Preventing Private Information Inference Attacks on Social Networks
Technical Report UTDCS-03-09

Raymond Heatherly, Murat Kantarcioglu,
and Bhavani Thuraisingham
Computer Science Department
University of Texas at Dallas

Jack Lindamood
Facebook

February 22, 2009

Abstract

On-line social networks, such as Facebook, are increasingly utilized by many people. These networks allow users to publish details about themselves and connect to their friends. Some of the information revealed inside these networks is meant to be private. Yet it is possible that corporations could use learning algorithms on released data to predict undisclosed private information. In this paper, we explore how to launch inference attacks using released social networking data to predict undisclosed private information about individuals. We then devise three possible sanitization techniques that could be used in various situations. Then, we explore the effectiveness of these techniques by implementing them on a dataset obtained from the Dallas/Fort Worth, Texas network of the Facebook social networking application and attempting to use methods of collective inference to discover sensitive attributes of the data set. We show that we can decrease the effectiveness of both local and relational classification algorithms by using the sanitization methods we described. Further, we discover a problem domain where collective inference degrades the performance of classification algorithms for determining private attributes.

1 Introduction

Social networks are online applications that allow their users to connect by means of various link types. As part of their offerings, these networks allow people to list details about themselves that are relevant to the nature of the network. For instance, Facebook is a general-use social network, so individual users list their favorite activities, books, and movies. Conversely, LinkedIn is a professional network; because of this, users specify details are related to their professional life (i.e. reference letters, previous employment, etc.)

This personal information allows social network application providers a unique opportunity; direct use of this information could be useful to advertisers for direct marketing. However, in practice, privacy concerns can prevent these efforts [2]. This conflict between desired use of data and individual privacy presents an opportunity for social network data mining – that is, the discovery of information and relationships from social network data. The privacy concerns of individuals in a social network can be classified into one of two categories: privacy after data release, and private information leakage.

Privacy after data release has to do with the identification of specific individuals in a data set subsequent to its release to the general public or to paying customers for specific usage. Perhaps the most illustrative example of this type of privacy breach (and the repercussions thereof) is the AOL search data scandal. In 2006, AOL released the search results from 650,000 users for research purposes. However, these results had a significant number of "vanity" searches – searches on an individual's name, social security number, or address – that could then be tied back to a specific individual.

Private information leakage, conversely, is related to details about an individual that is not explicitly stated, but, rather, is inferred through other details released and/or relationships to individuals who may express that trait. A trivial example of this type of information leakage is a scenario where a user, say John, does not enter his political affiliation because of privacy concerns. However, it is publicly available that he is a member of the College Democrats. Using this publicly available information regarding a general group membership, it is easily guessable what John's political affiliation is. We note that this is an issue both in live data (i.e. currently on the server) and in any released data.

This paper focuses on the problem of private information leakage for individuals as a direct result of their actions as being part of an online social network. We model an attack scenario as follows: Suppose Facebook wishes to release data to Electronic Arts for their use in advertising games to interested people. However, once Electronic Arts has this data, they want to identify the political affiliation of users in their data for lobbying efforts. This would obviously be a privacy violation of hidden details. We explore how the online social network data could be used to predict some individual private trait that a user is not willing to disclose (e.g. political or religious affiliation) and explore the effect of possible data sanitization approaches on preventing such private information leakage, while allowing the recipient of the sanitized data to do inference on non-private traits.

1.1 Our contributions

To the best of our knowledge, this is the first paper that discusses the problem of sanitizing a social network to prevent inference of social network data and then examine the effectiveness of those approaches on a real-world dataset. In order to protect privacy, we sanitize both details and link details. That is, deleting some information from a user's profile and removing links between friends. We then study the effect this has on combating possible inference attacks.

Additionally, we present a modification of the Naïve Bayes classification algorithm that will use details about a node, as well as the node's link structure, to predict private details. We then compare the accuracy of this new learning method against the accuracy of the traditional Naïve Bayes classifier.

2 Related Work

In this paper, we touch on many areas of research that have been heavily studied. The area of privacy inside a social network encompasses a large breadth, based on how privacy is defined. In [1], authors consider an attack against an anonymized network. In their model, the network consists of only nodes and edges. Trait details are not included. The goal of the attacker is to simply identify people. Further, their problem is very different than the one considered in this paper because they ignore trait details and do not consider the effect of the existence of trait details on privacy.

In [4] and [9], authors consider several ways of anonymizing social networks. However, our work focuses on inferring details from nodes in the network, not individually identifying individuals.

Other papers have tried to infer private information inside social networks. In [5], authors consider ways to infer private information via friendship links by creating a Bayesian Network from the links inside a social network. While they crawl a real social network, Livejournal, they use hypothetical attributes to analyze their learning algorithm. Also, compared to [5], we provide techniques that can help with choosing the most effective traits or links that need to be removed for protecting privacy. Finally, we explore the effect of collective inference techniques in possible inference attacks.

In [17], the authors propose a method of *link reidentification.* That is, they assume that the social network has various link types embedded, and that some of these link types are sensitive. Based on these assumptions, authors propose several methods by which these sensitive link types can be hidden. The general method by which they hide links is by either random elimination or by link aggregation. Instead of attempting to identify sensitive links between individuals, we attempt to identify sensitive traits of individuals by using a graph that initially has a full listing of friendship links. Also, instead of random elimination of links between nodes, we develop an heuristic for removing those links between individuals that will reduce the accuracy of our classifiers the most.

In [3], Gross and Acquisti examine specific usage instances at Carnegie Mellon. They also note potential attacks, such as node re-identification or stalking, that easily accessible data on Facebook could assist with. They further note that while privacy controls may exist on the user's end of the social networking site, many individuals do not take advantage of this tool. This finding coincides very well with the amount of data that we were able to crawl using a very simple

Name of value	**Variable**
Node numbered i in the graph	n_i
A single detail j	D_j
All details of person n_i	$D_{*,i}$
Detail j of person n_i	$d_{j,i}$
Friendship link between person n_i and n_k	$F_{i,k}$
The number of nodes with detail D_j	$\|D_j\|$
The weight of detail D_i	W_i
The weight of a friend link from n_i to n_j	$W_{i,j}$

Table 1: Common Notations Used in the Paper

crawler on a Facebook network. We extend on their work by experimentally examining the accuracy of some types of the Demographic Re-identification that they propose before and after santitization.

The Facebook platform's data has been considered in some other research as well. In [7], authors crawl Facebook's data and analyze usage trends among Facebook users, employing both profile postings and survey information. However, their paper focuses mostly on faults inside the Facebook platform. They do not discuss attempting to learn unrevealed traits of Facebook users, and do no analysis of the details of Facebook users. Their crawl consisted of around 70,000 Facebook accounts.

The area of link based classification is well studied. In [11], authors compare various methods of link based classification including loopy belief propagation, mean field relaxation labeling, and iterative classification. However, their comparisons do not consider ways to prevent link based classification. Belief propagation as a means of classification is presented in [16]. In [13], authors present an alternative classification method where they build on Markov Networks. However, none of these papers consider ways to combat their classification methods.

In [18], Zheleva and Getoor attempt to predict the private attributes of users in four real-world datasets: Facebook, Flickr, Dogster, and BibSonomy. In addition to using general relational classification, they introduce a group-based classification by taking advantage of specific types of attributes in each of the social networks. However, their work does not attempt to sanitize the data; it only reveals the problems we also describe herein.

Finally, in [8], we do preliminary work on the effectiveness of our Details, Links and Average classifiers and examine their effectiveness after removing some details from the graph. Here, we expand further by evaluating their effectiveness after removing details and links.

3 Learning Methods on Social Networks

We model a social network as an undirected graph, G = (V, E), where each node represents a person in the graph, and each link represents a friendship.

Each node n_i in the graph, has a set of details, $\{D_{1,i}, \ldots, D_{N,i}\}$. Each of these details is itself a set consisting of zero or more detail values. That is, suppose that we have two details: Hometown and Activities, which may be referred to as D_1 and D_2. Obviously Hometown is a binary attribute – one may only be born in one place, but a user also has a decision in whether to list it or not. Conversely, Activities could be a multivalued attribute – ”Programming, Video Games, and Reading”, for instance. In the facebook dataset that we use, these multivalued attributes are comma-separated. For clarity, we refer to a Detail as the actual category, say Activities. We represent the concept of *detail values* as the 2-tuple (Detail, expressed value). We further define a set of private details I, where any detail is private if $D_j \in I$. Consider the following illustrative example:

$$\begin{align}
I &= \{\text{political affiliation, religion}\} \tag{1}\\
N_1 &= (\text{Jane Doe}) \tag{2}\\
N_2 &= (\text{John Smith}) \tag{3}\\
D_1 &= (\text{Activities}) \tag{4}\\
D_{1,2} &= \{\text{John Smith's Activities }\} \tag{5}\\
d_{1,2} &= \{Activities, reading\} \tag{6}\\
F_{1,2} &\in E \tag{7}\\
F_{2,1} &\in E \tag{8}
\end{align}$$

That is, we define two details to be private, a person’s political affiliation and their religion (1). Then, say we have two people, named Jane Doe and John Smith respectively (2, 3). There is a single specified Activity detail (4) and John Smith has specified that one of the activities he enjoys is reading (6). Also, John and Jane are friends. Note that because our graph is undirected, examples 7 and 8 are interchangeable, and only one is actually recorded.

In order to evaluate the effect changing a person’s traits has on their privacy, we needed to first create a learning method that could predict a person’s private traits (for the sake of example, we assume that political affiliation is a private trait). For our purposes, we first attempted to use an SVM learning algorithm on the data, but the sheer size of the details involved makes this method less efficient for our purposes.[1] Since our goal is to understand the feasibility of possible inference attacks and the effectiveness of various sanitization techniques combating against those attacks, we initially used a simple Naïve Bayes classifier. Using Naïve Bayes as our learning algorithm allowed us to easily scale our implementation to the large size and diverseness of the Facebook dataset. It also has the added advantage of allowing simple selection techniques to remove detail and link information when trying to hide the class of a network node.

[1] Please see [6] for discussion of the SVM learning tool that we tried to use.

3.1 Naïve Bayes Classification

Naïve Bayes is a classifier that uses Bayes' Theorem to classify objects. Given a node, n_i, with m details and p classes to choose from, $C_1, \ldots, C_p$, Naïve Bayes determines which class C_x is more likely under the assumption that traits are independent. That is,

$$\begin{aligned} & argmax_x[P(C_x|D_{1,i} \ldots D_{m,i})] = \\ & argmax_x \left[\frac{P(C_x) \times P(D_{1,i}|C_x) \times \ldots \times P(D_{m,i}|C_x)}{P(D_{1,i}, \ldots, D_{m,i})} \right] \end{aligned} \tag{9}$$

Because $P(D_{1,i}, \ldots, D_(m,i))$ is a positive constant over all possible classes for any given user n_i, this factor becomes irrelevant when probabilities are compared. This reduces our more difficult original problem of $P(C_x|D_{1,i} \ldots D_{m,i})$ to the question of $P(D_{a,i}|C)$, for all $0 \leq a \leq m$.

3.2 Naïve Bayes on Friendship Links

Consider the problem of determining the class trait of person n_i given their friendship links using a Naïve Bayes model. That is, of calculating $P(C_x|F_{i,1}, \ldots, F_{i,m})$. Because there are relatively few people in the training set that have a friendship link to n_i, the calculations for $P(C_x|F_{i,j})$ become extremely inaccurate. Instead, we decompose this relationship. Rather than having a link from person n_i to n_j, we instead consider the probability of having a link from n_i to someone with n_j's traits. Thus,

$$\begin{aligned} P(C_x|F_{i,j}) &\approx P(C_x|L_1, L_2, \ldots, L_M) \\ &\approx \frac{P(C_x) \times P(L_1|C_x) \times \ldots \times P(L_m|C_x)}{P(L_1, \ldots, L_m)} \end{aligned} \tag{10}$$

where L_n represents a link to someone with detail D_n.

3.3 Weighing Friendships

There is one last step to calculating $P(C_i|F_aF_bF_c)$. Just like details can have weights, so can friendship links. In the specific case of social networks, two friends can be anything from acquaintances to close friends. While there are many ways to weigh friendship links, the method we used is very easy to calculate and is based on the assumption that the more traits two people are known to share, the more unknown traits they are likely to share. This gives the following formula for $W_{A,B}$, which represents the weight of a friendship link from n_A to node n_B:

$$W_{A,B} = \frac{|(D_{1,a}, ..., T_{N,a}) \cap (T_{1,b}, ..., T_{N,b})|}{(|D_{*,a}|)} \tag{11}$$

Equation 11 calculates the total number of traits n_a and n_b share divided by the number of traits of n_a.

Note that the weight of a friendship link is not the same for both people on each side of a friendship link. In other words, $W_{B,A} \neq W_{A,B}$. The final formula for person i becomes the following, where NOMR represents a normalization constant and $P(C_i|F_a)$ is calculated by equation 10.

$$\rho(C_i, F_aF_b...F_z) = \frac{P(C_i|F_a) * W_{a,I} + ... + P(C_i|F_z) * W_{z,I}}{\text{NOMR}} \tag{12}$$

The value $\rho(C_i, F_aF_b...F_z)$ is used as our approximation to $P(C_i|F_aF_b...F_z)$

3.4 Collective Inference

Collective Inference is a method of classifying social network data using a combination of node details and connecting links in the social graph. Each of these classifiers consists of three components: Local classifier, relational classifier, and Collective Inference algorithm.

Local classifiers are a type of learning method that is applied in the initial step of collective inference. Typically, it is a classification technique that examines details of a node and constructs a classification scheme based on the details that it finds there. For instance, the Naïve Bayes classifier we discussed previously is a standard example of Bayes classification. This classifier builds a model based on the details of nodes in the training set. It then applies this model to nodes in the testing set to classify them.

The relational classifier is a separate type of learning algorithm that looks at the link structure of the graph, and uses the labels of nodes in the training set to develop a model which it uses to classify the nodes in the test set. Specifically, in [10], Macskassy and Provost examine four relational classifiers: Class-Distribution Relational Neighbor (cdRN), Weighted-Vote Relational Neighbor (wvRN), Network-only Bayes Classifier(nBC), and Network-only Link-based Classification (nLB).

The cdRN classifier begins by determining a reference vector for each class. That is, for each class, C_x, cdRN develops a vector RV_x which is a description of what a node that is of type C_x tends to connect to. Specifically, $RV_x(a)$ is an average value for how often a node of class C_x has a link to a node of class C_a. To classify node n_i, the algorithm builds a class vector, CV_i, where $CV_i(a)$ is a count of how often n_i has a link to a node of class C_a. The class probabilities are calculated by comparing CV_i to RV_x for all classes C_x.

The nBC classifier uses Bayes Theorem to classify based only on the link structure of a node. That is, it defines

$$\begin{aligned} P(n_i = C_x|\mathcal{N}_\rangle) &= \frac{P(\mathcal{N}_i|n_i = C_x) \times P(n_i = C_x)}{P(\mathcal{N}_i)} \\ &= \prod_{n_j \in \mathcal{N}_i} \frac{P(n_j = C_a|n_i = C_x) \times P(n_i = C_x)}{P(n_j)} \end{aligned}$$

and then uses these probabilities to classify n_i.

The nLB classifier collects the labels of the neighboring nodes and by means of logistic regression, uses these vectors to build a model.

In the wvRN relational classifier, to classify a node n_i, each of its neighbors, n_j, is given a weight. The probability of n_i being in class C_x is the weighted mean of the class probabilities of n_i's neighbors. That is,

$$P(n_i = C_x|\mathcal{N}_i) = \frac{1}{Z} \sum_{n_j \in \mathcal{N}_i} [w_{i,j} \times P(n_j = C_x)]$$

where $\mathcal{N}_i$ is the set of neighbors of n_i.

As may be obvious, there are problems with each of the methods described above. Local classifiers consider only the details of the node it is classifying. Conversely, relational classifiers consider only the link structure of a node. Specifically, a major problem with relational classifiers is that while we may cleverly divide fully labeled test sets so that we ensure every node is connected to at least one node in the training set, real-world data may not satisfy this strict requirement. If this requirement is not met, then relational classification will be unable to classify nodes which have no neighbors in the training set. Collective Inference attempts to make up for these deficiencies by using both local and relational classifiers in a precise manner to attempt to increase the classification accuracy of nodes in the network. By using a local classifier in the first iteration, collective inference ensures that every node will have an initial probabilistic classification, referred to as a prior. The algorithm then uses a relational classifier to re-classify nodes. At each of these steps $i > 2$, the relational classifier uses the fully-labeled graph from step $i-1$ to classify each node in the graph.

The Collective Inference method also controls the length of time the algorithm runs. Some algorithms specify a number of iterations to run, while others converge after a general length of time. We choose to use Relaxation Labeling as described in [10]: a method which retains the uncertainty of our classified labels. That is, at each step i, the algorithm uses the probability estimates, not a single classified label, from step $i-1$ to calculate new probability estimates. Further, to account for the possibility that there may not be a convergence, there is a decay rate, called α set to 0.99 that discounts the weight of each subsequent iteration compared to the previous iterations. We chose to use Relaxation labeling because in the experiments conducted by Macskassy and Provost[10], Relaxation Labeling tended to be the best of the three collective inference methods.

Each of these classifiers, including a Relaxation Labeling implementation, is included in NetKit-SRL[2]. As such, after we perform our sanitization techniques, we allow NetKit to classify the nodes to examine the effectiveness of our approaches.

[2] Available at: http://netkit-srl.sourceforge.net/

4 Data Gathering

We wrote a program to crawl the Facebook network to gather data for our research. Written in Java 1.6, the crawler loads a profile, parses the details out of the HTML, and stores the details inside a MySQL database. Then, the crawler loads all friends of the current profile and stores the friends inside the database both as friendship links and as possible profiles to later crawl.

Because of the sheer size of Facebook's social network, the crawler was limited to only crawling profiles inside the Dallas/Forth Worth (DFW) network. This means that if two people share a common friend that is outside the DFW network, this is not reflected inside the database. Also, some people have enabled privacy restrictions on their profile which prevented the crawler from seeing their profile details. [3] The total time for the crawl was seven days.

Because the data inside a Facebook profile is free form text, it is critical that the input is normalized. For example, favorite books of "Bible" and "The Bible" should be considered the same detail. Often there are spelling mistakes or variations on the same noun.

The normalization method we use is based upon a Porter Stemmer presented in [14]. To normalize a detail, it was broken into words and each word was stemmed with a Porter Stemmer then recombined. Two details that normalized to the same value were considered the same for the purposes of the learning algorithm.

4.1 Data Overview

Table 2 gives an overview of the crawl's data. Our total crawl resulted in over 167,000 profiles, almost 4.5 million profile details, and over 3 million friendship links. In the graph representation, we had one large central group of connected nodes that had a maximum path length of 16. Only 22 of the collected users were not inside this group. As shown in table 3, a crawl of the Dallas regional network resulted in more conservatives than liberals, but not by a very large margin.

Common knowledge leads us to expect a small diameter in social networks [15]. To reconcile this fact with the empirical results of a 16 degree diameter in the graph, note that, although popular, not every person in society has a Facebook account and even those that do still do not have friendship links to every person they know.

5 Hiding Private Information

In this section, we first discuss the effectiveness of our modified Naïve Bayes classifier when compared to a traditional Naïve Bayes classifier. Next, we discuss how to reduce the effectiveness of our classifiers by manipulating the detail and

[3] The default privacy setting for Facebook users is to have all profile information revealed to others inside their network.

Diameter of the largest component	16
Number of nodes in the graph	167,390
Number of friendship links in the graph	3,342,009
Total number of listed details in the graph	4,493,436
Total number of unique details in the graph	110,407
Number of components in the graph	18

Table 2: General information about the data

Probability of being Liberal	.45
Probability of being Conservative	.55

Table 3: Odds of being Liberal or Conservative

link information, and then give an analysis of the experimental results of tests done on our real-world dataset.

5.1 Predicting Private details

In our experiments, we implemented four algorithms to predict the political affiliation of each user. The first algorithm is called "Details Only." This algorithm uses Equation 9 to predict political affiliation and ignores friendship links. The second algorithm is called "Links Only." This algorithm uses Equation 12 to predict political affiliation using friendship links and does not consider the details of a person. The third algorithm is called "Average." The Average algorithm predicts a node's class value based on the following equation:

$$P_A(N_i = C_a) \quad = \quad 0.5 * P_D(n_i = C_a) + 0.5 * P_L(n_i = C_a)$$

where P_D and P_L are the numerical probabilities assigned by the Details Only and Links Only algorithms, respectively. The final algorithm is a traditional Naïve Bayes classifier, which we used as a basis of comparison for our proposed algorithms.

5.2 Manipulating details

Clearly, details can be manipulated in three ways: adding details to nodes, modifying existing details and removing details from nodes.

The goal in the first case is to add details that may prevent learning algorithms from being able to infer a person's private details. In the second case, the goal is to prevent leakage of "accurate" information by modifying profile details (e.g., anonymization techniques). In the third case, the goal is to remove those details that most help a learning algorithm to predict a person's private details.

In the context of a social network, removing details does not introduce any misleading information. This follows from the implied nature of listed details inside a social network. If a detail is missing, it simply implies that the person failed to mention that detail. A missing detail does not imply that the detail does not describe the person. However, if a detail is mentioned, then it is implied that the detail does indeed describe the person. Unlike anonymization techniques such as k-anonymity, removing details could be easily done by each individual profile owner.

For instance, suppose there is a profile for a person named John Smith. On his profile, he specifies that he enjoys reading. He does not specify that he enjoys hiking. Because we specify that he likes reading, we know that this is factual information. However, when he does not specify a like for hiking, if we add the detail that John likes hiking, then this may be incorrect; he may, in fact, not like hiking. Conversely, we cannot add that he dislikes hiking for a similar reason. Clearly, John can delete the information about hiking from his profile easily.

Because of the reasons stated above, as a starting point, we focused on trying to sanitize a social network by removing details rather than by adding false details or modifying existing details. We leave the exploration of other sanitization techniques as a future work. [4]

The first question we need to deal with is how to choose which details to remove. Using Naïve Bayes as a benchmark makes the process of choosing which details to remove very simple.

Assume a person has the class value C_2 out of the set of classes C, and this person has public details $D_{*,x}$.

$$argmax_y[P(C_y) * P(D_{1,x}|C_y) * ... * P(D_{m,x}|C_y)] \tag{13}$$

Equation 13 identifies the learned class. Because we globally remove the most representative traits, we are able to find this based off of the equation

$$argmax_y[\forall C_x \in C : P(D_y|C_x)] \tag{14}$$

This allows us to find the single detail that be the most highly indicative of a class and remove it.

5.3 Manipulating Link Information

Links can be manipulated in the same way details can. For the same reasons given in section 5.2, we choose to evaluate the effects of privacy on removing friendship links instead of adding fake links.

Consider equation 12 for determining detail type using friendship links. Also assume that there are two classes for a node, and the true class is C_1. We want to remove links that will increase the likelihood of the node being in class C_2. Please note that we define a node to be in class C_2 if formula 15 is positive.

$$d = \rho(C_2, F_aF_b...F_z) - \rho(C_1, F_aF_b...F_z) \tag{15}$$

[4] Please also note that many anonymization methods such as k-anonymity [12] do not introduce any false information to data set.

Therefore, we would like to maximize the value of d as much as possible by removing links.

Define d_i as the new value for formula 15 if we remove friendship link i. We can compute d_i as

$$\begin{aligned} d_i &= \left(\rho(C_2, F_aF_b...F_z) - \frac{P(C_2|F_{j,i}) * W_{j,i}}{\text{NOMR}}\right) \\ &- \left(\rho(C_1, F_aF_b...F_z) - \frac{P(C_1|F_{j,i}) * W_{j,i}}{\text{NOMR}}\right) \\ &= d + \frac{(P(C_1|F_{j,i}) - P(C_2|F_{j,i})) * W_{j,i}}{\text{NOMR}} \end{aligned} \tag{16}$$

Because d and NORM are constants for all d_i, the best choice for i that maximizes d_i becomes one that maximizes $M_i = P(C_1|F_{j,i}) - P(C_2|F_{j,i})) * W_{j,i}$.

In our experiments, we order the links for each node based on the M_i values. When we remove links, we remove those with the greatest M_i values.

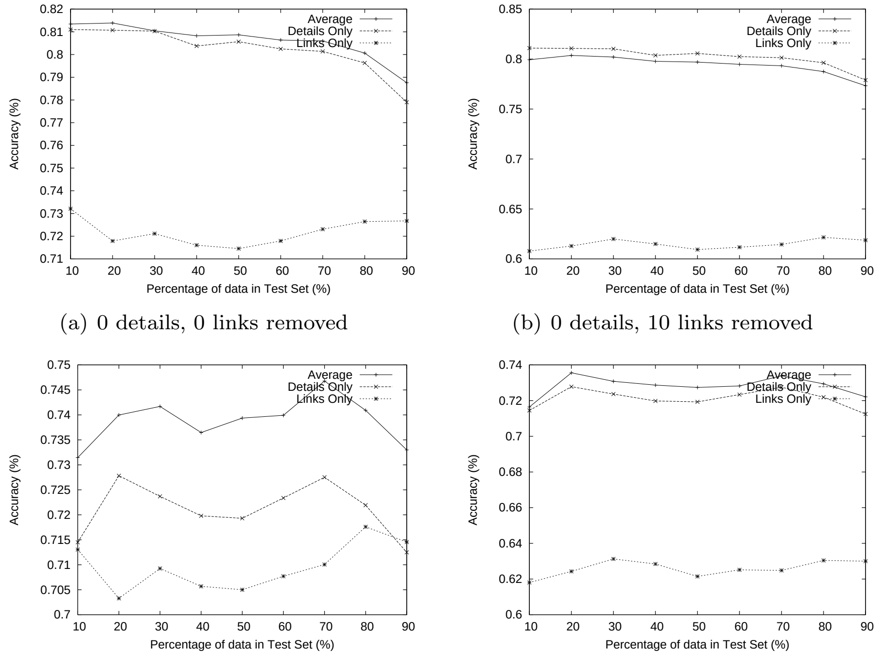
6 Experiments

We begin by pruning the total graph of 160,000 nodes down to only those nodes for which we have a recorded political affiliation. This reduces our overall set size to 35,000 nodes. Then, we use the ideas from Section 5.3 to remove the 10 most telling links from every node in the graph. This is done by use of Equation 16 to determine which K links connect a node to those nodes that are most similar, and delete those K links. Unlike removing details, which is done globally, removal of links is done locally. We believe that this is a reasonable method of sanitization because the data is sanitized and then released to an external party. Furthermore, we do not attempt to modify the existing public network. Similarly, we stress that while the data set we use was composed of individuals who had their profile open to the public, these methods should extend to work on the total, mostly-private social network.

We combine the Detail and Link removal methods and then generate test sets with both 10 details and 10 links removed from the graph. We refer to these sets as 0 details, 0 links; 10 details, 0 links; 0 details, 10 links; 10 details, 10 links removed, respectively. Following this, we want to gauge the accuracy of the classifiers for various ratios of labeled vs. unlabeled graphs. To do this, we collect a list of all of the available nodes, as discussed above. We then obtain a random permutation of this list using the Java function built-in to the *Collections* class. Next, we divide the list into a test set and a training set, based on the desired ratio. We focus on multiples of 10 for the accuracy percentages, so we generate sets of 10/90, 20/80, . . . , 90/10. We refer to each set by the percentage of data in the test set. We generate five test sets of each ratio, and run each experiment independently. We then take the average of each of these runs as the overall accuracy for that ratio.

Our results, as shown in Figure 1, indicate that the Average Only algorithm substantially outperformed traditional Nave Bayes and the Links Only algorithm. Additionally, the Average Only algorithm generally performed better than the Details Only algorithm with the exception of the (0 details, 10 links) experiments.

Also, as a verification of expected results, the Details Only classification accuracy only decreased significantly when we removed details from nodes, while the (0 details, *) accuracies are approximately equivalent. Similarly, the Link Only accuracies were mostly affected by the removal of links between nodes, while the (*, 0 links) points of interest are approximately equal. The difference in accuracy between (0 details, 0 links) and (10 details, 0 links) can be accounted for by the weighting portion of the Links Only calculations, which depends on the similarity between two nodes.



(a) 0 details, 0 links removed

(b) 0 details, 10 links removed

(c) 10 details, 0 links removed

(d) 10 details, 10 links removed

Figure 1: Local classifier prediction accuracies by percentage of nodes in test set

Figure 1 shows the results of our classification methods for various labeled node ratios. These results are generally consistent with what the initial results indicated: the Average classifier tends to outperform both the Links Only and the Details Only classifier, with the exception being Figure 1(c) , and in this case its error margin is only approximately 1% higher than that of Details only. Also, the results, with the exception of Figure 1(d) are generally consistent across all tests. The greatest variance occurs when we remove details alone. It may be unexpected that the Links Only classifier has such varied accuracies as a result

# details removed	# links removed	Before	After
10	0	52.78	52.81
0	10	52.75	52.30
10	10	52.72	52.81

Table 4: Gender classification test

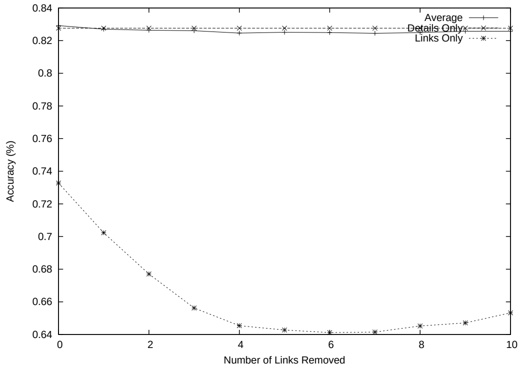


Figure 2: Local Classification accuracy by number of links removed

of removing details, but since our calculation of probabilities for that classifier uses a measure of similarity between people, the removal of details may affect that measure.

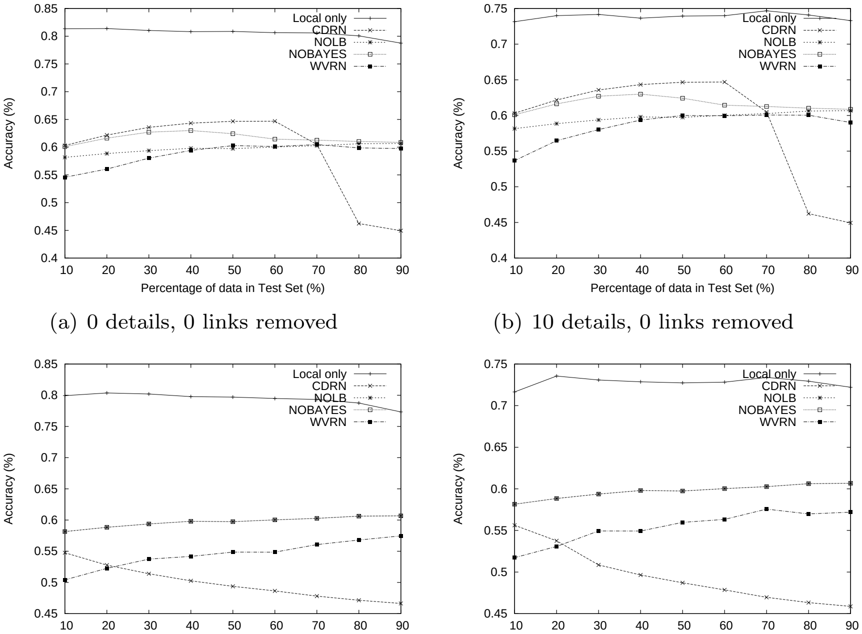
Next, we examine the effects of removing the links. We remove K links from each node, where $K \in [0, 10]$, and again partition the nodes into a test set and training set of equal size. We then test the accuracy of the local classifier on this test set. We repeat this five times and then take the average of each accuracy for the overall accuracy of each classifier after K links are removed. The results of this are shown in Figure 2. For for $K \in [1, 6]$, each link removal steadily decreases the accuracy of the classifier. Removing the seventh classifier has no noticeable effect, and subsequent removals only slightly increase the accuracy of the Links Only classifier. Also, due to space limitations, for the remainder of experiments we show only the results of the Average classifier.

Additionally, because our motivation is to hide private details while still allowing an interested party to be able to infer information about public details, we take each of our data sets and build a simple Naïve Bayes classifier to attempt to determine the gender of a user. The results of these tests are shown in Table 4. As we show, our sanitization approach does reduce the accuracy of inference methods on private data while preserving an interested party's ability to determine details that are not private. In fact, as can be seen from the (10, *) experiments, the gender classification became slightly more accurate after political affiliation-specific traits were removed.

6.1 Relaxation Labeling

We note that in the Facebook data, there are a limited number of ‘groups’ that are highly indicative of an individual’s political affiliation. When removing details, these are the first that are removed. We assume that conducting the collective inference classifiers after removing only one detail may generate results that are specific for the particular detail we classify for. For that reason, we continue to consider only the removal of 0 details and 10 details, the other lowest point on the classification accuracy. We also continue to consider the removal of 0 links and 10 links due to the marginal difference between the $[6, 7]$ region and removing 10 links.

For the experiments using relaxation labeling, we took the same varied ratio sets generated for the local classifiers in Section 6. For each, we store the predictions made by the Details Only, Links Only, and Average classifiers and use those as the priors for the NetKit toolkit. For each of those priors, we test the final accuracy of the cdRN, wvRN, nLB, and nBC classifiers. We do this for each of the five sets generated for each of the four points of interest. We then take the average of their accuracies for the final accuracy.



(a) 0 details, 0 links removed

(b) 10 details, 0 links removed

(c) 0 details, 10 links removed

(d) 10 details, 10 links removed

Figure 3: Prediction accuracy of Relaxation Labeling using the Average local classifier

Figure 3 shows the results of our experiments using Relaxation Labeling. In [10], Macskassy and Provost study the effects of collective inference on four real-world datasets: IMDB, CORA, WebKB, and SEC filings. While they do not discuss the difference in the local classifier and iterative classification steps of their experiments, their experiments indicate that Relaxation Labeling almost always performs better than merely predicting the most frequent class. Generally, it performs at near 80% accuracy, which is an increase of approximately 30% in their datasets. However, in our experiments, Relaxation Labeling typically performed no more than approximately 5% better than predicting the majority class. This is also substantially less accurate than using only our local classifier.

As shown previously, the Average and Details Only local classifiers were most strongly affected by removing details, while the Links Only classifier and all relational classifiers, to various degrees, were most strongly affected by the removal of links. More interestingly, however, was that the fewer nodes that were in the training sets, the better the result of Relaxation Labeling was on most classifiers. The cdRN relational classifier was the single exception to this trend. In any of the experiments where links were removed, cdRN's accuracy only decreased as the percentage of nodes in the test set increased. In the experiments with all links present, cdRN increased in accuracy until 60% of nodes were in the test set, after which its performance drastically declined.

Additionally, if we compare Figures 3(a) and 3(b) and Figures 3(c) and 3(d), we see that while the local classifier's accuracy is directly affected by the removal of details and/or links, this relationship is not shown by using relaxation labeling with the local classifiers as a prior. For each pair of the figures mentioned, the relational classifier portion of the graph remains constant, only the local classifier accuracy changes. From these, we see that the most 'anonymous' graph, meaning the graph structure that has the lowest predictive accuracy, is achieved when we remove both details and links from the graph.

7 Conclusion and Future Work

We addressed various issues related to private information leakage in social networks. For unmodified social network graphs, we show that using details alone, one can predict class values more accurately than using friendship links alone. We further show that using both friendship links and details together gives better predictability than details alone. In addition, we explored the effect of removing traits and links in preventing sensitive information leakage. In the process, we discovered situations in which collective inferencing does not improve on using a simple local classification method to identify nodes. When we combine the results from the collective inference implications with the individual results, we begin to see that removing trait details and friendship links together is the best way to reduce classifier accuracy. This is probably infeasible in maintaining the use of social networks. However, we also show that by removing only traits, we greatly reduce the accuracy of local classifiers, which give us the maximum accuracy that we were able to achieve through any combination of classifiers.

We also draw attention to the difference in our findings regarding collective inference and the findings of other researchers. While their research has generally noticed an increase in classifier accuracy after using collective inference, we notice a sharp decrease in accuracy. This could have extraordinary implications for the use of collective inference in general. While some networks are extremely popular for scientific research (such as Facebook because of its extreme popularity), not all networks can be studied so intensely. If there are specific types of social networks, or particular types of details that are naturally resistant to collective inference attacks, then further work could be done in an attempt to apply these attributes to other details.

We also assumed full use of the graph information when deciding which details to hide. Useful research could be done on how individuals with limited access to the network could pick which traits to hide. Similarly, future work could be conducted in identifying key nodes of the graph structure to see if removing or altering these nodes can decrease information leakage.

Another consideration is that social networks are vibrant, dynamic applications. We ignore temporal problems that may arise, such as those from repeated distributions of sanitized data over time. This would be another area of research that should be conducted.

References

[1] L. Backstrom, C. Dwork, and J. Kleinberg. Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 181–190, New York, NY, USA, 2007. ACM.

[2] Facebook Beacon, 2007. http://blog.facebook.com/blog.php?post=7584397130.

[3] R. Gross, A. Acquisti, and J. H. Heinz. Information revelation and privacy in online social networks. In *WPES '05: Proceedings of the 2005 ACM workshop on Privacy in the electronic society*, pages 71–80, New York, NY, USA, 2005. ACM Press.

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

[5] J. He, W. Chu, and V. Liu. Inferring privacy information from social networks. In Mehrotra, editor, *Proceedings of Intelligence and Security Informatics*, volume LNCS 3975, 2006.

[6] T. Joachims. Training linear SVMs in linear time. In *ACM SIGKDD International Conference On Knowledge Discovery and Data Mining (KDD)*, pages 217 – 226, 2006.

[7] H. Jones and J. H. Soltren. Facebook: Threats to privacy. Technical report, Massachusetts Institute of Technology, 2005.

[8] J. Lindamood, R. Heatherly, M. Kantarcioglu, and B. Thuraisingham. Inferring private information using social network data. In *WWW Poster*, 2009.

[9] K. Liu and E. Terzi. Towards identity anonymization on graphs. In *SIGMOD '08: Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 93–106, New York, NY, USA, 2008. ACM.

[10] S. A. Macskassy and F. Provost. Classification in networked data: A toolkit and a univariate case study. *Journal of Machine Learning Research*, 8:935–983, 2007.

[11] P. Sen and L. Getoor. Link-based classification. Technical Report CS-TR-4858, University of Maryland, February 2007.

[12] L. Sweeney. k-anonymity: a model for protecting privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, (5):557–570, 2002.

[13] B. Tasker, P. Abbeel, and K. Daphne. Discriminative probabilistic models for relational data. In *Proceedings of the 18th Annual Conference on Uncertainty in Artificial Intelligence (UAI-02)*, pages 485–492, San Francisco, CA, 2002. Morgan Kaufmann Publishers.

[14] C. van Rijsbergen, S. Robertson, and M. Porter. New models in probabilistic information retrieval. Technical Report 5587, British Library, 1980.

[15] D. J. Watts and S. H. Strogatz. Collective dynamics of small-world networks. *Nature*, 393(6684):440–442, June 4 1998.

[16] J. Yedidia, W. Freeman, and Y. Weiss. *Exploring Artificial Intelligence in the New Millennium.* Science & Technology Books, 2003.

[17] E. Zheleva and L. Getoor. Preserving the privacy of sensitive relationships in graph data. pages 153–171. 2008.

[18] E. Zheleva and L. Getoor. To join or not to join: the illusion of privacy in social networks with mixed public and private user profiles. In *WWW*, 2009.