Semantic-based Recommendation across Heterogeneous Domains

Deqing Yang*, Yanghua Xiao*, Yangqiu Song† and Wei Wang*

* School of Computer Science, Shanghai Key Laboratory of Data Science
† Fudan University, Shanghai, China P. R., 200433
Email: {yangdeqing, shawyh, weiwang}@fudan.edu.cn
†Department of Computer Science, UIUC, IL, USA
Email: yqsong@gmail.com

Abstract—Cross-domain recommendation has attracted wide research interest which generally aims at improving the recommendation performance by alleviating the cold start problem in collaborative filtering based recommendation or generating a more comprehensive user profiles from multiple domains. In most previous cross-domain recommendation settings, explicit or implicit relationships can be easily established across different domains. However, many real applications belong to a more challenging setting: recommendation across heterogeneous domains without explicit relationships, where neither explicit user-item relations nor overlapping features exist between different domains. In this new setting, we need to (1) enrich the sparse data to characterize users or items and (2) bridge the gap caused by the heterogeneous features in different domains. To overcome the first challenge, we proposed an optimized local tag propagation algorithm to generate descriptive tags for user profiling. For the second challenge, we proposed a semantic relatedness metric by mapping the heterogenous features onto their concept space derived from online encyclopedias. We conducted extensive experiments on two real datasets to justify the effectiveness of our solution.

Keywords—cross-domain recommendation, heterogeneous domains, semantic matching, user profiling

I. INTRODUCTION

Social media, such as Twitter and Facebook, have experienced fast growth in the last decade. One of particular research interests on social media is recommender system including friend recommendation, product recommendation to content recommendation [1], [2]. Most existing recommender systems focus on homogeneous settings [2], [3], that is, users and items come from either a single website or two websites in the same domain. More recently, cross-domain recommendation [4], [5] has attracted increasing research interests, where users and items come from two or more different domains. One important goal of these techniques is alleviating the cold start problem so that the matching between users and items across different domains is satisfactory in the beginning [6], [7], [8].

Many previous solutions for cross-domain recommendations are not feasible to an extreme heterogenous cross-domain setting where neither explicit relationships exist between users and items in different domains nor such relationships can be trivially established. For simplicity, we refer to this setting as RAHD-ER (recommendation across heterogeneous domains without explicit relationships). We illustrate the difference between RAHD-ER and traditional cross-domain recommendation setting in Fig. 1. Most of previous solutions mainly focused on two typical heterogeneous settings. In the first setting, either users or items are shared in both domains, as shown in Fig. 1 (a) and (b). The shared users or items play a role as bridge for the recommendation [9], [10]. In the second setting shown in Fig. 1 (c), both users and items have no overlap but the relations between users and items can be easily derived due to their shared features, such as the same tag space [4] or similar user-item interactive patterns [5]. All these solutions obviously can not be directly used to solve the problem setting as shown in Fig. 1 (d), where domain A only contains users and domain B only contains items and hence no shared features can be used to establish the link between users in domain A and items in domain B.

However, RAHD-ER is a popular and valuable setting in real world. For example, it will create a lot of business opportunities if we can accurately recommend products in eBay to a user in Twitter or Facebook. Thus, in this paper we focus on how to achieve RAHD-ER which is a more challenging setting than previous recommendation settings. There are two main obstacles.

1. Heterogeneous Features: It is difficult to build user-item relations across domains due to the heterogeneous features. For example, we may want to recommend Douban (a Chinese movie review website) movies to Weibo (the largest Chinese Twitter) users. However, the users or items in these domains are described by completely different tags (features). We found that Weibo tags generally characterize a user’s hobby, age, constellation and etc. In contrast, Douban tags are generally used to specify a movie’s production nation, year and category. It is very hard to lexically match these two groups of tags. To overcome this challenge, we propose a semantic matching approach based the concepts extracted from online encyclopedias.

2. Data Sparsity: The data to accurately profile users or items are usually sparse, especially for users due to their privacy concern. For example, nearly 45% of Weibo users do not publish any tag to introduce themselves. Data sparsity makes it difficult to build user-item relationships, which are usually used as a prerequisite in the previous cross-domain recommendation solutions. To overcome the data sparsity problem, we propose a tag propagation based algorithm to generate a set of personalized and informative user tags. In summary, the main contributions of our paper include:

This paper was supported by NSFC (No.61472085, 61171132, 61033010), by National Key Basic Research Program of China under No.2015CB358800, by Basic research project of Shanghai science and technology innovation action plan under No.15JC1400900, and by Shanghai Science and Technology Development Funds (13dz2260200, 13511504300). Corresponding author is Yanghua Xiao.
I. We propose a tag propagation based algorithm to overcome data sparsity of user profiling in social networks. Furthermore, we optimized our algorithm to improve its performance.

II. PROFILING USERS BY LOCAL TAG PROPAGATION

In this section, we introduce our algorithm of tag-based user profiling since it is the prerequisite of recommendation. In brief, it generates a user’s profile description by the tags of the similar users in the same social network according to homophily [11].

A. Preliminaries

Our solution relies on an influence graph which is constructed against the users of social media. An influence graph $G(V, E, w)$ is an edge-weighted directed graph, where $V$ is vertex (user) set and $E$ is influence edge set. Each directed edge $e_{uv} \rightarrow v$ indicates the social influence from user $u$ to user $v$. Furthermore, we assign a weight $w_{uv}$ to this edge to quantifies the extent to which $u$ can influence $v$ through it.

In this paper, we use the social relationships between Weibo users to construct the influence graph. The most fundamental social relationship in Weibo is following relationship. A Weibo user $u$ is $v$’s followee if $v$ follows $u$, or equivalently, $v$ is $u$’s follower. For better interpreting our algorithm, we assume that only followee can influence his/her followers resulting in tag propagation from followees to followers. Specifically, if and only if user $u$ follows $v$, there is an edge $e_{uv} \rightarrow v$ in the influence graph. We further set $w_{uv}$ as the frequency that $v$ retweets $u$ in a given period.

In formal, given $N$ candidate tags, our tag-based profiling algorithm for user $u$ aims to generate a tag score vector $\vec{R}_u = [r_1, ..., r_N] \in \mathbb{R}^N$ in which the score $r_j = \sum_{i = 1}^{N} e_{ij} w_{ij}$ quantifies the extent to which tag $j$ can describe $u$, i.e., the tendency of $u$ to tag $j$. Note that there are only 55% of Weibo users using tags to label themselves through our statistics, we call such original tags of a user as real tag in the following presentation. Our algorithm can generate tags for the users no matter whether they have real tags or not.

Homophily [11] is a frequently observed phenomenon in social media. It refers to the tendency for users to issue social ties with other similar users. In other words, users in a local social circle tend to share similar profiles. Thus, the tags most used by a user’s friends can also well characterize this user’s preference. Based on this observation, we use the tags of local neighborhoods to profile a specific user. In general, there are two options to specify the neighborhoods: 1) direct neighborhoods and 2) indirect neighborhoods. In the first option, we only consider the tags of the direct neighbors of a user in the influence graph. In the second option, we extend to consider the tags of indirect neighbors.

B. Direct Influence Algorithm

Given a user $u$ in $G$ with $|V| = M$, we first define $u$’s influence weight vector as $\vec{F}_u = [f_{u1}, ..., f_{uM}] \in \mathbb{R}^M$ where each $f_{ui}$ $(1 \leq i \leq M)$ characterizes the influence of user $i$ on $u$ in terms of tag propagation. $f_{ui}$ is set as

$$f_{ui} = \frac{w_{ui}}{\sum_{v \in u \rightarrow u} w_{vu}}$$  \hspace{1cm} (1)

where $v$ is $u$’s in-neighbor in $G$, i.e., $u$ follows $v$. According to aforementioned definition of influence weight, $f_{ui} = 0$ if $i$ is not $u$’s in-neighbor. Moreover, we define $f_{uu} = 1$ if $u$ has real tags, otherwise $f_{uu} = 0$. Accordingly, user $i$ can influence $u$ more if $u$ has retweeted $i$ more frequently than other out-neighbors of $i$. Suppose there are overall $N$ tags used by the users in $G$. The tag distribution vector of user $i$ is defined as $\vec{t}_i = [t_{i1}, ..., t_{iN}] \in \mathbb{R}^N$, where $t_j = 1/n(1 \leq j \leq N)$ if user $i$ uses tag $j$, otherwise $t_j = 0$. $n$ is the number of tags used by user $i$ and $\sum t_j = 1$. Thus, all users’ tag distribution vectors constitute an $M \times N$ matrix $\vec{T}$. According to the principle of homophily, the tags adopted by intimate friends are deserved to be selected as the profiling tags. Thus, $u$’s tag score vector can be set as

$$\vec{R}_u = \vec{F}_u \vec{T}$$  \hspace{1cm} (2)

Many general tags such as ‘movie’, ‘music’ and ‘tour’ are frequently used in Weibo. To suppress these tags, we import statistics, we call such original tags of a user as real tag in the following presentation. Our algorithm can generate tags for the users no matter whether they have real tags or not.

C. Refinement by Tag Propagation

Next, we refine above basic algorithm by considering the tags of indirect neighbors in the influence graph. There are two reasons for this refinement. First, in some cases, the direct neighbors may have rare tags either. The direct influence algorithm certainly will fail on these cases. Second, it is desired to increase the candidate tag sizes so that the tags can accurately reflect the collective preferences of a user’s neighborhoods. In general, the more tags sampled from a user’s neighborhoods, the more accurate the user profiling is. According to this basic idea, we propose a tag propagation algorithm to collect the tags from remote neighbors more than direct neighbors. This algorithm is an iterative algorithm running on the influence graph. In each iteration, each vertex (user) collects the tags of its direct neighbors to update its own tags and the corresponding tag scores. Accordingly, the tags of the indirect neighbors will be propagated to the objective vertex. The more iterations are computed, the more related tags will take effect. As the classic label propagation algorithm (LPA in short) [12], such tag propagation will converge to a stable state.

Let $\vec{R}$ be a tag score matrix, which is an $M \times N$ matrix and each row is the tag score vector of a user. Let $\vec{F}$ be an $M \times M$ influence matrix in which each entry $F_{uv}$ quantifies the influence of $v$ on $u$ in terms of tag propagation, i.e., $F_{uv}$ defined in Eq. (1). Based on Eq. (2), the computation of $\vec{R}$ after the $t$-th round can be evaluated as

$$\vec{R}^t = \vec{F} \vec{R}^{t-1}$$  \hspace{1cm} (3)

Then, let $\vec{R}$ be the $\vec{R}$ matrix after convergence, we finally return $\vec{R}^* \vec{D}$ as the result, where $\vec{D}$ is a diagonal matrix with each entry $d_i$ representing the IDF score of the tag $i$. Furthermore, we define $\vec{R}^0 = \vec{T}$, thus we have $\vec{R}^t = \vec{F} \vec{T}$ according to Eq. (2). We set $\vec{F}$’s diagonal elements, i.e., $F_{uu} = 1$ due to the intuition of tag propagation: each user $u$ tends to preserve his/her real tags although s/he may be influenced by his/her neighbors to accept their tags. Note that our algorithm is different to LPA in which only the nodes without labels will adopt new labels.

D. Reducing Computation Cost

Next, we first analyze the computation cost of Eq. (3) and then propose an optimization technique to reduce the computation cost.
1) Computation Cost Analysis: Suppose there are \( L \) users in \( G \) having real tags before tag propagation procedure. Then, we can rearrange these users’ tag distribution vectors as the top-\( L \) rows in \( T \), denoted by \( \bar{T}_i(1 \leq i \leq L) \). According to the following deduction

\[ R^t = FR^{t-1} = F : FR^{t-2} = \ldots = F^tR^0 = F^tT \] \hspace{1cm} (4)

we have the tag score vector of \( u \) after \( t \) rounds as

\[ \tilde{R}_t^u = \sum_{v=1}^{L} F^u \bar{T}_v \] \hspace{1cm} (5)

where \( F^t_{ij} \) (\( 1 \leq i, j \leq M \)) represents an entry of matrix \( F^t \). Obviously, the computation cost of \( F^t \) is \( O(LtM^3) \) which is unaffordable on a large influence graph. Next, we introduce how to reduce its computation cost.

2) Social Influence: We first introduce an equivalent expression for the entries of \( F^t \), i.e., \( F^t_{uv} \). The new expression can simplify the computation of Eq. (5). We first formalize the definition social influence in the context of tag propagation. It is similar to the influence concept proposed by Mashiach et al. for optimizing PageRank algorithm [13]. Given two nodes \( u \) and \( v \), \( v \)'s social influence on \( u \) is realized along the different paths from \( v \) to \( u \) in the graph. More formally, we give the definition as follows.

**Definition 1** [Social Influence] For a directed path \( p = (u_0, u_1, \ldots, u_t) \) in \( G \), we define the social influence along \( p \) from \( u_0 \) to \( u_t \) as

\[ si(p) = \prod_{i=0}^{t-1} w_{u_i,u_{i+1}} \sum_{u_{i+1}} w_{u_i,u_{i+1}} \] \hspace{1cm} (6)

where \( u \) is \( u_{i+1} \)'s in-neighbor in \( G \). Let \( P_i(v, u) \) be the set of all paths of length \( t \) from \( v \) to \( u \), thus the social influence of \( v \) on \( u \) at radius \( t \) is

\[ si_t(v, u) = \sum_{p \in P_i(v, u)} si(p). \] \hspace{1cm} (7)

Furthermore, we define \( si_0(v, u) = 1 = f_{uu} \) and for all \( v \neq u \) we set \( si_0(v, u) = 0 \). Then, the total social influence of \( v \) on \( u \) is \( si(v, u) = \sum_{t=0}^{\infty} si_t(v, u) \).

Based on the above definitions, we can establish the following theorem whose proof is omitted for space limitation.

**Theorem 1** For all \( u, v \) and \( t \), we have \( si_t(v, u) = F^t_{uv} \).

This theorem helps to transfer the computation of \( F^t_{uv} \) into computing social influence from \( v \) to \( u \) at radius \( t \). Thus,

\[ \tilde{R}_t^u = \sum_{v=1}^{L} si_t(v, u) \bar{T}_v \] \hspace{1cm} (8)

3) Optimization by Pruning: Based on above transformation through social influences, we further introduce some pruning strategies to decrease the overhead of our algorithm. Refer to Eq. (6) and Eq. (7), we can recursively compute the social influence of user \( v \) on \( u \) at radius \( t \) as

\[ si_t(v, u) = \sum_{x \in v \text{’s out-neighbor}} w_{v,x} \sum_{y \in u \text{’s out-neighbor}} w_{u,y} si_{t-1}(x, u) = \sum_{y \in u \text{’s out-neighbor}} f_{xy}si_{t-1}(x, u) \] \hspace{1cm} (9)

where \( x \) is \( v \)'s out-neighbor who has a path of \( t-1 \) length to \( u \) at least, and \( v \) is \( x \)'s in-neighbor. That is, the social influence of \( v \) on \( u \) at radius \( t \) equals to the weighted average influence of \( v \)'s out-neighbors on \( u \) at radius \( t-1 \). According to Eq. (8) and Eq. (9), \( \tilde{R}_t^u \) can be computed by the steps described in Alg. 1. Obviously, the computation cost of our algorithm is unaffordable if \( t \) is too large and there are too many users and words in \( G \). In order to further reduce the algorithm’s overhead, we introduce two pruning strategies to restrict \( t \) as well as the number of \( u \) and \( x \), respectively.

**Algorithm 1** Computing \( u \)'s tag score vector after \( t \) rounds.

**Input:** \( u, t \).

**Output:** \( \tilde{R}_t^u \).

1: \( \tilde{R}_0^u \leftarrow \emptyset \);
2: \( \text{layer}_0 \leftarrow u, \); 3: \( si(u, u) \leftarrow 1 \);
4: if \( u \) has real tags then \( \text{layer}_1 \leftarrow u \);
5: \( \tilde{R}_1^u \leftarrow \bar{T}_u \);
6: end if
7: for \( i=1 \) to \( t \) do
8: \( \text{layer}_i \leftarrow \{ \text{all in-neighbors of the nodes in layer}_{i-1} \} \);
9: for all \( v \in \text{layer}_i \) do
10: if \( v \) has real tags then
11: for each \( v \)'s out-neighbor \( x \) do
12: \( si_t(v, u) \leftarrow \sum_{x \in v \text{’s out-neighbor}} \sum_{y \in u \text{’s out-neighbor}} w_{v,x} si_{t-1}(x, u) \);
13: end for
14: \( \tilde{R}_t^u \leftarrow \tilde{R}_t^u + si_t(v, u) \bar{T}_v \);
15: end if
16: end for
17: end for
18: return \( \tilde{R}_t^u \);

As we declared before, the tags coming from heterogeneous domains are often lexically different, demanding to match two tag sets on semantic level rather than lexical level. In this section, we present how to match profile tags semantically by using millions of concepts in online encyclopedias.

A. Basic Idea

Representing the semantics of a tag is the basis of semantic matching of two tag sets. Most prior works resorted to well defined lexical resources such as WordNet [16]. However, lexical resources require lexicographic expertise and only cover a small fraction of language lexicons. Considering the substantial semantic information of online encyclopedias, some scholars used Wikipedia to enrich the semantic representation of words [17], [18], [19]. Since most tags of Weibo users are Chinese terms, we focus the Chinese counterparts of Wikipedia for our solution, i.e., Baidu encyclopedias (http://baike.baidu.com).

As Wikipedia, for each concept in Baidu, there exists an article page that describes the fact about the concept. In each article page, there are many hyperlinks referring to other concepts, namely reference concepts. A concept can be referenced to by many articles of other concepts. Meanwhile,
a reference concept can be mentioned more than once in an article. According to the intuition of Explicit Semantic Analysis (ESA for short) [17], two concepts are considered semantically related if they co-occur in one article page of the encyclopedia. The more such article pages can be found, the more semantically related the two concepts are.

**B. Semantic Interpretation**

In order to represent the semantics of a tag through the contexts provided by the encyclopedia, we should first map a tag into an article entry (i.e., a concept) in Baike. Given a tag $a$ and a concept $i$, we map $a$ into $i$ if $a$ and $i$ are exactly same or $i$’s string is a maximal substring of $a$’s string. Under this mapping scheme, we can find a mapped Baike concept for 88.7% of Weibo tags. Then, the semantics of tag $a$ in terms of user/item profiling can be interpreted by a concept vector. In formal, tag $a$’s concept vector is defined as $\vec{C}_a = [c_1, \ldots, c_C] \in \mathbb{R}^C$ where $C$ is the total number of concepts in the encyclopedia and each $c_j (1 \leq j \leq C)$ represents the semantic relevance of concept $j$ to concept $i$. We define $c_j = s_j(i) \times r_a$, where $s_j(i)$ is the TF-IDF score of concept $j$’s occurrence in $i$’s article and $r_a$ is $a$’s entry in the tag score vector which quantifies the extent that tag $a$ can profile a user or an item. For interpreting the semantics of a tag set, we use the sum of all tags’ concept vectors in the set. Accordingly, we transfer matching a user with an item into computing the similarity between the concept vectors of their tag sets.

**C. Similarity Metric**

Next, we propose a metric to compute the similarity of two tag sets’ concept vectors. By using the links between Wikipedia articles, Normalized Google Distance [20] has been proved as an effective measure on the semantic relatedness between two terms. However, it neglects the weight of each link. In our scenario, each tag in the profile set has different weight, i.e., the tag score computed by LTPA. Thereby, we propose a finer-grained metric, Weighted Normal Google Distance (WNGD for short), to achieve more accuracy on measuring the semantic relatedness of two tag sets. Finally, we asked each volunteer whether to accept the recommended movies or not. We took their average acceptance rates as the performance results of our approaches.

**2. Diabetes Datasets.** Considering that the human assessments of Weibo-Douban recommendation is not very convincing as real facts, we further demonstrated another recommendation against another dataset. Among various social network websites dedicated to diabetes patients, some of them might be open to all types of diabetes patients, such as http://www.tudiabetes.org, namely Diabetes1, which is for patients of Type I, Type II, and prediabetes. And some of them might focus on a specific type of diabetes patients, such as https://diabetessisters.org, namely Diabetes2, which is for female diabetes patients, especially for those with gestational diabetes. Due to the different vocabulary used by these two websites, we can view recommending recommending discussion threads (posts) from Diabetes2 to the users in Diabetes1 as a RAHD-ER problem. We randomly collected 5,000 users from Diabetes1. Each of them is profiled by a set of keywords extracted from his/her historical posts on this website. From Diabetes2, we collected 2,790 posts (threads), each of which is also characterized by a set of keywords. We used Diabetes datasets to justify the effectiveness of our WNGD-based semantic matching for RAHD-ER problem. To generate the ground truth of a diabetes patients preferred posts, we first generated a representative vector for each Diabetes1’s user and Diabetes2’s post. Given a Diabetes1’s user, we collected all his/her published posts as a corpus from which a set of keywords were extracted. Each entry of the user vector is the TF-IDF score of one keyword. So does a Diabetes2s post vector. Then, we ranked all posts according to the cosine similarity of the user vector and the post vectors. Finally, we assume that the user will only accept top 30% of the posts whose cosine similarity is greater than 0. These posts were considered as recommendation ground truth of the user. Since all Diabetes keywords are English words, we used Wikipedia’s concepts to compute the semantic similarity of keywords.

**IV. EXPERIMENTS**

**A. Experimental Setup**

1) **Datasets:** We first introduce the settings of our experiments. The relevant datasets and codes can be obtained from http://gdm.fudan.edu.cn/GDMWiki.

**1. Weibo-Douban Datasets.** As we declared before, recommending Douban movies to Weibo users is a typical RAHD-ER problem. Hence we demonstrate the effectiveness of our approach against the dataset fetched from Weibo and Douban website. We first randomly selected 300 users as seeds, and then crawled the seeds’ followers and the followees of seeds’ followees (FoF) before Sept. 2013. These users (more than 2M) constitute the vertices of the Weibo influence graph $G$ and all following relationships (more than 13M) constitute the graph’s edges. From Douban website, we collected 4,028 movies whose average rating scores are above 7.5 (10 is the best) and review numbers are greater than 2000 before Jan. 2012. For each movie, we also captured each tag’s occurrence frequency to this movie. Intuitively, we also normalized each tag frequency to constitute its tag score vector for the movie, which was used to generate concept vector for semantic matching. Since we can not directly fetch any information about Weibo users’ review or rating on Douban movies, we had to adopt human assessments to collect the ground truth of our recommendation. From the seed users in our Weibo dataset, we randomly selected 50 volunteers for whom we generated profile tags by LTPA algorithm and then recommended to them with some Douban movies through semantic matching of tag sets. Finally, we asked each volunteer whether to accept the recommended movies or not. We took their average acceptance rates as the performance results of our approaches.

**2. Baselines:**

1. **POP.** It is a classic recommendation baseline, i.e., recommending the popular items in random. For Douban movie recommendation, the movies having more than 15,000 reviews and more than 8.8 rating scores were recommended randomly. For Diabetes post recommendation, we randomly recommended the posts from those having more than 2 keywords shared in the keyword group of all users.

The next two baselines are used to compare user profiling, i.e., generating tag score vector, which were compared to
2. **TWEET.** Content-based profiling [21] is an effective recommendation mechanism where the tags/keywords specifying a user's preference are generally extracted from the user's posts on a Web site. To generate content-based profile for a Weibo user, we extracted top-k keywords from a user's recent 300 tweets based on TF-IDF scheme. The user's profile vector's entries are TF-IDF weights of the keywords.

3. **REALTAG.** For those Weibo users having real tags, we can directly use their real tags to profile them. Since Weibo users do not specify any preference on each real tag, we set the score of each real tag to 1/n where n is the number of real tags.

The last three baselines aim to match two tag sets from heterogenous domains. They were compared to justify the effectiveness of our WNGD-based semantic matching for resolving RAHD-ER.

4. **NGD.** This is classic Normalized Google Distance (NGD for short) [20]. For recommendation, the similarity between a user and an item can be represented by the NGD's value of their feature (tag/keyword) sets. Specifically, given a user u and an item v, we suppose $T_u$ represents the concepts which refer to the concepts mapped to u's tags/keywords in the encyclopedia, so does $T_v$ for v. Then the NGD similarity of u and v is computed as

$$GS(u, v) = 1 - \frac{\max \{ \log |T_u|, \log |T_v| \} - \log |T_u \cap T_v|}{\log C - \min \{ \log |T_u|, \log |T_v| \}} \quad (11)$$

where $C$ is the total number of concepts in the encyclopedia.

5. **LEXIS.** In this baseline, we measured the similarity of two tag sets by directly computing the cosine similarity of their respective tag score vectors. Obviously, the similarity is 0 if there are not any common tag existing in the two tag sets under this metric.

6. **ESA.** This baseline is actually a variant of original ESA, in which we use the hyperlinks of reference concepts instead of the occurrences of terms in one concept article to construct the concept vectors. It measure the semantic relatedness between two tag/keyword sets by the cosine similarity of their respective concept vectors.

In our experiments, we used Precision (Pr), Average Precision (AP) and nDCG to quantify the acceptance rate of the users to the recommended tags/Douban movies/Diabetes2 posts.

**C. Douban Movie Recommendation**

Our solution’s objectives are two-fold: 1) accurately profiling users in social networks by tags; 2) effectively correlating users to items through semantic matching. The latter is up to the effectiveness of the former. Thus, in order to justify the effectiveness of our approach, each rival method on Douban movie recommendation was conducted by two steps, i.e., tag-based Weibo user profiling followed by the matching of tag sets.

Since a Weibo user can only use 10 real tags at most according to Weibo setting, we only used the top-10 tags generated by LTPA for fair comparison with REALTAG. We compared our LTPA+WNGD with REALTAG+WNGD and TWEET+WNGD to justify the effectiveness of LTPA-based user profiling. As well, we compared LTPA+WNGD with LTPA+NGD, LTPA+ESA and LTPA+LEXIS to justify the effectiveness of WNGD-based semantic matching. Fig.3 plots the human assessing results of all rivals on Douban movie recommendation. There are many interesting results we can observe from the results. At first, POP performed worst due to the lack of personalization for the users. The superiority of LTPA+WNGD over REALTAG+WNGD shows that the tags generated by LTPA can profile a user more accurately in terms of recommendation although the real tags are specified by a user him/herself. The superiority of LTPA+WNGD over LTPA+NGD shows that the entry weights in concept vectors added in NGD can quantify the semantic relatedness between two tag sets more accurately. The superiority of LTPA+WNGD over LTPA+ESA indicates that WNGD is a more effective measure of semantic relatedness than ESA in terms of recommendation. The performance of LTPA+LEXIS is not very good due to that, 24% of volunteers were recommended to with randomly selected movies since their tag sets and movie tag sets are disjoint entirely. It is unsurprising that TWEET+WNGD was beaten by others because TWEET’s poor performance on user profiling.

**D. Diabetes Post Recommendation**

We first removed a fraction of shared keywords before running the algorithms in order to simulate the setting of RAHD-ER. Then, we tested rival algorithms under different situations when a specific proportion of shared keywords are deleted. As introduced in Subsec. IV-A, the ground truth of diabetes post recommendation was generated according to the principle of LEXIS. In addition, both Diabetes1 users deeply investigating users’ tweets, we found that many tweet keywords are colloquialisms, proper names or news locations, which can not accurately specify a user’s profile by these informal words.
Local Tag Propagation Algorithm of a Weibo user's neighbors to profile him/her through individuals in social networks. Specifically, we filter the tags of a Weibo user's neighbors to profile him/her through a Local Tag Propagation Algorithm. For discovering the semantic relations between the items and the users across two heterogenous domains, we propose a Weighted Normalized Google Distance metric to measure the similarity of two feature (tags/keyword) sets based on concept-based semantic interpretation. Experimental evaluations justify our approach’s effectiveness and superiority over the state-of-the-art competitors.

REFERENCES