Discovering Links among Social Networks

Francesco Buccafurri, Gianluca Lax, Antonino Nocera and Domenico Ursino

DIMET, University "Mediterranea" of Reggio Calabria, Via Graziella, Località Feo di Vito, 89122 Reggio Calabria, Italy
{bucca,lax,a.nocera,ursino}@unirc.it

Abstract. Distinct social networks are interconnected via bridge users, who play thus a key role when crossing information is investigated in the context of Social Internetworking analysis. Unfortunately, not always users make their role of *bridge* explicit by specifying the so-called `me` edge (i.e., the edge connecting the accounts of the same user in two distinct social networks), missing thus a potentially very useful information. As a consequence, discovering missing `me` edges is an important problem to face in this context yet not so far investigated. In this paper, we propose a common-neighbors approach to detecting missing `me` edges, which returns good results in real life settings. Indeed, an experimental campaign shows both that the state-of-the-art common-neighbors approaches cannot be effectively applied to our problem and, conversely, that our approach returns precise and complete results.

Keywords: Link Prediction, Link Mining, Social networks, Social Internetworking.

1 Introduction

In the last years (on-line) social networks have been showing an enormous development becoming probably the main actor of the Web 2.0. The rapid and revolutionary diffusion of social networks among all segments of the population has attracted the interest of many researchers from disparate fields [19], such as sociology, psychology, economy, computer science, etc., also for the applications which the analysis of involved data can enable. In this landscape, Social Network Analysis and Social Network Mining [9] have assumed an important role, since both the hugeness of data and their graph-based organization have enforced the development of specific models and methods allowing the study of social-network data to discover knowledge from them. Clearly, the graph-based data schema gives a great information power to links among data, since it allows people profiles, resources, activities, and so on, to be directly (and indirectly) related. The crucial role of relationships in the expression of an individual's personality as well as in her social identity, traditionally recognized by social sciences, is even strengthened in the field of virtual societies, where relationship links are the main form of expression of participation of individuals to the community. To make more challenging the analysis of this reality it concurs the fact

that the scenario where we operate is not the one of single, isolated, independent social networks, but a universe composed of a constellation of several social networks, each forming a community with specific connotations, but strongly interconnected with each other. It is a matter of fact that, despite the inherent underlying heterogeneity, the interaction among distinct social networks is the basis of a new emergent internetworking scenario enabling a lot of strategic applications, whose main strength will be just the integration of possibly different communities yet preserving their diversity and autonomy. Clearly, social mining and analysis approaches may strongly rely on this huge multi-network source of information, which also reflects multiple aspects of people personal life, thus enabling a lot of powerful discovering activities.

From this perspective, links among different social networks assume a fundamental role. They typically connect the same user among the social networks she belongs to, and derive from the explicit user's declaration (sometimes supported and encouraged by specific tools) consisting in the insertion of `me` edges [1]. Unfortunately, for disparate reasons, not always users make their role of *bridge* explicit by specifying the `me` edge, missing thus a potentially very useful information. As a consequence, we may view the overall underlying (social internetworking) graph as a graph where a big number of missed `me` edges exists, whose discovery thus represents a very important issue. In other words, an interesting problem of missing link detection arises, which partially overlaps with a link prediction issue, since we may expect that a portion of missing `me` edges will be inserted in a next stage in the graph.

In this paper, we deal with the above problem by proposing an effective solution experimentally tested in a real-life Social Internetworking Scenario (SIS, for short). To the best of our knowledge, the problem of detecting `me` edges has not been investigated in the literature, but the approach we adopt in this work, which exploits a recursive notion of common-neighbor similarity, suggested us to prior verify whether common-neighbor approaches for link prediction [21] can be directly applied to our problem. The answer to this question was definitely negative, as intuitively explained in Section 2 and experimentally confirmed in Section 5, thus motivating our work. Our solution is thus based on a notion of node similarity, whose exploitation allows us to detect whether a suitable threshold is exceeded and then a missing `me` edge between two nodes is detected. The similarity between two nodes is obtained by combining two contributions: a string similarity between the associated user names, and a contribution based on a suitable recursive notion of common-neighbor similarity. The latter is extremely important because, in the literature, it is well known that string similarity alone can lead to synonymy and homonymy errors [24,15]; the neighborhood similarity allows these errors to be detected and avoided. In order to motivate the above choice it is important to clarify that the problem we are addressing does not deal with the case in which a user voluntarily keeps two accounts separated in their respective social networks. In this case, she chooses account names very different from each other, she does not have common friends and, very probably, one of the two profiles is fake (i.e., it does not contain real information about her). In

this case any approach to detecting node similarity would presumably fail. This situation, which is closer to identity-management and security problems, is little relevant in our context, where we are interested in completing the real profile of users. For these users we may expect (and this is just what we found for `me`-connected accounts) that a user joining two (or more) social networks tends to have at least partially overlapping sets of friends in them. Therefore, neighbors are useful information to exploit in order to detect missing `me` edges.

The plan of this paper is as follows: in the next section, we examine related literature. In Section 3, we present our recursive notion of similarity. On the basis of this notion, we design the method we use to detect missing `me` edges. This is described in Section 4. In Section 5, we illustrate the experiments we have carried out to verify the performances of our technique. Finally, in Section 6, we draw our conclusions and sketch possible future evolutions of our research.

2 Related Work

The detection of `me` edges in a SIS can be seen as a special case of the problem of identifying users on the Web. As a matter of fact, it allows the features of bridge users to be detected. Identifying users on the Web has received a great attention in several application scenarios, such as personalization. A lot of work is devoted to verify whether user profile information can be sufficient to address this problem. In [13] the authors define and implement a framework that provides a common base for user identification for cross-system personalisation among Web-based user-adaptive systems. The corresponding user identification algorithm combines a set of identification properties, such as username, name, location or email address, and classifies a user as identified if such a combination exceeds a suitable threshold. In [18], a technique based on user profiles for identifying users across social systems is proposed. This technique has been successfully validated on three social tagging networks (Flickr, Delicious and StumbleUpon). The limit of this technique is that only few users make their profile available in social tagging platforms. A method to identify users on the basis of profile matching is proposed in [26]. In this paper data from two popular social networks are used to evaluate the importance of fields in the Web profile and to develop a profile comparison tool. The authors of [29] provide evidence on the existence of mappings among usernames across different communities. Starting from the observation of the data in BlogCatalog, they infer 7 hypotheses on the relationships among the usernames selected by a single person in different communities. On the basis of such hypotheses, they propose an approach that, given a username u in a source community and a target community c, generates a set of candidate usernames in c corresponding to u. The approach first generates a set of usernames from u by adding and removing suitable prefixes and suffixes. Then, it exploits a Web search on Google aimed at checking for the existence of each candidate username in such a way as to reduce the returned set of usernames.

From another point of view, the detection of `me` edges in a SIS is someway related to link prediction. Link prediction is a task of link mining aiming at predicting the (even future) existence of a link between two objects [21,6]. In the contest of social networks, it focuses on predicting friendships among users. Often, social networks are represented as graphs [11]. As a consequence, some link prediction approaches are totally based on the structural properties of these graphs [20]. A first possibility to perform this task consists in analyzing common neighbors. In order to decide whether two nodes are related, [5] exploits a similarity measure derived from the Jaccard coefficient. Based on *preferential attachment* [23], [8] experimentally verifies that the probability of a relationship between two nodes is proportional to the product of the number of their neighbors. Some approaches to link prediction rely on the notion of shortest-path distance which is computed by means of several similarity measures, like the Katz coefficient, PageRank and SimRank. Due to the high computational cost of these measures, some approximations have to be adopted in order to make them effective. In any case, whenever the number of nodes is considerable, the application of these methods may result in a too long running time. In conjunction with all the above techniques, some strategies may be used to enhance the accuracy of predictions. Also the use of *unseen bigrams* [14] can help in the link detection task. Here, the similarity between a node A and a node B is computed by taking into account the similarity between the nodes B and C, where this last one is the node most similar to A. Furthermore, the quality of link detection can be improved by means of clustering techniques aiming at identifying the graph components which introduce noise in the similarity computation [20]. [25] proposes the application of statistical relational learning to link prediction in the domain of scientific literature citations. In this approach statistical modeling and feature selection are integrated into a search mechanism over the space of database queries in such a way as to define feature candidates involving complex interactions among objects in a given relational database. [27] analyzes the localization in space and time of a large number of users by means of their call detail records. This analysis shows that users with similar movement routines are strongly connected in a social network and have intense direct interactions. This result allows implicit ties in the social network to be predicted with a significant accuracy starting from the analysis of the correlation between user movements (i.e., their mobile homophily). Other approaches to link detection come from the fields of *deduplication* and *disambiguation*. In particular, [7] proposes an algorithm for discovering duplicates in the dimensional tables of a Data Warehouse.

From the above analysis it emerges that our approach, in the above literature, can be related only with common-neighbors ones. However, despite their apparent closeness to ours, we can easily realize that they are not directly applicable to our context. Indeed, the notion of common-neighbors relies, in general, on the notion of common identity of the friends of a user. But discovering the common identity of users in different social networks is for us the output of the problem, leading to a sort of recursive definition of the problem itself. We have

experimentally confirmed the above claim by showing that the application of the state-of-the-art common-neighbors approaches to our problem returns very unsatisfying results. The results of these experiments are reported in Section 5.

3 The Notion of Similarity

Our approach operates in a Social Internetworking System (SIS) resulting from the interconnection of a number of distinct social networks. We start with the basic notion of underlying graph, which is the layer we deal with.

Definition 1. A *t-Social-Internetworking Graph (SIG)* is a directed graph $G = \langle N, E \rangle$, where N is the set of *nodes*, E is the set of *edges* (i.e., ordered pairs of nodes) and N is partitioned into t subsets $S_1, \ldots, S_t$. Given a node $a \in N$ we denote by $S(a)$ the social graph which a belongs to. E is partitioned into two subsets E_f and E_m. E_f is said the set of *friendship edges* and E_m is the set of `me` *edges*. E_f is such that for each $(a, b) \in E_f$, $S(a) = S(b)$, while E_m is such that for each $(a, b) \in E_m$, $S(a) \neq S(b)$[1]. Given a node a we denote by $\Gamma(a)$ the set of nodes in $S(a)$ such that for each $b \in \Gamma(a)$ $(a, b) \in E_f$. $\Gamma(a)$ is said the set of *neighbors* of a. The graphs corresponding to $S_1, \cdots, S_t$ are called *social graphs* and in *SIG* they are linked to each other by means of `me` edges. □

A t-Social-Internetworking Graph $G = \langle N, E_f \cup E_m \rangle$ is the graph underlying a SIS composed of t social networks. Each node a of G is associated with a user, joining the social network whose underlying graph is $S(a)$. An edge $(a, b) \in E_f$ means that the user b is a friend of the user a in the social network of $S(a)$. An edge $(c, c') \in E_m$ means that the user c in the social network of $S(c)$ has declared a `me` edge between herself and the user c' in the social network of $S(c')$. In other words, c is a bridge and this means that c and c' are associated to the same user. From now on, throughout this section, consider given a t-Social-Internetwork-ing Graph $G = \langle N, E_f \cup E_m \rangle$.

Our approach is based on a recursive notion of "inter-social-network" similarity aimed at detecting missing bridges of G. The similarity between two nodes a, b (belonging to two different social networks) is obtained by combining two contributions: a string similarity between the user names associated to a and b, and a contribution based on a suitable notion of common-neighbors similarity. The latter component leads to a recursive definition of the overall inter-social-network similarity since the common-neighbors notion has to necessarily rely on the same notion, because also neighbors belong to different social networks, and, thus, common nodes have to be detected too. Observe that this two-component philosophy has been successfully adopted in several application fields in the past; among these fields we cite schema matching [24,15], information retrieval [10], and logic programming [17].

Concerning the string similarity, several functions have been proposed in the literature, such as Jaro-Winkler, Levenshtein, QGrams, Monge-Elkan, Soundex

[1] Observe that the presence of E_m makes a t-Social-Internetworking Graph to not be a forest since, in this last case, the corresponding social networks should be disjoint.

[16]. One of them could be adopted to measure user name similarity in our approach, so that it can be considered parametric w.r.t. the string-similarity function. We have evaluated the application of the different functions in Section 5.2.

Before defining our notion of similarity, we have to introduce a preliminary notion, which the common-neighbors contribution relies on. Indeed, we detect a missing `me` edge between a and b if a suitable combination of the string similarity between the user names associated to them, according to the metric Q, and the (recursive) similarity of the top-k similar pairs each composed of a friend of a and a friend of b, is greater than a suitable threshold, for a given k.

Thus, we have preliminarily to define how to select such top-k pairs. This is related to the next definition.

Definition 2. Given a positive integer k_0, a pair of nodes $a, b \in N$ such that $S(a) \neq S(b)$, a string-similarity metric Q, and a non-negative integer n we inductively define $Top_Q^n(a, b, k_0)$ as follows:

1. $Top_Q^0(a, b, k_0)$ is any subset of $C = \{(x^a, y^b) \mid x^a \in \Gamma(a), y^b \in \Gamma(b)\}$ containing the top-k_0 elements of C w.r.t. the metric Q.
2. For any $0 < i \leq n$, $Top_Q^i(a, b, k_0)$ is any subset of $C = \{(x^z, y^w) \mid (z, w) \in Top_Q^{i-1}(a, b, k_0), x \in \Gamma(z), y \in \Gamma(w), (x^*, *) \notin Top_Q^j(a, b, k_0), (*, y^*) \notin Top_Q^j(a, b, k_0), 0 \leq j \leq i-1\}$ containing the top-k_i elements of C w.r.t. the metric Q, where $k_i = \lceil \frac{k_0}{(1+i)^{1+i}} \rceil$ and $*$ denotes any node in N. □

Concerning the contribution of neighbors, we have to consider a particular situation which could significantly affect the precision of our technique. Suppose we have two nodes z and w belonging to different social networks and consider $x \in \Gamma(z)$ and $y \in \Gamma(w)$. If x and y are power users (i.e., users having a very high degree in social networks) and z and w are not, it could happen that x (resp., y) belongs to $\Gamma(z)$ (resp., $\Gamma(w)$) only because it corresponds to a public figure (e.g., a V.I.P.). In this case, the presence of this node in $\Gamma(z)$ and $\Gamma(w)$ is not significant in defining the real life relationships of z and w. In order to prevent this effect, we introduce the following reduction coefficient $\gamma(x^z, y^w)$, which assumes a very powerful role in the above situation.

Definition 3. Let $x, y, z, w \in N$ be nodes of G such that $x \in \Gamma(z)$, $y \in \Gamma(w)$, and $S(z) \neq S(w)$. We define: $\gamma(x^z, y^w) = min(\delta(x^z), \delta(y^w))$ where $\delta(a^b) = \frac{max(|\Gamma(a)|, |\Gamma(b)|)}{max(|\Gamma(a)|, |\Gamma(b)|) + ||\Gamma(a)| - |\Gamma(b)||}$ for any pair of nodes $a, b \in N$. □

Now we are ready to define our similarity function. As said above, it is obtained as a combination of two contributions, namely, the string-similarity component and the common-neighbors one. The underlying intuition is that if two accounts, belonging to different social networks, are associated to the same user, even though there is no `me` edge between them, they will have user names someway related each other and, moreover, they share a (even low) number of friends. It is worth remarking that this condition, easily verifiable by exploring real-life

social networks, does not mean that the two users have a strong overlap of their neighbors, coherently to the diversity among the communities included in different social networks. However, we argue that it is highly probable that an even little neighbor overlap will occur[2].

Definition 4. Given a pair of nodes $a, b \in N$ such that $S(a) \neq S(b)$, a string-similarity metric Q, two integers $n \geq 0$ and $k_0 > 0$, we inductively define the *similarity operator* $T^n_Q(a, b, k_0)$ as follows:

1. $T^0_Q(a, b, k_0) = Q(a, b)$.
2. For any $0 < i \leq n$,

$$T^i_Q(a, b, k_0) = (1 - \beta_i) \cdot T^{i-1}_Q(a, b, k_0) + \beta_i \cdot \frac{\sum_{(x^z, y^w) \in Top^i_{\widetilde{Q}}(a,b,k_0)} \widetilde{Q}(x^z, y^w)}{|Top^i_{\widetilde{Q}}(a,b,k_0)|}$$

where $\beta_i = \frac{1}{(i+1)^{i+1}}$ and $\widetilde{Q}(x^z, y^w) = \gamma(x^z, y^w) \cdot Q(x, y)$, for any $x, y, z, w \in N$ nodes of G such that $x \in \Gamma(z)$, $y \in \Gamma(w)$, and $S(z) \neq S(w)$. □

The definition of similarity is recursive. At the basic step, only the direct string-similarity value concurs to define the similarity between two nodes a and b. At step i, the similarity is obtained as a linear combination of the similarity of the step $i-1$, and the new common-neighbors contribution. This is obtained as the average of the reduced (by γ – see Definition 3) string similarity between the top-k_i pairs w.r.t. the same metric. k_i is derived, at each step, as an exponential reduction of k_0, which is an input parameter allowing us to modulate the size of the neighbors overlapping considered relevant for the similarity computation. Observe that the above linear combination depends on the β_i parameter, which is exponentially decreasing as i increases, making quickly less important the common-neighbors contribution when leaving from the root nodes a and b.

Now we are ready to define the effective tool we provide to detect `me` edges, obviously based on the above notion of similarity. Indeed, Definition 4 would lead to an ineffective computation covering, in the worst case, the whole graph G. Anyway, we can observe that when, during the computation, we reach a step h whose contribution to the overall similarity is under a given small ϵ, we expect that from now on involved neighbors do not give us any meaningful information. Thus, we can stop here the iteration. This is encoded into the next definition.

Definition 5. Given a pair of nodes $a, b \in N$ such that $S(a) \neq S(b)$, a string-similarity metric Q, an integer number $k_0 > 0$, and a real number $\epsilon > 0$, we define the *ϵ-similarity* $S^\epsilon_Q(a, b, k_0)$ between a and b w.r.t. Q as $T^h_Q(a, b, k_0)$, where $h > 0$ is the least number (if any) such that $|T^h_Q(a, b, k_0) - T^{h-1}_Q(a, b, k_0)| < \epsilon$. □

Clearly, our approach is really effective if the ϵ-similarity $S^\epsilon_Q(a, b, k_0)$ between two nodes a and b always exists. This is ensured by the following theorem.

[2] Recall that we are not interested in cases of users who voluntarily keep two accounts separated in their respective social networks, where the above conditions are not verified, as explained in the Introduction.

Theorem 1. *Given a pair of nodes $a, b \in N$ such that $S(a) \neq S(b)$, a string-similarity metric Q, an integer number $k_0 > 0$, and a real number $\epsilon > 0$, then the ϵ-similarity $S^{\epsilon}_{Q}(a, b, k_0)$ between a and b w.r.t. Q exists.* □

Finally, in the next theorem we give a result related the complexity of our technique, showing that it is feasible.

Theorem 2. *Given a pair of nodes $a, b \in N$ such that $S(a) \neq S(b)$, a string-similarity metric Q, and an integer number $k_0 > 0$, then the number of visited nodes in G for the computation of $T^{h}_{Q}(a, b, k_0)$ is $O(d^2 \cdot k_0)$, where d is the maximum node degree.* □

The consequence of Theorem 2 is that, despite the potentially exponential explosion of visited nodes for the computation of the ϵ-similarity $S^{\epsilon}_{Q}(a, b, k_0)$ between two nodes a and b, due to the h-step iterative navigation of neighbors in G, the number of visited nodes is independent of the number of iterations h and is bounded by the number of visited nodes at the first two steps. As it can be easily verified by reading the proof of the theorem, this fact clearly depends on how k_i is defined in Definition 2. The above result ensures the feasibility of our method.

4 `Me`-Edge Detection

In this section, we present our method able to discover missing `me` edges, and, thus, new links among different social networks. Clearly, these links complement the `me` edges explicitly declared by users. Our technique is based on the notion of similarity presented in Section 3. In particular, the similarity function allows us to detect a missing `me` edge between two nodes, whether a suitable threshold is exceeded. We consider given a Social Internetworking System (SIS) composed of t social networks, as well as the underlying Social Internetworking Graph $G = \langle N, E_f \cup E_m \rangle$.

Since, in a SIS, the number of node pairs to consider for the possible presence of a `me` edge is enormous, we have identified a mechanism leading our approach to consider only a reasonable number of very promising pairs. In particular, from the examination of the explicitly declared `me` edges, we have found that, with a high probability, some of the nodes belonging to the neighbors of two nodes linked by a `me` edge are, in their turn, linked by a `me` edge. As a consequence, our approach starts from a set of already known `me` edges and examines only the neighbors of the nodes involved in these edges. Clearly, if this set of `me` edges is not available, our approach can work all the same by starting from any pair of nodes in G. Thus, our algorithm receives a set $M \subseteq E_m$ of existing `me` edges and returns a set M' of discovered `me` edges, such that, clearly $M' \cap (E_f \cup E_m) = \emptyset$. The function $detectMe$ implementing our approach is reported in Algorithm 1. It receives as arguments: the set M of starting `me` edges, two [0,1] real variables th_c and th_d, an integer $k_0 > 0$, and (a small) real $\epsilon > 0$. Recall that the

similarity between two nodes a, b defined in Section 3 is obtained by combining two contributions: a string similarity between the user names associated to a and b, and a contribution based on a suitable notion of common-neighbors similarity. Concerning the first contribution, as pointed out in Section 3, there exist several already defined functions for computing the similarity between two strings, each characterized by specific features (e.g., Jaro-Winkler, Levenshtein, QGrams, Monge-Elkan, Soundex, etc. [16]). The function $Q(a, b)$ (receiving two nodes a and b) in Algorithm 1 can be considered parametric w.r.t. the string-similarity function. We can choose one of the above functions on the basis of the desired target. For instance, QGrams is very severe and assigns quite low similarity degrees, Jaro-Winkler is more permissive whereas Soundex is very permissive. In Section 5.2 the application of the different functions in our technique is experimentally evaluated.

Synthetically, Algorithm 1 proceeds as follows. For each edge (a, b) in M, we take all pairs (a', b') such that $a' \in \Gamma(a)$ and $b' \in \Gamma(b)$. These pairs are candidate `me` edges. A pair (a', b') is discarded if the value of $Q(a', b')$ is lower than a suitable threshold th_c or if $(a', b') \in E_m$. Otherwise, a function S is called, which implements the computation of $S^{\epsilon}_{Q}(a', b', k_0)$ (see Definitions 4 and 5). If the value returned by this function is greater than a suitable threshold th_d, then (a', b') is detected as `me` edge and is inserted in M'. Otherwise, it is discarded. The function S is reported in Algorithm 2. It is recursive since it implements also the operator $T^{n}_{Q}(a, b, k_0)$ of Definition 5. It receives as arguments: an integer $k_0 > 0$, a (small) real $\epsilon > 0$, a list L of triplets $\langle a, b, s \rangle$, where a and b are nodes and s is a $[0, 1]$ real value, the value of S at the previous step (at the initial step of the recursion this value coincides with $Q(a, b)$), and an integer $i > 0$ representing the step of the recursion. To explain what is s, observe that, in order to implement $T^{n}_{Q}(a, b, k_0)$, we need as argument the list L which stores, at the step $i > 1$ of the recursion, the top-k_i node pairs w.r.t. the metric $\widetilde{Q}^{i}(a, b, k_0)$. Thus, in this case, s represents just $\widetilde{Q}^{i}(a, b, k_0)$. At the initial step of the recursion ($i = 1$), s is just the string similarity value $Q(a, b)$.

The correspondence between Algorithm 2 and Definitions 4 and 5 is quite clear. We just have to highlight that the result of the computation of the operator $Top^{i}_{\widetilde{Q}}(a, b, k_0)$ (see Definitions 2 and 5) is embedded in the list L, as described above.

The computation of $Top^{i}_{\widetilde{Q}}(a, b, k_0)$ is implemented by Algorithm 3. The function Top receives two nodes a and b and a positive integer k_i. It returns a list L of k_i triplets $\langle a', b', \widetilde{Q}^{i}(a', b', k_0) \rangle$, where $a' \in \Gamma(a)$, $b' \in \Gamma(b)$ and (a', b') is one of the top-k_i pairs w.r.t. the metric $\widetilde{Q}$. We remark that the metric here used includes the reduction factor γ of Definition 3.

5 Experiments

In this section we present our experimental campaign aimed at determining the performances of our approach. Since it operates on a SIS, we had to extract not

Algorithm 1 *detectMe*

Input M: the starting set of `me` edges
Input th_c: A candidate threshold
Input th_d: A detection threshold
Input k_0: an integer
Input ϵ: a real
Output M': the set of detected `me` edges
Variable L: a list of triplets of the form $\langle a, b, s\rangle$

1: $L := \emptyset$; $M' := \emptyset$
2: **for each** *edge (a,b)* $\in M$ **do**
3: **for each** *node pair (a',b')* $\in \Gamma(a) \times \Gamma(b)$ **do**
4: **if** $(Q(a', b') > th_c$ **and** *the edge (a',b')* $\notin E_m)$ **then**
5: insert the triplet $\langle a', b', Q(a', b')\rangle$ into L
6: **if** $(S(k_0, \epsilon, L, Q(a', b'), 1) > th_d)$ **then**
7: insert the `me` edge (a', b') in M'
8: **end if**
9: $L := \emptyset$
10: **end if**
11: **end for**
12: **end for**
13: **return** M'

Algorithm 2 S

Input k_0: an integer
Input ϵ: a real
Input L_i: a list of triplets $\langle a, b, s\rangle$
Input S_{i-1}: the similarity value of a' and b' at step $i-1$
Input i: an integer
Output S_i: the similarity value of a' and b' at step i
Variable β: a [0,1] real
Variable k_i: an integer
Variable $avgS$: a real
Variable L_{i+1}: a list of triplets $\langle a, b, s\rangle$

1: $L_{i+1} := \emptyset$; $\beta := \frac{1}{i^i}$; $k_i := \lceil k_0 \cdot \beta \rceil$
2: **if** (k=1) **then**
3: $S_i := S_{i-1}$
4: **else**
5: assign to $avgS$ the average value of the similarities of the node pairs of L_i
6: $S_i := (1-\beta) \cdot S_{i-1} + \beta \cdot avgS$
7: **end if**
8: **if** $(|S_i - S_{i-1}| \geq \epsilon)$ **then**
9: **for each** $\langle a, b, s\rangle$ in L_i **do**
10: add the list returned by $Top(a, b, k_i)$ to L_{i+1}
11: **end for**
12: **return** $S(i+1, L_{i+1}, S_i)$
13: **else**
14: **return** S_i
15: **end if**

only the connections among the accounts of different users in the same social network but also the connections among the accounts of the same user in different social networks. In order to handle these connections, two standards encoding human relationships are generally exploited. The former is XFN (XHTML Friends Network) [3]. It simply uses an attribute, called `rel`, to specify the kind of relationship between two users. Some possible values of `rel` are `me`, `friend`, `contact`, `co-worker`, `parent`, and so on. A (presumably) more complex alternative to XFN is FOAF (Friend-Of-A-Friend) [2]. In both of them information about `me` edges is the one explicitly declared by users. In our experiments, we

Algorithm 3 *Top*

Input a, b: a node
Input k_i: an integer
Output L: a list of triplets $\langle a, b, s\rangle$
Variable t: a triplet $\langle a, b, s\rangle$
Variable c: an integer
Variable $\delta_a, \delta_b, \widetilde{Q}_{(a,b)}$: a real

1: $L := \emptyset$; $c := 0$
2: **for each** a' in $\Gamma(a)$ **do**
3: **for each** b' in $\Gamma(b)$ **do**
4: $\delta_a := \frac{max(|\Gamma(a')|,|\Gamma(a)|)}{max(|\Gamma(a')|,|\Gamma(a)|)+||\Gamma(a')|-|\Gamma(a)||}$
5: $\delta_b := \frac{max(|\Gamma(b')|,|\Gamma(b)|)}{max(|\Gamma(b')|,|\Gamma(b)|)+||\Gamma(b')|-|\Gamma(b)||}$
6: $\widetilde{Q}_{(a,b)} := min(\delta_a, \delta_b) * Q(a', b')$
7: **if** $(c < k_i)$ **then**
8: insert the triplet $\langle a', b', \widetilde{Q}_{(a,b)}\rangle$ into L
9: sort L in a descending order
10: $c := c + 1$
11: **else**
12: $t := L.get(k_i - 1)$
13: **if** $(\widetilde{Q}_{(a,b)} > t.s))$ **then**
14: replace the triplet in the position $(k_i - 1)$ of L with the triplet $\langle a', b', \widetilde{Q}_{(a,b)}\rangle$
15: sort L in a descending order
16: **end if**
17: **end if**
18: **end for**
19: **end for**
20: **return** L

considered a SIS consisting of four social networks, namely Twitter, LiveJournal, YouTube and Flickr. Therefore, from now on we refer to a (real-life) t-Social Internetworking Graph such that $t = 4$. We argue that the relatively small number of involved social networks, as a first investigation, is adequate for our purpose, expecting that the results we obtain in this setting are still valid in a more complex environment. Anyway, the above social networks are highly representative (they are among the top-10 social networks in terms of population). As a matter of fact, they are largely analyzed in the past in Social Network Analysis [22,28].

For our experiments, we used a server equipped with a 2 Quad-Core E5440 processor and 16 GB of RAM with the CentOS 6.0 Server operating system. We performed our experiments from January 30, 2012 to April 5, 2012.

5.1 Application of the state-of-the-art common-neighbors approaches

As described in Section 2, we have to prior verify whether common-neighbor approaches for link prediction [21] can be directly applied to our problem. However, a first aspect has to be considered. In our scenario the notion of common neighbors cannot be the classical one, because the neighbors of examined pairs belong to different social networks. As a consequence, we cannot expect that two examined neighbors have some common nodes in strict sense. To overcome this drawback we have just to re-define the notion of node identity. Coherently with our setting, it simply suffices to consider as *identical* two nodes linked by

Index Name	*Definition*	*Sensitivity*
Salton Index (SAI)	$s_{ab}^{SAI} = \frac{\lvert\Gamma(a) \bigcap \Gamma(b)\rvert}{\sqrt{\lvert\Gamma(a)\rvert \times \lvert\Gamma(b)\rvert}}$	0.01
Jaccard Index (JAI)	$s_{ab}^{JAI} = \frac{\lvert\Gamma(a) \bigcap \Gamma(b)\rvert}{\lvert\Gamma(a) \bigcup \Gamma(b)\rvert}$	0.01
Sorensen Index (SOI)	$s_{ab}^{SOI} = \frac{2\lvert\Gamma(a) \bigcap \Gamma(b)\rvert}{\lvert\Gamma(a)\rvert + \lvert\Gamma(b)\rvert}$	0.01
Hub Promoted Index (HPI)	$s_{ab}^{HPI} = \frac{\lvert\Gamma(a) \bigcap \Gamma(b)\rvert}{min(\lvert\Gamma(a)\rvert, \lvert\Gamma(b)\rvert)}$	0.00
Hub Depressed Index (HDI)	$s_{ab}^{HDI} = \frac{\lvert\Gamma(a) \bigcap \Gamma(b)\rvert}{max(\lvert\Gamma(a)\rvert, \lvert\Gamma(b)\rvert)}$	0.01
Leicht-Holme-Newman Index (LHNI)	$s_{ab}^{LHNI} = \frac{\lvert\Gamma(a) \bigcap \Gamma(b)\rvert}{\lvert\Gamma(a)\rvert \times \lvert\Gamma(b)\rvert}$	0.01
Resource Allocation Index (RA)	$s_{ab}^{RA} = \sum_{z \in \Gamma(a) \bigcap \Gamma(b)} \frac{1}{\lvert\Gamma(z)\rvert}$	0.01
Local Path Index (LPI)	$s_{ab}^{LPI} = A^2 + \epsilon A^3$ (A is G's adjacency matrix)	0.03

Table 1. The tested common-neighbors approaches

a `me` edge. At this point, classical common-neighbor techniques can be directly applied.

Given two nodes a and b in G (such that $S(a) \neq S(b)$), the considered (state-of-the-art) techniques are those reported in Table 1, where we include, in the first and second columns, the definition of the similarity index which they rely on [21].

We tested all the above techniques in our SIS by preliminarily constructing a set M of 100 node pairs linked by a `me` edge and then by running them on M. For each technique, we obtained a set M' of detected `me` edges. Clearly, M' represents a set of true positives. Finally, we measured the *sensitivity* of the techniques as the ratio $\frac{|M'|}{|M|}$, obtaining the results reported in the third column of Table 1.

The analysis of such values clearly shows that no effective result can be obtained whether common-neighbors techniques are adopted. This has in fact motivated our further study, whose experimental validation is reported in the next sections.

5.2 Sample-driven method validation

A first experiment aims at determining the performance of our approach as well as at choosing the best function for computing the string similarity in our context. We started from the set M introduced in the previous section (i.e., a set of 100 real `me`-edge-connected pairs). Then, we find another set, denoted by $\neg M$, of 100 node pairs not connected by a `me` edge. To find two elements, say (c_1, d_1) and (c_2, d_2), of $\neg M$, we started from a `me`-edge-connected pair (a, b) and then we required that $c_1 = a$, $d_1 \in \Gamma(b)$, $c_2 = b$, $d_2 \in \Gamma(a)$. This way, both (c_1, d_1) and (c_2, d_2) are not linked by a `me`-edge, since d_1 is a friend of a user (i.e., b) who is identical to c_1. The dual situation occurs for (c_2, d_2).

Then, we applied our approach on the pairs of M and $\neg M$ and we obtained the sets TP, FP, and FN, which are true positives, false positives, and false negatives, resp. For clarity, an element in TP is a pair of M detected as `me`-edge-connected pair by our technique, an element in FP is a pair of $\neg M$ detected as

Function	*Precision*	*Recall*	*F-measure*
Jaro-Winkler	0.558	0.920	0.694
QGrams	0.908	0.690	0.784
Levenshtein	0.877	0.710	0.785
Smith-Waterman	0.840	0.790	0.814
Smith-Waterman-Gotoh	0.779	0.810	0.794
Monge-Elkan	0.779	0.810	0.794
Needleman-Wunch	0.500	1.000	0.667
Jaro	0.555	0.910	0.689
Soundex	0.500	0.990	0.664

Table 2. Precision, recall and F-Measure of our approach for each user name similarity function

`me`-edge-connected pair by our technique, and an element of FN is a pair of M not detected as `me`-edge-connected pair by our technique.

To compute the performance of our approach we adopted three classical measures [4], namely *precision* (as measure of correctness), *recall* (as measure of completeness) and *F-measure* (as the harmonic mean of precision and recall). They are defined as: $precision = \frac{|TP|}{|TP|+|FP|}$, $recall = \frac{|TP|}{|TP|+|FN|}$, and $F\text{-}measure = 2 \cdot \frac{precision \cdot recall}{precision+recall}$.

Since the behavior of our approach (and, consequently, the values of precision and recall) depends on the function adopted for computing string similarity, we considered the most common of these functions and, for each of them, we computed the precision and the recall of our technique. This way, we were able to determine the function(s) maximizing these measures. Obtained results are shown in Table 2.

The main conclusion we can draw from the analysis of this table is that our approach presents in general a very satisfying performance both in correctness and in completeness. Moreover, we observe that we are free to choose the string-similarity function in a rich set. Indeed, 5 functions led our approach to obtain a precision higher than 0.77 and 6 functions led it to obtain a recall higher than 0.81. However, among the considered functions, QGrams (resp., Needleman-Wunch) proved to be the one capable of assuring the best precision (resp., recall). The high performance level of our approach is even more evident if we compare Tables 1 and 2 and if we consider that, in this testbed, the definitions of recall and sensitivity coincide and, consequently, the corresponding columns can be compared[3].

5.3 Expert-based method validation

This experiment aims at computing the accuracy of our approach in a way different from the previous experiment. In this case, we want to benefit from the support of a human expert. We first applied a crawling technique to derive a sample of the SIS. This sample was necessary to have a starting set of `me` edges at

[3] Observe that, owing to the extremely low values of sensitivity, the computation of precision in Table 1 makes no sense.

disposal. In order to maximize the number of `me` edges occurring in the sample we applied BDS, a crawling technique specifically conceived to operate on a SIS, instead of on a single social network, which is highly capable of finding and returning explicitly declared `me` edges [12] (note that, to the best of our knowledge, no other technique with this feature is available in literature). Our sample consisted of 93,169 nodes and 146,325 edges. 745 out of 146,325 were `me` edges. This sample can be found at the URL `http://www.ursino.unirc.it/pkdd-12.html`. We randomly selected 160 `me` edges and put them in a set M. We gave this set as input to our technique. The adopted string-similarity function was QGrams, because it proved to assure the best precision. Our approach returned a set M' of 22 `me` edges and a set of 133 non-`me` edges, from which we randomly selected a set $\neg M'$ of 22 non-`me` edges in such a way that `me` edges and non-`me` edges had the same weight. After this, we asked the human expert to verify whether the elements of M' are actually `me` edges and the elements of $\neg M'$ are actually non-`me` edges. For each edge her possible answers were *true*, *false* and *unknown*. Observe that the value *unknown* reflects both uncertain cases and unreachable-page ones. At the end of the experiment we obtained that, as for M', she returned $t_p = 16$ *true*, $f_p = 4$ *false* and 2 *unknown*. As for $\neg M'$, she returned $t_n = 18$ *true*, $f_n = 2$ *false* and 2 *unknown*. Finally, we computed the *accuracy* as the ratio $\frac{t_p+t_n}{t_p+t_f+t_n+f_n}$, obtaining the value 0.85, which denotes a very good performance of our method.

6 Conclusion and Future Work

In this paper, we have studied the problem of discovering missing `me` edges in a Social Internetworking Scenario. The most evident information we can use to detect missing `me` edges, besides user names, concerns neighbors. This might lead us to conclude that we are in front of a classical problem of link prediction where one of the state-of-the-art common-neighbors techniques can be applied in order to solve it. The first conclusion we have drawn in this work, on the basis of a number of experiments, is that this is definitely not true, showing the need of studying the problem as new and finding a specific solution. To do this, we have defined a suitable notion of "inter-social-network" similarity, which is recursive since the common-neighbors notion has to necessarily rely on the same notion, because also neighbors belong to different social networks. On the basis of this notion, we have defined an algorithm able to detect whether there is a missing `me` edge between two given nodes. The experimental analysis of this method on a real-life data set has shown its correctness and completeness, thus validating it. In the future, this analysis could be further empowered in several directions. Some of them could be: *(i)* the analysis of our approach running time; *(ii)* the analysis of the mutual role of the two components of our similarity definition; *(iii)* the analysis of the stability and sensitivity of our approach.

The results obtained in this paper can be useful for further investigations in the direction of social internetworking. Indeed, the role of `me` edges is relevant for any phenomenon of information crossing through different social networks,

so that discovering new `me` edges may strongly enrich the analysis capabilities of social data, also strengthening multi-context analyses of people profile. We thus believe that a number of future directions of our research can be undertaken, especially in the context of behavioral analysis, but also at the "lower" level of graph analysis, where the discovery of new `me` edges may improve the capability of crawlers to explore a Social Internetworking Scenario.

Acknowledgment

This work was partially funded by the Italian Ministry of Research through the PRIN Project EASE (Entity Aware Search Engines).

References

1. Google Social Graph. `http://code.google.com/p/itswhoyouknow/wiki/SocialGraph`, 2012.
2. The Friend of a Friend (FOAF) project. `http://www.foaf-project.org/`, 2012.
3. XFN - XHTML Friends Network. `http://gmpg.org/xfn`, 2012.
4. ISO 5725-1:1994/Cor 1:1998. Accuracy (trueness and precision) of measurement methods and results - Part 1: General principles and definitions - Technical Corrigendum 1. 1998.
5. L. Adamic and E. Adar. Friends and Neighbors on the Web. *Social Networks*, 25(3):211–230, 2003.
6. M. Al Hasan and M.J. Zaki. A Survey of Link Prediction in Social Networks. In *Social Network Data Analytics*, pages 243–276. Elsevier, 2011.
7. R. Ananthakrishna, S. Chaudhuri, and V. Ganti. Eliminating fuzzy duplicates in Data Warehouses. In *Proc. of the International Conference on Very Large Data Bases (VLDB '02)*, pages 586–597, Hong Kong, China, 2002. VLDB Endowment.
8. A.L. Barabási, H. Jeong, Z. Neda, E. Ravasz, A. Schubert, and T. Vicsek. Evolution of the social network of scientific collaborations. *Physica A: Statistical Mechanics and its Applications*, 311(3-4):590–614, 2002.
9. M. Berlingerio, M. Coscia, F. Giannotti, A. Monreale, and D. Pedreschi. As time goes by: Discovering eras in evolving social networks. In *Proc. of the Int. Conf. on Advances in Knowledge Discovery and Data Mining (PAKDD 2010)*, pages 81–90, Hyderabad, India, 2010. Springer.
10. D.T. Bollegala. *A Study on Attributional and Relational Similarity between Word Pairs on the Web*. PhD Thesis, University of Tokyo, 2009.
11. M. Brocheler, A. Pugliese, and V.S. Subrahmanian. Probabilistic Subgraph Matching on Huge Social Networks. In *Proc. of the International Conference on Advances in Social Networks Analysis and Mining (ASONAM'11)*, pages 271–278, Kahosiung, Taiwan, 2011.
12. F. Buccafurri, G. Lax, A. Nocera, and D. Ursino. Crawling Social Internetworking Systems. In *Proc. of the International Conference on Advances in Social Analysis and Mining (ASONAM 2012)*, Istanbul, Turkey, Forthcoming.
13. F. Carmagnola and F. Cena. User identification for cross-system personalisation. *Information Sciences*, 179(1-2):16–32, 2009.
14. I. Dagan, F. Pereira, and L. Lee. Similarity-based estimation of word cooccurrence probabilities. In *Proc. of the annual meeting on Association for Computational Linguistics*, pages 272–278. Association for Computational Linguistics, 1994.

15. P. De Meo, G. Quattrone, G. Terracina, and D. Ursino. Integration of XML Schemas at various "severity" levels. *Information Systems*, 31(6):397–434, 2006.
16. A.K. Elmagarmid, P.G. Ipeirotis, and V.S. Verykios. Duplicate Record Detection: A Survey. *IEEE Transactions on Knowledge and Data Engineering*, 19(1):1–16, 2007.
17. S. Ferilli, T. M. A. Basile, N. Di Mauro, M. Biba, and F. Esposito. A New Similarity Measure for Guiding Generalizations Search. In *Proc. of the International Conference on Inductive Logic Programming (ICLP'06)*, Seattle, Washington, USA.
18. T. Iofciu, P. Fankhauser, F. Abel, and K. Bischoff. Identifying users across social tagging systems. In *Proc. of the International Conference on Weblogs and Social Media (ICWSM'11)*, Barcelona, Catalonia, Spain, 2011. The AAAI Press.
19. J. Kleinberg. The convergence of social and technological networks. *Communications of the ACM*, 51(11):66–72, 2008.
20. D. Liben-Nowell and J. Kleinberg. The link-prediction problem for social networks. *Journal of the American society for information science and technology*, 58(7):1019–1031, 2007.
21. L. Lü and T. Zhou. Link Prediction in Complex Networks: A Survey. *Physica A: Statistical Mechanics and its Applications*, 390(6):1150–1170, 2011.
22. A. Mislove, H.S. Koppula, K.P. Gummadi, F. Druschel, and B. Bhattacharjee. Growth of the Flickr Social Network. In *Proc. of the International Workshop on Online Social Networks (WOSN08)*, pages 25–30, Seattle, Washington, USA, 2008. ACM.
23. M.E.J. Newman. The structure and function of complex networks. *SIAM review*, pages 167–256, 2003.
24. L. Palopoli, D. Saccà, G. Terracina, and D. Ursino. Uniform techniques for deriving similarities of objects and subschemes in heterogeneous databases. *IEEE Transactions on Knowledge and Data Engineering*, 15(2):271–294, 2003.
25. A. Popescul and L.H. Ungar. Statistical relational learning for link prediction. In *Proc. of the International Workshop on Learning Statistical Models from Relational Data*, volume 149, pages 172–179, Acapulco, Mexico, 2003.
26. J. Vosecky, D. Hong, and V.Y. Shen. User identification across multiple social networks. In *Proc. of the International Conference on Networked Digital Technologies (NDT'09)*, pages 360–365, Ostrava, the Czech Republic, 2009.
27. D. Wang, D. Pedreschi, C. Song, F. Giannotti, and A.L. Barabási. Human mobility, social ties, and link prediction. In *Proc. of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'11)*, pages 1100–1108, San Diego, California, USA, 2011. ACM.
28. S. Ye, J. Lang, and F. Wu. Crawling online social graphs. In *Proc. of the International Asia-Pacific Web Conference (APWeb'10)*, pages 236–242, Busan, Korea, 2010. IEEE.
29. R. Zafarani and H. Liu. Connecting corresponding identities across communities. In *Proc. of the International Conference on Weblogs and Social Media (ICWSM'09)*, San Jose, California, USA, 2009. The AAAI Press.