Sensitive attribute prediction for social networks users

Younes Abid  
Lorraine University  
Nancy, France  
younes.abid@loria.fr

Abdessamad Imine  
Lorraine University, Cnrs, Inria  
Nancy, France  
abdessamad.imine@loria.fr

Michaël Rusinowitch  
Lorraine University, Cnrs, Inria  
Nancy, France  
michael.rusinowitch@loria.fr

ABSTRACT
Social networks are popular means of data sharing but they are vulnerable to privacy breaches. For instance, relating users with similar profiles an entity can predict personal data with high probability. We present SONSAI a tool to help Facebook users to protect their private information from these inferences. The system samples a subnetwork centered on the user, cleanses the collected public data and predicts user sensitive attribute values by leveraging machine learning techniques. Since SONSAI displays the most relevant attributes exploited by each inference, the user can modify them to prevent undesirable inferences. The tool is designed to perform reasonably with the limited resources of a personal computer, by collecting and processing a relatively small relevant part of network data.

1 INTRODUCTION
Data published on social networks profiles can be mined for inferring sensitive information about users. For instance it was shown in [10] how musical tastes allow one to predict educational level. To increase user awareness about these privacy threats we have designed a tool, SONSAI, for Facebook users to audit their profiles. The system crawls the network around the user and predicts its sensitive attributes values using a machine learning engine. The results provided by SONSAI, also shows the public attributes of the user that have oriented the learning algorithm towards a particular sensitive attribute value. The user can therefore modify these public attributes to prevent inference.

For the approach to be feasible several problems have to be solved: First, data collection by crawling is limited both by the social network and by country regulations. Hence the crawler exploration strategy has to focus only on meaningful representative network nodes. Since attributes are numerous, for the learning program to scale one has to select only the most relevant ones for inferring sensitive attribute values. Hence the second problem is to find an attribute relevance measure that is both accurate and easy to compute. Note that we cannot rely on semantic proximity since we notice that a user that hides a sensitive attribute probably will hide semantically related ones, too. Moreover for fully anonymised datasets the attributes semantics is hard to recover. Therefore we follow an alternative approach by modelling attributes as bipartite graphs and measuring relevance of attributes by comparing their bipartite graph structures.

For specific attributes such as gender and relationship status, the sets of values are much smaller than for other attributes like music and movie. Consequently, the graphs that model these attributes have higher connectivity than the other graphs. For instance, the density of the graph that models gender (as most users publish their gender) is close to 0.5. In order to infer hidden links in such graphs we need to learn from highly connected graphs. However most of the available learning graphs are sparse. To cope with this last problem, we derive new graphs by merging several learning graphs.

All the proposed methods have been implemented for Facebook, however they can be applied to many other social networks. The system has been tested by several volunteer users for auditing their Facebook profiles. In each case a dataset was built from real profiles collected in the user neighborhood network.

Related works. In [15] the authors propose algorithms to detect whether a sensitive attribute value can be inferred from the neighborhood of a target user in a social network. Heatherly et al. [12] infer attribute values in social network with bayesian classification techniques. For the same purpose Estivill-Castro et al. [7] employ decision-tree learning algorithms. In these works learning is performed off-line on large datasets. In order to perform attribute inference from sparser datasets collected in short time by our tool user in his ego-network, we rather use random walk-based learning. The random walk technique has been applied to social representations in [14] and [11] where the authors analysed friendships and used a skip-gram model. In these works, user profiles that have similar friends will be mapped to similar representations. This model helps to detect communities and can predict a set of potential friends for a given user profile. Skip-gram model has also been applied recently to infer social relationship from mobility data [5]. Our work uses a more adapted Continuous Bag of Words (CBOW) model to predict the most likely values for a user sensitive attribute. In our setting friends are considered as an attribute among others. Moreover we first determine automatically a relevant subset of the social network for optimizing random walks. In [3] the relevance of attributes is computed by Bayesian optimisation which is much less efficient than the graph comparison approach adopted here. In particular [3] does not succeed in reasonable time (on standard PC) with attributes like gender and relationship status. Let us note that (unlike [1]) our approach does not need any ontology to perform semantic correlation between attributes. Finally we notice that the proposed system SONSAI is close under some aspects to a recommendation system: an item suggestion can be viewed as an attribute value prediction [4]. However unlike recommendation systems SONSAI also provides explanations for the predicted values, namely an ordered list of attributes that have played a significant role in the computation. For enforcing privacy, our final goal will be
to reduce the “recommendation accuracy” by acting on this list of attributes.

## 2 ARCHITECTURE OF SONSAI

### Figure 1: Architecture of SONSAI.

The architecture of SONSAI is overviewed in Figure 1. The Facebook crawler (about 5k lines of Java 8) drives a Firefox 58.0b4 navigator through a Selenium 3.5.3 server. Collected information from each profile, group and page are stored in separated XML files. The Anonymizer, Cleanser, Random walker and Ranker components are written in Python 2.7 (about 2.5k lines of code). The Anonymizer component parses all the XML files, generates anonymized graphs and stores them as TSV files. Then, the Cleanser selects the most relevant graphs and stores them as adjacency lists. The Random walker component browses the adjacency lists and stores the resulting walks in a text document. We use the Python gensim implementation of Word2Vec to parse the text document and compute a vectorial representation of social network nodes encountered in the walk. Finally, the Ranker component classifies the sensitive nodes according to their similarity to the target user profile.

## 3 SAMPLING FACEBOOK

We have designed a crawler that explores the social network around a user (to some distance given as a parameter) and collects public information from the visited web pages (i.e., friends, liked pages . . .) in order to build a representative subnetwork. We distinguish two types of Facebook nodes: user profiles (u) and pages (p) and two types of links: like-ships between user profiles and pages, and friendships between user profiles. Given a node c, c.n denotes the set of nodes that are linked to c. A discovered node is a node whose URL is known by the crawler. For instance, if the crawler retrieves a user profile and collects its public friends list, then all the friends of that particular user profile are discovered. Algorithm 1 crawls at most nc nodes at distance ≤ d from the target node ut. Each iteration of the outer loop samples a node, crawls it and update the sets of discovered and crawled nodes. The sampling is done by random walks of length ≤ d with a transition probability π designed to crawl with higher priority closer nodes and to favour neighbour nodes according to their type. Function sinks(j) returns the set of sinks, i.e. crawled nodes such that all discovered nodes at distance ≤ j are also crawled. Sinks are avoided by the random walks to guarantee that the final node has not been crawled yet.

### Algorithm 1: Crawling nodes around a target user.

1. Procedure crawl_nodes(ut, d, nc, π)
2.   crawl_node(ut);
3.   while |crawled_nodes| < nc do
4.     c ← ut; r ← {};
5.     for i ← 1 to d by 1 do
6.       r.addall(sinks(d − i));
7.       c ← random_select_in((c ∪ c.n) \ r);
8.   end
9.   crawl_node(c);

### Figure 2: Example of pages like-ship graphs.

### 4 SOCIAL NETWORK MODEL

#### Modelling friendship relations. Since friendship on Facebook is symmetric, we model friendship between user profiles by an undirected graph \((U, F)\) where \(U\) is a set of users’ profiles and \(F\) is a set of friendship links between them.

#### Modelling page like-ship. We model like-ship between user profiles and pages by several bipartite graphs \((U, F, L)\) where \(U\) is a set of users profiles, \(P\) is a set of pages (a type) and \(L\) is a set of like-ship links between them. Figure 2 shows an example of page like-ship modelled by two graphs. Graph (a) models liked pages of music type and Graph (b) models liked pages of book type. We note that user profiles can like several pages of the same type.

Anonymizing the social network graphs. For ethical and regulation reasons Facebook identifiers are replaced by fresh identifiers. Each node in the network is then identified by a unique integer ID replacing its Facebook ID. The anonymized IDs are sorted according to the node types. Anonymized graphs are saved under the tab-separated value (TSV) format, one of the most general delimiter-separated values format (DSV). TSV is widely used in

---

2[https://radimrehurek.com/gensim/index.html](https://radimrehurek.com/gensim/index.html)
3Icon made by Smashicons from www.flaticon.com
the sensitive graph. And the graph that models the link-ship between user profiles and pages of musics is the learning graph. The Jaccard index between two user nodes that have a null degree in the learning graph or in the sensitive graph represents the confidence rate \( \theta_{cr} \) that publish their links in the second part of the learning graph. The ratio of user profiles that publish their links in the sensitive graph contains user profiles that hide their links in the sensitive graph to recover. On the contrary we will rely on comparing attributes graph structures.

In the following we assume a fixed sensitive graph \( s \).

Step 1: Computing the learning and the confidence rates of learning graphs. In order to compare the structure of a given learning graph to the structure of the sensitive graph, we first split each graph in two parts. The first part contains user profiles that hide their links in the sensitive graph. The ratio of user profiles that publish their links in the first part of the learning graph represents the learning rate \( lr \). The second part contains user profiles that publish their links in the sensitive graph. The ratio of user profiles that publish their links in the second part of the learning graph represents the confidence rate \( cr \).

The function \( \text{connected_profiles_in}(l) \) returns the set of user profiles that publish their links in the graph given as argument. Given \( l, s \) respectively a learning graph and a sensitive graph and \( U \) the set of user profiles in all graphs, we define:

\[
\begin{align*}
U_l &= \text{connected_profiles_in}(l) \\
U_s &= \text{connected_profiles_in}(s) \\
lr(l) &= \frac{\text{size}(U_l) \cap (U \setminus U_s)}{\text{size}(U \setminus U_s)} \\
cr(l) &= \frac{\text{size}(U_l) \cap U_s}{\text{size}(U_s)}
\end{align*}
\]

Figure 3 depicts an example of splitting two graphs for comparison. The graph that models the link-ship between user profiles and pages of politicians is the sensitive graph. And the graph that models the link-ship between user profiles and pages of musics is the learning graph. The learning rate \( lr \) for this example is equal to 50%. And the confidence rate \( cr \) is equal to 75%.

Step 2: Computing the distance between a learning graph and the sensitive graph. In this step, we discard user profiles that have a null degree in the learning graph or in the sensitive graph. The Jaccard index between two user nodes \( u_1 \) and \( u_2 \) in a given graph \( A \) is computed as follows, where the function \( \text{links}_A(u_j) \) returns the set of nodes to which user node \( u_j \) is connected in the graph \( A \).

\[
J_A(u_1, u_2) = \frac{\text{size}(\text{connected_profiles_in}(u_1) \cap \text{connected_profiles_in}(u_2))}{\text{size}(\text{connected_profiles_in}(u_1) \cup \text{connected_profiles_in}(u_2))}
\]

The Hamming distance \( H \) between graphs \( l \) and \( s \) is defined by:

\[
H(l) = \max_{u_k, u_j \in U_l \cap U_s} |J_s(u_k, u_j) - J_s(u_k, u_j)|
\]

In order to compare learning graphs with different sets of common connected profiles \( U_l \cap U_s \), we normalize this distance by the maximal Hamming distance that can be obtained on such a set. Hence we define the Hamming rate: \( hr(l) = H(l)/M(l) \) where \( M(l) \) is

\[
M(l) = \max_{u_k, u_j \in U_l \cap U_s} \text{size}(\text{connected_profiles_in}(u_k) \cup \text{connected_profiles_in}(u_j))
\]

Step 3: Selecting most relevant graphs for learning sensitive attribute values. We first discard the learning graphs that have a learning rate \( lr \) lower than threshold \( \theta_{hr} \), since they do not convey enough information. We then discard the graphs that have a confidence rate \( cr \) lower than \( \theta_{cr} \), since they are considered as unreliable. Finally, from the remaining graphs we only select graphs that have a Hamming rate \( hr \) higher than \( \theta_{hr} \) since they are the most similar to the sensitive graph.

Densifying graphs

For some sensitive attributes such as gender, age and relationship status, user profiles are linked to at most one value. Moreover the sets of values for these particular attributes are much smaller than for other attributes. Consequently, the graphs that model these attributes are denser than the other ones. In this case, for improving the random-walk based learning process (see Section 6) we need to merge several learning graphs in order to obtain a denser one. We explain the method with a simple example of gender prediction: we select attribute graphs with high \( lr \), \( cr \) rates

---

4https://www.kaggle.com/netflix-inc/netflix-prize-data
but with also a good rate of discrimination between genders, that is the gender of connected users in the graph is unbalanced between male and female. For instance we can select jewelry and fast-food graphs. We merge these graphs by grouping all fast-foods in a unique node and similarly for jewelries as shown in Figure 4 to obtain a new learning graph.

\begin{figure}[h]
\centering
\includegraphics[width=0.25\textwidth]{merging_graphs.png}
\caption{Merging graphs.}
\end{figure}

6 RANDOM WALK-BASED ATTRIBUTE LEARNING

Representation generation. We plan to translate latent information from the selected graphs (in previous section) to a document. The resulting document holds information about paths in the graphs and their frequencies that can be exploited for inferring proximity of a user node to some sensitive attribute value node. Following \cite{13}, paths in the social graph are sampled by random walks. In our case, the walks are executed only in the subset of selected graphs.

Let \( G_1, G_2, \ldots, G_n \) be the list of selected learning graphs. Let \( U \) be the set of users in all graphs. We assume that the friendship graph is selected and \( G_1 = (U, F) \) (otherwise we can simply adapt the computation below). The other graphs are bipartite and we pose \( G_i = (U, V_i, L_i) \) for \( i > 1 \). We introduce quotas to quantify the importance of each graph \( G_r \) for inferring secret values of the target sensitive attribute. Each selected graph \( G_r \) is assigned a 3 dimensional vector \( V_{G_r} = [v_{r}(G_r), \ c_{r}(G_r), \ 1 - hr(G_r)] \). The quota of \( G_r \) is given by its Mahalanobis distance to the null vector \([0,0,0]\). It is computed as follows:

\[
q(G_r) = \sqrt{V_{G_r}^T \Sigma^{-1} V_{G_r}}
\]

with \( \Sigma \) the \( 3 \times 3 \) covariance matrix over the set of selected graph vectors.

To specify the random walk transitions we first define the probability \( p_{u,y} \) that being in node \( u \) the next node in the random walk is in a selected graph \( G_y \):

\[
p_{u,y} = \begin{cases} 
q(G_y) & \text{if } deg_y(u) > 0 \\
0 & \text{otherwise}
\end{cases} \quad (4)
\]

where \( deg_y(u) \) is the degree of user \( u \) in graph \( G_y \). A value node \( v \) is followed by a user node chosen uniformly at random from the ones connected to \( v \). Assuming the node following user node \( u \) is in \( G_y \), then it will be chosen uniformly at random from the nodes in \( G_y \) that are connected to \( u \). Therefore, the transition probabilities are \( (G_1 \) is the friendship graph):

\[
\begin{align*}
&u \rightarrow v : \frac{p_{u,y}/deg_y(u)}{deg_y(u)} & \text{if } y \neq 1 \text{ and } (u,v) \in L_y \\
v \rightarrow u : \frac{1}{deg_y(v)} & \text{if } y \neq 1 \text{ and } (u,v) \in L_y \quad (5) \\
u \rightarrow u' : \frac{p_{u,1}/deg_1(u)}{deg_1(u)} & \text{if } (u, u') \in F
\end{align*}
\]

As illustrated in Figure 5 the document is constructed by connecting all graphs through random jumps between their nodes (see also \cite{14}). At each step the walker state changes and a new word is written in the text document. One step amounts to select a graph where the current node occurs with non null degree, and then to select a node that is connected to the current node in the selected graph. The selected node then becomes the new current node.

In this example we aim to predict liked pages of politicians masked by user profile u3. The sensitive graph is Graph 2 and the learning graphs are Graph 1 and Graph 3. Since the values of the sensitive attributes (the pages of politicians) are labelled (each value belongs to a unique cluster), they are represented by the label of their clusters in the final document. We use a greedy clustering algorithm \cite{3} to define size similar clusters. Pages of politicians that share many common 'likers' end up in the same cluster. For instance the first walk depicted by Figure 5 is \([u_1, u_4, v_2,3, u_4]\). But for efficiency the walk \([u_1, u_4, c_{2,2}, u_4]\) is stored instead in the document since the value \( v_{2,3} \) belongs to the cluster \( c_{2,2} \).

Applying word2vec to compute node representations. We have performed multi-graph random walks in the social network and generated a text document. Walks in the document can be interpreted as sentences, where the words are network nodes. Hence, inferring a link between a user node and an attribute value node is similar to the problem of estimating the likelihood of words co-occurrence in a corpus. We use word2vec \cite{9, 13} to map one-hot encoded vectors that represent words in a high-dimensional vocabulary space to low-dimensional vectors (see \cite{6}). Word2vec
is employed with the Continuous Bag of Words (CBOW) model for predicting a word given its context defined by the $c-1$ words surrounding it ($c$ is the window size of the context). Inputs of the model in our case are users and published attribute value. Since there is no order between the attribute values CBOW model is more adequate than Skipgram. The output of the CBOW model is a vector of size $v$ representing a probability distribution of co-occurrence between all the words of the context and each word from the vocabulary within a window of size $c$.

Ranking sensitive attribute values. We measure semantic similarity between two nodes by the cosine of the angle formed by the vectors representing the nodes. Cosine similarity is known to take greater account of context information. We rank all the sensitive values by their cosine similarity to the target user profile. The values that have the lowest rank are most likely the sensitive attribute secret values of the target user profile, where secret values are actually the true values of the target user but are not published by him on the social network.

The Figure 6 depicts an example of 2-dimensional vectors that encode 8 nodes: 3 user profiles ($u_1$, $u_2$ and $u_3$), 2 pages of musics ($v_3$,1 and $v_3$,2) and 3 clusters of pages of politicians ($c_2$,1, $c_2$,2 and $c_2$,3). The clusters of the pages of politicians are the sensitive values and their vectors are red. The node $u_1$ is the target user profile and its vector is blue. The clusters of pages of politicians will be ranked according to their distances to $u_1$. And the inference algorithm will infer as most probable pages of politicians to be liked by $u_1$, the pages of politicians of the cluster that has the smallest rank (the closest cluster to $u_1$).

In [16] Schakel et al. show that word2vec unsupervised learning algorithm encodes word semantics by affecting vectors in the same direction for co-occurring words during training. Besides, the magnitude of a vector reflects both the frequency of appearance of related words in the corpus and the homogeneity of contexts. Where a context is a set of words that have high co-occurrence probability in the corpus.

In fact, the words that appear in the same contexts have small angular distances between them. The less overlapping the contexts are, the larger the angular distances between their different words are. However, words that appear in many contexts are represented by vectors that average vectors pointing in many contexts directions. Hence, the vectors magnitude generally decreases with respect to the number of contexts. Moreover, the higher the word frequency is, the higher the chance that it is used in different contexts is. Consequently, the vector magnitude also decreases with respect to frequency. From these remarks, we conclude that the euclidean distance is not a good measure for our inference purpose. Actually, words that appear in many contexts have low magnitude. As a result, their euclidean distances will be small and, using this criteria, they would be considered close even if they do not appear in any common context. For instance, the euclidean distance between the cluster of pages of the most popular politicians will be small even if they are rivals. In the example depicted by Figure 6 the politicians of the clusters $c_2$,1 and $c_2$,2 are rivals. The angular distance between those two clusters is big. However, the euclidean distance is small. Moreover, the euclidean distance between a user that has many friends, for instance the user $u_3$ in the Figure 6, and a popular music like “despacito”, for instance the page of music $v_3$,3 in the Figure 6, will be small. But popular users do not necessarily like popular musics.

7 EXPERIMENTS

To build datasets we have crawled Facebook profiles of people that live in North-East France and in Ile-de-France (Paris). Table 1 gives statistics about the two crawled datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>$#$ Attributes</th>
<th>$#$ Attribute values</th>
<th>$#$ Crawled user profiles</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>1 929</td>
<td>1 022 860</td>
<td>15 012</td>
</tr>
<tr>
<td>D2</td>
<td>1 296</td>
<td>298 617</td>
<td>6 550</td>
</tr>
</tbody>
</table>

Table 1: Details about the datasets.

7.1 Political orientation

From each node $n$ we perform a random-walk of 800 steps. The dimension of the node representation is taken to be 512. The dimension is usually taken between 100 and 300 for natural languages. However the size of the vocabulary in social networks (equal to the number of nodes) is much higher than in natural languages.

In Dataset 1 the sensitive graph represents the links between 2554 user profiles and 4589 politician pages. For each experiment we generate a new social graph from the dataset by selecting the user profiles that publish their preferences concerning the sensitive attribute (pages of politicians) and at least another attribute. Then we remove all the links in the sensitive graph of 10% of the selected user profiles. The algorithm makes sure that all the nodes in the resulting social graph remain connected. The experiments have consisted then in inferring the hidden links based on information from the learning graphs.

Among the 1928 learning graphs, we selected the ones with learning rate greater than $\theta_{f} = 0.05$, confidence rate greater than $\theta_{cr} = 60\%$ and Hamming rate lower than $\theta_{hr} = 4\%$.

Table 2 details the 23 selected graphs relevance measures.

<table>
<thead>
<tr>
<th>Attribute graph</th>
<th>IT (in %)</th>
<th>CP (in %)</th>
<th>AV (in %)</th>
<th>Number of Users</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Movies</td>
<td>44.97</td>
<td>98.47</td>
<td>1.75</td>
<td>8 118</td>
<td>137 338</td>
</tr>
<tr>
<td>Communities</td>
<td>38.58</td>
<td>91.38</td>
<td>2.03</td>
<td>7 411</td>
<td>84 762</td>
</tr>
<tr>
<td>PublicFigures</td>
<td>32.86</td>
<td>92.44</td>
<td>1.80</td>
<td>6 455</td>
<td>28 289</td>
</tr>
<tr>
<td>News/Organizations</td>
<td>31.85</td>
<td>86.57</td>
<td>1.72</td>
<td>6 180</td>
<td>28 474</td>
</tr>
<tr>
<td>Artists</td>
<td>30.65</td>
<td>84.22</td>
<td>1.92</td>
<td>5 970</td>
<td>31 684</td>
</tr>
<tr>
<td>Companies</td>
<td>30.05</td>
<td>85.94</td>
<td>1.75</td>
<td>5 939</td>
<td>20 750</td>
</tr>
<tr>
<td>Websites</td>
<td>29.57</td>
<td>83.94</td>
<td>1.78</td>
<td>5 829</td>
<td>17 931</td>
</tr>
<tr>
<td>Music</td>
<td>29.8</td>
<td>83.61</td>
<td>2.31</td>
<td>5 778</td>
<td>11 976</td>
</tr>
<tr>
<td>Entertainment</td>
<td>29.28</td>
<td>76.20</td>
<td>2.84</td>
<td>5 669</td>
<td>8 319</td>
</tr>
<tr>
<td>Media/News</td>
<td>29.23</td>
<td>87.27</td>
<td>1.82</td>
<td>5 871</td>
<td>14 042</td>
</tr>
<tr>
<td>Organizations</td>
<td>27.52</td>
<td>80.93</td>
<td>1.86</td>
<td>5 496</td>
<td>15 986</td>
</tr>
<tr>
<td>News/Media</td>
<td>27.44</td>
<td>83.43</td>
<td>1.86</td>
<td>5 550</td>
<td>9 247</td>
</tr>
<tr>
<td>Organizations</td>
<td>26.78</td>
<td>80.77</td>
<td>1.63</td>
<td>5 328</td>
<td>14 738</td>
</tr>
<tr>
<td>Movies</td>
<td>26.09</td>
<td>75.17</td>
<td>2.28</td>
<td>5 171</td>
<td>16 282</td>
</tr>
<tr>
<td>LocalBusinesses</td>
<td>24.91</td>
<td>78.58</td>
<td>1.69</td>
<td>5 111</td>
<td>17 321</td>
</tr>
<tr>
<td>Clothing</td>
<td>23.99</td>
<td>68.12</td>
<td>1.96</td>
<td>4 729</td>
<td>18 090</td>
</tr>
<tr>
<td>Books/comics</td>
<td>23.52</td>
<td>71.73</td>
<td>2.24</td>
<td>4 763</td>
<td>8 422</td>
</tr>
<tr>
<td>Actors/Directors</td>
<td>23.12</td>
<td>74.54</td>
<td>2.78</td>
<td>4 785</td>
<td>10 425</td>
</tr>
<tr>
<td>Magazine</td>
<td>22.83</td>
<td>73.96</td>
<td>1.69</td>
<td>4 733</td>
<td>9 955</td>
</tr>
<tr>
<td>Athletes</td>
<td>22.68</td>
<td>68.79</td>
<td>2.35</td>
<td>4 583</td>
<td>14 123</td>
</tr>
<tr>
<td>ApplicationPages</td>
<td>21.68</td>
<td>66.36</td>
<td>2.04</td>
<td>4 396</td>
<td>4 244</td>
</tr>
<tr>
<td>Sports/Teams</td>
<td>21.48</td>
<td>63.93</td>
<td>2.35</td>
<td>4 309</td>
<td>10 843</td>
</tr>
</tbody>
</table>

Table 2: Selected learning graphs in D1 for politicians.
We note that the communities graph has the second greatest learning rate \( lr = 44.97\% \), which means that it holds much latent information about users who hide their likes in the sensitive graph. It also has the maximal confidence rate \( cr = 98.47\% \) and the fifth lowest Hamming rate \( hr = 1.75\% \) among the 23 selected relevant graphs, which means that its structure is very similar to the structure of the politicians graph. The friendship graph (Users) has the maximal learning rate \( lr = 88.37\% \) and a high confidence rate \( cr = 83.98\% \) since 87.62\% of users are connected to this graph. However, this graph has a Hamming rate \( hr = 2.08\% \) greater than the average \( hr \) of selected graphs which means that learned information from this graph is less reliable than learned information from the communities graph.

We use the area under the ROC curve (AUC) as defined in [8] to measure the accuracy of the inferred links. For the defined thresholds (\( \theta_{lr} = 20\% \), \( \theta_{cr} = 60\% \), \( \theta_{hr} = 4\% \)) the precision is equal to 0.79. However, the inference accuracy when the 23 relevant graphs are selected randomly is only 0.41. We conducted more tests by selecting manually 3 graphs that are semantically close to politics as follows. Graph \( G_1 \) models the links between 1246 user profiles and 2357 political organizations, \( G_2 \) models the links between 1120 user profiles and 1758 political parties and \( G_3 \) models the links between 39 user profiles and 41 political ideologies. Although the selected graphs seem promising, the inference accuracy is only 0.46. This can be explained by the fact that the selected graphs are very sparse and users are vigilant when publishing their preferences about those attributes. Consequently, the algorithm cannot learn well from them.

We note that the musicians/bands graph was automatically selected by our relevance-based selection method confirming that music and politics are correlated as it was empirically discovered in previous studies [17].

Table 3 summarizes the results of the conducted experiments.

<table>
<thead>
<tr>
<th>Selection based on</th>
<th>Accuracy</th>
<th>Targets</th>
<th>Deleted links</th>
<th>Nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relevance (23 graphs)</td>
<td>0.79</td>
<td>252</td>
<td>409</td>
<td>558125</td>
</tr>
<tr>
<td>Random (23 graphs)</td>
<td>0.41</td>
<td>204</td>
<td>351</td>
<td>11200</td>
</tr>
</tbody>
</table>

**Table 3: Experimental results**

### 7.2 Gender and relationship status

To produce a text document from the social graph, we perform random-walks of 80 steps. Each random walk starts from a different node. The dimension of the node vectorial representation is taken to be 128 since in that case the number of sensitive values is smaller. Moreover learning graphs are obtained by grouping several attribute values in the merging operation (see Section 5). For the experiments, we have generated a new social graph from the dataset by randomly selecting 10\% of user profiles that publish the value of their sensitive attribute and we have masked it. Then SONSAI has tried to infer the masked values of the sensitive attribute for each user based on the selected attributes by the cleanser module.

**Relationship status.** The sensitive graph models the relationship status of user profiles. To simplify the presentation we define two meta-relationship status as follows:

- \( R_1 = \{ \text{Single, Divorced, Separated, Widowed, Complicated} \} \)
- \( R_2 = \{ \text{Domestic partnership, Married, Engaged, Relationship, Civil union, Open relation} \} \)

We aim to infer the meta-relationship status of users. Table 4 gives more details about the selected attributes from dataset D2.

We notice that discriminant attributes toward \( R_1 \) are focused around educations and leisures. On the other hand, discriminant attributes toward \( R_2 \) are focused around business. The accuracy in AUC of inferring the meta-relationship status is higher than 0.7 in both datasets D1.

![Figure 6: Example of 2-dimensional vectors that encode 8 nodes.](image-url)
and D2 as soon as the target publishes values concerning at least 4 selected attributes by the cleanser.

Gender. The sensitive graph models the gender of user profiles. We notice that discriminant attributes toward male are focused around sports, games and software. On the other hand, discriminant attributes toward female are focused around health, home and luxury. The accuracy in AUC of inferring the gender is higher than 0.83 in dataset D1 and higher than 0.67 in dataset D2 as soon as the target publishes values concerning at least 2 selected attributes by the cleanser.

Table 4: Selected attributes in D2 for relationship status.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Importance</th>
<th>Discrimination</th>
</tr>
</thead>
<tbody>
<tr>
<td>Education</td>
<td>2.70</td>
<td>58.41 % R1</td>
</tr>
<tr>
<td>Community College</td>
<td>2.74</td>
<td>90.02 % R1</td>
</tr>
<tr>
<td>Consulting Agency</td>
<td>2.71</td>
<td>90.70 % R2</td>
</tr>
<tr>
<td>Sports &amp; Recreation</td>
<td>2.56</td>
<td>91.18 % R2</td>
</tr>
<tr>
<td>Home &amp; Garden Website</td>
<td>2.40</td>
<td>91.89 % R2</td>
</tr>
<tr>
<td>Automotive, Aircraft &amp; Boat</td>
<td>2.48</td>
<td>92.86 % R2</td>
</tr>
<tr>
<td>Locality</td>
<td>2.47</td>
<td>92.59 % R2</td>
</tr>
<tr>
<td>Corporate Office</td>
<td>2.40</td>
<td>91.18 % R2</td>
</tr>
<tr>
<td>News &amp; Media Website</td>
<td>2.42</td>
<td>90.32 % R2</td>
</tr>
<tr>
<td>Financial Service</td>
<td>2.41</td>
<td>90.00 % R2</td>
</tr>
<tr>
<td>Industrial Company</td>
<td>2.40</td>
<td>89.29 % R2</td>
</tr>
<tr>
<td>Educational Consultant</td>
<td>2.02</td>
<td>75.00 % R1</td>
</tr>
<tr>
<td>Playground</td>
<td>1.80</td>
<td>66.67 % R1</td>
</tr>
<tr>
<td>Phone/Tablet</td>
<td>1.70</td>
<td>63.64 % R1</td>
</tr>
<tr>
<td>Plastic Surge</td>
<td>1.60</td>
<td>60.00 % R1</td>
</tr>
<tr>
<td>Consulate &amp; Embassy</td>
<td>1.60</td>
<td>60.00 % R2</td>
</tr>
<tr>
<td>School Sports Team</td>
<td>1.53</td>
<td>52.00 % R1</td>
</tr>
<tr>
<td>Dive Bar</td>
<td>1.40</td>
<td>54.55 % R1</td>
</tr>
<tr>
<td>Video</td>
<td>1.44</td>
<td>51.00 % R1</td>
</tr>
<tr>
<td>Playlist</td>
<td>1.41</td>
<td>53.04 % R1</td>
</tr>
</tbody>
</table>

7.4 Parameter sensitivity analysis

Let us investigate the impact of the cleansing parameters \( lr \), \( cr \) and \( hr \). All experiments detailed in this section are conducted on dataset D1 to infer users’ political orientation.

Table 7 shows that only 3 graphs among the 1928 available graphs have a learning rate \( lr \) higher than 30%. Based on those graphs, inference accuracy can be very low. For instance, inference accuracy based on gender attribute is only 0.36. Based on only the users (i.e. friendship) graph accuracy is getting better to 0.64. The communities graph gives high accuracy of 0.74 for inferring political views. However, we notice that the best accuracy is obtained when selecting graphs with learning rate between 10% and 40%. Table 7 shows that the learning rate parameter \( lr \) is important to select the best graphs for inference. However, accuracy does not depend only on this parameter since some graphs such as gender graph that have high learning rate may lead to very low accuracy results.

Table 8 shows that when the Hamming rate \( hr \) decreases, accuracy increases. However, most graphs have a low Hamming rate because only a small part of them can be compared to the sensitive graph, as few users publish their preferences in both graphs. Hence, their structure is not fairly comparable to the politicians’ graph structure. To cope with this problem we compute a third parameter, the confidence rate \( cr \), that indicates how reliable the structure comparison is.

Table 9 shows that the confidence rate, \( cr \), does not give information about the best graph to select when it is considered isolately but it must be coupled with other parameters. For instance, if a given graph \( g \) has a high confidence rate but a low Hamming rate, that means that it is a good graph for inference. However, if a given graph \( g \) has a high confidence rate and high Hamming distance rate that means that \( g \) is probably a bad graph for inferring the sensitive attribute. But a given graph \( g \) could be interesting for inference if it has low \( cr \) and high \( hr \).

8 CONCLUSION

SONSAI application enables users to predict their sensitive links in social networks from relatively small amount of data and computing resources. Indeed, sensitive data inferences are fast and accurate on typical personal attributes. It should be noted that the friendship graph was not selected among important ones to deduce both the gender and the relationship status of users. This probably means that alternative techniques based solely on homophily would be inaccurate in this context. Moreover, we have observed that the privacy of users is threatened as soon as they start publishing at least three important attributes. These ones are automatically brought to light by SONSAl, regardless...
their semantics and only through a structural analysis of the social network graph. As future work, we plan to incorporate countermeasures into our tool to protect users against attacks that might compromise their privacy. We also plan to enhance SONSAI prediction engine with a tool that permits one to disclose hidden friendship links using adequate combinations of queries provided by the social network [2].

Acknowledgments. This work is supported by MAIF Foundation5.

REFERENCES


[7] We refer to a list of 1000 cities and countries, which we used for the experiments.

5www.fondation-maif.fr/