Supervised Rank Aggregation Approach for Link Prediction in Complex Networks

Manisha Pujari
LIPN CNRS UMR-7030
Université Paris Nord
93430-Villetaneuse, France
manisha.pujari@lipn.univ-paris13.fr

Rushed Kanawati
LIPN CNRS UMR-7030
Université Paris Nord
93430-Villetaneuse, France
rushed.kanawati@lipn.univ-paris13.fr

ABSTRACT

In this paper we propose a new topological approach for link prediction in dynamic complex networks. The proposed approach applies a supervised rank aggregation method. This functions as follows: first we rank the list of unlinked nodes in a network at instant t according to different topological measures (nodes characteristics aggregation, nodes neighborhood based measures, distance based measures, etc). Each measure provides its own rank. Observing the network at instant $t+1$ where some new links appear, we weight each topological measure according to its performances in predicting these observed new links. These learned weights are then used in a modified version of classical computational social choice algorithms (such as Borda, Kemeny, etc) in order to have a model for predicting new links. We show the effectiveness of this approach through different experimentations applied to co-authorship networks extracted from the DBLP bibliographical database. Results we obtain, are also compared with the outcome of classical supervised machine learning based link prediction approaches applied to the same datasets.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

General Terms

Graph theory, social network analysis

Keywords

link prediction, ranked list aggregation

1. INTRODUCTION

Analyzing dynamic large-scale networks is a major emerging topic in different research areas. Actually, many real-world systems can be readily modeled as an evolving network of interacting *actors*. This is namely the case of on-line social networks, collaboration networks (such as academic co-publishing networks, product co-purchasing, etc), biological systems (such as protein interaction networks) and computer science networks as the Internet and peer-to-peer networks.

One of the major problems in studying dynamic evolution of complex networks, is the problem of *link prediction* [18, 20]. This refers to the problem of finding associations (edges) in the network at a given point of time t when provided with the information about the network's temporal history before time t. The problem has a wide variety of applications : recommender systems, identification of probable professional or academic associations in scientific collaboration networks or e-commerce sites, identification of structures of criminal networks and structural analysis in the field of microbiology or biomedicine etc. All these demand for much more efficient and versatile approaches for link prediction and thereby making it an important and scientifically attractive research topic.

A variety of approaches has been proposed in the scientific literature. Recent surveys on the topic can be found in [20, 3]. A major trend is composed of topological approaches : these are approaches, based merely on mining topological evolution of the network history in order to predict the appearance of new links [18]. Such approaches are inherently application-field independant. They spare the need for any specific knowledge about the actors (i.e. nodes) of the studied network. Meanwhile, these approaches can be combined with node content approaches for enhancing prediction performances [13].

Since the work of *Liben-Nowell* and *Kleinberg* [18] showing that topological measures can be effectively used for predicting links, a number of works has been published focusing on how to combine different topological metrics in order to enhance prediction performances. One popular approach is using supervised machine learning algorithms. A short survey of existing approaches is provided in Section 2. However, surprisingly enough, no work has attempted to combine the prediction power of individual topological metrics by applying computational social choice algorithms (or what is also known by rank agreggation methods) [9]. In this work, we report on applying a *supervised* version of classical social choice algorithms to link prediction. The basic idea of a social choice algorithm is to merge preferences of different *voters* in order to obtain a preference vector minimizing the distance to all individual preferences. In this work, we modify these classical approaches by learning weights to attribute to each individual voter (here the voter is a topological measure) in order to enhance the overall performances of the prediction task.

The reminder of the paper is organized as follows. Section 2 contains a short overview of the related work in this field. Section 3 gives a detailed description of what actually we mean by rank aggregation and the existing methods. In section 4 , we describe our proposed approach of link pre-

WWW 2012 Companion, April 16–20, 2012, Lyon, France.
ACM 978-1-4503-1230-1/12/04.

diction by supervised rank aggregation. Section 5 contains a brief description of experiments done so far and the preliminary results obtained. We conclude in section 6 with a short description of future perspectives.

2. RELATED WORK

As mentioned earlier, we are only interested here in topological approaches for link prediction. Different approaches similar to this has been proposed in the scientific literature. These can be broadly classified into two classes : *Dyadic* and *structural* approaches. In a dyadic approach the main question is to learn a model or to compute a score that evaluate the probability of establishing a link between two unlinked nodes in the network. Examples of dyadic approaches are given in [18, 5, 14].

On the other side, structural approaches try to learn models of the evolution of the whole graphs or to learn rules of evolution of sub-graphs in the studied network. Example of work of the first type is presented in [1] where authors apply a matrix factorization based method in order to learn a model for the evolution of the network adjacency matrix. In [17] an interesting sub-graphs rule evolution miner is presented. While structural approaches may allow to predict the formation of different links at a time, they scale much poorly (at least existing methods) to be applied to large-scale graphs.

In this paper we present a new dyadic topological approach. One historical dyadic approach is the work presented in [18] where authors propose to rank unlinked pair of nodes based on their topological characteristics. In this work authors have considered many different types of topological attributes to characterize a pair of unlinked nodes but mostly concentrating on proximity based attributes. They rank the pair of unlinked nodes by different attributes and compute the individual performance in link prediction task. Our approach is different from this approach in a sense that here authors consider each topological attribute individually and calculate their effectivity in predicting new links where as we try to aggregate the ranking information obtained from each attribute by using a rank aggregation algorithm before making any prediction.

Work provided in [5] is a temporal link prediction approach based on supervised machine learning where link prediction is done by using decision tree algorithm with boosting. The authors have improved the prediction result by considering the dynamic aspects of the network. Another approach is given in [14]. In this paper also, the authors propose to use supervised machine learning algorithms for the purpose of link prediction.They make a comparative analysis on the suitability of many learning algorithms to be used in link prediction. One of the interesting study that they made, was to use rank algorithms to compare the attributes used in order to judge their relative strength in a link prediction task.

Looking into the work on rank aggregation techniques, we can say that rank aggregation methods were very much a part of social choice theory and were mostly applied to political and election related problems [10, 6, 23]. It was not until recent years that these methods found an application outside, especially in metasearch [4, 8], multiple search, similarity search[11] etc. All these works apply unsupervised rank aggregation algorithms.

A significant work on supervised rank aggregation has been done in [19] where authors propose supervised aggregation by Markov chain to enhance the ranking result on meta-searches. Another very recent work is described in [22] where authors apply supervised rank aggregation to find influential nodes and future links.

3. RANK AGGREGATION

Before describing our approach, we provide here a brief account about *ranked list aggregation/rank aggregation* process and the existing methods.

List merging or list aggregation refers to the process of combining a number of lists with same or different elements in order to get a single list with all elements in it. In rank or preference aggregation, the order or rank of elements in input lists is also taken into consideration. The input lists can be categorized as *Full* lists, *Partial* lists or *Disjoint* lists. Full lists are the lists that contain exactly the same elements but with a different ordering, partial lists may have some of the elements in common but not all and disjoint lists have completely different elements. For example we consider four lists :

$$L_1 = [A, B, C, D]$$
$$L_2 = [B, D, A, C]$$
$$L_3 = [A, B, C, D, E]$$
$$L_4 = [E, F, G, H]$$

Here, L_1 and L_2 are full or complete lists, L_1 and L_3 are partial lists, and L_1 and L_4 are discrete/disjoint lists.

In ranked list aggregation or simply rank aggregation, distance metrics are used to find the disagreement between two lists/rankings. Two well-known distance measures are

- Spearman Footrule distance: This computes the distance between two ranked lists by computing the sum of differences in rankings of each element. Formally, it is given by

$$F(L_1, L_2) = \sum_{i \in n} \mid L_1(i) - L_2(i) \mid \quad (1)$$

- Kendall Tau distance: This counts the number of pairs of elements that have opposite rankings in the two input lists i.e. it calculates the pairwise disagreements.

$$K(L_1, L_2) = \mid (i, j)\ s.t.\ L_1(i) \leq L_2(j)\ \&\ L_1(i) \geq L_2(j) \mid \quad (2)$$

where L_1 and L_2 are the input lists and $L_1(i)$ and $L_2(i)$ represent the ranks of element i in the two lists correspondingly.

Rank aggregation methods can be categorized into two types: score-based and order-based. Score- based aggregation methods use score information from individual rankers while order-based methods use only the rank information [19].

Three standard methods are described in [21] namely Borda's method, Markov chain method and Median rank method. Borda's method is a truly positional method as it is based on the absolute positioning of the ranked elements rather than their relative rankings. A Borda score is calculated for each element in the lists and based on this score the elements are ranked in the aggregated list. For a set of complete ranked lists $L = [L_1, L_2, L_3, ..., L_k]$, the Borda's score for a element i and a list L_k is given by:

$$B_{L_k}(i) = \{count(j) | L_k(j) < L_k(i) \& j \in L_k\} \quad (3)$$

The total Borda's score for an element is given as:

$$B(i) = \sum_{t=1}^{k} B_{L_t}(i) \quad (4)$$

Borda's method is mostly applicable to complete lists and is not very suitable for partial lists.

Markov's chain methods represent the elements of ranked lists as nodes of graph and the transitional probabilities from one node to other is defined by the relative rankings of the elements in different input lists. This method is suitable for aggregation of complete as well as partial lists but does not guarantee a most optimized solution.

The third method, Median rank aggregation, makes use of *MEDRANK* algorithm to find an aggregation of a set of complete rankings.This method may give an optimal aggregation but is limited in use by its applicability to full lists.

In [21] author makes use of item similarity in order to enhance the performance of the standard methods for list aggregation. Another approach proposed in [8], is *Kemeny optimal aggregation* which makes use of Kendall Tau distance to find the optimal aggregation. The first step is to find a initial aggregation of input lists using any standard method. The second step is to find all possible permutations of the elements in the initial aggregation. For each permutation, a score is computed which is equal to the sum of distances between this permutation and the input lists. The permutation having the lowest score is considered as optimal solution. For example, for a collection of input rankings $\tau_1, \tau_2, \tau_3, ..., \tau_k$ and an aggregation π , the score is given by:

$$SK(\pi, \tau_1, \tau_2, \tau_3, ..., \tau_k) = \sum_{i \in k} K(\pi, \tau_i) \quad (5)$$

In spite of all advantages Kemeny optimal aggregation is computationally hard to implement. So we look for an alternative solution that will give similar kind of aggregation but being computationally feasible. That leads us to another approach named *Local kemenization* [8]. A full list π is locally Kemeny optimal aggregation of parial lists $\tau_1, \tau_2, \tau_3, ..., \tau_k$, if there is no full list π' that can be obtained from π by performing a single transposition of a single pair of adjacent elements and for which

$$SK(\pi', \tau_1, \tau_2, \tau_3, ..., \tau_k) < SK(\pi, \tau_1, \tau_2, \tau_3, ..., \tau_k)$$

In other words it is impossible to reduce the total distance of an aggregation by flipping any adjacent pair of elements in the aggregation.

4. OUR APPROACH

4.1 Supervised rank aggregation

The existing methods for rank aggregation described in the previous section, usually give equal weights to all experts or rankers who provide the input rankings. But sometime, it may happen that these rankers have different importance in identifying the correct order of elements. For example, it may happen that some of the rankers are biased or like in meta-search, the ranking lists are generated from different sources which have their different capacity and accuracies. These facts motivate us to think that attaching a weight with each ranker may enhance the aggregation results significantly. We thus propose two ways to introduce weights into Borda's method and local Kemeny optimal method.

Supervised Borda: We have tried to introduce weights into Borda's method in the following way. Suppose $(w_1, w_2, \ldots, w_n)$ are the weights for n rankers (and thus for the ranked lists provided by them), the Borda score for individual element can be obtained as follows:

$$B(i) = \sum_{t=1}^{n} w_i * B_{L_t}(i) \quad (6)$$

Supervised Local Kemeny Aggregation: We also try to introduce weight into the local Kemeny aggregation method. Algorithm 1 describes our proposed approach for finding supervised local Kemeny aggregation.

Algorithm 1 Supervised local kemeny aggregation

Input: $T = [\tau_1, \tau_2, \ldots, \tau_r]$ where $\tau_i = [e_1, e_2, \ldots, e_m]$ for r rankers and m elements
$W = [w_1, w_2, \ldots, w_r]$ where w_i is the weight for ranker i and $w_T = \sum_{i=1}^{r} w_i$
$\mu = \tau_1$ where μ can be considered as initial aggregation
Output: π : an aggregated list of elements

```
Initialize an empty matrix M
for element x = 1 to m − 1 do
  for element y = 1 to m do
    score = 0
    for τ_i ∈ T do
      xPREFy = { 0  if τ_i(x) > τ_i(y)
               { 1  if τ_i(x) < τ_i(y)
      score = score + (w_i * xPREFy)
    end for
    if score > 0.5 * w_T then
      M_xy ⇐ true
      M_yx ⇐ false
    else
      M_xy ⇐ false
      M_yx ⇐ true
    end if
  end for
end for
Bubble sort μ using M.
if M_xy = false then
  swap(x,y)
end if
Return μ
```

4.2 Applying supervised rank aggregation to link prediction

Our approach proposes a novel way to predict links in dynamic graphs having temporal sequence of graphs. The whole sequence is divided into three parts: *training*, *labeling* and *testing* or *validation*. Three graphs namely G_{learn}, G_{label} and G_{test} are generated by making union of the temporal sequences of the graphs for corresponding time slots. The training data is constructed as follows. An example will be generated for each couple of nodes (x, y) that are not linked in G_{learn} but both belonging to the same connected component. The class labeling is obtained by checking whether the couple of nodes is indeed connected in G_{label}. If such a connection exists then it will be a positive example in the supervised learning task and if no connection exists,

it will be a negative example [5].Thus, examples are generated from these graphs for both training and validation. These examples are also characterized by a given number of topological attributes.

Each attribute of an example has the capacity to provide some unique information about the data when considered individually. The training examples are ranked based on the attribute values. So, for each attribute we will get a ranked list of all examples. Considering only the *topk* ranked examples and with an assumption that when we rank the examples according to their attribute values, the positive examples should be ranked on the top, we compute the performance of each attribute. This performance is measured in terms of either *precision* (maximization of identification of positive examples) or *false positive rate* (minimization of identification of negative examples) or a combination of both. Based on the individual performances, a weight is assigned to each attribute.

For validation, we use examples obtained from the validation graph characterized by same attributes and try to rank all examples based on their attribute values. So for n different attributes we shall have n different rankings of the test examples. These ranked lists are then merged using a *supervised rank aggregation* method and the *weights of the attributes* obtained during learning process. The *topk* ranked examples in the aggregation are taken to be the predicted list of positive examples.Using this predicted list, we calculate the performance of our approach. k in this case is equal to the number of positive examples in the validation graph.

Computation of attribute weights: Weights of the attributes are computed based on the following criteria :

- **Maximization of positive precision:** Based on maximization of identification of positive examples the attribute weight is calculated as

$$W_{a_i} = n * Precision_{a_i} \tag{7}$$

where n is the total number of attributes and $Precision_{a_i}$ is the *precision* of attribute a_i based on identification of positive examples.

- **Minimization of false positive rate:** By minimizing the identification of negative examples we get a weight as below

$$W_{a_i} = \frac{n}{FPR_{a_i}} \tag{8}$$

where n is the total number of attributes and FPR_{a_i} is the $false\ positive\ rate$ of attribute a_i based on identification of negative examples examples.

5. EXPERIMENTATION

We evaluated our approach using data obtained from DBLP [1] databases. DBLP is a scientific bibliography website containing a large database of articles mostly related to computer science. Our network consists of authors and their publications providing us a bipartite structure of graph. The data used corresponds to a time span of 1970 to 1980. This data is divided into three datasets containing information for different years each having a training set and a test or validation set. Following the procedure described in the previous section, we generate examples for each dataset. Table.1 summarizes information about the three datasets created from DBLP data.

Before giving a brief account of the attributes used, here are some basic notations that are applied afterwards. We denote by $\Gamma_G(x)$ the set of direct neighbors of a node x belonging to a graph G. The set of neighbors of a node x is denoted $\Gamma(x)$ when there is no ambiguity concerning the considered graph. $\parallel E \parallel$ refers to the cardinality of set E. Graphs used in our study are non oriented or undirected and the degree of a node x in a graph G is equal to $\parallel \Gamma_G(x) \parallel$.

The attributes characterizing each example in each dataset are:

- **Neighborhood-based attributes:**
  - Common neighbors : $CN(x,y))=\parallel \Gamma(x) \cap \Gamma(y) \parallel$
  - Jaccard's coefficient : $JC(x,y))=\frac{\parallel \Gamma(x) \cap \Gamma(y) \parallel}{\parallel \Gamma(x) \cup \Gamma(y) \parallel}$
  - Adamic Adar: $AD(x,y)=\sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{log \parallel \Gamma(z) \parallel}$ [2]
  - Preferential attachment: $PA(x,y)=\parallel \Gamma(x) \times \Gamma(y) \parallel$ [15]
- **Distance-based attributes:**
  - Shortest path distance(Dis)
  - Katz: $Katz(x,y) = \Sigma_{l=1}^{\infty} \beta^{\ell} \times \parallel path_{x,y}^{(\ell)} \parallel$, where $path_{x,y}^{(\ell)}$ is the number of paths between x and y of length ℓ and β is a positive parameter which favours shortest paths [16]
  - Maximum forest algorithm (MFA): It makes use of *Laplacian* matrix and identity matrices of the graph to find distance between all the nodes of a graph.A matrix is obtained as $M = (I + M_L)^{-1}$ where I is an identity matrix and M_L is the *Laplacian* of the graph being studied. So, $MFA(x,y) = M(x,y)$ [12].
- **Centrality-based attributes:**
  - Product of PageRank (PPR)[7]
  - Product of degree centrality (PCD)
  - Product of clustering coefficient (PCF)

These attributes can be computed directly from the bipartite graph and/or also from the projected graphs. Projected graph refers to the unimodal graphs obtained by projecting the bipartite graph over one of its node sets [5]. The attributes computed from projected graphs are called indirect attributes.

Before using the rank aggregation method, we ranked all the test examples by their attribute values. Considering only the top-k examples and taking the number of positive examples in each dataset as the corresponding value of k, we compute the performance of each attribute in identifying the real positive link in the *topk* positions. Table.2 summarizes this information.

5.1 Results

In the first part of experimentation we applied our approach to the complete datasets. For rank aggregation, we have used supervised Borda's method. We also tried

[1]http://www.dblp.org

Datasets	Training Time	Validation Time	Training examples		Test examples	
			Positive	Total	Positive	Total
Dataset 1	[1970,1973,1974,1975]	[1971,1974,1975,1976]	30	1693	41	3471
Dataset 2	[1972,1975,1976,1977]	[1973,1976,1977,1978]	87	19332	82	18757
Dataset 3	[1974,1977,1978,1979]	[1975,1978,1979,1980]	102	35190	164	60046

Table 1: DBLP Datasets

Attributes	Dataset1	Dataset2	Dataset3
Katz	0	0	0.0061
MFA	0.0244	0.0732	0.0488
PPR	0.0244	0.0244	0
PCF	0.0732	0.0244	0
PCD	0	0	0
VC	0.5122	0.4268	0.1829
JC	0.2195	0.1707	0.0488
AD	0.1463	0.1463	0.1463
AP	0.0488	0	0
Dis	0	0.0122	0.0244
Indirect Dis	0.6098	0.5366	0.8171
Indirect AD	0.0976	0.0488	0.0366
Indirect AP	0.0244	0	0
Indirect PCD	0.0244	0.0122	0.0061
Indirect MFA	0.0488	0.0732	0.0427
Indirect PCF	0.4878	0.4756	0.4207
Indirect JC	0.0488	0.1098	0.0549
Indirect PPR	0.0488	0	0
Indirect VC	0.0488	0.0488	0.0183
Indirect Katz	0.1220	0.1098	0.0488

Table 2: Results(average precision) obtained by ranking the test examples by attribute values

to compare our approach with link prediction approaches using basic machine learning algorithms like Decision tree, Naive bayes and k-Nearest neighbors algorithm. We named our approaches as Supervised Borda_prec and Supervised Borda_fpr based on how the attribute weights are computed. We will follow the same convention to represent supervised local Kemeny also. Figure-1 shows the results obtained on the complete datasets in terms of F1-measure. F-measure is defined by the harmonic mean of both precision and recall.

$$F = \frac{Precision * Recall}{Precision + Recall} \tag{9}$$

For dataset 1, our approach based on maximization of positive precision, gives a comparatively better performance in terms of F1-measure as compared to most of the other methods. For dataset 3, both our methods perform remarkably well.

As observed, the number of negative examples in all three datasets is remarkably higher than the number of positive examples. So in the next part of the experiment, we created random samples of learning examples from these datasets, keeping all the positive examples but limiting the number of negative examples. We wanted to see the effect on the result if we train our model by limiting the number of negative examples. If N is the number of negative examples and P is the number of positive examples in a sample, then $N = m * P$. P is always equal to the number of positive examples in the original training example sets. Hence, any sample for training contains all the positive examples and N randomly chosen negative examples. Under this criteria, experiment was made on learning samples created by increasing the number of negative examples gradually. Figure-2 gives a graphical comparison of results for different approaches in terms of average precision over 10 learning samples.With the increasing number of negative examples our method Supervised Borda_prec outperforms all other classical methods except for in the end where the performance of KNN algorithm is very close to ours.

Datasets	Training examples		Test examples	
	Positive	Total	Positive	Total
Dataset 1	30	1693	41	246
Dataset 2	87	19332	82	492
Dataset 3	102	35190	164	984

Table 3: DBLP Datasets

In order to apply supervised local Kemeny aggregation without facing any computational difficulties, for the preliminary experimentation we tried to reduce the size of the set of validation/test examples. This enabled us to use supervised local Kemeny aggregation to get a very quick result. The new test samples were generated by keeping all positive examples intact but limiting the number of negative examples to five times of the positive examples. So now the new set of examples are presented in Table.3.

We compared our supervised rank aggregation based methods with other machine learning based approaches like before and also with the approach based on using Decision tree with boosting proposed in [5]. These results are measured in terms of F1-measure. Figure-3 shows the results obtained on dataset 1 by using supervised and unsupervised Borda and local Kemeny aggregation approaches and it clearly shows that application of weights on attributes have significantly improved the prediction result.

Figure-4 gives the comparative result obtained in terms of F1 measure for our rank aggregation based methods and other baseline approaches using decision tree and decision tree with boosting. Clearly, our approach gives a comparatively better result as compared to other machine learning based methods.

6. CONCLUSION

In this paper we propose a novel approach for solving the problem of link prediction network. Our approach is based on supervised rank aggregation and is motivated by the belief that each attribute can provide us with some unique information which can be aggregated in the end to make a better prediction of association between two unconnected entities in a network. First we have come up with a new way

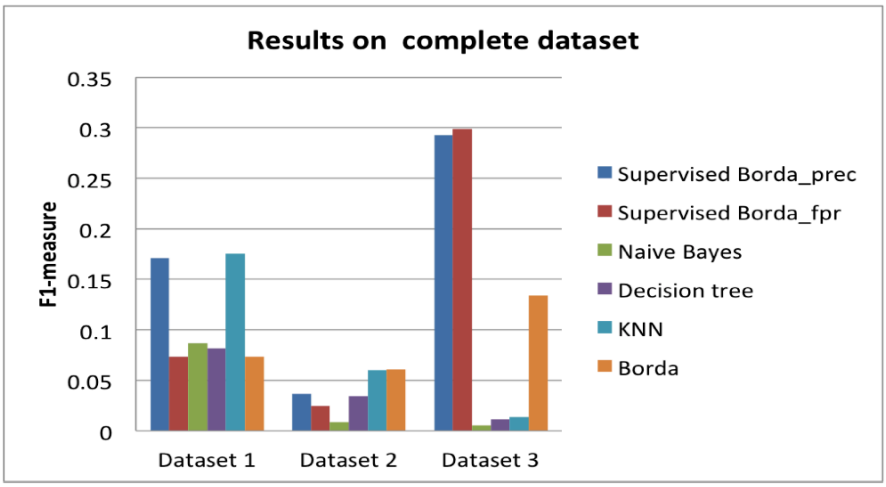

Figure 1: F1-measure for complete test set by learning on complete datasets

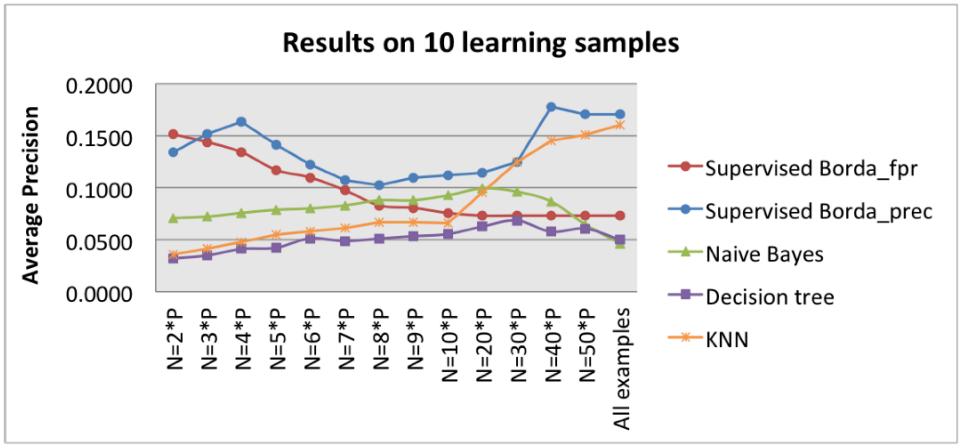

Figure 2: Average precision for learning on varying samples of Dataset 1

of introducing weights in the two well known rank aggregation methods. And secondly, we have proposed to apply this approach for the purpose of link prediction in real complex graphs. We evaluated our approach on a co-authorship network obtained from DBLP database. The experimental results are quite encouraging as in many cases, our method seems to perform better than the approaches using classical machine learning algorithms for link prediction.

Based on the results obtained so far, we intend to continue our experiments on other type of data. We are also interested to research further with supervised local Kemeny aggregation based approach in order to apply it further on bigger datasets.

7. REFERENCES

[1] E. Acar, D. M. Dunlavy, and T. G. Kolda. Link Prediction on Evolving Data Using Matrix and Tensor Factorizations. In *ICDM Workshops*, pages 262–269, 2009.

[2] L. A. Adamic, O. Buyukkokten, and E. Adar. A social network caught in web. In 8, editor, *First Monday*, number 6, June 2003.

[3] Al Hasan Mohammad and Zaki Mohammed J. A SURVEY OF LINK PREDICTION IN SOCIAL NETWORKS. In Charu C. Aggarwal, editor, *Social network Data Analysis*, chapter 9, pages 243–275. Springer, 2010.

[4] J. A. Aslam and M. Montague. Models for metasearch. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '01, pages 276–284, New York, NY, USA, 2001. ACM.

[5] N. Benchettara, R. Kanawati, and C. Rouveirol. Supervised machine learning applied to link prediction in bipartite social networks. In *ASONAM'10*, pages 326–330, 2010.

[6] D. Black, R. Newing, I. McLean, A. McMillan, and B. Monroe. *The Theory of Committees and Elections by Duncan Black, and Revised Second Editions Committee Decisions with Complementary Valuation by Duncan Black*. Kluwer Academic Publishing, 2nd edition.

[7] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks*

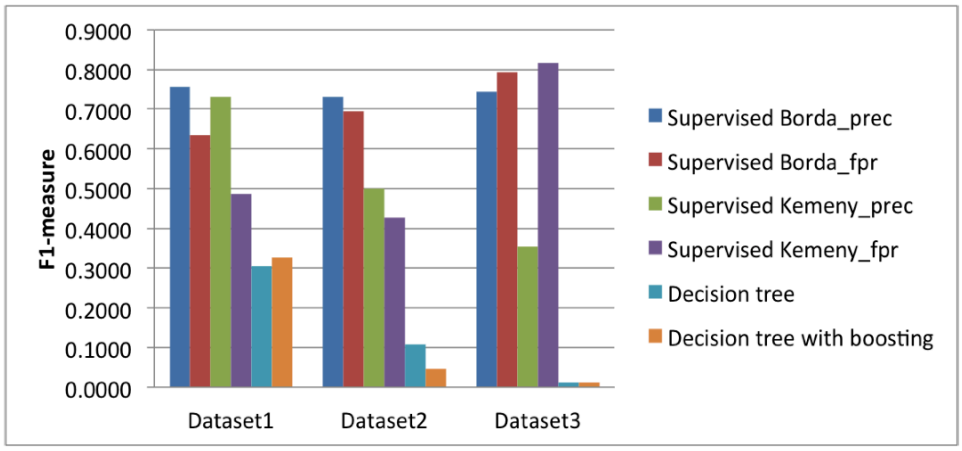

Figure 4: F1-measure on reduced test set by learning on complete training set of Dataset 1

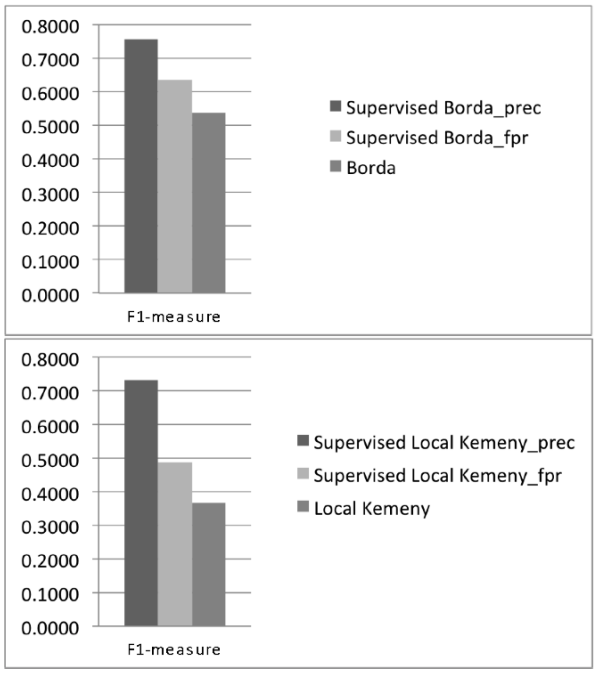

Figure 3: F1-measure on reduced test set by learning on complete training set of Dataset 1

and ISDN Systems, 30:107 – 117, 1998. lt;ce:title>Proceedings of the Seventh International World Wide Web Conferencelt;/ce:title>.

[8] C.Dwork, R.Kumar, M.Naor, and D.Sivakumar. Rank aggregation methods for web. In *Proceedings of the 10th international conference on World Wide Web*, WWW '01, pages 613–622, Hong Kong, 2001. ACM.

[9] Y. Chevaleyre, U. Endriss, J. Lang, and N. Maudet. A short introduction to computational social choice. *SOFSEM 2007: Theory and Practice of Computer Science*, pages 51–69, 2007.

[10] J. de Borda. Memoire sur les elections au scrutin. 1781.

[11] R. Fagin, R. Kumar, and D. Sivakumar. Efficient similarity search and classification via rank aggregation. In *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, SIGMOD '03, pages 301–312, New York, NY, USA, 2003. ACM.

[12] F. Fouss, A. Pirotte, J.-M. Renders, and M. Sarens. Random-Walk Computation of Similarities between Nodes of a Graph with Application to Collaborative Recommandation. *IEEE Transactions on knowledge and data engineering*, 19(3):355–369, 2007.

[13] S. Gao, L. Denoyer, and P. Gallinari. Temporal link prediction by integrating content and structure information. In *Proceedings of the 20th ACM international conference on Information and knowledge management - CIKM '11*, page 1169, New York, New York, USA, 2011. ACM Press.

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

[15] Z. Huang, X. Li, and H. Chen. Link prediction approach to collaborative filtering. In *Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries*, JCDL'05, pages 141–142, New York, USA, 2005. ACM.

[16] L. Katz. A new status index derived from sociometric analysis. In *Psychometrika*, pages 39–43, March 1953.

[17] M. Lahiri and T. Y. Berger-Wolf. Structure Prediction

in Temporal Networks using Frequent Subgraphs. In *CIDM*, pages 35–42. IEEE, 2007.

[18] 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, pages 556–559, New York, NY, USA, 2003. ACM.

[19] Y.-T. Liu, T.-Y. Liu, T. Qin, Z.-M. Ma, and H. Li. Supervised rank aggregation. In *Proceedings of the 16th international conference on World Wide Web*, WWW '07, pages 481–490, New York, NY, USA, 2007. ACM.

[20] L. Lü and T. Zhou. Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications*, 390(6):1150–1170, Mar. 2011.

[21] D. Sculley. Rank aggregation for similar items. In *Proceedings of the Seventh SIAM International Conference on Data Mining (SDM)*, April 2007.

[22] K. Subbian and P. Melville. Supervised rank aggregation for predicting influence in networks. In *In the proceedings of the IEEE Conference on Social Computing (SocialCom-2011).*, Boston, October 2011.

[23] H. Young and A. Levenglick. A consistent extension of condorcet's election principle. *SIAM Journal on Applied Mathematics*, 35(2), 1978.