All liaisons are dangerous when all your friends are known to us

Daniel Gayo-Avello
University of Oviedo
Despacho 57, planta baja, Edificio de Ciencias
C/Calvo Sotelo s/n 33007 Oviedo (SPAIN)
dani@uniovi.es

ABSTRACT

Online Social Networks (OSNs) are used by millions of users worldwide. Academically speaking, there is little doubt about the usefulness of demographic studies conducted on OSNs and, hence, methods to label unknown users from small labeled samples are very useful. However, from the general public point of view, this can be a serious privacy concern. Thus, both topics are tackled in this paper: First, a new algorithm to perform user profiling in social networks is described, and its performance is reported and discussed. Secondly, the experiments –conducted on information usually considered sensitive– reveal that by just publicizing one's contacts privacy is at risk and, thus, measures to minimize privacy leaks due to social graph data mining are outlined.

Categories and Subject Descriptors

G.2.2 [**Discrete Mathematics**]: Graph Theory—*Graph algorithms,Graph labeling*; I.5.2 [**Pattern Recognition**]: Design Methodology—*Classifier design and evaluation*; K.4.1 [**Computers and Society**]: Public Policy Issues—*Privacy*

General Terms

Algorithms, Experimentation, Human Factors, Legal Aspects

Keywords

Online Social Networks, Twitter, graph labeling, privacy

1. INTRODUCTION AND MOTIVATION

Graph Labeling is the task of assigning labels to the vertices or edges of a graph. Because social networks are usually represented as graphs, vertex and edge labeling algorithms can be applied to them straightforwardly. In the former case the individuals in the network are labeled, while in the later the labels are assigned to the relationships between them.

In that context, labeling algorithms can exploit a property of social networks: the tendency of people to relate more likely with those sharing similar traits, or *homophily*. This phenomenon is pervasive to very different social networks, and it has been revealed that a number of personal characteristics –such as race and ethnicity, age, religion, education, occupation, or sex– induce homophilous relationships [13].

Thereby, homophily can be used both to cluster similar individuals within a network, or to infer attribute values for every individual from his neighbors' characteristics. In the first case –community detection– it is not necessary to know anything about the members of the social network except for their relations. In the second, the attributes of interest are needed and, hence, part of the individuals in the network must have known values for them –i.e. they must be labeled. Thus, from a machine learning perspective, the first is an unsupervised problem while the second is semi-supervised.

It is well-known that online social networks (OSNs) ask their users for personal information and that many of those users happily provide it. Hence, part of the users in OSNs are labeled and semi-supervised approaches can be employed to label the rest of the members of the network.

This paper describes a new semi-supervised algorithm to perform user profiling in social networks. A number of experiments conducted on Twitter data are reported, and the privacy implications discussed. Certainly, there exist a number of semi-supervised methods to label partially labeled social graphs. Hence, a later section reviews those most highly related to the method here proposed, and points out the main differences between this and them. Moreover, previous works regarding the privacy implications of social graph mining are also discussed. In that sense, this paper focuses on active measures the users can adopt and, thus, it outlines a protocol to minimize leakages due to graph data mining.

2. THE McC-SPLAT ALGORITHM

McC-Splat[1] is an iterative algorithm to perform vertex labeling on a partially labeled social network. It is a multiclass classifier, that is, each attribute can have more than two classes and, in fact, in addition to the predefined attribute values –e.g. *female* and *male* for *sex*– an extra class, *unknown*, is also required for each attribute.

Needless to say, individuals have a number of different attributes –e.g. sex, age, or marital status– and, hence, each

[1] Mnemonic for Multiclass Classification using Soft Labeling Propagation and Automatic Thresholding.

person would have got a *profile* comprising such attributes with their corresponding values. The *McC-Splat* algorithm can simultaneously propagate the values for each attribute in the users' profiles but, for the sake of clarity, the following description just covers the single attribute case.

McC-Splat works on a directed graph $G = \{V, E, C, A\}$ where V is the set of vertices –i.e. individuals, E denotes the edges –i.e. relationships between those individuals, $C = \{c_1, \ldots, c_m\}$ are the different classes each individual's attribute can take, and finally A is the set of attribute weight vectors for the individuals in V.

By using attribute weight vectors it would be possible to model overlapping classes; nevertheless, all of the experiments reported here were conducted with disjoint classes. Moreover, the weights can be seen as a proxy for the user's likelihood to belong to a given class –including the *unknown* one– although weights are not probabilities in a strict sense. The following formalization describes the A set:

$$A \subset [0,1]^{m+1} \, | m = |C| \, , \forall \mathbf{a} \in A : |\mathbf{a}| = 1$$

Given that G is partially labeled, V is divided in two disjoint sets: the set V^K of known vertices –i.e. those individuals with a known class value for the attribute– and the set of unknown vertices V^U –i.e those belonging to the *unknown* class. Because the attribute values can be taken from m different classes this can be formalized as:

$$V = V^K + V^U$$

$$V^K = \{v_i \in V | \exists j : 1 \leqslant j \leqslant m, \mathbf{a}_{\mathbf{i}j} = 1\}$$

$$V^U = \{v_i \in V | \neg \exists j : 1 \leqslant j \leqslant m, \mathbf{a}_{\mathbf{i}j} = 1\}$$

Finally, a definition for the neighborhood of each vertex is needed. In this regard it must be noted that (1) this algorithm assumes directed graphs, and (2) it only considers as neighbors of a person those people related to the first one via a relationship started by that person.

For instance, in a phone network those numbers a user makes calls to would be neighbors, but not the numbers from which he receives calls; in the blogosphere the neighborhood would comprise the blogs a given blog links to, but not those linking to that blog; in Twitter the neighbors would be those users a given user is following, but not his followers.

Thereby, the neighborhood for vertex $v_i \in V$ would be:

$$N_i = \{v_j \in V | \exists e_{ij} \in E\}$$

All of this defines the input graph but not the way in which the algorithm works on it. As it has been said, it is an iterative algorithm and, hence, at its core there is an operation to compute new weights for each vertex attribute vector from its neighbors weights in the previous iteration. It must be noted that only the weights for vertices belonging to V^U are updated, those from the originally labeled set V^K are not assigned new weights:

$$\forall v_i \in V^U : \mathbf{a_i}^{(t)} = \frac{1}{Z} \sum_{v_j \in N_i} \mathbf{a_j}^{(t-1)}$$

$$\forall v_i \in V^K : \mathbf{a_i}^{(t)} = \mathbf{a_i}^{(0)}$$

In the previous formalization Z is a normalizer.

McC-Splat, like other graph iterative algorithms, converges after relatively few iterations. Hence, once weight vectors have stabilized –or after a predefined number of iterations– a large part of vertices in V^U have got weight vectors for the attribute of interest. Other algorithms would then assign to each vertex the label with the highest weight within the vector, or would require an *ad hoc* threshold to be defined for each class value. *McC-Splat*, instead, introduces two extra steps which can be used to achieve automatic thresholding in a number of ways.

First of all, a fictitious sink vertex can be introduced. Such a vertex would represent an individual related to every single person within the social network. The weights for that vertex are computed after the last iteration and they provide a measure of which weights could be expected for a user without homophilous relationships. The usefulness of such an approach is clear when a large majority of people belongs to a single class; if that prevalence is not taken into account most of the unknown individuals would be incorrectly assigned to the majority class. This equation defines the weight vector for such a sink vertex:

$$\mathbf{s}^{(T+1)} = \frac{1}{Z} \sum_{v_i \in V} \mathbf{a_i}^{(T)}$$

Once the sink vertex weights are computed they can be used in two ways: (1) vertices from V^U can be assigned the label with the highest weight which is *also* above the corresponding weight in the sink vector; or (2) vertices from V^U can be assigned the label with the weight which most largely departs –in percentage value– from the corresponding weight in the sink vector.

The second approach to automatic thresholding requires to compute an alternative weight vector for the members of V^K. As it has been said, those vertices' vectors have got one single component with a unity value –i.e. the component corresponding to the class each individual belongs to– and their vectors are not modified as the algorithm iterates. However, it is possible to compute from their corresponding neighborhoods the weights they would have whether they had belonged to V^U:

$$\forall v_i \in V^K : \mathbf{a'}_{\mathbf{i}}^{(T+1)} = \frac{1}{Z} \sum_{v_j \in N_i} \mathbf{a_j}^{(T)}$$

By doing that it is possible to produce a reverse-ordered ranking of individuals for each of the class values the attribute can take. That way, instead of defining an *ad hoc* threshold to decide if a weight is high enough to accept the induced label, it is possible to find different weights at different percentile values.

Thereby, when using *McC-Splat* it is not needed to take *ad hoc* weight thresholds; instead, the confidence required from the labeled output can be chosen. For instance, by choosing the 90th percentile only those members of V^U whose weights were above 90% of the weights of V^K members would appear in the output labeled set.

So, in short, *McC-Splat* comes in the following flavors: (1) *Plain-vanilla*, the class with the highest weight is assigned[2]. (2) *Sink-absolute*, the class with the highest weight

[2] The *unknown* class is ignored, otherwise all of the vertices would remain unknown unless the number of labeled examples surpassed the number of unknown vertices.

and above the corresponding weight within the sink node is assigned. (3) *Sink-relative*, the class with the highest positive difference against the corresponding weight within the sink node is assigned. (4) *Percentile*, the class with the highest percentile –according to the labeled individuals– is assigned. Optionally, a minimum value –e.g. 90%– can be forced or, otherwise, the *unknown* class is assigned.

Now, the algorithm's name should be self-explained: it is a multiclass classifier which iteratively propagates weight vectors to every node from its neighborhood; because each node's vector roughly represent its likelihood of belonging to each class, the labeling is performed in a "soft" rather than in a "hard" way; moreover, the algorithm provides alternatives to automatically determine the most reliable class for each node, making *ad hoc* thresholds unnecessary.

3. EXPERIMENTAL EVALUATION

3.1 Dataset Description

Social network data was needed to test the performance of *McC-Splat*. The graph depicting relationships between individuals was essential but, in addition to that, a part of those users had to be labeled.

Hence, the Twitter[3] dataset collected in [2] was used. It comprises 27.9 million English-written tweets published from January 26 to August 31, 2009 by 4.98 million users.

Followers and followees for each of the users in that dataset were also collected. Links to users not appearing in the dataset were disregarded, and isolated users were removed. Furthermore, a substantial amount of user accounts were suspended at the moment of the graph crawl and, hence, no information on them was available. Lastly, because of the unavoidable network problems, coupled with the fact that the API was pushed a little too far, the information for a noticeable amount of users was not eventually crawled.

Thus, the user graph consisted of 1.8 million users with their corresponding links and profiles –i.e. full name, short biography, location, etc. Given that at the moment of collecting the dataset, the number of Twitter users in the U.S. was estimated between 14 and 18 millions[4], and that most of the crawled users were supposed to be from the U.S. it can be considered a rather substantial sample.

3.2 Labeling Twitter users

Unlike other OSNs such as Facebook, Twitter profiles do not provide highly structured information; there is no way, for instance, to indicate the user's sex or age. Instead, Twitter profiles consist of the user's full name, location, website, and a short biography. All of these fields are free text and there is a high disparity in their use. For example, 62.31% of the users in the dataset provide a location string, but only 36.46% provide their full personal name [2].

This does not mean that no personal information can be extracted from Twitter profiles. Quite to the contrary, using the location, full name, and biography strings, half of the users in the dataset were geolocated, the sex of one third of them was found, in addition to the age for about 11,000 [2]. Needless to say, the data was noisy, and the labeling methods a bit rough; though, anecdotal evidence revealed a quite accurate big-picture of Twitter demographics.

Therefore, a similar approach was employed to label users according to a number of personal traits. In addition to sex and age, the following attributes[5] were also chosen: *political orientation*, *religious affiliation*, *race and ethnicity*, and *sexual orientation*. All of them are usually considered sensitive information, and most countries have enacted laws against discrimination based on any of such attributes. In spite of this, many people still feel the need to hide those personal details. Thus, it is important to find out the degree in which such individuals can be inadvertently exposed because of their acquaintances.

All of the classes, except those corresponding to sex, were determined by means of pattern matching (see table 1 for the patterns applied). Firstly, each class name was used to obtain a initial list of users. For instance, the patterns `democrat*` and `republican*` were used to find users self-defined as Democrats or Republicans. Once there was a preliminary list of users for each class, their biographies were mined to find the most frequent keywords which could be considered indicative of class belonging. That way, for example, patterns such as `lib-dem*` or `dems*` were found for Democrats, and `conservat*` or `tea party` for Republicans.

Certainly, such a labeling method is error prone but the goal was to obtain the largest[6] possible labeled set for each class and attribute. Because of the nature of *McC-Splat*, it was assumed that large although noisy data was preferable to cleaner but small samples. After all, should the results be encouraging, better labeling approaches could be used.

Finally, the labeled sets were split into *training* and *tests* partitions: the former consisted of a random selection of 80% of the users in each class and was used as input for the algorithm; the later comprised the remaining 20% of the users and was left out for evaluation.

3.3 Results

McC-Splat was applied to the Twitter graph in each of its four different "flavors" just considering the users in the training partitions. That way, labels were obtained for the rest of the users in the graph including those in the test partitions. Then, by comparing the algorithm's class assignments for those users with the actual class belonging according to their biographies, precision and recall figures were computed (see table 4). For comparison purposes, the performance of a random classifier based on the proportion of each of the different classes is shown in table 2.

In addition to that first experiment, a second one was conducted on another independently labeled set. To that end, data was collected from *WeFollow*[7] which is a Twitter

[3] Twitter is a microblogging and social networking service. Users publish short text messages (*tweets*) which are shown to all of their followers. Relationships in Twitter are asymmetrical and, thus, a user has got followers and *followees.*

[4] `http://www.socialtimes.com/2009/04/twitter-14-million/`, `http://mashable.com/2009/09/14/twitter-2009-stats/`

[5] All of the chosen attributes, except for *race/ethnicity*, appear in Facebook profiles and, thus, it would not be surprising to find information on them in Twitter biographies.

[6] Table 1 reveals that the number of labeled examples obtained was rather low for all of the attributes and, unsurprisingly, the more sensitive the attribute, the fewer users disclose information about it in their biographies.

[7] `http://wefollow.com`

Table 1: Classes for each of the six personal attributes along the rules applied to label Twitter users according to them. All of the labels, except for *sex* were obtained by pattern-matching the users' biographies. The age intervals were those used by [5].

Attribute	Class	Rule or pattern	# users
sex	female	User name had to be composed of first and last name from the U.S. Census. Sex was assigned according to frequency of use of the first name in U.S. population.	271,539
	male		384,574
age	teenage	Age was extracted from the user's bio looking for the patterns *year-old* or *years old* preceded by a number or a numeral. Then, ages <18, 18-24, 25-34, 35-49, and >49 were assigned to each class.	3,483
	youngster		4,562
	young		1,911
	mid-age		663
	elder		296
political orientation	democrat	**democrat***, lib-dem*, libdem*, dems*	248
	republican	conservat*, gop, g.o.p., palin, pro-life*, prolife*, **republican***, right-wing, rightish, tcot, tea-party, teaparty	2,040
religious affiliation	atheist	agnost*, anti-theis*, antitheis*, ateus, **atheis***, athiest*, empiricist*, godless*, heathen*, humanism*, humanist*, irreligion*, non-believer*, non-theist*, nonbeliever*, nontheist*, pagan*, rational*, sceptic*, secular*, skepchicks, skeptic*	330
	buddhist	**buddh***, dhamma*, dharma*, sangha, twangha, vipassana, yoga*, yoginis, yogis, zen	204
	christian	adventist*, anglican*, baptist*, cathol*, cattolici, **christ***, church*, evangelical, gospel*, jesus*, lutheran*, methodist*, minister*, ministries*, ministry*, pastor*, pentecostal*, preacher*, presbyterian*, priest*	8,103
	jewish	circumcision, israel*, jerusalem, jew, **jewish**, jews, judaism, jude, kosher, rabbi, sephardic, synagogue*, torah, yiddish, zion*	458
	muslim	imam, islam*, isulamic*, mosque*, **muslim***, quran, salaam, tweeplims	171
race/ethnicity	asian-american	**asian***, chinese-american, filipin*, hindu*, india, indian-american, japan, japanese-american, korea*, taoism, vietnam*	65
	black	africa*, **black**, black-american, black-man, black-woman, hip-hop*, hiphop*	202
	hispanic	amigo, belleza, familia, favoritos, gente, **hispanic**, latina, latino, mexico	6
	native-american	aboriginal, alaska-native, american-indian, first-nation, firstnation, indigenous*, native american, **native-american**	80
	native-hawaiian	aloha, hawaii*, honolulu, native hawaiian, **native-hawaiian**, oahu, ohana	4
	white	caucasian, **white**, white-american, white-man, white-woman	24
sexual orientation	heterosexual	**hetero***	15
	homosexual	bisexual*, gay*, glbt, glsen, gltb, homo-*, **homosex***, l-word, lesbian*, lgbt, lgbtq, marriage-equality, queer, transgender	1,471

user directory where users classify themselves according to the topics they are interested in. Each topic is represented by a tag, and a list of users following each tag can be obtained[8].

Hence, most of the patterns from table 1 were employed to obtain lists of users from WeFollow[9]. Needless to say, not every user in those lists appeared in the Twitter user graph and, therefore, those users not appearing in the graph, in addition to those already labeled –i.e. appearing in the training and test partitions– were removed.

Performance results on this second dataset for both the random classifier and the *McC-Splat* algorithm can be seen in tables 3 and 5, respectively.

3.4 Discussion of Results

As it can be seen from tables 4 and 5 the performance of *McC-Splat* was notably high. Average precision and accuracy figures were quite similar, implying that performance across classes within the same attribute is comparable and, thus, there was no much bias towards the prevalent classes.

Attributes such as *religious affiliation*, *political orientation*, *sexual orientation*, and *race and ethnicity* achieved above 95% precision when evaluating on the test partitions. Results in the WeFollow dataset were very similar, except for *race/ethnicity* where precision dropped to 50% and accuracy to 71%.

The poorest results were achieved when assigning *sex* and *age*: 62% and 43% macro-averaged precision, respectively. With regards to *age*, maybe it was problematic because it is actually a continuous variable. After reviewing the actual classifications it was found that most of the errors were due to assigning users to nearby classes –e.g. classifying *teenagers* as *youngsters*, *youngsters* as *youngs*, etc.

All in all, *McC-Splat* clearly outperformed the random classifier by an exceedingly large margin although, certainly, when an attribute has got a clearly prevalent class it is much more difficult to outperform it. In the presence of such prevalent classes the random classifier achieved good accuracy but also poor macro-averaged precision; *McC-Splat*, instead, was not very affected by such prevalent classes and it exhibited comparable precision across classes.

Regarding the different "flavors", the *Plain-vanilla* version did not outperform the random classifier for prevalent classes (e.g. *male* vs *female*, *young* vs the rest of *age* intervals, and *christians* vs the rest of *religious affiliations*), and it even underperformed when classifying *homosexual* individuals. The rest of the "flavors" clearly outperformed the random classifier –even for prevalent classes– and they consistently achieved high performance figures. Therefore, *Plain-vanilla* could be disregarded and additional experiments are required to find which of the other three alternatives can be the best choice. In this regard, better labeled data –in particular for large majority classes– is also needed.

4. RELATED WORK

As it has been said, *McC-Splat* is a semi-supervised graph labeling algorithm based on label propagation. There are other algorithms which are somewhat similar and, hence, those most highly related are to be briefly reviewed.

Maybe the best known iterative graph algorithm is PageRank [15], it computes for each vertex –generally a web page– a score which corresponds to its relevance within the network. Its popularity has spurred the use of similar methods in many other scenarios –e.g. to fight spam in the Web [4].

With regards to the use of the graph structure to perform classification, one of the earliest works was a hypertext classifier [1]. In this case, however, the links were used to improve the classifier but other clues –such as the documents content– were also required.

Much more related to *McC-Splat* are the works described in [11, 14]. In [14] it is described an iterative application of Bayesian classifiers where the objects attributes were modified from the inferences made on their neighbors in each iteration. In [11] the so-called wvRN[10] method is described. That algorithm works on undirected weighted graphs and just relies on the objects labels and relationships. It estimates the probability of an object belonging to a given class as the weighted proportion of its neighbors that belong to that class and, then, the majority label is assigned after each iteration.

Although related, there are several differences between wvRN and *McC-Splat*: the later works on unweighted directed graphs, labels are not assigned by majority vote but, instead, weight vectors are propagated. Besides, in the absence of labeled neighbors wvRN assigns label on the basis of the class priors –i.e. a random classifier– while *McC-Splat* assigns the *unknown* class. Finally, the use of a sink node and the estimation of weight vectors for the labeled examples to perform auto-thresholding are novel additions which could be compared to cautious classification [12].

As it has been said, data mining users' relationships in OSNs raises some concerns and, in fact, this study have exposed the privacy risks due to the public nature of those relationships. Hence, this work has got some points of similarity with a number of recent studies on privacy in OSNs.

It has been shown, for instance, that different kind of attacks can be conducted on the basis of known relationships and group memberships [18, 19], and a number of studies have provided additional support for those findings in Facebook –e.g. [6, 8].

It has been stated that privacy attacks can be successful when *"as much as half of the profiles are private"* [18]. However, this study has revealed that the number of required known users is, in fact, much lower –well below 1% for a sample of 1.8 million users– and the achieved precision is much higher than the one reported in [18]. Thereby, privacy issues because of publicizing acquaintances in OSNs should be a major concern for their users.

Finally, a few pertinent works on measures to improve privacy in OSNs are referenced to provide context for the protocol described in the last section.

At least two different Facebook applications relying on public key cryptography to store obfuscated information in the OSN servers have been proposed [9, 10]. By doing that users can still make use of the OSN services but their personal information is decrypted on the client side and, thus,

[8] For instance, `http://wefollow.com/twitter/democrat` gives access to a list of users self-defined as Democrat, while `http://wefollow.com/twitter/republican` provides a list of Republican users.

[9] Sex and age were not able to be tested with data from WeFollow.

[10] Weighted-vote Relational Network classifier.

Table 2: Performance of a random classifier based on the proportion of each class in the labeled data and working on the same labeled data.

Attribute	P=R=F_1		Class	P=R=F_1
sex	Micro-avg.	0.5148	female	0.4139
	Macro-avg.	0.5	male	0.5861
age	Micro-avg.	0.3116	teenage	0.3191
			youngster	0.4180
			young	0.1751
	Macro-avg.	0.2	mid-age	0.0607
			elder	0.0271
religious affiliation	Micro-avg.	0.7693	atheist	0.0356
			budhist	0.0220
			christian	0.8745
	Macro-avg.	0.2	jewish	0.0494
			muslim	0.0185
political orientation	Micro-avg.	0.8068	democrat	0.1084
	Macro-avg.	0.5	republican	0.8916
sexual orientation	Micro-avg.	0.9798	heterosexual	0.0101
	Macro-avg.	0.5	homosexual	0.9899
race/ethnicity			asian-american	0.1706
	Micro-avg.	0.3586	black	0.5302
			hispanic	0.0157
			native-american	0.2100
	Macro-avg.	0.1667	native-hawaiian	0.0105
			white	0.0630

Table 3: Performance of a random classifier based on the proportion of each class in the labeled data and working on the WeFollow dataset.

Attribute		P	R	F_1	Class	P	R	F_1
religious affiliation	Micro-avg.		0.6843		atheist	0.0872	0.0356	0.0506
					budhist	0.0513	0.0220	0.0308
					christian	0.7741	0.8745	0.8213
	Macro-avg.		0.2		jewish	0.0484	0.0494	0.0489
					muslim	0.0389	0.0185	0.0250
political orientation	Micro-avg.		0.7966		democrat	0.1213	0.1084	0.1145
	Macro-avg.		0.5		republican	0.8787	0.8916	0.8851
sexual orientation	Micro-avg.		0.9899		heterosexual	0	1	0
	Macro-avg.	0.5	0.9950	0.6655	homosexual	1	0.9899	0.9949
race/ethnicity					asian-american	0.6022	0.1706	0.2659
	Micro-avg.		0.2119		black	0.1853	0.5302	0.2746
					hispanic	0.1735	0.0157	0.0289
					native-american	0.0391	0.2100	0.0659
	Macro-avg.	0.1667	0.4878	0.2484	native-hawaiian	0	1	0
					white	0	1	0

it is inaccessible for the OSN operator. Needless to say, encrypted text is relative easy to detect and, thus, a "hostile" OSN operator could disable accounts using such a measure.

Because of that, it has been proposed to use instead a dictionary known to the members of a group [3]. Such a dictionary would provide a way to replace "atoms" of personal information with atoms from other users. For instance, the name, age, or sex of a user would be stored in such a way that they still resemble personal information but cannot be linked to the actual individual. By using the dictionary, the group members could translate that fake information into the actual attributes of their acquaintance. Purportedly, this measure is much more difficult to detect than cryptography and, thereby, it could be applied even when using the services provided by "hostile" OSN operators [3].

5. IMPLICATIONS AND CONCLUSIONS

5.1 Implications for Users Privacy

Users sensitive information, such as political or religious beliefs, race and ethnicity, or sexual orientation can be determined with notable precision from their neighbors with rather simple algorithms. Thereby, it does not matter if users do not self-disclose personal traits, they can be inadvertently exposed because of acquaintances who do not conceal such information.

Most works on privacy in OSNs have mainly focused on ways to guarantee that released datasets do not put at risk the users' privacy –e.g. [17]. Certainly, such anonymization measures may dispel some concerns the operators of OSNs can have about releasing data for research purposes. However, it is not at all necessary to obtain the data from the operator of the OSN, but it is relatively easy to collect using the available APIs. Therefore, in spite of anonymization methods, users of OSNs are fully exposed to any third party aiming to data mine social graphs.

A trivial solution for that problem would be, of course, to disable the APIs. This, however, is unlikely to happen because it would be contrary to the interests of the operators of the OSNs. In addition to that, it would just make difficult[11] for third parties to mine the users data but would not prevent the operator of the OSN and licensed third parties from doing it.

A number of works, some of them referenced in the previous section, propose users to encrypt the information they submit to the system. Needless to say, making the users information opaque for the OSN would put at risk their current business models which, to a great or lesser extent, revolve around marketing and personalization. Thereby, it does not seem unreasonable to assume that if encryption went mainstream among OSN users, the operators of the services would force users to use plain text.

5.2 Minimizing Data Mining Risks

So, to sum up, graph anonymization is an unreliable passive[12] measure, and heavy use of cryptography, an active user's measure, could be easily disallowed by the operators of the OSNs. Hence, procedures to minimize privacy leakages should be active and keep the use of cryptography to a minimum; some hints on such a prophylactic protocol are provided here.

First of all, the following protocol has been devised for asymmetrical social networks in general, and Twitter in particular. Secondly, users are responsible for the information they disclose on themselves; that is, the purpose of this protocol is not to protect their privacy regardless of their actions, but to minimize the likelihood of being exposed because of their relationships. In third place, users cannot control who is following them but who they follow. It has been shown that these relationships are risky and, thus, identifiable accounts cannot be used to follow anybody.

Needless to say, the network is useless if users are isolated and, thereby, they need a mechanism to follow other users. To that end, a second account is to be used. The nickname should be a totally random string, and no information should be provided other than a public key. This *anonymous* account –in contrast to the previous *identified* account– would not be used to post messages other than mentions to followees, and it would not accept followers.

Obviously, using two different accounts would be pointless if they can be linked to each other by means of the IP address. Therefore, the anonymous account should connect to the service through an anonymizing service such as *Tor*[13] or *I2P*[14] while this is not necessary for the identified account.

With regards to message publishing, those not mentioning any account or mentioning an identified account could be published unencrypted. After all, users are responsible for what they publish on themselves, and cannot control the messages other users address to them. However, if the message is a reply from an identified account to an anonymous account it should be fully encrypted using the public key corresponding to the anonymous account. The reason for this is to avoid eavesdroppers to find out implicit links starting on identified accounts.

The most cumbersome part would be the one regarding the exchange of *credentials* between anonymous and identified accounts. Such an exchange would be needed to allow users to follow their followers. As it has been said, anonymous accounts are not for publishing messages and, thus, they would be of no interest. However, after receiving a new follower, that anonymous account is the only piece of information the user has got to reach the follower's identified account. Hence, the user receiving a new follower should publish his or her public key encrypted with the public key of the new follower. The follower would publish, in return, the nickname for his or her identified account encrypted with the public key of the followee. At that point, the followee could use his anonymous account to start following the identified account of his new follower.

Clearly, that chain of actions would allow an eavesdropper to link anonymous and identified accounts. Thus, to avoid it, the exchange of credentials could be made at pre-scheduled hours. In addition to this, it must be clear that this protocol does not aim to maintain users anonymous from each other but to conceal their relationships from third parties observing the social network –including its operators.

[11] Several ways in which an attacker can obtain information on network relationships by compromising a number of user accounts are described in [7].

[12] Passive, that is, from the point of view of the users.

[13] `https://www.torproject.org/`

[14] `http://www.i2p2.de/`

Finally, all of these measures should be implemented by client software in such a way that the user could use the OSN transparently.

5.3 Final Remarks and Future Work

A new algorithm to perform user profiling in social networks, *McC-Splat*, has been described. The new method is related to other known algorithms but, unlike them, it does not require *ad hoc* thresholds but, instead, it provides a number of alternatives to perform auto thresholding from the input labeled data.

A number of experiments were conducted to test its performance. Results from those experiments have been reported, revealing that *McC-Splat* largely outperforms a random classifier and, in fact, achieves a notably high precision for very different classes and attributes. Nevertheless, further experiments are needed to determine which of the different "flavors" of the algorithm is the best choice, in addition to test the algorithm on data from OSNs other than Twitter, and labeled by different means.

The attributes employed for the experiments are usually considered sensitive personal information and, thus, the experiments had an additional outcome: exposing the risk that acquaintances suppose for users which can be exposed even without revealing any personal information on themselves.

Thereby, a prophylactic protocol to minimize leakages due to graph data mining was outlined. Further work is needed in this regard: a prototype implementation is highly needed; in addition to field studies regarding its use by real users, and analyzing its sensitiveness to different kind of attacks –mainly those based on infiltration.

6. REFERENCES

[1] Soumen Chakrabarti, Byron Dom, and Piotr Indyk. Enhanced hypertext categorization using hyperlinks. *SIGMOD Rec.*, 27(2):307–318, 1998.

[2] Daniel Gayo-Avello. Nepotistic relationships in twitter and their impact on rank prestige algorithms. *CoRR*, abs/1004.0816, 2010.

[3] Saikat Guha, Kevin Tang, and Paul Francis. NOYB: privacy in online social networks. In *WOSP '08: Proceedings of the first workshop on Online social networks*, pages 49–54, New York, NY, USA, 2008. ACM.

[4] Zoltán Gyöngyi, Hector Garcia-Molina, and Jan Pedersen. Combating web spam with trustrank. In *VLDB '04: Proceedings of the Thirtieth international conference on Very large data bases*, pages 576–587. VLDB Endowment, 2004.

[5] Jian Hu, Hua J. Zeng, Hua Li, Cheng Niu, and Zheng Chen. Demographic prediction based on user's browsing behavior. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 151–160, New York, NY, USA, 2007. ACM.

[6] Carter Jernigan and Behram F. T. Mistree. Gaydar: Facebook friendships expose sexual orientation. *First Monday*, 14(10), 2009.

[7] Aleksandra Korolova, Rajeev Motwani, Shubha U. Nabar, and Ying Xu. Link privacy in social networks. In *Proceeding of the 17th ACM conference on Information and knowledge management*, CIKM '08, pages 289–298, New York, NY, USA, 2008. ACM.

[8] Jack Lindamood, Raymond Heatherly, Murat Kantarcioglu, and Bhavani Thuraisingham. Inferring private information using social network data. In *WWW '09: Proceedings of the 18th international conference on World wide web*, pages 1145–1146, New York, NY, USA, 2009. ACM.

[9] Matthew M. Lucas and Nikita Borisov. FlyByNight: mitigating the privacy risks of social networking. In *WPES '08: Proceedings of the 7th ACM workshop on Privacy in the electronic society*, pages 1–8, New York, NY, USA, 2008. ACM.

[10] Wanying Luo, Qi Xie, and Urs Hengartner. Facecloak: An architecture for user privacy on social networking sites. *Computational Science and Engineering, IEEE International Conference on*, 3:26–33, 2009.

[11] S. Macskassy and F. Provost. A simple relational classifier. In *Workshop on Multi-Relational Data Mining in conjunction with KDD-2003 (MRDM-2003), Washington, DC, 2003*, pages 64–76, 2003.

[12] Luke K. McDowell, Kalyan M. Gupta, and David W. Aha. Cautious Collective Classification. *J. Mach. Learn. Res.*, 10:2777–2836, 2009.

[13] Miller McPherson, Lynn Smith-Lovin, and James M Cook. Birds of a feather: Homophily in social networks. *Annual Review of Sociology*, 27(1):415–444, 2001.

[14] J. Neville and D. Jensen. Iterative classification in relational data. In *In Proc. AAAI*, pages 13–20. AAAI Press, 2000.

[15] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank Citation Ranking: Bringing Order to the Web. Technical report, Stanford Digital Library Technologies Project, 1998.

[16] Karen Spärck-Jones. Automatic indexing. *Journal of Documentation*, 30:393–432, 1974.

[17] Xiaowei Ying and Xintao Wu. On link privacy in randomizing social networks. In Thanaruk Theeramunkong, Boonserm Kijsirikul, Nick Cercone, and Tu-Bao Ho, editors, *Advances in Knowledge Discovery and Data Mining*, volume 5476 of *Lecture Notes in Computer Science*, pages 28–39. Springer Berlin / Heidelberg, 2009.

[18] Elena Zheleva and Lise Getoor. How friendship links and group memberships affect the privacy of individuals in social networks. Technical report, University of Maryland, College Park, 2008.

[19] Elena Zheleva and Lise Getoor. To join or not to join: the illusion of privacy in social networks with mixed public and private user profiles. In *WWW '09: Proceedings of the 18th international conference on World wide web*, pages 531–540, New York, NY, USA, 2009. ACM.

Table 4: Performance figures for the six attributes and the four different "flavors" of the *McC-Splat* algorithm working on the Twitter dataset. Details for each individual class are provided in addition to aggregated figures: both micro- and macro-averaged. Micro-averaged precision is equivalent to the accuracy of the classifier for each attribute. Figures in bold correspond to "material" performance improvements against the random classifier –i.e. larger than 10%, according to the criterion proposed by [16].

Attribute	Class	Plain-vanilla			Sink-absolute			Sink-relative			Percentile		
		P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
sex	female	**0.6307**	0.1091	0.186	**0.6149**	0.3135	0.4153	**0.6125**	0.3356	0.4337	**0.6174**	0.2432	0.3489
	male	0.6039	0.8852	0.718	**0.6803**	0.4792	0.5623	**0.6907**	0.4679	0.5579	**0.6765**	0.3318	0.4452
	Macro-avg.	**0.6173**	0.4972	0.5507	**0.6476**	0.3964	0.4917	**0.6516**	0.4018	0.497	**0.6469**	0.2875	0.3981
	Micro-avg	**0.606**	0.564	0.5842	**0.6582**	0.4106	0.5057	**0.6623**	0.4132	0.5089	**0.6551**	0.2951	0.4069
age	teenage	**0.533**	0.1392	0.2207	**0.5112**	0.1636	0.2478	**0.5398**	0.175	0.2644	**0.5989**	0.1564	0.248
	youngster	0.4438	0.8697	0.5877	**0.5**	0.2267	0.312	**0.5375**	0.1961	0.2873	**0.5464**	0.1742	0.2641
	young	**0.3607**	0.0574	0.0991	**0.2825**	0.1305	0.1786	**0.2458**	0.1149	0.1566	**0.2411**	0.0705	0.1091
	mid-age	**0.3**	0.0226	0.042	**0.1441**	0.1278	0.1355	**0.135**	0.1654	0.1486	**0.1897**	0.0827	0.1152
	elder	**0.5**	0.0167	0.0323	**0.0857**	0.1	0.0923	**0.0792**	0.1333	0.0994	**0.0476**	0.0167	0.0247
	Macro-avg.	**0.4275**	0.2211	0.2915	**0.3047**	0.1497	0.2008	**0.3075**	0.1569	0.2078	**0.3247**	0.1001	0.153
	Micro-avg.	**0.4486**	0.4195	0.4336	**0.3932**	0.1802	0.2472	**0.3743**	0.1715	0.2353	**0.4623**	0.1404	0.2154
religious affiliation	atheist	**1**	0.2576	0.4096	**0.4719**	0.6364	0.5419	**0.4699**	0.5909	0.5235	**0.6579**	0.3788	0.4808
	budhist	**1**	0.2195	0.36	**0.6667**	0.6341	0.65	**0.4143**	0.7073	0.5225	**0.5769**	0.3659	0.4478
	christian	0.9174	0.9186	0.918	**0.9899**	0.7896	0.8785	**0.9926**	0.7409	0.8485	**0.9928**	0.768	0.8661
	jewish	**0.9592**	0.5109	0.6667	**0.6495**	0.6848	0.6667	**0.617**	0.6304	0.6237	**0.8154**	0.5761	0.6752
	muslim	**1**	0.4571	0.6275	**0.8**	0.6857	0.7385	**0.2747**	0.7143	0.3968	**0.8571**	0.5143	0.6429
	Macro-avg.	**0.9753**	0.4727	0.6368	**0.7156**	0.6861	0.7005	**0.5537**	0.6768	0.6091	**0.78**	0.5206	0.6245
	Micro-avg.	**0.9207**	0.8507	0.8843	**0.927**	0.7736	0.8434	**0.8734**	0.7288	0.7946	**0.9658**	0.731	0.8322
political orientation	democrat	**1**	0.26	0.4127	**0.85**	0.34	0.4857	**0.6905**	0.58	0.6304	**0.7045**	0.62	0.6596
	republican	0.9157	0.9583	0.9365	0.944	0.9093	0.9263	0.973	0.8848	0.9268	**0.9808**	0.875	0.9249
	Macro-avg.	**0.9579**	0.6092	0.7447	**0.897**	0.6247	0.7365	**0.8318**	0.7324	0.7789	**0.8427**	0.7475	0.7922
	Micro-avg.	**0.9182**	0.8821	0.8998	**0.9395**	0.8472	0.8909	**0.9443**	0.8515	0.8955	**0.951**	0.8472	0.8961
sexual orientation	heterosexual	1	0	0	1	0	0	1	0	0	1	0	0
	homosexual	*0.9892*	0.9418	0.9649	1	0.8116	0.896	1	0.8116	0.896	1	0.8116	0.896
	Macro-avg.	**0.9946**	0.4709	0.6392	**1**	0.4058	0.5773	**1**	0.4058	0.5773	**1**	0.4058	0.5773
	Micro-avg.	0.9892	0.9322	0.9599	1	0.8034	0.891	1	0.8034	0.891	1	0.8034	0.891
race/ethnicity	asian-american	**0.8571**	0.4615	0.6	**0.8571**	0.4615	0.6	**0.75**	0.4615	0.5714	**0.8571**	0.4615	0.6
	black	**0.9412**	0.7805	0.8533	**0.9412**	0.7805	0.8533	**0.9412**	0.7805	0.8533	**0.9412**	0.7805	0.8533
	hispanic	1	0	0	1	0	0	1	0	0	1	0	0
	native-american	**0.9231**	0.75	0.8276	**0.9231**	0.75	0.8276	**0.9167**	0.6875	0.7857	**0.9231**	0.75	0.8276
	native-hawaiian	1	0	0	1	0	0	1	0	0	1	0	0
	white	1	0	0	1	0	0	1	0	0	1	0	0
	Macro-avg.	**0.9536**	0.332	0.4925	**0.9536**	0.332	0.4925	**0.9347**	0.3216	0.4785	**0.9536**	0.332	0.4925
	Micro-avg.	**0.9259**	0.641	0.7576	**0.9259**	0.641	0.7576	**0.9074**	0.6282	0.7424	**0.9259**	0.641	0.7576

Table 5: Performance figures of the four "flavors" of the *McC-Splat* algorithm working on the *WeFollow* dataset. Bold figures correspond to performance differences above 10% when comparing against the random classifier (see table 3).

Attribute	**Class**	**Plain-vanilla**			**Sink-absolute**			**Sink-relative**			**Percentile**		
		P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
religious affiliation	atheist	**0.9795**	0.1496	0.2595	**0.8615**	0.3117	0.4577	**0.8355**	0.3062	0.4481	**0.8735**	0.2326	0.3673
	budhist	**0.95**	0.0759	0.1406	**0.7349**	0.1625	0.2661	**0.5447**	0.1864	0.2778	**0.7034**	0.1358	0.2277
	christian	**0.8705**	0.3084	0.4555	**0.9878**	0.2643	0.417	**0.9982**	0.2411	0.3897	**0.9983**	0.2518	0.4021
	jewish	**0.9672**	0.1664	0.284	**0.6524**	0.2144	0.3227	**0.6329**	0.2116	0.3171	**0.768**	0.1961	0.3124
	muslim	**0.973**	0.0633	0.1188	**0.6667**	0.0984	0.1715	**0.2331**	0.109	0.1485	**0.6849**	0.0879	0.1558
	Macro-avg.	**0.948**	0.1527	0.2631	**0.7807**	0.2103	0.3313	**0.6489**	0.2111	0.3185	**0.8056**	0.1808	0.2954
	Micro-avg.	**0.8799**	0.2662	0.4088	**0.9361**	0.2543	0.4	**0.8768**	0.2382	0.3746	**0.9566**	0.2351	0.3774
political orientation	democrat	**1**	0.0478	0.0913	**0.9429**	0.0686	0.1279	**0.7899**	0.1954	0.3133	**0.7638**	0.2017	0.3191
	republican	0.9178	0.3717	0.5291	0.9338	0.3602	0.5199	**0.9778**	0.3536	0.5194	**0.9831**	0.3502	0.5164
	Macro-avg.	**0.9589**	0.2098	0.3442	**0.9384**	0.2144	0.349	**0.8839**	0.2745	0.4189	**0.8735**	0.276	0.4194
	Micro-avg.	**0.9191**	0.3324	0.4882	**0.934**	0.3248	0.482	**0.9616**	0.3344	0.4963	**0.9627**	0.3322	0.4939
sexual orientation	heterosexual	1	1	1	1	1	1	0	1	0	0	1	0
	homosexual	1	0.2685	0.4234	1	0.2415	0.389	1	0.2402	0.3874	1	0.2402	0.3874
	Macro-avg.	1	0.6343	0.7762	1	0.6208	0.766	0.5	0.6201	0.5536	0.5	0.6201	0.5536
	Micro-avg.	1	0.2685	0.4234	1	0.2415	0.389	0.9947	0.2402	0.387	0.9965	0.2402	0.3871
race/ethnicity	asian-american	**0.8571**	0.035	0.0672	**0.8571**	0.035	0.0672	**0.8571**	0.035	0.0672	**0.8913**	0.0341	0.0658
	black	**0.6818**	0.1624	0.2623	**0.6818**	0.1624	0.2623	**0.6919**	0.161	0.2613	**0.6722**	0.1637	0.2633
	hispanic	1	0	0	1	0	0	1	0	0	1	0	0
	native-american	**0.4828**	0.0897	0.1514	**0.4828**	0.0897	0.1514	**0.4375**	0.0897	0.1489	**0.4483**	0.0833	0.1405
	native-hawaiian	0	1	0	0	1	0	0	1	0	0	1	0
	white	0	1	0	0	1	0	0	1	0	0	1	0
	Macro-avg.	**0.5036**	0.3812	0.4339	**0.5036**	0.3812	0.4339	**0.4978**	0.381	0.4316	**0.502**	0.3802	0.4327
	Micro-avg.	**0.7078**	0.0547	0.1015	**0.7078**	0.0547	0.1015	**0.7045**	0.0544	0.101	**0.7036**	0.0541	0.1006