Sensitive attribute prediction for social networks users

Younes Abid, Abdessamad Imine, Michael Rusinowitch

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

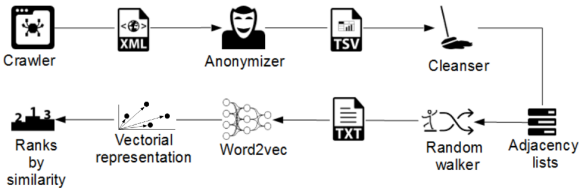

Figure 1: Architecture of SONSAI.

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 [1]. 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 [2] 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 n_c nodes at distance $\leq d$ from the target node u_t. 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 $\leq d$ 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 $\leq j$ are also crawled. Sinks are avoided by the random walks to guarantee that the final node has not been crawled yet.

[1] http://www.seleniumhq.org/
[2] https://radimrehurek.com/gensim/index.html

```
1  Procedure crawl_nodes(u_t, d, n_c, π)
2      crawl_node(u_t);
3      while |crawled_nodes| < n_c do
4          c ← u_t; 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);
10     end
```

Algorithm 1: Crawling nodes around a target user.

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, P, 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[3]. 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.

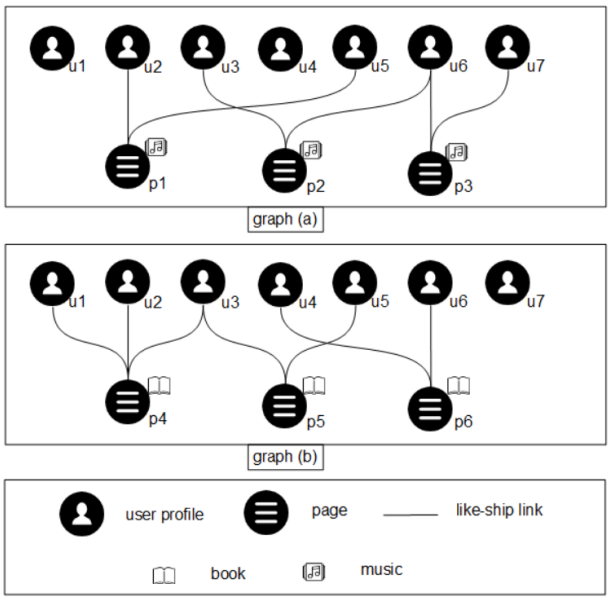

Figure 2: Example of pages like-ship graphs.

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

[3] Icon made by Smashicons from www.flaticon.com

graph exchange. In contrast to the dataset released by Netflix [4] where only user IDs are anonymised (but not movies title, rating, date of rating), all attribute values are anonymised in our datasets.

In the following a *sensitive graph* is a graph that models an attribute that is considered sensitive by the user (i.e. the user does not want its value to be predictable). The *learning graphs* are attribute graphs available for the learning module in order to predict hidden links in the sensitive graph.

5 MODEL CLEANSING

The task of predicting a sensitive attribute from the other ones is made difficult by the size of datasets. Therefore, we suggest to apply the inference process only on a subset of most relevant attributes for the task. Our relevance notion does not rely on semantic proximity since we noticed that i) a user that hides a sensitive attribute will probably hide other semantically-related attributes, and moreover ii) for fully anonymised datasets the attributes semantics is hard 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 `connected_profiles_in()` 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:

$$Ul = \texttt{connected\_profiles\_in}(l)$$
$$Us = \texttt{connected\_profiles\_in}(s)$$
$$lr(l) = \texttt{size}(Ul \cap (U \setminus Us)) \ / \ \texttt{size}(U \setminus Us)$$
$$cr(l) = \texttt{size}(Ul \cap Us) \ / \ \texttt{size}(Us)$$

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

[4] https://www.kaggle.com/netflix-inc/netflix-prize-data

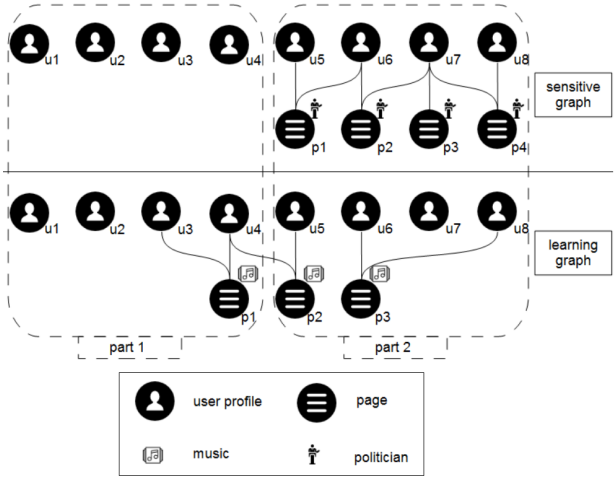

Figure 3: Example of cutting graphs for structure comparison.

the function $links_A(u_j)$ returns the set of nodes to which user node u_j is connected in the graph A.

$$J_A(u_1, u_2) = \frac{links_A(u_1) \cap links_A(u_2)}{links_A(u_1) \cup links_A(u_2)} \quad (1)$$

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

$$H(l) = \sum_{\substack{u_k, u_j \in Ul \cap Us \\ k \neq j}} |J_l(u_k, u_j) - J_s(u_k, u_j)| \quad (2)$$

In order to compare learning graphs with different sets of common connected profiles $Ul \cap Us$, 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

$$\sum_{\substack{u_k, u_j \in Ul \cap Us \\ k \neq j}} |Max(J_s(u_k, u_j), 1 - J_s(u_k, u_j))| \quad (3)$$

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 θ_{lr} since they do not convey enough information. We then discard the graphs that have a confidence rate cr lower than θ_{cr} since they are considered as unreliable. Finally, from the remaining graphs we only select graphs that have a Hamming rate hr higher than θ_{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

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.

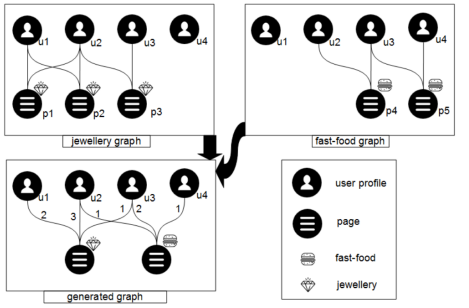

Figure 4: Merging graphs.

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 infering proximity of a user node to some sensitive attribute value node. Following [14], 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 $V_{G_r} = [lr(G_r),\ cr(G_r),\ 1 - hr(G_r)]$. The quota of G_r is given by its Mahalanobis distance to the null vector [0,0,0]. It is computed as follows:

$$q(G_r) \quad = \sqrt{V_{G_r}^T \Sigma^{-1} V_{G_r}}$$

with Σ the 3×3 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:

$$p_{u,y} = \begin{cases} \dfrac{q(G_y)}{\sum_{\{x|deg_x(u)>0\}} q(G_x)} & \text{if } deg_y(u) > 0 \\ 0 & \text{otherwise} \end{cases} \qquad (4)$$

where $deg_x(u)$ is the degree of user u in graph G_x. 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):

$$\begin{array}{lll} u \to v : & p_{u,y}/deg_y(u) & \text{if } y \neq 1 \text{ and } (u,v) \in L_y \\ v \to u : & 1/deg_y(v) & \text{if } y \neq 1 \text{ and } (u,v) \in L_y \\ u \to u' : & p_{u,1}/deg_1(u) & \text{if } (u,u') \in F \end{array} \qquad (5)$$

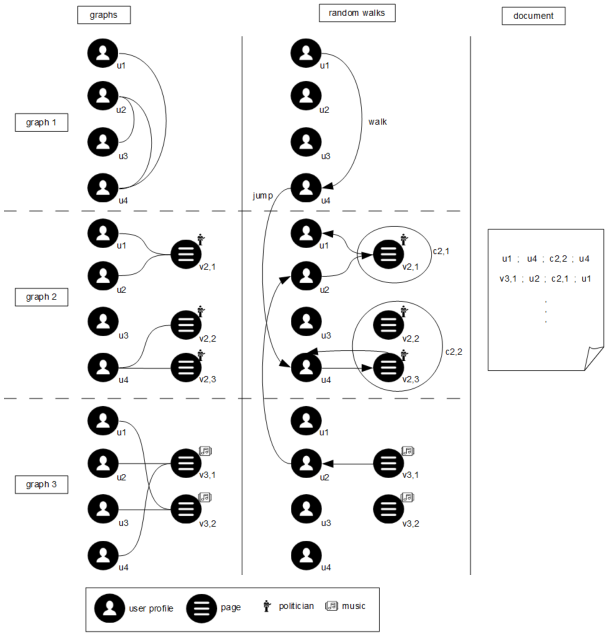

Figure 5: Example of multi graph random walk.

As illustrated in Figure 5 the document is constructed by connecting all graphs through random jumps between their nodes (see also [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 [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 [9, 13] to map one-hot encoded vectors that represent words in a high-dimensional vocabulary space to low-dimensional vectors (see [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 Skip-gram. 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-occurrent 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_1 in the Figure 6, and a popular music like "despacito", for instance the page of music $v_{2,1}$ 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 Île-de-France (Paris). Table 1 gives statistics about the two crawled datasets.

	♯ Attributes	♯ Attribute values	♯ Crawled user profiles
Dataset 1 (D 1) North-East France	1 929	1 022 860	15 012
Dataset 2 (D 2) Île-de-France	1 296	298 617	6 550

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 θ_{lr} = 20%, confidence rate greater than θ_{cr} = 60% and Hamming rate lower than θ_{hr} = 4%.

Table 2 details the 23 selected graphs relevance measures.

Attribute graph	lr (in %)	cr (in %)	hr (in %)	Number of Users	Number of Values
Users	88.37	83.98	2.08	13155	13155
Communities	44.97	98.47	1.75	8118	137338
Musicians/Bands	38.58	91.38	2.03	7141	84762
PublicFigures	32.86	92.44	1.80	6455	28289
NonProfitOrganizations	31.85	86.57	1.72	6180	25847
Artists	30.65	84.22	1.92	5970	31681
Companies	30.05	85.94	1.75	5939	20750
Websites	29.57	83.94	1.78	5829	17931
TVShows	29.48	82.41	2.31	5778	11876
EntertainmentWebsites	29.26	79.20	2.84	5669	8319
Media/NewsCompanies	29.23	87.27	1.82	5871	14042
Products/Services	27.52	80.93	1.86	5496	15986
News/MediaWebsites	27.44	83.43	1.86	5550	9247
Organizations	26.20	80.77	1.63	5328	14738
Movies	26.09	75.17	2.26	5171	16282
LocalBusinesses	24.91	78.58	1.69	5111	17321
Clothings	23.99	68.12	1.96	4729	16090
Gastronomies	23.52	71.73	2.24	4763	8422
Actors/Directors	23.12	74.54	2.78	4785	10425
Magazines	22.82	73.96	1.69	4733	9955
Athletes	22.68	68.79	2.35	4583	14123
ApplicationPages	21.68	66.36	3.04	4396	4244
SportsTeams	21.48	63.93	2.35	4309	10433

Table 2: Selected learning graphs in D1 for politicians.

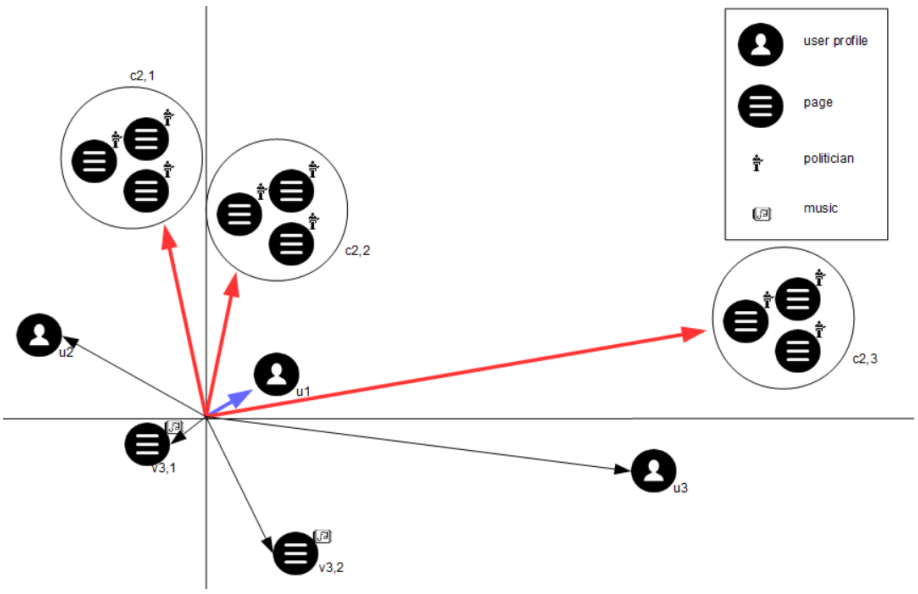

Figure 6: Example of 2-dimensional vectors that encode 8 nodes.

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 (θ_{lr} = 20%, θ_{cr} = 60%, θ_{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.

Selection based on	accuracy	targets	deleted links	nodes
Relevance (23 graphs)	0.79	252	409	558125
Random (23 graphs)	0.41	204 (average)	351 (average)	11200 (average)

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:

$R1$= $\{Single, Divorced, Separated, Widowed, Complicated\}$
$R2$= $\{Domestic\ partnership, Married, Engaged, Relationship, Civil\ union, Open\ relation\}$

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 $R1$ are focused around educations and leisures. On the other hand, discriminant attributes toward $R2$ are focused around business. The accuracy in AUC of inferring the meta-relationship status is higher than 0.7 in both datasets D1

Attribute	Importance	Discrimination
Education	2.75	88.41 % R1
Community College	2.74	90.02 % R1
Consulting Agency	2.71	90.70 % R2
Sports & Recreation	2.56	91.18 % R2
Home & Garden Website	2.49	91.89 % R2
Automotive, Aircraft & Boat	2.48	92.86 % R2
Locality	2.47	92.59 % R2
Corporate Office	2.46	91.18 % R2
News & Media Website	2.42	90.32 % R2
Financial Service	2.41	90.00 % R2
Industrial Company	2.40	89.29 % R2
Educational Consultant	2.02	75.00 % R1
Playground	1.80	66.67 % R1
Phone/Tablet	1.70	63.64 % R1
Plastic Surgeon	1.60	60.00 % R1
Consulate & Embassy	1.60	60.00 % R2
School Sports Team	1.53	52.00 % R1
Dive Bar	1.45	54.55 % R1
Video	1.44	51.00 % R1
Playlist	1.41	53.04 % R1

Table 4: Selected attributes in D2 for relationship status.

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.

Attribute	Importance	Discrimination
Sports League	4.22	75.97 % Male
Recreation & Sports Website	3.80	77.09 % Male
Video Game	3.66	80.16 % Male
Cars	3.25	73.15 % Male
Amateur Sports Team	3.03	72.86 % Male
Sport	2.80	73.07 % Male
Jewelry/Watches	2.72	56.26 % Female
Electronics	2.68	73.19 % Male
Software	2.52	77.23 % Male
Outdoor & Sporting Goods	2.35	77.19 % Male
Women's Clothing Store	2.35	77.28 % Female
Home Decor	2.29	54.60 % Female
Stadium, Arena & Sports Venue	2.28	74.45 % Male
Baby Goods/Kids Goods	2.14	66.61 % Female
Kitchen/Cooking	2.08	55.93 % Female
Bags/Luggage	2.04	59.16 % Female
Beauty, Cosmetic & Personal Care	2.03	60.59 % Female
Cosmetics Store	1.98	66.25 % Female
Hair Salon	1.92	61.44 % Female
Home & Garden Website	1.72	55.18 % Female

Table 5: Selected attributes in D1 for gender.

7.3 Processing times

Table 6 displays the processing times. The processor clock is 2.3 GHz. Cleansing and random walk algorithms are not parallelized. Cleansing takes more time than the other processes in the case of gender inference since it handles hundreds of thousands of nodes, compares hundreds of graphs to the sensitive graph and computes their importance. The random walk, in the case of gender inference, is performed on a small graph containing only a few dozen of super-values and a few thousands of user profiles. On the other hand, in the case of politicians inference, the task is performed on larger graphs containing dozens of thousands of values. The machine disposes only of 8GB of RAM memory. Each chunk of 5k steps is stored separately in a text file of about 25MB. Those files are then parsed by Word2Vec. Word2Vec speed depends on the size of the document vocabulary. It is fast in the case of gender inference since the vocabulary is limited to user profiles, super-values (i.e. clusters of values) and sensitive values.

Ranking task has to compute cosine similarity between a few vectors that represent the sensitive values and the target user vector. Cleansing task selects most important attributes and reduces the vocabulary. Consequently, this process accelerates the inference tasks (relying on Random walk and Word2Vec). Moreover, Cleansing increases accuracy by discarding irrelevant information.

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 only on 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 infering 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 SONSAI, regardless

Process			Cleansing	Random walk	Word2Vec	Ranking
Time (in seconds)	Gender inference	D1	423	34	50	0.12
		D2	243	25	30	0.12
	Politicians inference	D1	782	523	924	1
		D2	574	451	733	1

Table 6: Processing times.

lr (in %)	Inference accuracy in AUC	♯ selected graphs	♯ attacked targets	♯ masked links
[0, 10[	0.61	1873	252	409
[10, 20[	0.80	31	254	411
[20, 30[	0.86	16	254	418
[30, 40[	0.80	5	253	410
[40, 50[	0.74	1 (Communities)	251	408
[70, 80[	0.36	1 (Genders)	213	353
[80, 90]	0.64	1 (Users)	214	350

Table 7: Impact of lr on inference accuracy.

hr (in %)	Inference accuracy in AUC	♯ selected graphs	♯ attacked targets	♯ masked links
[0, 5[	0.68	1744	253	410
[5, 10[	0.59	87	177	304
[10, 20[	0.53	58	87	167
[20, 30[	0.45	11	83	129
[30, 40[	0.42	13	11	21
[40, 50[	0.42	5	2	3
[50, 100]	0.41	10	211	351

Table 8: Impact of hr on inference accuracy.

cr (in %)	Inference accuracy in AUC	♯ selected graphs	♯ attacked targets	♯ masked links
[0, 10[	0.63	1711	245	409
[10, 20[	0.43	94	246	407
[20, 30[	0.74	37	245	405
[30, 40[	0.54	28	248	404
[40, 50[	0.72	16	247	410
[50, 60[	0.38	14	250	407
[60, 70[	0.63	8	248	403
[70, 80[	0.60	6	248	393
[80, 90[	0.65	11	255	419
[90, 100]	0.82	3	253	410

Table 9: Impact of cr on inference accuracy.

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 posts 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 Foundation[5].

REFERENCES

[1] Chaabane Abdelberi, Gergely Ács, and Mohamed Ali Kâafar. 2012. You are what you like! Information leakage through users' Interests. In *19th Annual Network and Distributed System Security Symposium, NDSS 2012, San Diego, California, USA, February 5-8, 2012*. https://www.ndss-symposium.org/ndss2012/you-are-what-you-information-leakage-through-users-interests

[2] Younes Abid, Abdessamad Imine, Amedeo Napoli, Chedy Raïssi, and Michaël Rusinowitch. 2016. Online Link Disclosure Strategies for Social Networks. In *Risks and Security of Internet and Systems - 11th International Conference, CRiSIS 2016, Roscoff, France, September 5-7, 2016, Revised Selected Papers*. 153–168. https://doi.org/10.1007/978-3-319-54876-0_13

[3] Younes Abid, Abdessamad Imine, Amedeo Napoli, Chedy Raïssi, and Michaël Rusinowitch. 2017. Two-Phase Preference Disclosure in Attributed Social Networks. In *Database and Expert Systems Applications - 28th International Conference, DEXA 2017, Lyon, France, August 28-31, 2017, Proceedings, Part I*. 249–263. https://doi.org/10.1007/978-3-319-64468-4_19

[4] Anton Alekseev and Sergey I. Nikolenko. 2017. Word Embeddings for User Profiling in Online Social Networks. *Computación y Sistemas* 21, 2 (2017). http://www.cys.cic.ipn.mx/ojs/index.php/CyS/article/view/2734

[5] Michael Backes, Mathias Humbert, Jun Pang, and Yang Zhang. 2017. walk2friends: Inferring Social Links from Mobility Profiles. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*. 1943–1957. https://doi.org/10.1145/3133956.3133972

[6] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A Neural Probabilistic Language Model. *Journal of Machine Learning Research* 3 (2003), 1137–1155.

[7] Vladimir Estivill-Castro, Peter Hough, and Md Zahidul Islam. 2014. Empowering users of social networks to assess their privacy risks. In *2014 IEEE International Conference on Big Data, Big Data 2014, Washington, DC, USA, October 27-30, 2014*. 644–649. https://doi.org/10.1109/BigData.2014.7004287

[8] Fei Gao, Katarzyna Musial, Colin Cooper, and Sophia Tsoka. 2015. Link Prediction Methods and Their Accuracy for Different Social Networks and Network Metrics. *Scientific Programming* 2015 (2015), 172879:1–172879:13. https://doi.org/10.1155/2015/172879

[9] Yoav Goldberg and Omer Levy. 2014. word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method. *CoRR* abs/1402.3722 (2014).

[10] Virgil Griffith. 2007. music that makes you dumb. (2007). http://musicthatmakesyoudumb.virgil.gr/

[11] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*. 855–864. https://doi.org/10.1145/2939672.2939754

[12] R. Heatherly, M. Kantarcioglu, and B. Thuraisingham. 2013. Preventing Private Information Inference Attacks on Social Networks. *IEEE Transactions on Knowledge and Data Engineering* 25, 8 (Aug 2013), 1849–1862. https://doi.org/10.1109/TKDE.2012.120

[13] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. *CoRR* abs/1310.4546 (2013).

[14] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: online learning of social representations. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*. 701–710.

[15] Eunsu Ryu, Yao Rong, Jie Li, and Ashwin Machanavajjhala. 2013. curso: protect yourself from curse of attribute inference: a social network privacy-analyzer. In *Proceedings of the 3rd ACM SIGMOD Workshop on Databases and Social Networks, DBSocial 2013, New York, NY, USA, June, 23, 2013*. 13–18. https://doi.org/10.1145/2484702.2484706

[16] Adriaan M. J. Schakel and Benjamin J. Wilson. 2015. Measuring Word Significance using Distributed Representations of Words. *CoRR* abs/1508.02297 (2015).

[17] John Street. 2012. *Music & Politics*. Polity Press.

[5] www.fondation-maif.fr/