Project Proposal, Anonymized Social Networks- CS6150 Advanced Algorithms

Paper: ‘Wherefore Art Thou R3579X? Anonymized Social Networks, Hidden Patterns, and Structural Steganography’ - Backstrom, Dwork, Kleinberg, 2007

Team Members:

Scott Gale, u1203422, scottdgale@gmail.com

Sudie Roweton, u1210082, sudie.roweton@gmail.com

Dennis Njeru, u1076700, kansla@yahoo.com

1. Introduction

Anonymization of social networks allows researchers the valuable opportunity to study the structure of social networks while protecting the privacy of the social entities or users of those networks. The question becomes, is anonymization enough to prevent adversaries from compromising privacy? This paper discusses attacks that an adversary can utilize to compromise the privacy of the social entities that exist in an anonymized copy of a social network. Compromising privacy in this context refers to discovering whether a relationship exists between targeted users.

The paper presents two main categories of attacks: Active and passive. Active attacks are those in which the adversary creates fake user accounts and establishes links from those newly created accounts to a targeted group of users that already exist in the network thus creating a subgraph in the network. In contrast, passive attacks are those in which the adversary already exists in the network and colludes with a group of people that already exist in the network; thus, the subgraph exists in the network without having to create new user accounts. The goal of the passive attack is to uncover relationships among users that the adversaries are already linked to. There exists another type of attack, the semi-passive attack, that is a natural extension to the passive attack. The semi-passive attack is similar to the passive attack except the colluding adversaries each connect to specific targeted user(s). The idea is, if the adversaries can find their subgraph, then, by extension, they would have found the targeted user(s). Walk-based and cut-based attacks are different active attack approaches to recover that subgraph in the network. The walk-based active attack will be the focus of our project.

2. Formal description of the algorithmic problem

The structure of a social network is a graph where the nodes represent the users and the edges represent relationships between the users. The algorithmic problem presented in the paper centers around how to determine if two (or more) anonymous nodes in a graph have a relationship (edge), thus, compromising privacy. Formally, there is a targeted set of users *w1* , *w2* , *w3*, . . . ,*wn* in an anonymized copy of the social network, *G.* Then, the problem to be solved is to determine whether or not the edge (*wi*, *wj*) exists for every *i, j*.

This is done by creating a relatively small graph and implanting it into a much larger graph. The small graph is connected to targeted nodes, *w1*, *w2*, *w3*, . . . , *wn*, in the large graph prior to the graph being anonymized. Once the graph is anonymized, the goal is to locate our small implanted graph so we can determine relationships that exist between the targeted nodes in the larger graph.

3. Algorithm

Overview

Let $G = (V, E)$ be a n-node anonymized undirected graph.

Choose set of targeted users: $W = \{w_1, w_2, w_3, \ldots w_b\}$ where w_i is an individual node in a network and where $b = O(log^2 n)$.

Goal: Identify targeted nodes in G denoted by $W = \{w_1, w_2, w_3, \ldots w_k\}$. Try to discover if the edge (w_i, w_j) exists . . . looking for relationships between nodes in W.

Algorithm - Active "Walked-Based Attack"

1. Create k new nodes: $X = \{x_1, x_2, x_3, \ldots, x_k\}$ where $k = (2 + \delta)logn$; where $\delta > 0$ (small constant).
2. Create a random graph H (defined by X) to insert into G; H will connect to G in a clever way such that we can discover post anonymization.
3. H is connected to G by creating an edge from a node in H to a node in G. Thus creating edges (x_i, w_i) .
4. Deterministically create edges between nodes (x_i, x_{i+1}) - this allows traversal of X in order.
5. Choose two constants $d_0 \leq d_1 = O(logn)$.
6. For each $i = 1, 2, 3, \ldots, k$ we choose an external degree $\Delta_i \in [d_0, d_1]$ specifying the number of edges x_i will have to nodes in $G - H$. These are chosen uniformly at random from the interval $[d_0, d_1]$.
7. Choose a set $N_j \subseteqq \{x_1, x_2, x_3, \ldots, x_k\}$ where each x_i is unique and $N_j \leq c$ where c is constant $(c = 3)$ in this example. Each x_i can appear in at most Δ_i of the sets N_j.
8. Construct links to w_j from each $x_i \in N_j$.
9. Create random edges in H where the edge (x_i, x_j) has $\frac{1}{2}$ probability of existing.
10. $\Delta_i' =$ the degree of x_i of all nodes (from edges in both G and H).

-------------------------------Network gets anonymized---

Recovering H given G

1. Search G for every node with degree Δ_1'.
2. Successively try and add nodes with degree Δ_i' that also have the correct edges to other corresponding nodes in X.
3. Once H is discovered, discover $W = \{w_1, w_2, w_3, \ldots w_k\}$ from the unique relationship between each w_j and N_j.

4. Other baseline algorithms

Cut-Based Attack:

Another active attack, the “cut-based attack”, is constructed by attaching H to G using very few edges and recovering H using Gomory-Hu cut trees. However since H is “thinly” attached to G, H will be unique not only to the attacker, but also potentially to an observer (i.e. easier to detect). The cut-based attack has the advantage of matching the tight theoretical bound on the number of nodes needed to recover H. However the use of Gomory-Hu trees makes the recovery algorithm more expensive than that of the walk-based attack. Additionally, the cut-based attack can only compromise O(k) users as compared to O(k2) users for the walk-based attack.

Passive Attacks:

In a passive attack, regular users are able to discover their locations in G using their knowledge of the local structure of the network around them. An example of this kind of attack is where a small coalition of passive attackers collude to discover their location. By doing so, they compromise the privacy of some of their neighbors: those connected to a unique subset of the coalition, and hence unambiguously recognizable once the coalition is found. This passive attack is analogous to the walk-based attack, except that the structure of H occurs organically as a natural function of individuals using the system. Since the coalition X has not specifically targeted any nodes, it is possible that although they can uniquely find themselves, they cannot locate any specific users other than themselves. However, empirical evidence suggests that once a coalition is moderately sized, it can compromise the privacy of at least some users. The primary disadvantage of this attack as compared to the active-based attack is that it does not allow one to compromise the privacy of arbitrary users.

5. Project Proposal

Objective: Implement the Active Walk-Based Algorithm as discussed in section 3 then deterministically remove a certain percentage of edges to determine at what point the algorithm becomes ineffective. We will use the Java programming language to implement this project. We have decided to code our own graph class to maintain maximum flexibility in performing multiple simulations.

Step 1: Create Datasets. We found several large anonymized social network datasets on the Stanford Network Analysis Project. We will create our own datasets modeled after these datasets by replicating the number of nodes, edges, average number of edges per node, etc. We will do this for two different sizes of networks - small (4K+ nodes / 80K+ edges - modeled after the "ego-Facebook" dataset) and large (100K+ nodes / 100M+ edges - modeled after the "gemsec-Facebook").

Step 2: Implement the "Walk-Based Attack" algorithm. Once we load our datasets we will implement the "Walk-Based" algorithm on our dataset. We want to ensure the algorithm is working properly and we are reproducing the probability of recovering our implanted graph in accordance with published data from the paper. Once we have successfully implemented the algorithm and it is working properly we will move on to step 3.

Step 3: Deterministically remove edges to discover the effectiveness of the algorithm. We will start by randomly cutting 1% of all edges in the graph. This cut will include edges in G and in H (the complete anonymized graph). We are replicating something an entity might due to protect their anonymized data from an attack such as this. We will record our results from cutting 1% of the edges. We will then cut 2% of the edges and so on and so forth until we reach a threshold where the algorithm fails with high probability.

Step 4: Recording and analysis. We will record all of our results in text and in graph form and describe the correlation between edge cuts and successfully conducting the attack. We will highlight any differences discovered between the different sizes of networks - if a larger network is more resilient to such attacks (or not). Lastly, we will summarize the effectiveness, based on our analysis, of cutting edges to further ensure that anonymized data remains anonymized.