Link prediction in bipartite graphs using *internal links* and weighted projection

Oussama Allali, Clémence Magnien and Matthieu Latapy

LIP6 – CNRS and Université Pierre et Marie Curie (UPMC – Paris 6)
4 place Jussieu
75252 Paris cedex 05– France
firstname.lastname@lip6.fr

***Abstract*—Many real-world complex networks, like client-product or file-provider relations, have a bipartite nature and evolve during time. Predicting links that will appear in them is one of the main approach to understand their dynamics. Only few works address the bipartite case, though, despite its high practical interest and the specific challenges it raises. We define in this paper the notion of internal links in bipartite graphs and propose a link prediction method based on them. We describe the method and experimentally compare it to a basic collaborative filtering approach. We present results obtained for two typical practical cases. We reach the conclusion that our method performs very well, and that internal links play an important role in bipartite graphs and their dynamics.**

I. Introduction

Many real-world complex networks have a natural bipartite structure and may therefore be modeled as bipartite graphs [1], *i.e.* two sets of nodes with links only between nodes in different sets. Typical examples include peer-to-peer file-provide graphs [2] where peers are linked to the files they provided; and client-product graphs where clients are linked to the products they bought [3].

Most of these networks are dynamic: they evolve during time, with node and link additions and removals. One approach for studying such dynamics is *link prediction*, which consists in predicting the links that will probably appear in the future, given a snapshot of the considered graph at a given time [4].

We address here the problem of link prediction in bipartite graphs. To do so, we define a special kind of links in bipartite graphs, which we call *internal links*. We then propose an approach based on these links and compare it to a basic classical approach. We study the performance of our method on two real-world datasets. We show that this method reaches very good performances and that internal links play a key role in the dynamics of real-world bipartite graphs.

The paper is organized as follows. We review related work in Section II and present the bipartite framework, including the notion of internal links, in Section III. We formally state the considered problem and its assessment in Section IV. We present in Section V our prediction method, and our experiments in Section VI. We discuss our conclusions and perspectives in Section VII.

II. Related work

Link prediction is a key research problem in dynamic network analysis. Several works study this problem on classical (non-bipartite) graphs [4], [5], [6], but they are not directly applicable to or appropriate for bipartite graphs. For instance because they rely on the presume of triangles in the graph. Up to our knowledge, only two works target this problem [7], [8]. The authors adapt some topological measures used in classical graphs for predicting links in bipartite graphs. In addition, they consider two transformations of the bipartite graph into a classical one, and they use a supervised learning algorithm to perform link prediction.

Another research problem is closely related to link prediction in bipartite graphs: the recommendation problem [9]. Recommendation systems are used to suggest items to users, such as products to customers for instance. Notice however that the two problems are quite different: recommendation aims typically at finding products of interest for *all* customers; prediction aims at finding links that will appear in the future. Predicting a huge number of new links for a given node and no links for the other nodes will therefore be of little interest regarding recommendation but may be a great success regarding prediction.

The most successful and widely used approach for recommendation is collaborative filtering [10], [11], [3], which consists in ranking the most relevant items for a given user in order of decreasing interest, and then in recommending the top N items to this user. We will use it in this paper for the purpose of comparison with our method.

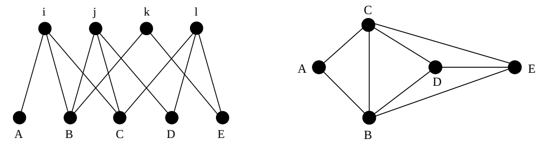


Fig. 1: An example of bipartite graph G (left), and its $\perp$-projection $G_\perp$(right).

III. Bipartite graphs, weighted projections, and internal links

We present here bipartite graphs and their transformations into (weighted) classical graphs, called projection. We also introduce a new class of links in bipartite graphs, which we call *internal links*. These links are at the core of our work.

A bipartite graph $G = (\perp, \top, E)$ is defined by a set $\perp$ of bottom nodes, a set $\top$ of top nodes and a set $E \subseteq \perp \times \top$ of links. The key point is that links exist only between a node in $\perp$ and one in $\top$. We denote by $N(u) = \{v \in (\perp \cup \top),\ (u, v) \in E\}$ the neighborhood of a node u in G. If $u \in \perp$ then $N(u) \subseteq \top$, and conversely.

The $\perp$-projection of G is the graph $G_\perp = (\perp, E_\perp)$ in which $(u, v) \in E_\perp$ if u and v have at least one neighbor in common in G: $N(u) \cap N(v) \neq \emptyset$. See Figure 1 for an example. We denote by $N_\perp(u)$ the neighborhood of a node u in $G_\perp$: $N_\perp(u) = \{v \in \perp,\ (u, v) \in E_\perp\} = N(N(u))$. The $\top$-projection of G, denoted by $G_\top$, is defined dually.

As explained for instance in [1], $G_\perp$ contains much less information than G. In particular, the fact that u and v are linked in $G_\perp$ means that they have *at least* one neighbor in common in G but says nothing on their *number* of common neighbors. One way to capture such information is to use a *weighted projection* in which a weight $\omega(u, v)$ is associated to each link $(u, v) \in E_\perp$. We present such weight functions in Section VI-A.

We now introduce a special class of links, called *internal links*, which play a key role in the whole paper.

Definition 1 (internal links): Let us consider a bipartite graph $G = (\perp, \top, E)$ and the bipartite graph $G' = (\perp, \top, E \cup \{(u, v)\})$ obtained by adding the link $(u, v) \in \perp \times \top$ to G, with $(u, v) \notin E$. The link (u, v) is *internal* if $G_\perp = G'_\perp$.

In other words, an internal link in a bipartite graph G is a pair of nodes (u, v) such that adding the link (u, v) to G does not change its $\perp$-projection. In Figure 1 for instance, (B, l) is an internal link. Indeed, all neighbors of l in G, namely $N(l) = \{C, D, E\}$, are already linked to B in $G_\perp$: the pairs of bottom nodes (B, C), (B, D) and (B, E) already have a neighbor in common in G, respectively, i, j and k. Adding link (B, l) to G increases their number of common neighbors to 2 and thus does not change $\perp$-projection.

We finally introduce the notion of induced links.

Definition 2 (induced links): Given a bipartite graph $G = (\perp, \top, E)$, the set of links induced by any pair of nodes (u, v) in $(\perp \times \top)$ is: $\perp(u, v) = \{u\} \times N(v) = \{(u, w), w \in N(v)\}$.

In Figure 1, for instance, $\perp(A, j) = \{A\} \times N(j) = \{(A, B), (A, C), (A, D)\}$. Notice that $E_\perp = \bigcup_{(u,v)\in E} \perp(u, v)$: the links of the $\perp$-projection of G are the links induced by all the links of G. By definition, a pair of nodes $(u, v) \in (\perp \times \top) \setminus E$ is an internal link if and only if all the links it induces are already in $G_\perp$. In Figure 1, for instance, $\perp(B, l) = \{(B, C), (B, D), (B, E)\} \subseteq E_\perp$ and therefore (B, l) is an internal link.

IV. The bipartite link prediction problem

Let us consider a dynamic bipartite graph defined by a set of n timestamped links $D = \{(t_i, u_i, v_i), i = 1...n\}$. Let $G = (\perp, \top, E)$ be the graph observed from a given instant a to another instant $b > a$: $\perp = \{u, \exists(t, u, v) \in D \text{ s.t. } a \leqslant t < b\}$, $\top = \{v, \exists(t, u, v) \in D \text{ s.t. } a \leqslant t < b\}$ and $E = \{(u, v), \exists(t, u, v) \in D \text{ s.t. } a \leqslant t < b\}$. We call G the *reference graph* and $[a, b[$ the *reference period*.

Now let us consider an instant $c > b$. This induces a set E' of links added to G during the period $[b, c[$, which we call the *prediction period*: $E' = \{(u, v), \exists(t, u, v) \in D \text{ s.t. } b \leqslant t < c\} \cap (\perp \times \top \setminus E)$. Notice that we consider only the links between nodes of G (we ignore new nodes appearing in the period $[b, c[$) which are not present in G (we consider links in $\perp \times \top \setminus E$ only).

In this framework, the goal of a link prediction method is to find a set P of *predicted links* which contains many of the links in E' but only few which are not in E'. Notice that in the extreme case where one predicts *all* possible links, *i.e.* $P = \perp \times \top \setminus E$, then one succeeds in predicting all links of E' but also predicts many links which are not in E'. Conversely, predicting no link at all, *i.e.* $P = \emptyset$, trivially does not predicting links not in E' but fails in predicting any link in E'.

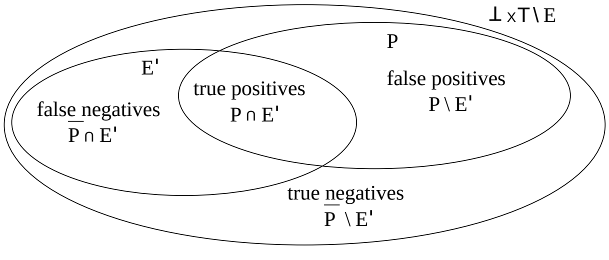


Fig. 2: A prediction method divides the set of possible links $\perp \times \top \setminus E$ into four categories: true positives, $P \cap E'$; true negatives, $\overline{P} \setminus E'$; false positives, $P \setminus E'$; and false negatives, $\overline{P} \cap E'$.

Let us denote by $\overline{P}$ the set of links that the method predicts will not appear: $\overline{P} = (\bot \times \top \setminus E) \setminus P$. Link prediction divides $\bot \times \top \setminus E$ the into four classes (see Figure 2): the set $P \cap E'$ of *true positives* is the set of appearing links that the method successfully predicts; the set $\overline{P} \setminus E'$ of *true negatives* is the set of unpredicted links which indeed do not appear; conversely, the *false positives* are the links in $P \setminus E'$, *i.e.* the links which we predicted but do not appear, and the *false negatives* are the links in $\overline{P} \cap E'$.

The aim of a link prediction method is to maximize the number of true positives and negatives while minimizing the number of false positives and negatives. This is classically captured by two quantities [12], called *precision*, *i.e.* $\frac{|P \cap E'|}{|P|}$ and *recall i.e.* $\frac{|P \cap E'|}{|E'|}$.

V. Internal link prediction

In this section, we introduce our link prediction method for bipartite graphs, which we call *internal link prediction*.

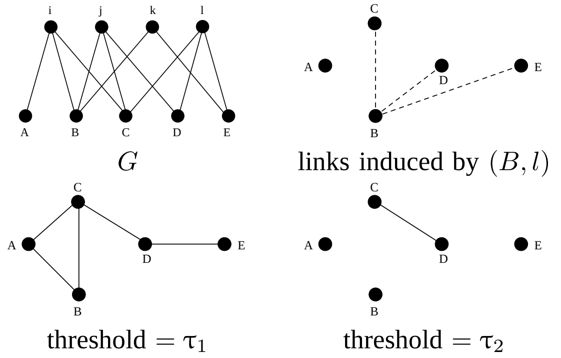
The key feature of our prediction method is that it focuses on internal links: it predicts internal links only (of which there are much less than possible links between $\top$ and $\bot$ nodes). The underlying intuition is that two bottom nodes which already have a common neighbor in G (*i.e.* they are linked in $G_\bot$) will probably acquire more in the future. Instead, if two nodes have no common neighbor in G, then they will probably still have none in the future. The links that can be added to G which fit both criteria are precisely internal links.

Going further, two bottom nodes with many common neighbors in G will probably have more in the future. We will capture this in a weight function (defined in the next section), with the expectation that the links that will appear are the internal links inducing $\bot$-links with high weights.

This leads to the following prediction method, which we call *internal links prediction*. Let us consider a weight function ω, and a given weight threshold τ. We denote by $E_{\bot\tau} = \{(u,v) \in E_\bot,\ \omega(u,v) \geq \tau\}$ the set of links in the projection that have a weight larger than or equal to τ. We then predict all the internal links which induce at least one link in $E_{\bot\tau}$.

Figure 3 shows an example of internal link prediction. The set of internal links of G is $\{(B,l),(C,k),(D,k),(E,j)\}$; let us focus on the internal link (B,l). It induces (B,C), (B,D), and (B,E). Given a threshold τ we predict (B,l) if one of these links has weight at least τ. For instance (see Figure 3):

- if $\tau = \tau_1$, only 5 links in the projection have weight larger than or equal to τ, including (B,C), which is induced by (B,l); we therefore predict (B,l);
- if $\tau = \tau_2$, only one link has the weight larger than or equal to τ, and it is not a link induced by (B,l); therefore we do not predict (B,l).



G links induced by (B,l)

threshold = τ_1 threshold = τ_2

Fig. 3: **Example of internal link prediction**. First row (left to right): a bipartite graph G and the links of $G_\bot$ induced by the internal link (B,l). Second row (left to right): the links $E_{\bot\tau_1}$ and $E_{\bot\tau_2}$ with weight at least thresholds τ_1 and τ_2 respectively, with $\tau_2 > \tau_1$, for a given weight function.

The complexity of this method is the same as the one of the collaborative filtering approach that we use for comparison, and it is bounded by the computation of $G_\bot$: it is in time $\mathcal{O}(\Delta_\bot |E|)$, where $\Delta_\bot = \max_{u \in \bot} |N_\bot(u)|$ is the largest degree in $G_\bot$, and space $\mathcal{O}(|\top| + |\bot|)$ in addition to the space needed for storing G.

VI. Experimental results

The performances of link prediction methods depend on various parameters, in particular the reference and prediction periods durations, and the weight function. In this paper we focus on the weight functions. We first describe three classical weight functions used in the literature. We then describe real-world datasets for our experiments. We show that the amount of internal links in them is high, which ensures the relevance of predicting internal links. Finally, we compare the performances of our approach for link prediction to the ones of the collaborative filtering approach, a classical recommendation technique.

A. Weight functions

Several approaches are used for weighting the links of the $\bot$-projection in order to capture more information than raw projections. We present the main ones in this section.

First, the weight of link (u,v) may be defined as the number of (top) neighbors that u and v have in common in the bipartite graph, called *sum* [13]:

$$\sigma(u,v) = |N(u) \cap N(v)|.$$

Notice that if u and v both have many neighbors, then $\sigma(u,v)$ will naturally tend to be high. Conversely, if u and v have only few neighbors but these neighbors

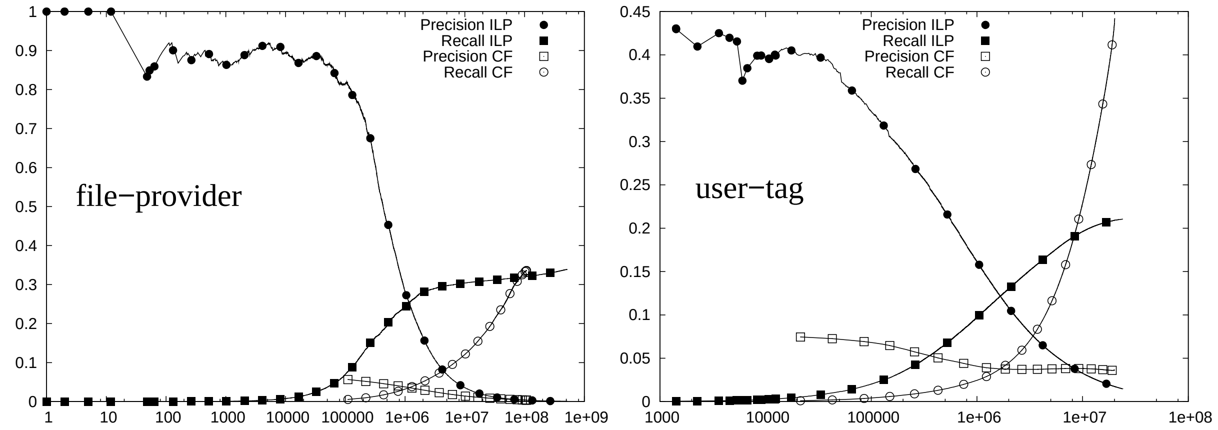


Fig. 4: Performances of the two considered link prediction methods *internal link prediction* (ILP) and collaborative filtering (CF) (left: file-provider bipartite graph; right: user-tag bipartite graph) when the weight function is the Jaccard coefficient. We plot precision and recall (vertical axis) as functions of the number of predicted links (horizontal axis).

are the same, then $\sigma(u, v)$ is low, which does not reflect the fact that u and v are very similar. To capture this, one may use the *Jaccard* coefficient [14]:

$$\gamma(u, v) = \frac{|N(u) \cap N(v)|}{|N(u) \cup N(v)|}.$$

The value of $\gamma(u, v)$ may however be strongly biased if one of the two nodes has many neighbors and the other one only few: the value would then be very low, even if all neighbors of one node are neighbors of the other. From this point of view, though, nodes play an unbalanced role: a $\top$-node x has an influence on the similarity between $\frac{|N(x)| \times (|N(x)|-1)}{2}$ pairs of $\bot$-nodes. When $N(x)$ is large, this is huge; on the contrary, if a $\top$-node only has two neighbors then it probably indicates a significant similarity between them. To capture this, one may consider that each $\top$-node *votes* for the similarity between its neighbors and that the sum of its votes is only one (it has only one voice to distribute). This leads to the *delta* function [15]:

$$\delta(u, v) = \sum_{x \in N(u) \cap N(v)} \frac{2}{|N(x)| \times (|N(x)| - 1)}.$$

All weighting functions presented above are natural and capture relevant informations about a bipartite graph. Each has its own strengths and weaknesses, and up to our knowledge there has been only limited comparison between them until now. By comparing their performance in the context of link prediction below, we expect to give some insight on their respective relevance in this context.

B. Data

Evaluating our method in practice requires the availability of large scale bipartite data *with their dynamics*. We use for our experiments a file-provider graph from a peer-to-peer measurement [16] and user-tag graph from delicious.com [17]. For each dataset, we choose reference and prediction periods which are representative of wide ranges of values for these parameters. Basic features of the reference graph G and the new links E' appearing during the prediction period are presented in Table I, for the two datasets.

	file-provider	user-tag
number of $\top$-nodes	$1,920,353$	$13,851$
number of $\bot$-nodes	$122,599$	$21,398$
number of links in E	$4,502,704$	$435,830$
number of links in E'	$1,170,504$	$1,663,799$
fraction of internal link in E'	34%	21%

TABLE I: Number of $\top$-nodes, $\bot$-nodes, links in the bipartite graph G, new links in the prediction period and fraction of internal links among them, for file-provider and user-tag bipartite graphs.

The fraction of internal links among the new links E' appearing during the prediction period is very high, 34% and 21% for file-provider and user-tag bipartite graphs respectively. This motivates our approach of focusing on this special class of links.

C. Impact of the number of predicted links

In order to illustrate the performances of our link prediction method, we observe the impact of the number of predicted links $|P|$ on the two prediction methods. We compute the precision and recall for all possible values of $|P|$ and plot them in Figure 4 as a function of $|P|$.

Note that high values of $|P|$ correspond to small values of the threshold τ for internal links prediction. If $\tau = 0$ then all possible internal links are predicted,

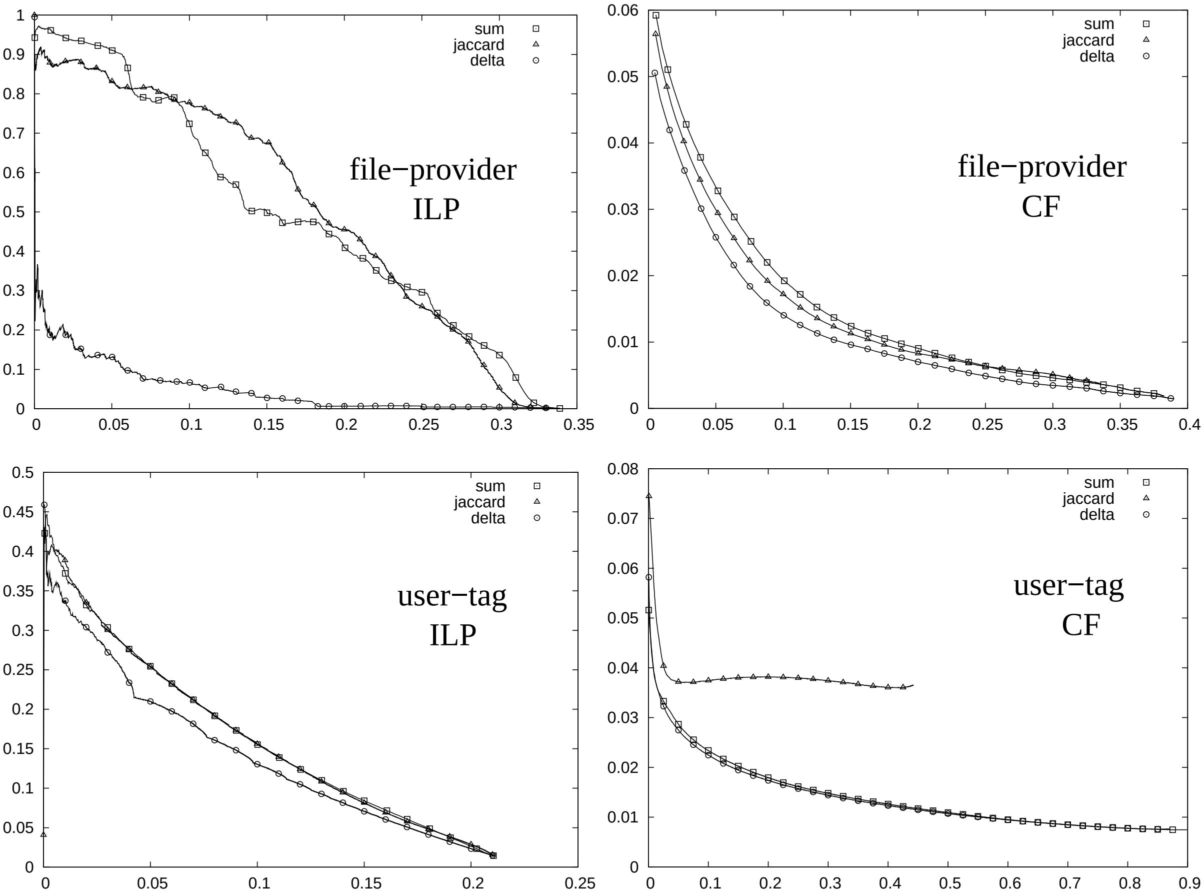


Fig. 5: Precision (vertical axis) as a function of recall (horizontal axis), for the three weight functions: sum, Jaccard and delta. First row: file-provider bipartite graph. Second row: user-tag bipartite graph. Left: *internal link prediction* (ILP); right: collaborative filtering (CF). Each point corresponds to the precision and recall obtained for a given value of τ or N.

which corresponds in this example to 34% and 21% of all appearing link for file-provider and user-tag bipartite graphs respectively. However, many of these links do not actually appear, and so the corresponding precision is almost zero. Instead, if a very high threshold is used then only few internal links are predicted, and so the obtained recall is almost zero. However, most of these few links do appear, which corresponds to a precision of almost 100% and 45% for file-provider and user-tag bipartite graphs respectively.

More generally, the number of predicted links $|P|$ has a strong impact on the performance of the prediction methods. Figure 4 shows that precision decreases and recall increases when $|P|$ increases, as expected. In practice, one has to choose a tradeoff between the two performance indicators. Figure 4 also shows that *internal link prediction* surpasses significantly collaborative filtering, but we do not detail this here.

D. Impact of the weight function

Let us now observe the impact of weight functions on both considered prediction methods and real-world datasets. We compute the precision and recall for all possible values of the threshold τ for *internal link prediction* and all possible values of N for collaborative filtering; we plot the obtained precision as a function of the obtained recall in Figure 5.

For *internal link prediction* (Figure 5, left), a first important observation is that the considered weight functions clearly split into two classes for the file-provider graph (first row): sum and Jaccard reach very high values of precision, and are also able to reach very good compromises between precision and recall (like a precision of 50% and a recall of 20%); instead, delta leads to poor performances. In the user-tag graph (second row), the three weight functions give good compromises between precision and recall.

No such behavior is observable for collaborative filtering (Figure 5, right), and for all weight functions internal link prediction performs much better than collaborative filtering (notice that the vertical axes are at different scales to help readability).

VII. Conclusion

In this paper, we introduce a new class of links in bipartite graphs, which we call *internal links*, and propose a method which uses them for solving the link prediction problem. We evaluate the relevance of this method by comparing it to a classical collaborative filtering approach and perform experiments on two datasets.

Our link prediction method has the following advantages. First, it performs very well, much better than a collaborative filtering approach, where no other method was previously available. Moreover, our method is purely structural: it relies on the identification of a specific kind of links which will probably appear in the future; this gives much insight on the properties of the underlying dynamics. Finally, the use of weight functions allows to tune the method in order to reach target tradeoffs in the quality of the prediction: one may use small thresholds to have excellent precision at the cost of a poorer recall, and conversely.

Our work may be extended in several ways. In particular, other (maybe more specific) weight functions may be introduced and tested. One may also predict internal links that induce *only* links with weight above the threshold (inducing *one* such link is sufficient in our current algorithm), or use both $\top$- and $\bot$-projections (our current algorithm only uses the $\bot$- one). There is therefore room for improving the method and its results.

Likewise, it would be interesting to conduct more experimentations and compare results on different datasets. Comparing our method with others, in particular machine learning approaches like the one presented in [7] is also appealing. Last but not least, our work calls for the development of link prediction methods for *external* links (those links which are not internal).

Another interesting direction would be to modify our approach in order to perform recommendation. As already explained, link prediction and recommendation are quite different problems, but they are strongly related. Just like we adapted collaborative filtering for link prediction in bipartite graphs, one may adapt our method and evaluate its relevance for recommendation.

Finally, we think that the notion of *internal links* introduced in this paper is fundamental and may be used as a building block in a much wider scope, in particular analysis of bipartite graphs in general. Although different, it is close to the notion of *redundancy* proposed in [1], which is one of the main statistics currently used for studying real-world bipartite graphs. The fraction of internal links in any bipartite graph and similar statistics based on internal links may be used for this same purpose, and have significant advantages over redundancy (in particular, it is not a local measure, and is related to the graph dynamics). We consider this as one of the main perspectives of our work.

References

[1] M. Latapy, C. Magnien, and N. D. Vecchio, "Basic notions for the analysis of large two-mode networks," *Social Networks*, 2008.

[2] J.-L. Guillaume, M. Latapy, and S. Le-Blond, "Statistical analysis of a P2P query graph based on degrees and their time-evolution," in *Proceedings of the 6-th International Workshop on Distributed Computing (IWDC'04)*, 2004.

[3] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: item-to-item collaborative filtering," *Internet Computing, IEEE*, 2003.

[4] D. Liben-Nowell and J. Kleinberg, "The link prediction problem for social networks," in *Proceedings of the twelfth international conference on Information and knowledge management(CIKM '03)*, 2003.

[5] M. A. Hasan, V. Chaoji, S. Salem, and M. Zaki, "Link prediction using supervised learning," in *Proceedings of SDM 06 workshop on Link Analysis, Counterterrorism and Security*, 2006.

[6] J. O'Madadhain, J. Hutchins, and P. Smyth, "Prediction and ranking algorithms for event-based network data," *ACM SIGKDD Explorations Newsletter*, 2005.

[7] N. Benchettara, R. Kanawati, and C. Rouveirol, "Supervised machine learning applied to link prediction in bipartite social networks," *Social Network Analysis and Mining, International Conference on Advances in*, 2010.

[8] Z. Huang, X. Li, and H. Chen, "Link prediction approach to collaborative filtering," in *Proceedings of the Joint Conference on Digital Libraries (JCDL05). ACM*, 2005.

[9] P. Resnick and H. Varian, "Recommender systems," *Communications of the ACM*, 1997.

[10] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon, and J. Riedl, "Grouplens: applying collaborative filtering to usenet news," *Communications of the ACM*, 1997.

[11] M. Deshpande and G. Karypis, "Item based top-n recommendation algorithms," *ACM Transactions on Information Systems*, 2004.

[12] P. R. Christopher D. Manning and H. Schütze, *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[13] M. E. J. Newman, "Clustering and preferential attachment in growing networks," *Physical Review Letters E*, 2001.

[14] S. Le-Blond, J.-L. Guillaume, and M. Latapy, "Clustering in P2P exchanges and consequences on performances." in *Proceedings of the 4th International Workshop on Peer-To-Peer Systems (IPTPS'05)*, 2005.

[15] L. A. Adamic and E. Adar, "Friends and neighbors on the web," *Social Networks*, 2003.

[16] F. Aidouni, M. Latapy, and C. Magnien, "Ten weeks in the life of an eDonkey server," in *Proceedings of the Sixth International Workshop on Hot Topics in Peer-to-Peer Systems (Hot-P2P'09)*, 2009.

[17] O. Görlitz, S. Sizov, and S. Staab, "Pints: Peer-to-peer infrastructure for tagging systems," in *Proceedings of the Seventh International Workshop on Peer-to-Peer Systems, IPTPS*, 2008.