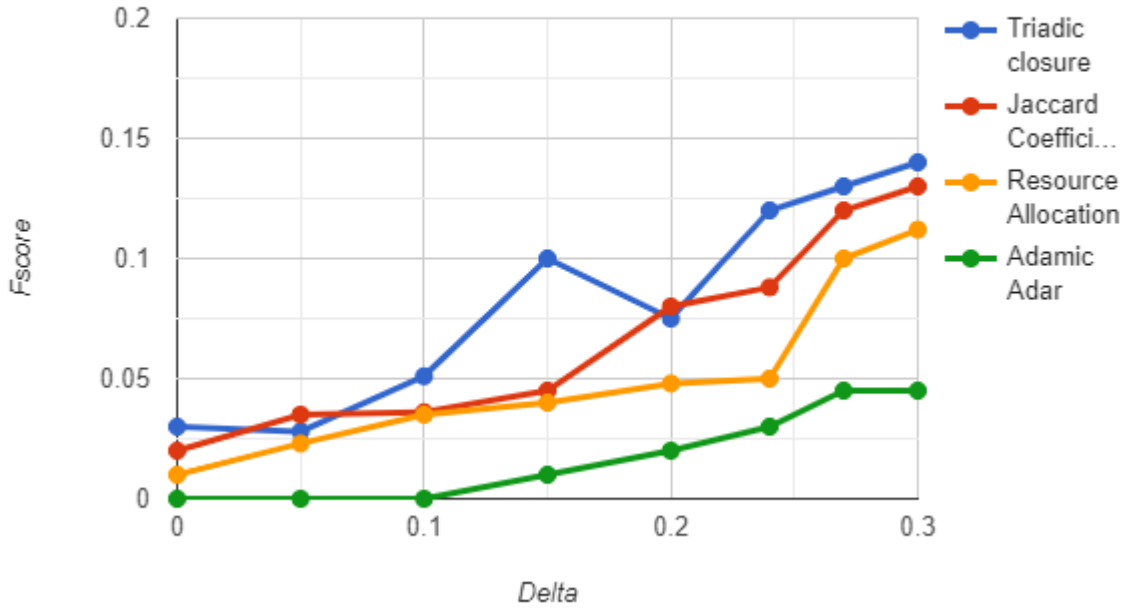


Figure 5.3.6 F-score Vs local classifiers after using S-kdp

5.3.5 Dataset-1 (100 node)

5.3.5.1 Profile attribute utility

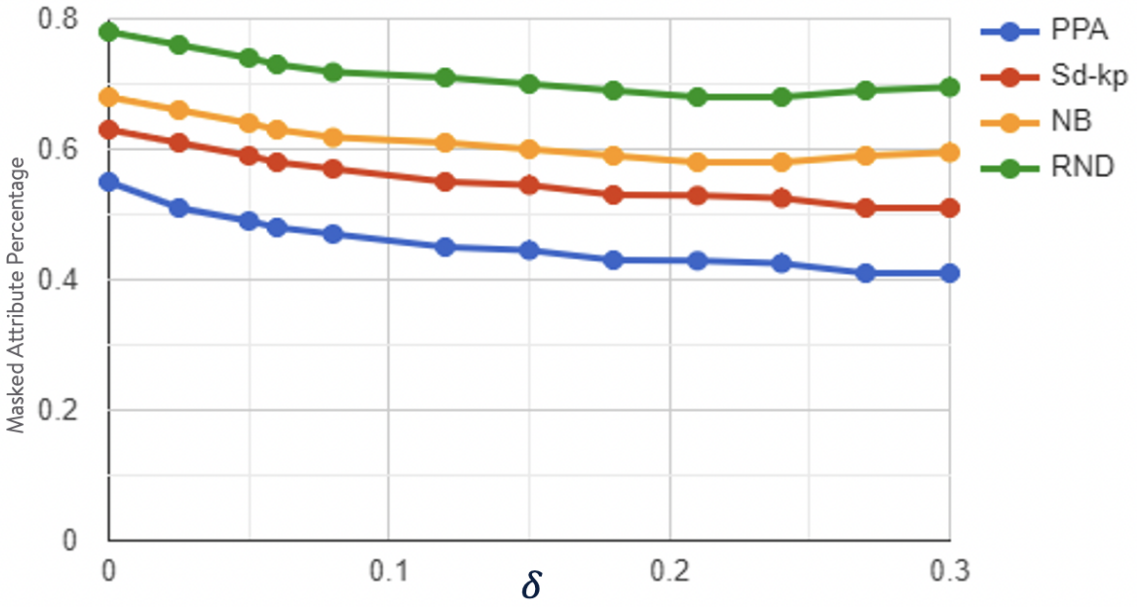


Figure 5.3.7 Masked Attribute - δ

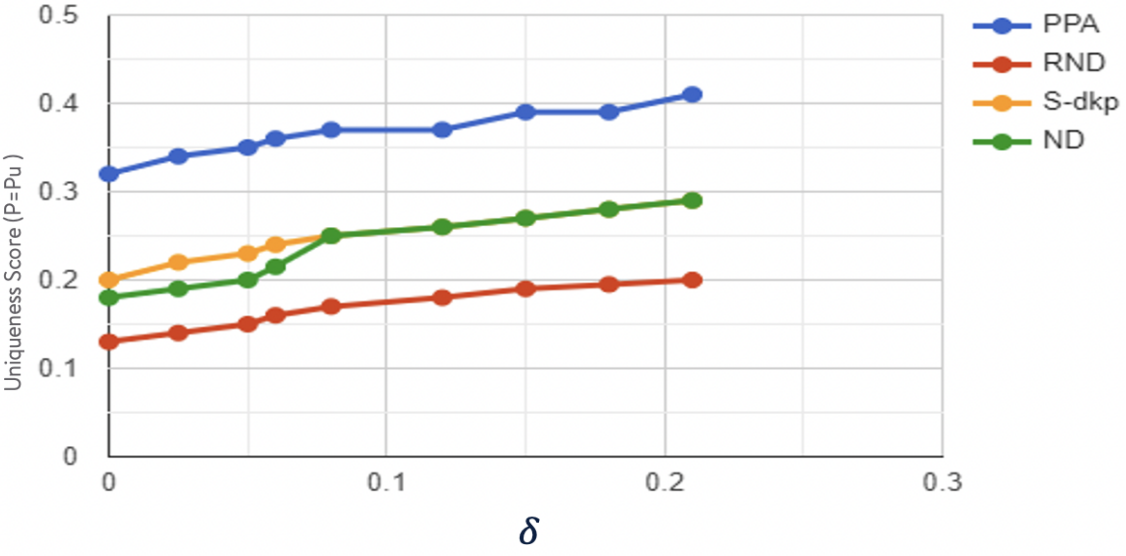


Figure 5.3.8 Uniqueness Score - δ

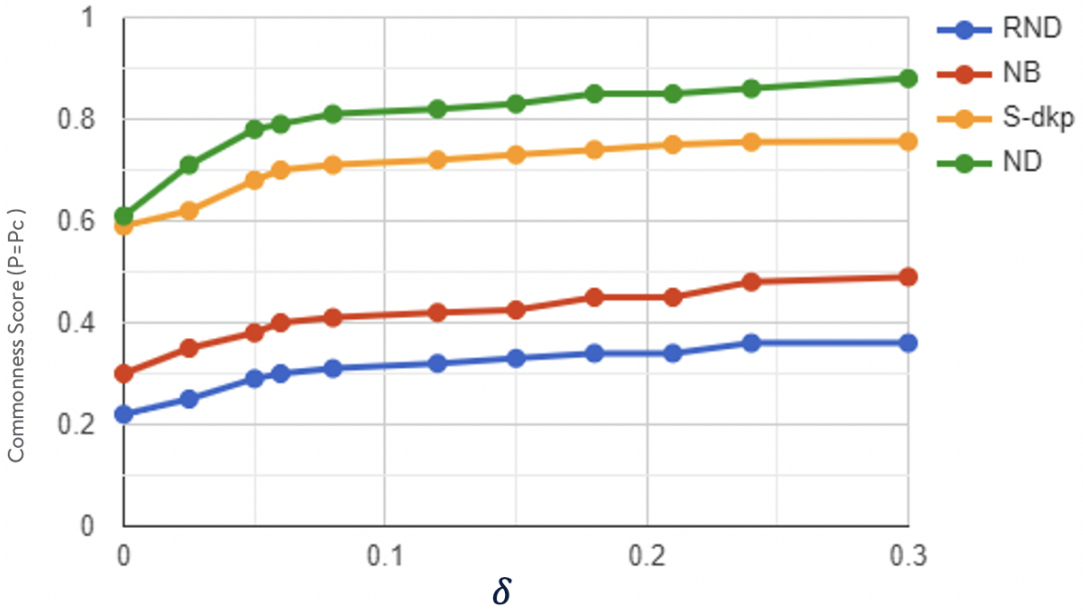


Figure 5.3.9 Commonness Score - δ

5.3.5.2 Social relation utility

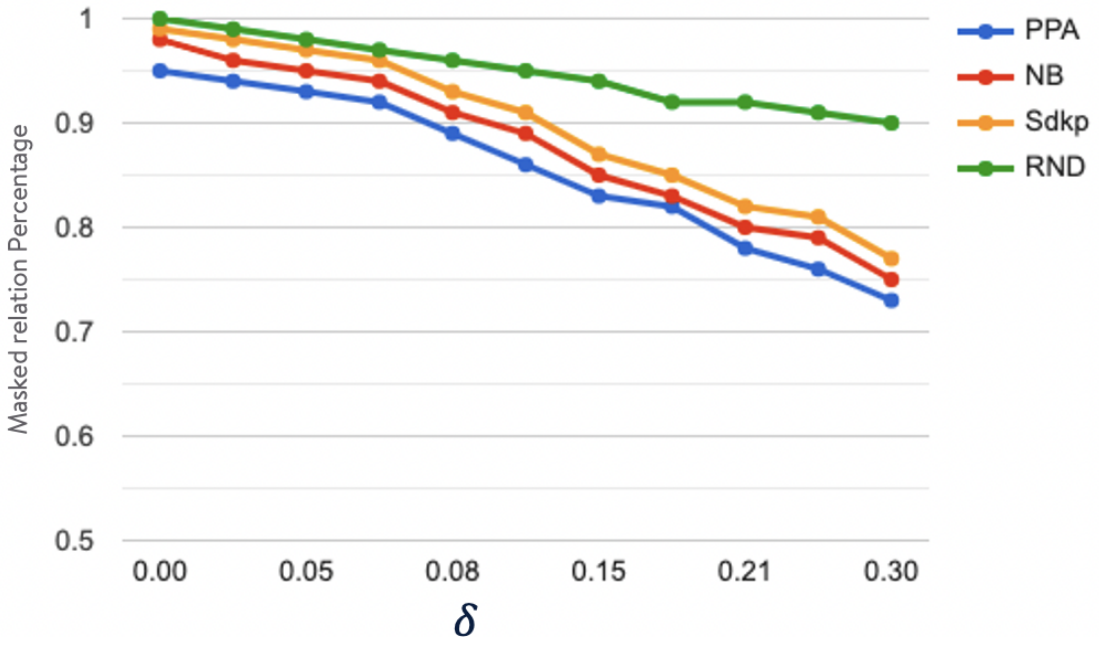


Figure 5.3.10 Masked Relation - δ

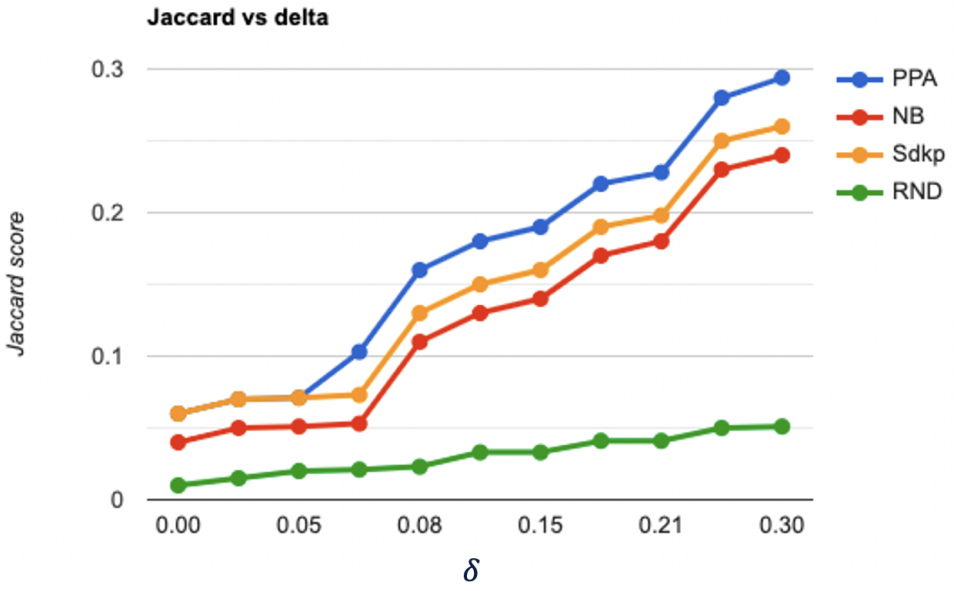


Figure 5.3.11 Jaccard Score - δ

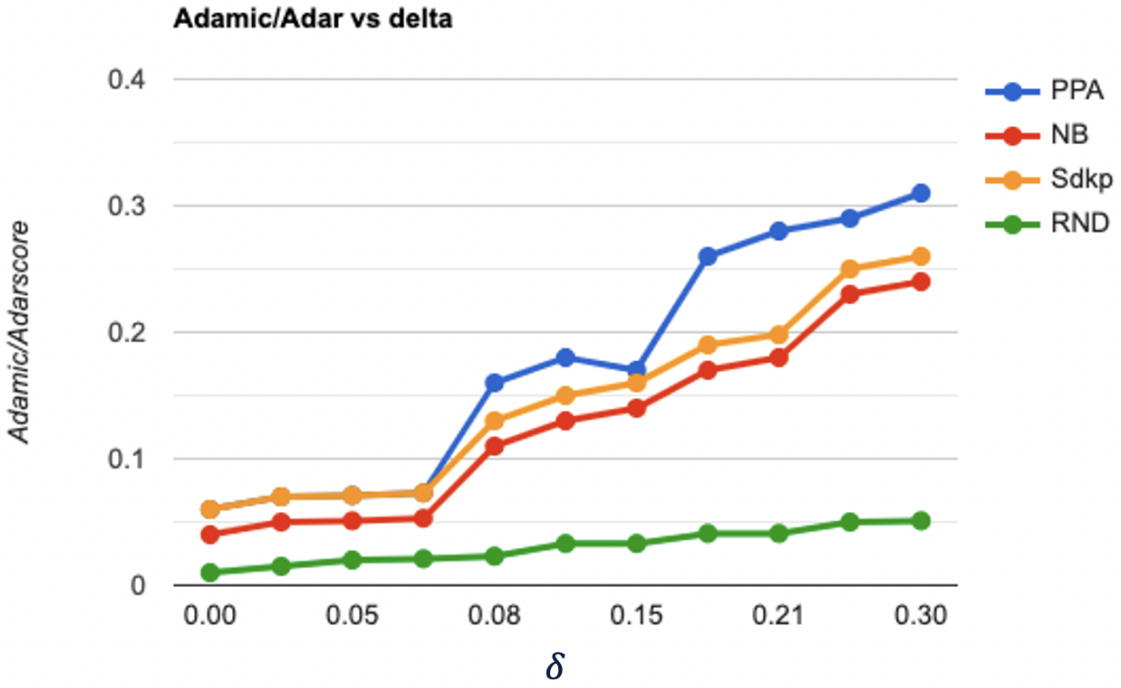


Figure 5.3.12 Adamic/Adar Score - δ

5.3.6 Dataset-2 (500 node)

5.3.6.1 Profile attribute utility

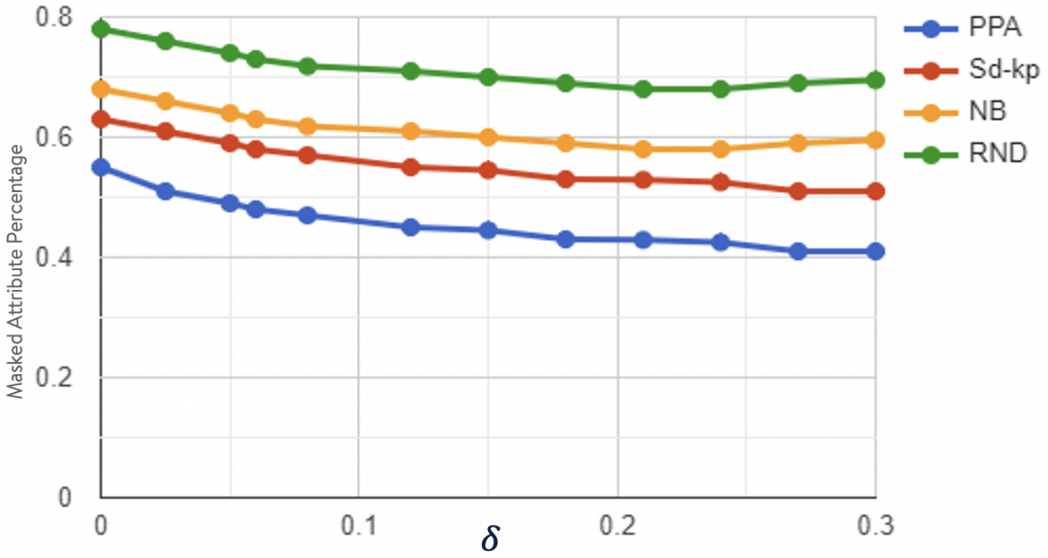


Figure 5.3.13 Masked Attribute - δ

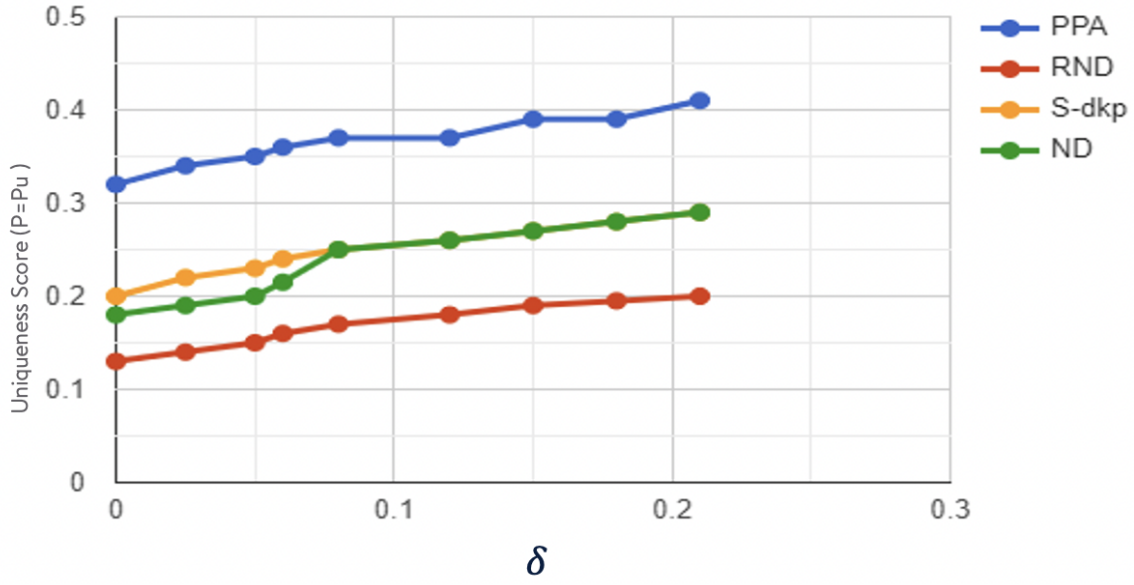


Figure 5.3.14 Uniqueness Score - δ

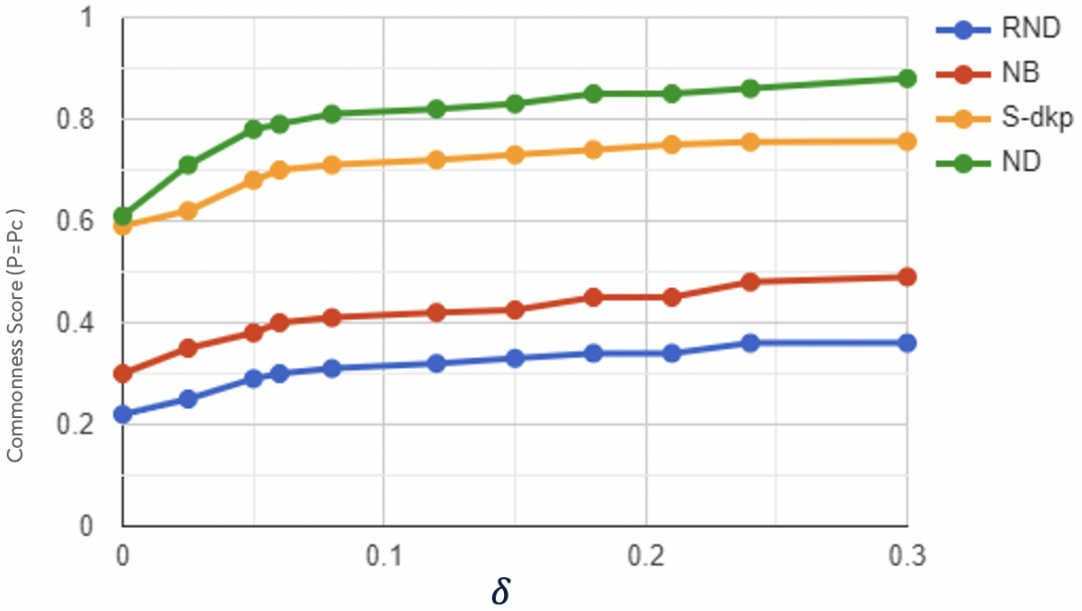


Figure 5.3.15 Commonness Score - δ

5.3.6.2 Social relation utility

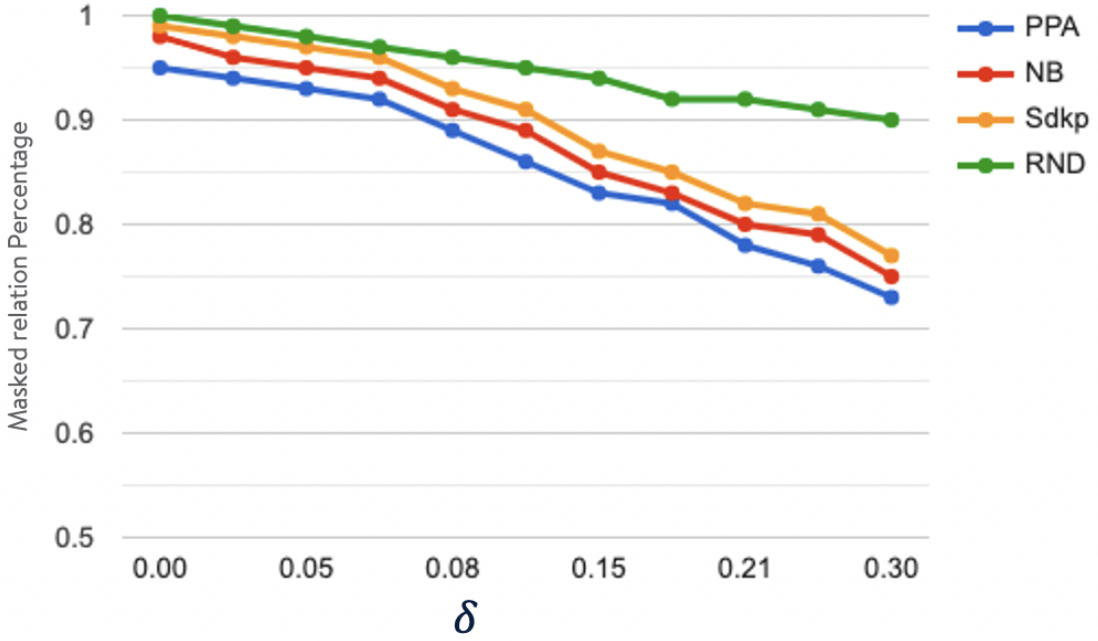


Figure 5.3.16 Masked Relation - δ

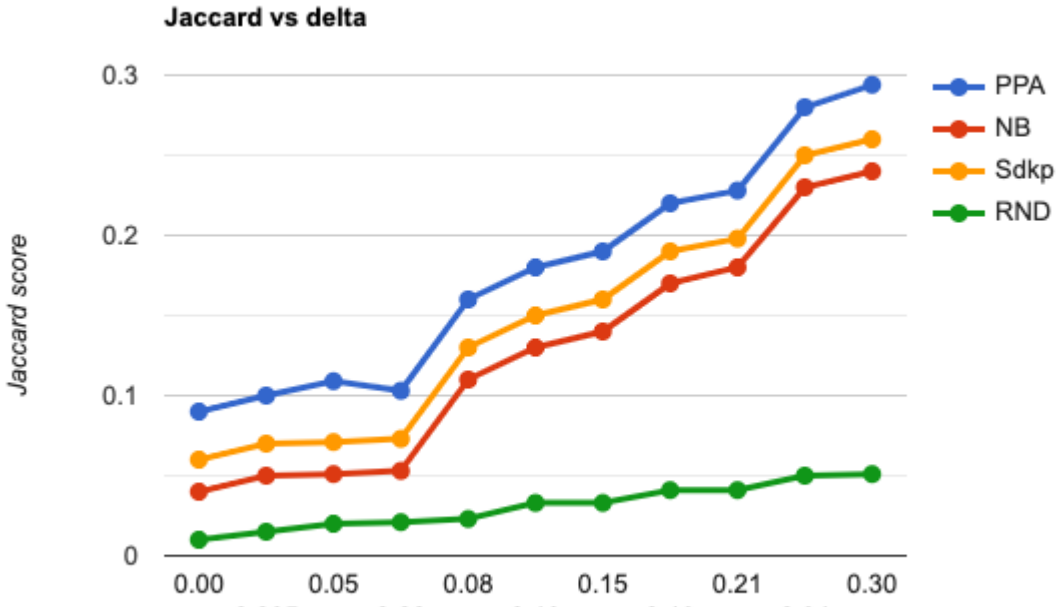


Figure 5.3.17 Jaccard Score - δ

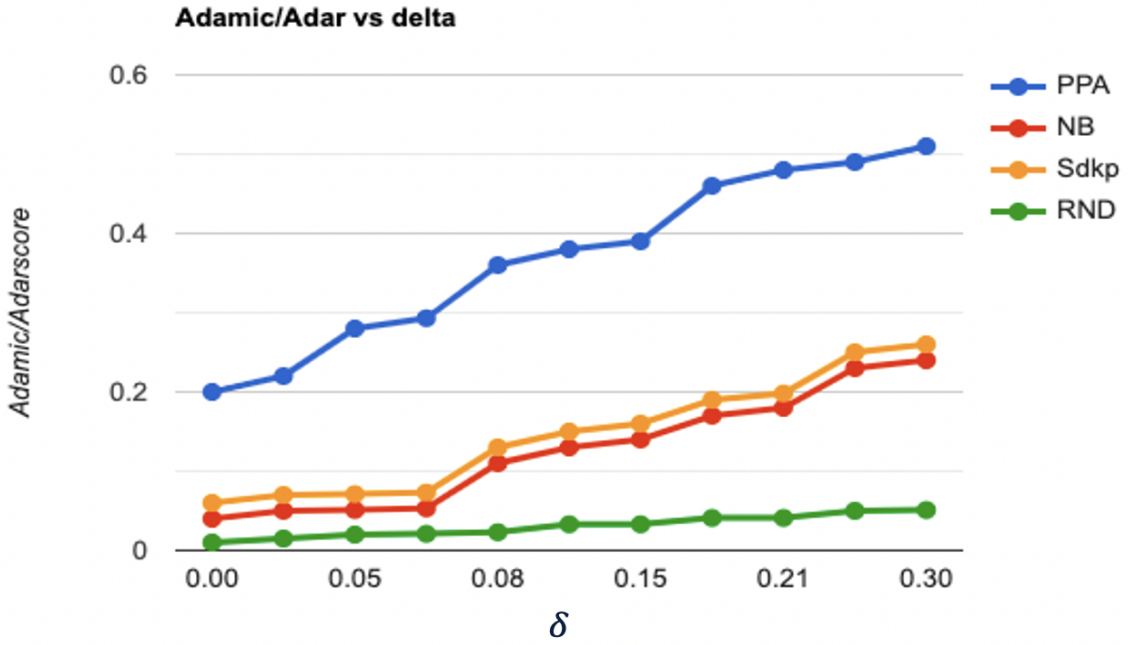


Figure 5.3.18 Adamic/Adar Score - δ

Chapter-6

Conclusions and Future Enhancement

REFERENCES

[1] Weihao Li and Hui Li Xidian University, Xi'an, China," LRDM: Local Record-Driving Mechanism for Big Data Privacy Preservation in Social Networks " IEEE First International Conference on Data Science in Cyberspace , 2016.

[2] Na Li CSE Department The University of Texas at Arlington , Nan Zhang CS Department George Washington University, Sajal K. Das CSE Department The University of Texas at Arlington," Relationship Privacy Preservation in Publishing Online Social Networks", IEEE International Conference on Privacy, Security, Risk, and Trust, and IEEE International Conference on Social Computing,2011.

[3] Yasmeen Alufaisan Department of Computer Science Northern Kentucky University Highland Heights, USA, Alina Campan Department of Computer Science Northern Kentucky University Highland Heights, USA ," Preservation of Centrality Measures in Anonymized Social Networks", SocialCom/PASSAT/BigData/EconCom, 2013.

[4] JINQUAN ZHANG College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China , XIAO WANG1 , YANFENG YUAN1 , AND LINA NI,Key Laboratory of the Ministry of Education for Embedded System and Service Computing, Tongji University, Shanghai 201804, China: RcDT," Privacy Preservation Based on R-Constrained Dummy Trajectory in Mobile Social Networks".

[5] Mayank Singh Shishodia Sumeet Jain B. K. Tripathy Vellore Institute of Technology, Vellore, India 632014 ," GASNA: Greedy Algorithm for Social Network Anonymization", IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining , 2013.

[6] Lorenz Schwittmann, Matthäus Wander, Christopher Boelmann, and Torben Weis University of Duisburg-Essen," Privacy Preservation in Decentralized Online Social Networks", Published by the IEEE Computer Society 1089-7801/14/$31.00 © IEEE,2014.

[7] DAN YIN, YIRAN SHEN , (Member, IEEE), AND CHENYANG LIU Department of Computer Science and Technology, Harbin Engineering University, Harbin 150001,

China,” Attribute Couplet Attacks and Privacy Preservation in Social Networks”, November 2, 2017.

[8] KAH MENG CHONG AND AMIZAH MALIP Faculty of Science, Institute of Mathematical Sciences, University of Malaya, Kuala Lumpur 50603, Malaysia Corresponding author: Amizah Malip,”Trace Me If You Can: An Unlinkability Approach for Privacy-Preserving in Social Networks”.

[9] Kainan Zhang, Zhi Tian, Zhipeng Cai, and Daehee Seo,” Link-Privacy Preserving Graph Embedding Data Publication with Adversarial Learning”, Number 2, April 2022.

[10] HONG ZHU , XIN ZUO , AND MEIYI XIE School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China,”DP-FT: A Differential Privacy Graph Generation With Field Theory for Social Network Data Release”, November 21, 2019.

[11] Haiping Huang , Member, IEEE, Dongjun Zhang, Fu Xiao , Member, IEEE, Kai Wang, Jiateng Gu, and Ruchuan Wang,” Privacy-Preserving Approach PBCN in Social Network With Differential Privacy”, VOL. 17, NO. 2, JUNE 2020.

[12] Borui Yang, Jianxin Li, Yingjie Cao, Hua Wei, Peiyuan Sun, Nannan Wu and Bo Li School of Computer Science & Engineering Beihang University Beijing, China, Lu Liu College of Engineering and Technology University of Derby Derby, UK,”ShutterRoller: Preserving Social Network Privacy towards High-Speed Domain Gateway”, IEEE International Conference on Computer and Information Technology, Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing,2015.

[13] Qin Liu, Guojun Wang, Member, IEEE, Feng Li, Shuhui Yang, Member, IEEE, and Jie Wu, Fellow, IEEE,”Preserving Privacy with Probabilistic Indistinguishability in Weighted Social Networks”,IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 28, NO. 5, MAY 2017.

[14] Youyang Qu , Member, IEEE, Shui Yu , Senior Member, IEEE, Wanlei Zhou , Senior Member, IEEE, Shiping Chen , Senior Member, IEEE, and Jun Wu, Senior Member, IEEE ,” Customizable Reliable Privacy-Preserving Data Sharing in Cyber-Physical Social Networks”, 1, JANUARY-MARCH 2021

[15] Qian Wang , Member, IEEE, Yan Zhang, Xiao Lu, Zhibo Wang, Member, IEEE, Zhan Qin, Student Member, IEEE, and Kui Ren, Fellow, IEEE," Real-Time and Spatio-Temporal Crowd-Sourced Social Network Data Publishing with Differential Privacy", JULY/AUGUST 2018.