Inferring Private Information Using Social Network Data

Jack Lindamood and Murat Kantarcioglu
Computer Science Department
University of Texas at Dallas

July 23, 2008

Abstract

Online social networks, such as Facebook, are increasingly utilized by many users. These networks allow people to publish details about themselves and connect to their friends. Some of the information revealed inside these networks is private and it is possible that corporations could use learning algorithms on the released data to predict undisclosed private information. In this paper, we propose an effective, scalable inference attack for released social networking data to infer undisclosed private information about individuals. We then explore the effectiveness of possible sanitization techniques that can be used to combat such an inference attack.

1 Introduction

Social networks are platforms that allow people to publish details about themselves and connect to other members of the network through friendship links. Recently, the popularity of such on-line social networks is increasing significantly. For example, Facebook now claims to have more than 80 million active users. [1] The existence of on-line social networks that can be easily mined for various reasons creates both interesting opportunities and challenges. For example, social network data could be used for marketing products to the right customers. At the same time, privacy concerns can prevent such efforts in practice [2]. Therefore, for future social network applications, privacy emerges as an important concern. In social network data mining, we analyze two cases in which users inside a social network may want to protect their privacy.

- **Privacy After Data Release:** The first privacy concern occurs when a social networking site releases the social networking data to the public, either for research purposes or through an open social networking API. In

[1] http://www.facebook.com/press/info.php?statistics

this case, using some available information, it may be possible to identify individuals by either link structures[1] or some other information. The worst case scenario of this is the AOL search data scandal of 2006, where AOL released the search results of 650,000 users over a 3 month period for research purposes. These search results contained social security numbers, addresses, and pornographic search queries that could sometimes be tied back to specific individuals. This action resulted in the resignation of AOL's CTO, a lawsuit, and a position on CNNMoney's list of "101 Dumbest Moments in Business." While releasing social networking data would be of great use to researchers, it would need to be done in a way that can guarantee the privacy of the individuals that consent to the release.

- **Private Information Leakage:** Another aspect of privacy is how the people inside a social network can protect themselves from the prying eyes of the company that runs the social network itself. Social networking sites can use learning algorithms to detect private traits a person chooses not to reveal. For example, if a person chooses not to reveal their political affiliation on Facebook because they consider it private but at the same time join a group titled "1,000,000 strong for Barack Obama", they have in essence revealed their political affiliation through their membership in this group. This brings up the question of how can people obfuscate or hide their details in order to keep sensitive information private and still get the benefits of being a part of on-line social networks.

In this paper, we focus on the problem of individual private information leakage due to being part of an on-line social network. More specifically, we explore how the on-line 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 alternatives on preventing such private information leakage.

1.1 Our contributions

To our knowledge this is the first paper that discusses the problem of inferring private traits using social network data and possible sanitization approaches to prevent such inference. First, we present a modification of Naive Bayes classification that is suitable for classifying large amount of social network data. Our modified Naive Bayes algorithm predicts privacy sensitive trait information using both node traits and link structure. We compare the accuracy of our learning method based on link structure against the accuracy of our learning method based on node traits. In order to improve privacy, we modify both trait (e.g., deleting some information from a user's on-line profile) and link details (e.g., deleting links between friends) and explore the effect they have on combating our inference attack.

2 Previous 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. In any case, 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], authors consider perturbing network data in order to preserve privacy. While their method considers graph structure, it ignores any extra details or traits that a node inside the social network may possess.

Our research is not the first to specifically consider the Facebook platform's data. 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.

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. They also do not distinguish between private information that a person would want to hide and public information a person would not mind sharing.

The area of link based classification is well studied. In [8], 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 [13]. In [10], authors present an alternative classification method where they build on Markov Networks. However, none of these papers consider ways to combat their classification methods.

3 Learning Methods on Social Networks

The form of a social network that we will analyze in this paper follows the Facebook model. A social network is an undirected graph $G = (V, E)$ where each node represents a person in the graph and each link a friendship. Each node N_i in the graph has a set of traits, $T_1^i ... T_N^i$. Each trait is a (tn, tv) pair of strings, where tn represents a trait name and tv represents a trait value. There is a set of private trait names I where any trait (tn, tv) is private if $tn \in I$. Every trait that is not private is considered public. Two nodes i and j may be connected by an undirected edge $F_{i,j}$ which represents a friendship link between

Name of value	**Variable**
Node numbered i in the graph	N_i
A single trait	T_j
Trait j of person N_i	T_j^i
Unknown class of the object	C_N
Friendship link between person N_i and N_j	$F_{i,j}$
The number of nodes with trait T_i	$\|T_i\|$
The number of nodes with trait T_i and T_j	$\|T_i \cap T_j\|$
The weight of trait T_i	W_i
Indicator variable to represent a link to someone with trait T_i	L_i
The weight of a friend link to node j from node i	W_j^i

Table 1: Common Notations Used in the Paper

nodes i and j.

$$T_1^5 = (\text{name}, \text{John Doe}) \tag{1}$$

$$T_2^5 = (\text{political affiliation}, \text{liberal}) \tag{2}$$

$$I = \{\text{political affiliation}, \text{religion}\} \tag{3}$$

$$F_{5,1} \in E \tag{4}$$

$$F_{5,2} \in E \tag{5}$$

As an illustrative example, consider 1 through 5, which is a subset of a sample network. It shows that person 5 in the graph is John Doe, who is a liberal. The fact that John Doe is a liberal is considered private information while the fact that his name is John Doe is not considered private. John Doe has two friends in this graph.

3.1 Learning method selection

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 a SVM learning algorithm on the data, but the sheer size of the details involved make this method impractical. [2] The learning method we finally used is based on Naive Bayes. Using Naive Bayes as our learning algorithm allows us to easily scale our implementation to the large size and diverseness of the Facebook dataset. It also has the added advantage of allowing $O(N)$ selection of which traits to remove when trying to hide the class of a network node.

[2] The program we tried to use is referenced at [6]

3.1.1 Naive Bayes Classification

Naive Bayes is a classifier that uses Bayes Theorem to classify objects. Given an object with N traits and two classes to pick from, C_1 and C_2, Naive Bayes determines which class, C_1 or C_2, is more likely under the assumption that traits are independent.

$$\begin{aligned} P(C_1|T_1...T_N) &= \frac{P(C_1) * P(T_1|C_1) * ... * P(T_N|C_1)}{P(T_1, ..., T_N)} \\ P(C_2|T_1...T_N) &= \frac{P(C_2) * P(T_1|C_2) * ... * P(T_N|C_2)}{P(T_1, ..., T_N)} \end{aligned}$$

Because $P(T_1, ..., T_N)$ is a positive constant for both equations, it will cancel out when the equations are compared. This modification reduces the much larger problem of $P(C_1|T_1...T_N)$ into the simpler problem of $P(T_1|C_1)$.

3.2 Our Modification of Naive Bayes Classification

As an example, let us assume that we want to determine $P(A|B)$ and that one has as training data every possible object in the universe of objects that has trait B. If this were the case, then $P(A|B)$ could be estimated precisely from the training data using equation 6.

$$P(A|B) = \frac{|A \cap B|}{|B|} \tag{6}$$

However, in real world data one does not possess every possible object that has trait B. The further $|B|$ is from the true count of the number of objects with trait B in the universe, the less one can trust equation 6.

Since training data will likely contain many hundreds of thousands of examples, one can confidently calculate $P(A)$, but because $|A|$ may be small, one cannot use the equation 6 for $P(A|B)$ directly. We have based our modification upon the idea that if one can trust the information given by trait A then $P(A|B)$ is very close to equation 6. However, if one cannot trust the information given by trait A, $P(A|B)$ is assumed to be close to $P(A)$.

Consider the following equation:

$$\begin{aligned} P_1 &= P(A) \\ P_2 &= \frac{|A \cap B|}{|B|} \\ W_A &= \frac{\log(|A|)}{\log(|Q|)} \\ \phi(A, B) &= P_2 * W_A + P_1 * (1 - W_A) \end{aligned} \tag{7}$$

Here we use $\phi(A, B)$ as our approximation of $P(A|B)$. The value W_A is the weight of trait A. Trait Q is whichever trait is the most popular. Using the Facebook data as an example, Q would be the trait "'Bible"' when asking about "Favorite Books". Empirical results showed that this method performed much better than the usual formula for $P(A|B)$ that uses a Laplace correction. (Please see section 5 for more discussion.)

This approximation for $P(A|B)$ has the following desirable qualities:

- It is guaranteed to return results in (0...1]
- It acknowledges that if comparing two traits A and B, where $|A| = 1$ and $|B| = 10$, trait B is much more trustworthy than trait A. Meanwhile, if $|A| = 100$ and $|B| = 109$, then trait A and trait B are almost equally trustworthy.

This formula works extremely well in worlds where there are many traits and most of the traits are mentioned very rarely, as is the case for user-inputted data. As a result, we use equation 7 to calculate $P(A|B)$ when applying Naive Bayes.

3.3 Naive Bayes on Friendship Links

Consider the problem of determining a person's class trait given their friendship links using a Naive Bayes model, or calculating $P(C_1|F_{i,1}...F_{i,n})$. Because there will be very few people in the training set that have a friendship link to node N_i, one cannot calculate $P(C_1|F_{i,j})$ using friendship links alone. Instead, it is best to decompose the event of having a link to N_j into having a link to someone with N_j's traits. This becomes the following:

$$\begin{aligned} P(C_1|F_{i,j}) &\approx P(C_1|L_1L_2L_3...L_N) \\ &\approx \frac{P(C_1) * P(L_1|C_1) * .. * P(L_N|C_1)}{P(L_1...L_N)} \end{aligned} \tag{8}$$

Here L_N represents a link to someone with trait T_N. This just leaves the problem of calculating $P(L_i|C_j)$. This can be calculated and weighted in the same manner as equation 7.

$$\begin{aligned} P(L_i) &= \frac{|L_i|}{|E|} \\ P_2 &= \frac{|C_j \cap L_i|}{|A|} \\ W_i &= \frac{\log(|L_i|)}{\log(|Q|)} \\ \Gamma(L_i, C_j) &= P_2 * W_i + P_1 * (1 - W_i) \end{aligned} \tag{9}$$

Here $\Gamma(L_i, C_j)$ is used as our approximation for $P(L_i|C_j)$. As in equation 7, Q in this case is the trait that is linked to the most, or the largest value for L_k considering all L and variable E represents the size of the edge set.

3.4 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_B^A, which represents the weight of a friendship link from node A to node B:

$$W_B^A = \frac{|(T_1^A, ..., T_N^A) \cap (T_1^B, ..., T_N^B)|}{(|A|)} \tag{10}$$

Equation 10 calculates the number of traits nodes A and B share divided by the number of traits of node 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 8. The value $\rho(C_i, F_aF_b...F_z)$ is used as our approximation to $P(C_i|F_aF_b...F_z)$

$$\rho(C_i, F_aF_b...F_z) = \frac{P(C_i|F_a) * W_a^I + ... + P(C_i|F_z) * W_z^I}{\text{NOMR}} \tag{11}$$

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 and 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 trait. Often there are spelling mistakes or variations on the same noun.

The normalization method we use is based upon a Porter Stemmer presented in [11]. To normalize a trait, it was broken into words and each word was stemmed with a Porter Stemmer then recombined. Two traits that normalized

[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 traits in the graph	4,493,436
Total number of unique traits 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

to the same value were considered the same trait 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. All but 22 of the people crawled were inside one, large component of diameter 16. 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.

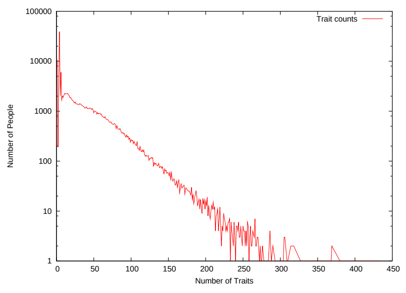
Common knowledge is to expect a small diameter in social networks [12]. To reconcile this fact with the empirical results of a 16 degree diameter in the graph, note that, although popular, not every person in society has a Facebook account and even those that do still do not have friendship links to every person they know.

The graph is very sparse and the average number of traits listed per node is very small compared to the total number of unique traits. Because of this situation, general formulas for $P(A|B)$ produce inaccurate results. This is what prompted us to consider equation 7.

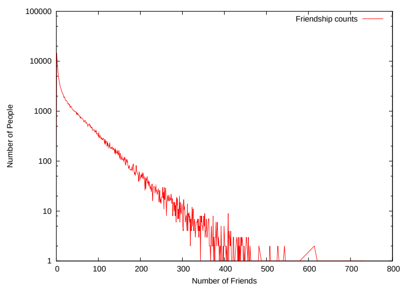
Figures 1(a) and 1(b) show how many people have the specified number of traits or friendship links. For example, the point $(x = 4, y = 6100)$ for figure 1(b) means that 6,100 people have 4 friendship links. The point $(x = 4, y = 38979)$ for figure 1(a) means that 38,979 people have 4 listed details. It is important to note that both figures have a logarithmic Y scale. This shows that the vast majority of nodes in the graph have few traits and friendship links.

Figure 1(c) shows the number of profiles where the most popular activities were listed. For example, the point $(x = 1, y = 2373)$ means that the most popular trait of the "Activities" type was listed inside 2,373 profiles while the point $(x = 4, 1482)$ means that the fourth most popular activity was listed 1,482 times.

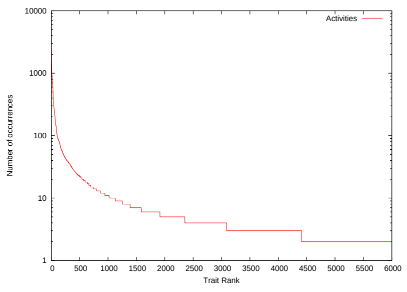
Figure 1(c) is actually only part of the graph. The X axis can extend to the point 94,146 but was cut off at 6,000 for readability purposes. Combined with



(a) Number of traits per any person



(b) Number of friendship links per any person



(c) Frequency of the activities trait

Figure 1: Distribution Information for Various Social Network Attributes

the fact that the Y axis has a a logarithmic Y scale, this shows that the few most popular traits are represented strongly in the data set.

5 Hiding Private Information

In this section, we first discuss the effectiveness of our modified naive Bayes classifier on predicting the private traits. Later on, we discuss how to reduce the effectiveness of our naive Bayes classifier by manipulating either trait or link information.

5.1 Predicting Private Traits

In our experiments, we used three algorithms to predict the political affiliation of each user. The first algorithm is called "Details only". This algorithm uses equation 7 to predict political affiliation and ignores friendship links. The second algorithm is called "Links only". This algorithm uses equation 11 to predict political affiliation using friendship links and does not consider the trait details of a person. The third algorithm is called "Average". This algorithm calculates the equally weighted average value of equation 7 and 11 to predict political affiliation.

We used 10-fold cross validation to validate the effectiveness of the three algorithms discussed above. [4] During each validation run 10% of the nodes that have a known political affiliation were modified to have their political affiliation removed. Afterwords, we evaluate the nodes to see if the original political affiliation can be detected. All graphs are fitted to a best fit polynomial using the gnuplot [3] "fit" command.

First, we explored the relationship between the number of friends a person has versus the effectiveness of the learning algorithms. Figure 2 shows that after having twenty friendship links, more friends did not make trait prediction easier. Furthermore, "Average" algorithm has approximately 80% prediction accuracy. One interesting thing to note about the figure 2 is that after having more than 70-80 friends, there is a slight drop in the accuracy of the "link" only prediction method. This result indicates that after a certain point, having more friends do not increase the prediction accuracy.

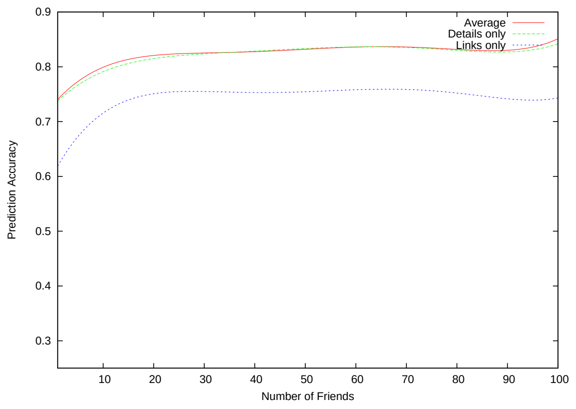


Figure 2: Unmodified private trait predictability per friendship links

Second, we explored the relationship between the number of traits a person is revealing on his or her profile versus the prediction accuracies of the learning algorithms. Again revealing ten to twenty traits on ones profile is good enough for accurately predicting political affiliation.

Overall, the algorithm "Average", which considered both friendship links and trait details, performed about 1% better than the algorithm "Details only" for unmodified graphs. It also performed better in every anonymized instance (see the remainder of this section for more details). This result suggests that it is better to consider both trait details and friendship links together when considering data that may have noise, or may be tainted.

In addition, we tested the effect of using Laplace correction versus our

[4] Unless otherwise stated all experiments used 10-fold cross validation.

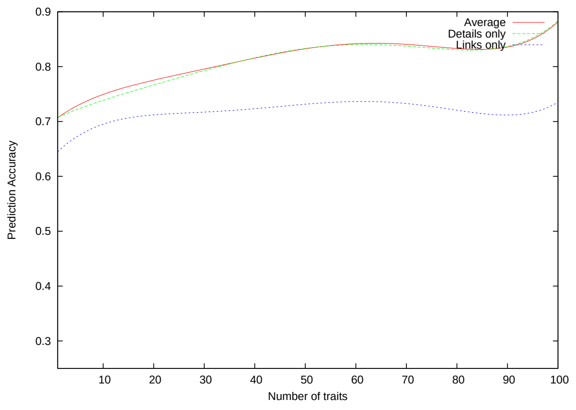


Figure 3: Unmodified private trait predictability per profile details

method for estimating the conditional probabilities. As figure 4 suggests, on average, our method of calculating conditional probabilities produced more accurate classifiers than Laplace correction.

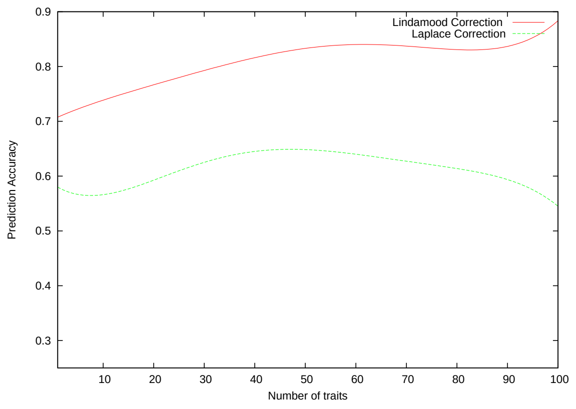


Figure 4: Our learning method VS using Laplace Correction

5.2 Manipulating traits

First line of defense against any classifier that predicts sensitive trait value is to manipulate publicly available trait information. Clearly, traits can be manipulated in two ways:

- Adding traits to nodes
- Removing traits from nodes

The goal in the first case is to add traits that may prevent learning algorithms from being able to pinpoint a person's true value for their important traits. In the second case, the goal is to remove the traits that most help learning algorithm to predict a person's private trait.

Considering the context of a social network, removing traits will not introduce any misleading information. This follows from the implied nature of listed traits inside a social network. If a trait is missing, it simply implies that the person failed to mention that trait. A missing trait does not imply that the trait does not describe the person. However, if a trait is mentioned, then it is implied that the trait does indeed describe the person.

As a real world example, if someone mentions on their profile that they like pizza, then you interpret that as meaning the person really does like pizza. However, if that same person did not mention spaghetti at all, you would not take that to mean the person does not like spaghetti.

Because of this, we focused on trying to anonymize a network by removing traits rather than by adding false traits. [5]

5.2.1 Choosing the Best Traits to Remove

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

Assume a person has the private trait C_2 out of the set of private traits C_1 and C_2, and this person has public traits $T_1 ... T_N$.

$$\begin{aligned} E_1 &= P(C_1) * P(T_1|C_1) * ... * P(T_N|C_1) \\ E_2 &= P(C_2) * P(T_1|C_2) * ... * P(T_N|C_2) \\ V &= \frac{E_1}{E_2} \end{aligned} \tag{12}$$

The learned class is then identified by equation 12. If V is greater than one, the true class is thought to be C_1, and if V is less than one, the true class is thought to be C_2.

Because the true class of the person is C_2, we want V to be as large as possible in order to hide this true class. Variable V_i represents the new value of V if we remove trait T_i. Our goal is to quickly calculate the V_i that has the largest value.

[5] Please note that many anonymization methods such as k-anonymity [9] do not introduce any false information to data set.

Trait Name	**Trait Value**	**Weight**
group member	legalize same sex marriage	46.160667
group member	every time i find out a cute boy is conservative a little part of me dies	39.685994
group member	equal rights for gays	33.837868
favorite music	ani difranco	17.36825
favorite movies	sicko	17.280959

Table 4: A sample of the most liberal traits

Trait Name	**Trait Value**	**Weight**
group member	george w bush is my homeboy	45.88831329
group member	bears for bush	30.86484689
group member	kerry is a fairy	28.50250433
favorite music	delirious	18.85227471
favorite movies	end of the spear	14.53703765

Table 5: A sample of the most conservative traits

$$\begin{aligned} V_i &= \frac{\frac{E_1}{P(T_i|C_1)}}{\frac{E_2}{P(T_i|C_2)}} \\ &= \frac{E_1 * P(T_i|C_2)}{E_2 * P(T_i|C_1)} \qquad (13) \\ &= V * \frac{P(T_i|C_2)}{P(T_i|C_1)} \qquad (14) \end{aligned}$$

Variable V in formula 14 is a constant for all V_i, so it can be removed for comparison purposes. Given a constant number of private trait classes, the best public trait to remove can be calculated in $O(N)$ time where N is the number of public traits a person has.

Tables 4 and 5 list the most liberal or conservative traits using equation 14. For example the most liberal trait, as shown in table 4, is being a member of the group “legalize same sex marriage”. Equation 14 gives a value that means someone with that trait is 46 times more likely to be a liberal than a conservative. Tables 6 and 7 show the most liberal and conservative traits for each trait type.

Trait Name	**Trait Value**	**Weight**
activities	college republicans	5.846955271
favorite books	redeeming love	6.348153362
favorite movies	end of the spear	14.53703765
favorite music	delirious	18.85227471
favorite tv shows	fox news	7.753312932
grad school	sw seminary	2.749648395
group member	george w bush is my homeboy	45.88831329
interests	hunting and fishing	7.614995442
relationship status	married	1.667495517
religious views	christian	2.441063037
sex	male	1.087798286

Table 6: Most conservative traits for each trait name

Trait Name	**Trait Value**	**Weight**
activities	amnesty international	4.659100601
favorite books	middlesex	4.841749269
favorite movies	hedwig and the angry inc	24.80050378
favorite music	deerhoof	22.94603913
favorite tv shows	queer as folk	9.762900035
grad school	computer science	1.698146579
group member	legalize same sex marriage	46.16066789
interests	vegetarianism	11.76878725
looking for	whatever i can get	1.703651985
relationship status	in an open relationship	1.617950632
religious views	agnostic	3.15756412
sex	female	1.103484182

Table 7: Most liberal traits for each trait name

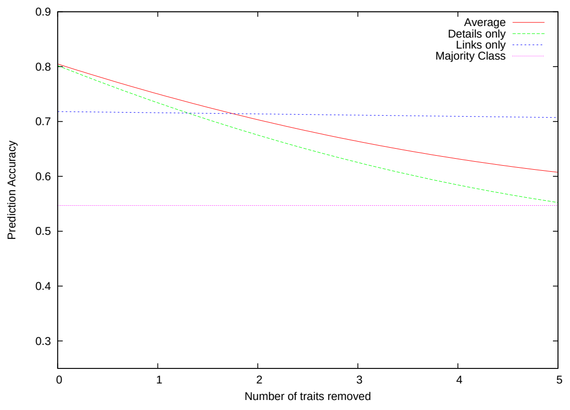


Figure 5: Reliability after hiding traits

5.2.2 Effect of Trait Removal on Inferring Private Trait values

Using the Facebook data, we removed the top K attributes from each person and reevaluated if we were still able to correctly predict their political affiliation. We repeated this for up to five removed traits. After five, our learning algorithm became worse than guessing the majority case each time.

Three learning algorithms are evaluated at each step of the anonymization process. Figure 5 shows the number of traits removed from each profile on the X axis and the prediction accuracy on the Y axis. After removing five traits, the predictive accuracy of the learning algorithm "Details only" was worse than if we had simply predicted the majority class, conservative, each time. Even though the link based learning algorithm depends upon counting shared traits to determine link weight, removing details from profiles seemed to have a very small effect on how accurate link based prediction was.

If we remove five traits, as we see in Figure 7, predictability becomes very poor for the vast majority of cases. Surprisingly, traits with a few details can still be reliably predicted using learning algorithm "Links only". For profiles with many traits, such as those with more than 80 details, using profile details is actually more predictive than links even after removing the top five traits. Figure 6 shows that even after removing five traits from each profile, having many friendship links did not increase predictability after around 20 friendship links.

On average, using friendship links alone becomes more useful than considering trait details after removing two traits, as figure 5 shows. In all graphs we see that considering the joint "Average" algorithm to predict class values is rarely worse than considering details alone, and in the cases where traits are hidden it is actually much more predictive.

Another idea follows from tables 4 and 5. We see that membership in a group effects our learning algorithm much more than any other trait type. This stems from the way groups and other traits are identified inside the Facebook platform.

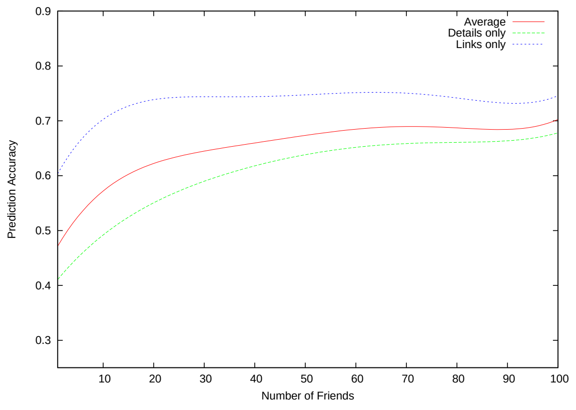


Figure 6: Private trait predictability after removing five traits

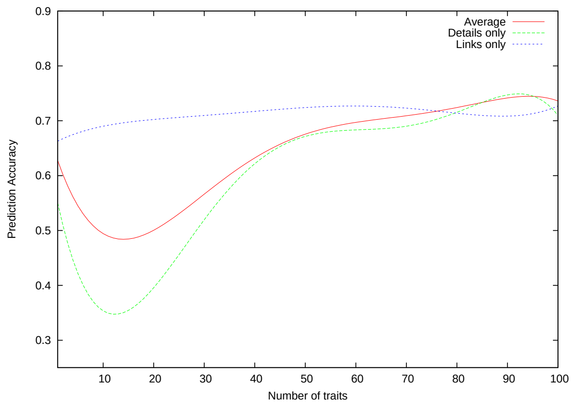


Figure 7: Private trait predictability after removing five traits

When users are inputting their traits, such as "Activities" and "Favorite Books", they are typed in without suggestion of what to put. Not only can this lead to misspellings and variations of phrasing the same idea, both of which can confuse learning algorithms, it can also lead to people simply forgetting to include a trait that actually does identify them. On the other hand, people join groups not by typing in the group name, but by selecting the group from a list or by being prompted to join the group from friends. This makes knowledge of group membership much more accurate.

5.3 Manipulating Link Information

Links can be manipulated in the same way traits 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 the equation 11 for determining trait 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 in class C_2 if formula 15 is positive.

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

Therefore, we would like to increase 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

$$\begin{aligned} d_i &= \left(\rho(C_2, F_aF_b...F_z) - \frac{P(C_2|F_i) * W_i^I}{\text{NOMR}}\right) \\ &\quad - \left(\rho(C_1, F_aF_b...F_z) - \frac{P(C_1|F_i) * W_i^I}{\text{NOMR}}\right) \\ &= d + \frac{(P(C_1|F_i) - P(C_2|F_i)) * W_i^I}{\text{NOMR}} \end{aligned}$$

Because d and NORM are constants for all d_i, the best choice for i that maximizes d_i becomes one that maximizes $M_i = P(C_1|F_i) - P(C_2|F_i)) * W_i^I$.

In our experiments, we order the links for each node based on the M_i values. After removing the most telling friendship link from each node, we see a drop in predictability between graphs 2 and 8. While the drop is more noticeable after removing five friendship links from each node, as seen in figure 9, we still see that after a certain number of friendship links, having more friendship links does not help predictability. It is also interesting to note that our "Average" algorithm, which depends upon link structure, still performs relatively well even after removing the top five friendship links.

6 Conclusion and Future Work

In this paper, we addressed various issues related to private information leakage in social networks. For unmodified social network graphs, we show that using

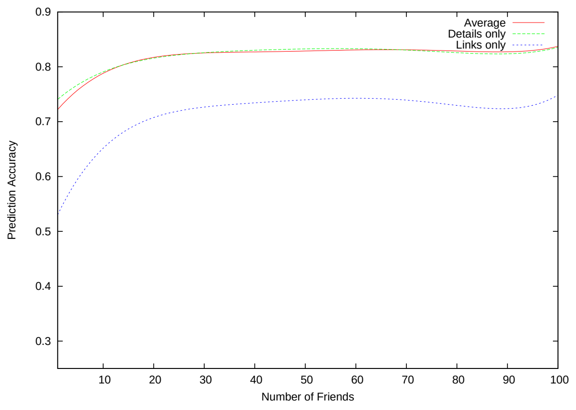


Figure 8: Private trait predictability versus number of friends after removing one friendship link from each node

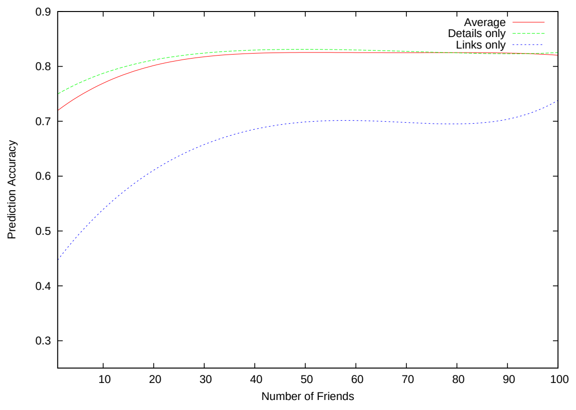


Figure 9: Private trait predictability versus number of friends after removing five friendship links from each node

details alone one can predict class values more accurately than using friendship links alone, and that using both friendship links and details together does give better predictability than details alone. In addition, we explored the effect of removing traits and links in preventing sensitive information leakage.

In this paper, we 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.

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] Gnuplot 4.2.3, 2008. http://www.gnuplot.info.

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

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

[10] 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.

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

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

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