A Proximity Measure for Link Prediction in Social User-Item Networks

Chun-Hao Fu, Cheng-Shang Chang and Duan-Shin Lee
Institute of Communications Engineering
National Tsing Hua University
Hsinchu 300, Taiwan, R.O.C.
u9611111@oz.nthu.edu.tw, cschang@ee.nthu.edu.tw, lds@cs.nthu.edu.tw

Abstract

Recommendation systems based on historical action logs between users and items are usually formulated as link prediction problems for user-item bipartite networks, and such problems have been studied extensively in the literature. With the advent of on-line social networks, social interactions can also be recorded and used for predicting user's future actions. As such, the link prediction problem based on the union of a social network and a user-item bipartite network, called a social user-item network in this paper, has been a hot research topic recently. One of the key challenges for such a problem is to identify and compute an appropriate proximity (similarity) measure between two nodes in a social user-item network. To compute such a proximity measure, in this paper we propose using a random walk with two *different jumping probabilities toward different neighboring nodes. Unlike the simple random walk, our method is able to assign different weights to different paths and thus can lead to a better proximity measure by optimizing the two jumping probabilities. To test our method, we conduct various experiments on the DBLP dataset [21]. With a 3-5 year training period, our method performs significantly better than random guess in terms of minimizing the root mean squared error.*

keywords: social networks, user-item networks, personal recommendation, link prediction

1 Introduction

Recommendation systems based on historical action logs between users and items are usually formulated as link prediction problems for user-item bipartite networks, and such problems have been studied extensively in the literature (see e.g., [5, 9, 22, 10, 1]). In practice, there are also many commercial websites such as Amazon [24] and Netflix [19] that have implemented recommendation systems to provide personal recommendations. In a recommendation system, there are a set of users and a set of items. The interaction between a user and an item, such as buying, renting, or rating, is represented by an edge (with a certain weight) between them, and the collection of these interactions are then represented by a bipartite network, called a *user-item* network. Based on such a bipartite network, a recommendation system then comes up with a list of personal recommendations for a specific user. In the literature, there are several methods for doing this. Among them, collaborative filtering (see e.g., [10] for a survey) is one of the most frequently used techniques. Collaborative filtering can be further separated into two categories: neighborhood-based methods [6, 12, 24, 16] and model-based methods [11, 26]. The neighborhood-based methods use the ratings of those users (resp. items) that are *similar* to the target user (resp. item) to come up with the recommendations for the target user (resp. item). As such, one key step for a neighborhood-based method is to compute the proximity (similarity) measure [23] between two nodes in the bipartite network. On the other hand, model-based methods (see e.g., [20, 2] for surveys of these methods) use the bipartite network to learn predictive models that can be easily represented by *latent* characteristics, e.g., the principal component analysis technique [13, 19] can be used to convert the original user-item bipartite network into another matrix with a much lower dimension, and thus greatly reduces the computation efforts for link prediction. Model-based methods in general have better performance than neighborhood-based methods as model-based methods incorporate the whole bipartite network into consideration instead of only using proximity (similarity) measures in neighborhood-based methods. However, the drawback of a model-based approach is that latent characteristics in general do not have clear physical meanings and it is difficult to see how a model-based approach can be further improved.

As on-line social networks become popular, the information of social interactions can also be recorded and used for personal recommendations. As such, constructing personal recommendation systems that take on-line social in-

teractions into consideration has been a hot research topic [3, 16, 17, 32, 29, 28, 35]. In addition to the user-item network that collects historical interactions between users and items, there is another social network (sometimes called trust network) that characterizes the interactions among users. The main advantage of using a social network is that *cold-start* users who have very limited interactions on items can be predicted by their trusted friends. As pointed out in [17], combining the prediction by a traditional collaborative filtering method and that by a trust-based method can lead to much better performance than using a single method alone.

Instead of using a combined approach, in this paper we ask the question whether prediction can be made *directly* from a social user-item network (that consists of a social network and a user-item bipartite network). As for neighborhood-based collaborative filtering methods, the key challenge is to identify and compute an appropriate proximity measure between two nodes in a social user-item network. There are many methods addressed in [23] for computing the proximity (similarity) measures for a pair of two nodes in a network, including methods based on node neighborhoods and methods based on the ensemble of all paths. However, these methods cannot be directly applied as now we have two graphs in a social user-item network. Since every node in a social user-item network can now have two kinds of neighbors, a user or an item, these have to be treated differently. In view of this, we propose a random walk method on a social user-item network for computing the proximity measure. Such a random walk is characterized by two parameters α and β that specifies the jumping probabilities to these two kinds of neighbors. These two parameters can then be chosen to optimize the performance of our recommendation system. To test our method, we conduct various experiments on the DBLP dataset [21] that collects years of papers published in various conferences. In this dataset, coauthors of papers are used for constructing the social network and the numbers of papers published by authors in conferences are used for constructing the user-item (author-conference) bipartite network. With a 3-5 year training period, our method performs significantly better than random guess in terms of minimizing the root mean squared error. Also, the value of the precision is close to 0.25 for our top-1 predictor, and the value of recall is close to 0.6 for our top-50 predictor.

The rest of this paper is organized as follows: In Section 2, we describe the formulation of the problem. We then propose the method for computing the proximity measure in a social user-item network via a random walk in Section 3. The experimental setup and results for the DBLP dataset are presented in Section 4. Finally, we conclude the paper in Section 5, where we discuss possible future research topics.

2 Problem Formulation

In this section, we describe the problem formulation of our link prediction problem. In the literature, a user-item network is in general referred to as a bipartite network with *users* as one side of nodes and *items* as the other side of nodes. In this paper, we consider a user-item network G_b with the set of users U=$\{u_1, u_2, \ldots, u_N\}$ and the set of items V=$\{v_1, v_2, \ldots, v_M\}$, where N is the number of users and M is the number of items. The bipartite network G_b is characterized by the $N \times M$ *weighted* biadjacency matrix $B = (B_{ij})$, where B_{ij} represents the edge weight between user u_i and item v_j. In practice, the edge weight could represent the price that a certain user buys a particular product, the rating that a user gives to a specific movie, or the number of papers that an author publishes in a conference. In addition to the user-item network, we also assume there is a social network G_s with the set of users U as its nodes. The social network G_s is characterized by the $N \times N$ adjacency matrix $A = (A_{i,j})$, where $A_{i,j} = 1$ if user u_i and user u_j are friends of each other, and 0 otherwise. A *social* user-item network G in this paper is then defined as the union of the user-item network G_b and the social network G_s, i.e, $G = G_b \cup G_s$.

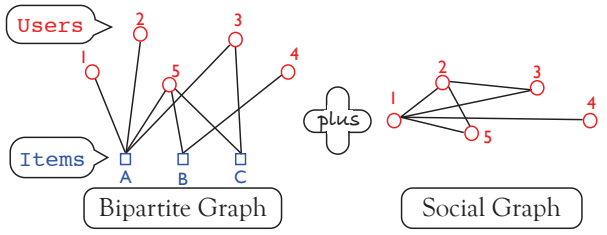

Figure 1: An example of a social user-item network with 5 users and 3 items.

To illustrate this, we show in Fig. 1 an example of a social user-item network with 5 users and 3 items. On the left hand side of Fig. 1, there is a user-item bipartite network of 5 users and 3 items. On the right hand side of Fig. 1, there is a social network of these 5 users. The adjacency matrix A and the biadjacency matrix B are as follows:

$$A = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix}, \qquad (1)$$

$$B = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix}. \quad (2)$$

As in [23], the link prediction problem of the social user-item network considered in this paper is to infer new interactions that are most likely to occur between a user and an item in the future given a snapshot of the social user-item network. Our approach for the link prediction problem is also based on measures for analyzing "proximity" of nodes in a network [23]. Specifically, we compute every pair of a user and an item a *score* based on their proximity (similarity) measure and then predict (or recommend) those with high scores.

There are many methods addressed in [23] for computing the proximity (similarity) measures for a pair of two nodes in a network, including methods based on node neighborhoods and methods based on the ensemble of all paths. However, these methods cannot be directly applied as now we have two graphs in a social user-item network. Specifically, every node in a social user-item network can now have two kinds of neighbors, a user or an item. As such, methods based on node neighborhoods such as the number of common neighbors and Jaccard's coefficient [15] are not applicable as they do not distinguish these two kinds of neighbors. On the other hand, methods based on the ensemble of all paths such as Katz's similarity [18] do not take these two kinds of nodes in the path into account. In view of all these, a good proximity measure for a social user-item network must treat these two kinds of nodes differently. Our approach for this is to use a random walk method that assigns different jumping probabilities when the random walker are in different kinds of nodes. Such a method will be described in details in the next section.

We note that the random walk method has been successfully used in many previous works for link prediction (see e.g., [7, 4, 16, 25, 29, 31]). There are several advantages of using the random walk method to compute the proximity measure in a social user-item network. These are summarized below:

(i) *Weighted paths*: A random walk method with different jumping probabilities could be easily used for assigning different weights to all the paths between two nodes.

(ii) *Sparsity*: Neighborhood-based methods generally require a node to have several neighboring nodes to achieve a good prediction. For nodes with few neighboring nodes, the random walk method [14, 34] is known to be able to alleviate the sparsity problem.

(iii) *Mathematical analysis and physical meanings*: Like PageRank [7], the proximity measures derived from random walks can be computed by solving the steady state probabilities of Markov chains. As such, the proximity measures by using random walks have clear physical meanings than other latent methods [27, 33].

3 Proximity Measure by Using a Random Walk

In this section, we propose using a random walk method to compute the proximity measure between two nodes in a social user-item network. Recall that a social user-item network consists of a social graph of users and a bipartite graph with users as one sides of nodes and items as the other side of nodes. We use the $N \times N$ matrix $A = (A_{i,j})$ for the adjacency matrix of the social network and the $N \times M$ matrix $B = (B_{i,j})$ for the biadjacency matrix of the user-item network. Now consider a random walker on a social user-item network. The jumping probabilities for the random walker are specified as follows:
(R1) The walker is on a user node:

- With probability α, the walker will walk to another ***user node***, which is selected *uniformly* among all its neighboring users.
- With probability $1-\alpha$, the walker will walk to another ***item node***, which is randomly selected among all its neighboring items with the probability proportional to the edge weight between that user and that item.

(R2) The walker is on an item node:

- With probability β, the walker will walk to a ***user node***, which is randomly selected among all its neighboring users with the probability proportional to the edge weight between that item and that user.
- With probability $1-\beta$, the walker will ***stop*** at this item node.

Clearly, if we let $X(t)$ be the node that the random walker visits at time t. Then $\{X(t), t \geq 0\}$ forms an absorbing Markov chain. Let $P_1(u, v)$ be the absorbing probability that a random walker is started from a user node u and is absorbed by an item node v. Also, let $P_2(v, v')$ be the absorbing probability that a random walker is started from an item node v and is absorbed by an item node v'. Intuitively, a large absorbing probability between a user node u and an item node v indicates that it is very likely for user u to reach item v. Thus, we use the absorbing probability $P_1(u, v)$ as the proximity measure between user u and item v. Similarly, we also use the absorbing probability $P_2(v, v')$ as the proximity measure between item v and item v'.

To compute the absorbing probability $P_1(u,v)$, we consider the following two events: (i) the first step of the random walker is toward a *user* node j and (ii) the first step of the random walker is toward an *item* node ℓ. According to Rule (R1), the probability for the first event is $\alpha \cdot \frac{A_{u,j}}{\sum_{s=1}^{N} A_{u,s}}$ and the probability for the second event is $(1-\alpha) \cdot \frac{B_{u,\ell}}{\sum_{s=1}^{M} B_{u,s}}$. If the first step is toward a user node j (resp. an item node ℓ), the probability that the random walker will be absorbed by the item node v is $P_1(j,v)$ (resp. $P_2(\ell,v)$). This then leads to

$$\begin{aligned} P_1(u,v) &= \alpha \cdot \sum_{j=1}^{N} \frac{A_{u,j}}{\sum_{s=1}^{N} A_{u,s}} P_1(j,v) \\ &\quad + (1-\alpha) \cdot \sum_{\ell=1}^{M} \frac{B_{u,\ell}}{\sum_{s=1}^{M} B_{u,s}} P_2(\ell,v). \quad (3) \end{aligned}$$

Similarly, we also have from Rule (R2) that

$$\begin{aligned} P_2(v,v') &= \beta \cdot \sum_{\ell=1}^{N} \frac{B_{\ell,v}}{\sum_{s=1}^{N} B_{s,v}} P_1(\ell,v') \\ &\quad + (1-\beta) \cdot \delta(v,v'), \quad (4) \end{aligned}$$

where $\delta(x,y)$ is the delta function that is equal to 1 if $x = y$ and 0 otherwise. Let

$$W^{U,U}(u,j) = \frac{A_{u,j}}{\sum_{s=1}^{N} A_{u,s}}, \quad (5)$$

$$W^{U,I}(u,\ell) = \frac{B_{u,\ell}}{\sum_{s=1}^{M} B_{u,s}}, \quad (6)$$

and

$$W^{I,U}(v,\ell) = \frac{B_{\ell,v}}{\sum_{s=1}^{N} B_{s,v}}. \quad (7)$$

Define the three matrices $W^{U,U} = (W^{U,U}(u,j))$, $W^{U,I} = (W^{U,I}(u,\ell))$, and $W^{I,U} = (W^{I,U}(v,\ell))$. By substituting (5), (6), and (7) into (3) and (4), we can write $P_1(u,v)$ and $P_2(v,v')$ in the following matrix forms:

$$\begin{aligned} P_1 &= \alpha \cdot W^{U,U} \cdot P_1 + (1-\alpha) \cdot W^{U,I} \cdot P_2, \quad (8) \\ P_2 &= \beta \cdot W^{I,U} \cdot P_1 + (1-\beta) \cdot I. \quad (9) \end{aligned}$$

By solving these two matrix equations, we can then obtain P_1 as follows:

$$\begin{aligned} P_1 &= (1-\alpha)(1-\beta) \cdot \\ &\quad \Big(I - \big((1-\alpha)\beta \cdot W^{U,I} \cdot W^{I,U} + \alpha \cdot W^{U,U}\big)\Big)^{-1} \cdot \\ &\quad W^{U,I}. \quad (10) \end{aligned}$$

This can then be used to obtain P_2 from (9).

Note that we have to calculate the inverse of an $N \times N$ matrix in (10). The computational complexity of inverting a matrix is $O(N^3)$ by using the Gauss-Seidel elimination. Since N is the number of users in a recommender system, its value could be very large and thus computing the inverse of such a matrix could be numerically challenging. Such a problem is commonly known as the proximity inversion problem [30]. As described in [30], one way to compute a good approximation for the inverse of a matrix $(I - \delta C)^{-1}$ with $\delta << 1$ and C being a sparse matrix is to use the (truncated) power series representation, i.e.,

$$(I - \delta C)^{-1} = \sum_{k=0}^{\infty} \delta^k C^k \approx \sum_{k=0}^{K_{\max}} \delta^k C^k,$$

where $K_{\max}$ is a constant that is large enough to guarantee a small error. Then one can exploit the sparsity of the matrix C to compute C^k. However, we will not explore this further in this paper. On the other hand, if we only would like to make personal recommendations for a specific user u, then we only need to compute $P_1(u,\cdot)$ and there is no need to compute the whole matrix P_1. In that regard, it might be more efficient to use simulation to estimate $P_1(u,\cdot)$. In this paper, we will simulate the random walks for 10000 times for each user node u and estimate $P_1(u,v)$ by the ratio of the number of times that the walk is absorbed by the item node v to 10000. The simulation algorithm is described below.

Algorithm 1 Estimating $P_1(u,v)$ by simulation

```
input: G = G_s ∪ G_b, (α, β)
for user u ← 1 to N do
    for TIMES ← 1 to 10000 do
        Record the absorbing item
    end for
    P_1(u,v) ← 1/10000 (the number of TIMES that user u
                        is absorbed by item v)
end for
```

4 Experiments

4.1 Experimental Setup

In this section, we conduct various experiments on the Digital Bibliography & Library Project (DBLP) dataset [21]. We first parse the dataset for publications of type "*in-proceedings*" from 1959 to 2011 (first half-year). During this period, there are 1,020,406 papers, 716,812 authors, and 4,773 conferences in this dataset. By filtering out the authors who published less than 10 papers over the past 10 years, the number of authors is reduced to 30,758.

Table 1: Number of papers published in the 4,773 conferences from 2003 to 2008

Year	2003	2004	2005	2006	2007	2008
Papers	48,348	62,408	72,271	80,741	86,564	91,520

Our objective for the link prediction problem is to use the historical data in the DBLP dataset to predict in which conference(s) a specific author will publish his/her paper(s). To do this, we choose year 2008 as the test year and the years from 2003 to 2008 as the training years (for generating the social user-item network). In TABLE 1, we show the number of papers published in the 4,773 conferences from 2003 to 2008. To generate the social user-item network from the data in the training period, we represent authors as user nodes and conferences as item nodes. To construct the social graph G_s, we add an edge between two authors if they coauthored at least one paper together. On the other hand, for the user-item (author-conference) network G_b, the edge weight between an author and a conference is the number of papers published by that author in that conference. These two graphs are processed in C by using the library of the igraph [8]. Once the social user-item network is generated, we then run Algorithm 1 to estimate the proximity measures $P_1(u, v)$ for the $N = 30,758$ authors and the $M = 4,773$ conferences. For the two parameters α and β, we run all the cases for α from 0 to 0.9 and β from 0.1 to 0.9, where the steps of both are set to 0.1. By doing so, we obtain 90 matrices for the proximity measures. Among these 90 matrices, we then compare their performance and identify the best choice of the two parameters α and β.

4.2 Root Mean Squared Error

In the literature, the Root Mean Squared Error (RMSE) [34] is a frequently used measure of the differences between the predicted values and the real values. Let $T(u, v)$ be the number of papers that author u published in conference v in the test year of 2008, and $T(u) = \sum_{v=1}^{M} T(u, v)$ be the number of papers that author u published in the test year of 2008. Since $P_1(u, v)$ is the absorbing probability of the random walk from user u to item v in the social user-item network, it can be viewed as the probability that author u will publish a paper in conference v in the future. As such, a reasonable estimate for the number of papers that author u published in conference v in the test year of 2008 is $\hat{T}(u, v) = P_1(u, v) \cdot T(u)$. Then, we compute the root mean squared error by using the following equation:

$$\mathbf{RMSE} = \sqrt{\sum_{u=1}^{N}\sum_{v=1}^{M} |\hat{T}(u, v) - T(u, v)|^2}, \qquad (11)$$

If our prediction is good, at least the root mean squared error in (11) should be much smaller than that from random guess. The root mean squared error from random guess is to assign each conference with an equal probability $1/M$, i.e.,

$$\mathbf{RMSE(Random)} = \sqrt{\sum_{u=1}^{N}\sum_{v=1}^{M} |\frac{1}{M}T(u) - T(u, v)|^2} = 396.36. \qquad (12)$$

In Fig. 2, we show the RMSE for various α's and β's by using year 2007 as the training set. The results are not good. For some values of α and β (when β is small), their RMSE is even worse than that from random guess. There are many reasons for this negatively correlated prediction: (i) an author who published a paper in a *prestigious* conference (with a low acceptance ratio) may not be able to publish another one in that conference in the following year, and (ii) an author who went to a boondoggle conference may not want to attend the same conference in the following year. In view of this, we extend the training period to three years. In Fig. 3, we show the RMSE for various α's and β's by using years 2005-2007 as the training set. As shown in Fig. 3, the RMSE for various α's and β's are significantly lower than that from random guess. The improvement is intuitive as now we use more information for our prediction. We then further extend the training period to five years. In Fig. 4, we show the RMSE for various α's and β's by using years 2003-2007 as the training set. The RMSE in Fig. 4 for various α's and β's are comparable to their counterparts in Fig. 3. This shows that additional information may not lead to further improvement. In fact, we also evaluate our link prediction method for various training periods of more than five years and their RMSEs could be worse when the length of the training period is too long. It seems that using 3-5 years for the training period is relatively good for this DBLP dataset. One possible explanation for this is that researchers might change their research interest every 3-5 years and thus change their home conferences.

A generalization of our method is to assign different weights to different years. However, it is in general very difficult to find appropriate weight assignments. By conducting various experiments with different weight assignments, we find that the improvement for RMSE is quite minor. In Fig. 5, we show the RMSE for various α's and β's by setting the weights for the years 2003-2007 to be $2, 5, 4, 10, 8$, respectively. As shown in Fig. 5, the improvement of RMSE is only 0.7% when compared with the assignment with equal weights, i.e., $1, 1, 1, 1, 1$. All these suggest that using a window of 3-5 years might be good enough for the DBLP dataset. Also, choosing $\alpha \in [0.2, 0.4]$ and $\beta \in [0.3, 0.6]$ leads to a low RMSE in Fig. 3 and Fig. 4.

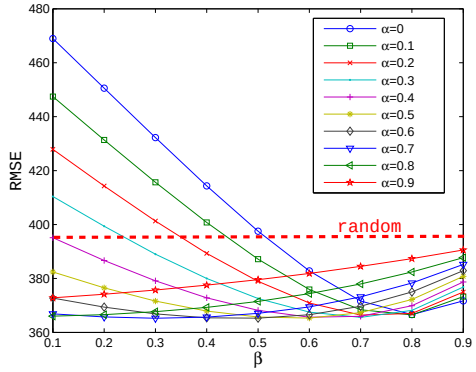

Figure 2: RMSE for various α's and β's by using year 2007 as the training set.

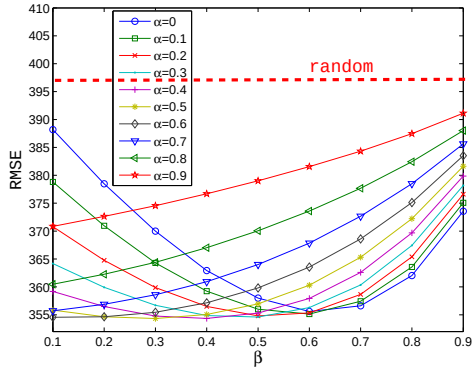

Figure 3: RMSE for various α's and β's by using years 2005-2007 as the training set.

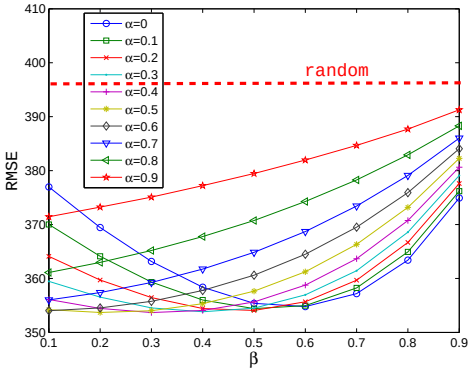

Figure 4: RMSE for various α's and β's by using years 2003-2007 as the training set.

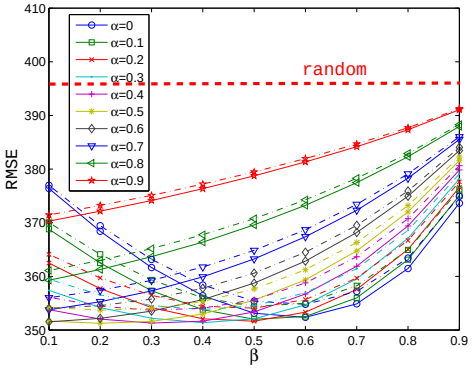

Figure 5: RMSE for various α's and β's by using different weights for years 2003-2007 as the training set: (i) the solid lines for the weights $2, 5, 4, 10, 8$ and (ii) the dash lines for equal weights.

4.3 Precision and Recall

In addition to RMSE, precision and recall are commonly used to evaluate the top-n predictor that predicts the n most likely conferences for a specific author to publish his/her papers in the test year. For the top-n predictor, we rank the conferences for each author u according to the proximity measure $P_1(u, v)$ and recommend the n largest ones. Also, precision and recall for the top-n predictor are defined as follows:

$$precision(n) = \frac{\#\text{ matches of top–n}}{\text{n}}, \qquad (13)$$

and

$$recall(n) = \frac{\#\text{ matches of top–n}}{\text{n}'}, \qquad (14)$$

where n' is the number of conferences in which a specific author published a paper in the test year of 2008. For example, if we use the top-10 predictor and there are 3 conferences that match the result, the precision is $3/10 = 0.3$. However, if the author actually published in 6 conferences in this test year, then the recall in this situation is $3/6 = 0.5$. Once the precision and the recall for each user is computed, the precision and the recall is then computed by averaging over the precision and the recall for each user.

In our experiments, we evaluate the precision and recall of the top-n predictor for $n = 1, 2, \ldots, 50$. For these experiments, we set $\alpha = 0.2$ and $\beta = 0.4$. In Fig. 6 and Fig. 7, we show the results for precision and recall as a function of n, respectively. There are four curves in these two figures. The one marked with "2007" (resp. "3 years", and "5 years") is generated by using year 2007 (resp. years 2005-2007, and years 2003-2007) for the training period. The one marked with "5 years with weights" is generated with

the weight assignment 2,5,4,10,8 for the five years 2003-2007, respectively. As expected, all the curves for precision in Fig. 6 is decreasing in n, while all the curves for recall in Fig. 7 is increasing in n. As the results for RMSE, there is a significant improvement of precision and recall by using 3-5 years for the training period in comparison with that by using year 2007 alone. Also, the three curves of precision and recall marked with "3 years", "5 years", and "5 years with weights" are almost identical. In particular, the value of the precision is close to 0.25 for the top-1 predictors, and the value of recall is close to 0.6 for the top-50 predictors. These also suggest that using a window of 3-5 years for the training period is good enough for the DBLP dataset.

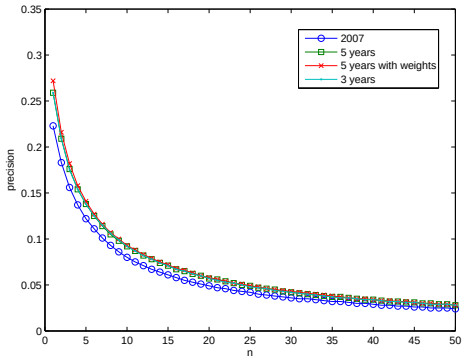

Figure 6: Precision of top-n predictors.

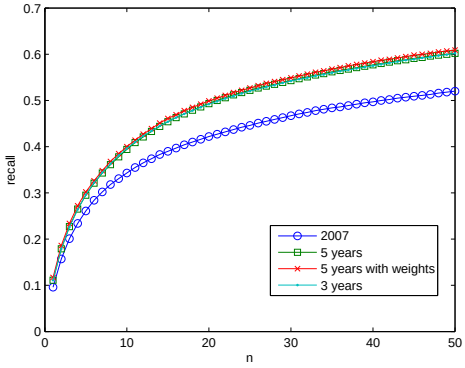

Figure 7: Recall of top-n predictors.

To further understand the effect of the two parameters α and β, we evaluate the precision and recall for various top-n predictors by using years 2003-2007 as the training period. In Fig. 8, we show the best choices of the two parameters α and β for these top-n predictors. The range of each bar in this figure indicates the corresponding value of α or β. For example, choosing $\alpha = 0.5$ and $\beta = 0.4$ leads to the best result for the top-5 predictor. One interesting finding is that one should choose large (resp. small) α and β for small (resp. large) n.

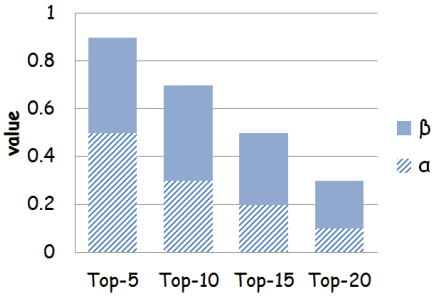

Figure 8: The best choices of the two parameters α and β for various top-n predictors.

5 Conclusion

In this paper, we consider the link prediction problem for social user-item networks. In order to apply the neighborhood-based collaborative filtering methods, one has to identify and compute an appropriate proximity measure between two nodes in such a network. The main contribution of this work is to propose a random walk method that uses *two* different jumping probabilities to compute the proximity measure of a social user-item network. Unlike the simple random walk, our method is able to assign different weights to different paths and thus can lead to a better proximity measure by optimizing the two jumping probabilities. We test our method by using the DBLP dataset and our experimental results show significant improvement in comparison with random guess. In addition to link prediction, we believe our proximity measure can also be applied for computing centralities of nodes (that can be used for ranking nodes in a social user-item network). Another possible extension of our work is to use the proximity measure for community detection in a social user-item network.

References

[1] C. C. Aggarwal. *An introduction to social network data analytics*. Springer, 2011.

[2] M. Al Hasan and M. J. Zaki. A survey of link prediction in social networks. In *Social network data analytics*, pages 243–275. Springer, 2011.

[3] R. Andersen, C. Borgs, J. Chayes, U. Feige, A. Flaxman, A. Kalai, V. Mirrokni, and M. Tennenholtz. Trust-based recommendation systems: an axiomatic approach. In *Proceedings of the 17th international conference on World Wide Web*, pages 199–208. ACM, 2008.

[4] S. Baluja, R. Seth, D. Sivakumar, Y. Jing, J. Yagnik, S. Kumar, D. Ravichandran, and M. Aly. Video suggestion and discovery for youtube: taking random walks through the view graph. In *Proceedings of the 17th international conference on World Wide Web*, pages 895–904. ACM, 2008.

[5] T. Bogers. Movie recommendation using random walks over the contextual graph. In *Proc. of the 2nd Intl. Workshop on Context-Aware Recommender Systems*, 2010.

[6] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, pages 43–52. Morgan Kaufmann Publishers Inc., 1998.

[7] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, 30(1):107–117, 1998.

[8] G. Csardi and T. Nepusz. The igraph software package for complex network research. *InterJournal*, Complex Systems:1695, 2006.

[9] J. Davidson, B. Liebald, J. Liu, P. Nandy, T. Van Vleet, U. Gargi, S. Gupta, Y. He, M. Lambert, B. Livingston, et al. The youtube video recommendation system. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 293–296. ACM, 2010.

[10] C. Desrosiers and G. Karypis. A comprehensive survey of neighborhood-based recommendation methods. In *Recommender systems handbook*, pages 107–144. Springer, 2011.

[11] D. M. Dunlavy, T. G. Kolda, and E. Acar. Temporal link prediction using matrix and tensor factorizations. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 5(2):10, 2011.

[12] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, 1992.

[13] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, 4(2):133–151, 2001.

[14] Z. Huang, D. Zeng, and H. Chen. A link analysis approach to recommendation under sparse data. In *Proc. 2004 Americas Conf. Information Systems*, 2004.

[15] P. Jaccard. *Etude comparative de la distribution florale dans une portion des Alpes et du Jura*. Impr. Corbaz, 1901.

[16] M. Jamali and M. Ester. Trustwalker: a random walk model for combining trust-based and item-based recommendation. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 397–406. ACM, 2009.

[17] M. Jamali and M. Ester. Using a trust network to improve top-n recommendation. In *Proceedings of the third ACM conference on Recommender systems*, pages 181–188. ACM, 2009.

[18] L. Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, 1953.

[19] Y. Koren. The bellkor solution to the netflix grand prize. *Netflix prize documentation*, 2009.

[20] Y. Koren and R. Bell. Advances in collaborative filtering. In *Recommender Systems Handbook*, pages 145–186. Springer, 2011.

[21] M. Ley. Dblp: some lessons learned. *Proceedings of the VLDB Endowment*, 2(2):1493–1500, 2009.

[22] M. Li, B. M. Dias, I. Jarman, W. El-Deredy, and P. J. Lisboa. Grocery shopping recommendations based on basket-sensitive random walk. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1215–1224. ACM, 2009.

[23] 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.

[24] G. Linden, B. Smith, and J. York. Amazon. com recommendations: Item-to-item collaborative filtering. *Internet Computing, IEEE*, 7(1):76–80, 2003.

[25] W. Liu and L. Lü. Link prediction based on local random walk. *EPL (Europhysics Letters)*, 89(5):58007, 2010.

[26] H. Ma, H. Yang, M. R. Lyu, and I. King. Sorec: social recommendation using probabilistic matrix factorization. In *Proceedings of the 17th ACM conference on Information and knowledge management*, pages 931–940. ACM, 2008.

[27] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. King. Recommender systems with social regularization. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 287–296. ACM, 2011.

[28] D. O'Doherty, S. Jouili, and P. Van Roy. Towards trust inference from bipartite social networks. In *Proceedings of the 2nd ACM SIGMOD Workshop on Databases and Social Networks*, pages 13–18. ACM, 2012.

[29] S. Shang, S. R. Kulkarni, P. W. Cuff, and P. Hui. A random walk based model incorporating social information for recommendations. *CoRR*, abs/1208.0787, 2012.

[30] H. H. Song, T. W. Cho, V. Dave, Y. Zhang, and L. Qiu. Scalable proximity estimation and link prediction in online social networks. In *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*, pages 322–335. ACM, 2009.

[31] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. 2011.

[32] P. Symeonidis, E. Tiakas, and Y. Manolopoulos. Product recommendation and rating prediction based on multi-modal social networks. In *Proceedings of the fifth ACM conference on Recommender systems*, pages 61–68. ACM, 2011.

[33] X. Yang, Y. Guo, Y. Liu, and H. Steck. A survey of collaborative filtering based social recommender systems. *Computer Communications*, 41:1–10, 2014.

[34] H. Yildirim and M. S. Krishnamoorthy. A random walk method for alleviating the sparsity problem in collaborative filtering. In *Proceedings of the 2008 ACM conference on Recommender systems*, pages 131–138. ACM, 2008.

[35] X. Yu, X. Ren, Y. Sun, B. Sturt, U. Khandelwal, Q. Gu, B. Norick, and J. Han. Recommendation in heterogeneous information networks with implicit user feedback. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 347–350. ACM, 2013.