A Random Walk Model for Item Recommendation in Social Tagging Systems

ZHU ZHANG, Chinese Academy of Sciences
DANIEL D. ZENG, Chinese Academy of Sciences and University of Arizona
AHMED ABBASI, University of Virginia
JING PENG, University of Pennsylvania
XIAOLONG ZHENG, Chinese Academy of Sciences

Social tagging, as a novel approach to information organization and discovery, has been widely adopted in many Web 2.0 applications. Tags contributed by users to annotate a variety of Web resources or items provide a new type of information that can be exploited by recommender systems. Nevertheless, the sparsity of the ternary interaction data among users, items, and tags limits the performance of tag-based recommendation algorithms. In this article, we propose to deal with the sparsity problem in social tagging by applying random walks on ternary interaction graphs to explore transitive associations between users and items. The transitive associations in this article refer to the path of the link between any two nodes whose length is greater than one. Taking advantage of these transitive associations can allow more accurate measurement of the relevance between two entities (e.g., user-item, user-user, and item-item). A PageRank-like algorithm has been developed to explore these transitive associations by spreading users' preferences on an item similarity graph and spreading items' influences on a user similarity graph. Empirical evaluation on three real-world datasets demonstrates that our approach can effectively alleviate the sparsity problem and improve the quality of item recommendation.

Categories and Subject Descriptors: H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information filtering*

General Terms: Design, Algorithms, Experimentation, Performance

Additional Key Words and Phrases: Recommender systems, random walk, sparsity, social tagging

ACM Reference Format:
Zhang, Z., Zeng, D. D., Abbasi, A., Peng, J., and Zheng, X. L. 2013. A random walk model for item recommendation in social tagging systems. *ACM Trans. Manage. Inf. Syst.* 4, 2, Article 8 (August 2013), 24 pages.
DOI:http://dx.doi.org/10.1145/2499962.2499966

1. INTRODUCTION

In recent years, social tagging has become increasingly popular in many Web 2.0 applications, including social bookmarking (e.g., Delicious, CiteULike), music

An earlier version of this article was presented at the 21st Workshop on Information Technologies and Systems.
This research was partially funded through NNSFC Grants #71025001, #91024030, #70890084, #71103180, #91124001, and MOH Grants #2013ZX10004218 and #2012ZX10004801.
Authors' addresses: Z. Zhang and X. L. Zheng, State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, China; D. D. Zeng (corresponding author), State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, China and Department of Management Information Systems, The University of Arizona; email: dajun.zeng@ia.ac.cn; A. Abbasi, McIntire School of Commerce, University of Virginia; J. Peng, Department of OPIM, University of Pennsylvania.

DOI:http://dx.doi.org/10.1145/2499962.2499966

recommendation (e.g., Last.fm), and video sharing (e.g., YouTube). Social tagging allows users to annotate and categorize a variety of resources (e.g., Web pages, songs, videos), generally referred to as items. Users can annotate items with descriptive words of their own choice, providing a novel mechanism for organizing and discovering resources. The semantic information embedded in tags constitutes an additional information source pertaining to the interaction between users and items. As such, how to best leverage tag information to enhance item recommendation performance is a topic that has been attracting greater attention from the recommender systems research community.

Several different algorithms have been proposed for tag-based (or tag-aware) item recommendation. These algorithms can be divided into two main kinds. The first kind treats tags as new features for describing user preferences and item characteristics. These new features are then incorporated into traditional Collaborative Filtering (CF) [Goldberg et al. 1992] methods without using the three-dimensional correlations among users, tags, and items [Peng et al. 2010a; Tso-Sutter et al. 2008; Wetzker et al. 2009; Zheng and Li 2011]. The second type of approaches keeps the three-dimensional correlations by using a 3rd-order tensor to model social tagging data, and then applies tensor decomposition methods to reveal the latent semantic associations among users, tags, and items [Peng et al. 2010b, 2011; Rendle et al. 2009; Symeonidis et al. 2010].

While a number of tag-based item recommendation methods have been proposed in the literature, the data sparsity problem, which largely inhibits the performance of recommender systems, has not yet been sufficiently addressed. In the context of item recommendation, data sparsity can be attributable to the fact that most users only interact with a small percentage of items, resulting in limited user-item interactions. The situation is exacerbated since users only provide a small number of tags when annotating items they have interacted with, resulting in limited user-item-tag ternary interactions. As shown in Table III, the data densities (percentage of non-zero entries in the user-item matrix) of the Delicious and CiteULike datasets in this study are less than 5%, even after heavily pruning infrequent users, items, and tags. User-item sparsity is caused by the fact that the items chosen by users account for only a very small proportion of the whole item set in real-world social tagging applications, resulting in sparse user-item matrices. Similarly, item-tag sparsity is caused by the fact that a user often intends to annotate an item with only a few tags ($3\sim4$ on average). The user-item matrix will be used to compute the inter-user and inter-item similarities in this article, so user-item sparsity can lead to the insufficiently accurate measure of the inter-user and inter-item similarities. Moreover, the item-tag matrix will also be used to compute the inter-item similarities, so item-tag sparsity can lead to the insufficiently accurate measure of the inter-item similarities. Intuitively, using item-tag matrix to calculate the inter-item similarities is better than using user-item matrix, because tags are more semantic and descriptive than users when used as the features of items. These are the main differences between user-item sparsity and item-tag sparsity.

In the context of social tagging, many tag-based item recommendation models that are based on traditional CF methods, including similarity-based and model-based methods, are susceptible to data sparsity issues [Ma et al. 2011]. Under sparse data, similarity-based [Jin et al. 2004; Linden et al. 2003; Ma et al. 2007] recommendation methods may fail to find a sufficient number of similar neighbors. Model-based CF algorithms [Hofmann 2003, 2004; Salakhutdinov and Mnih 2008; Si and Jin 2003] also have difficulty with users that have rated only a few items. This problem becomes even salient in tensor decomposition algorithms [Cai et al. 2011] as it requires users, items, and tags to co-occur simultaneously, whereas users could bookmark items without assigning tags and subscribe to tags without specifying items.

To alleviate the data sparsity problem in the context of social tagging, we propose a random-walk-based item recommendation model that exploits the transitive associations among users, items, and tags. Specifically, we first construct an item graph and a user graph, in which the edges linking two items and two users are weighted by their similarities, respectively. Throughout this article, we define the term "similarity" as a value measuring the distance between two nodes (users or items). Random walks are an effective way to explore transitive associations between nodes in a graph [Gori and Pucci 2007; Yildirim and Krishnamoorthy 2008], as well as to compute the similarity between nodes [Fouss et al. 2007]. Accordingly, we design a PageRank-like [Page et al. 1999] algorithm to apply multistep random walks on the item graph and user graph, so as to capture the transitive associations among users, tags, and items and obtain personalized item rankings for each user. Empirical evaluation on three real-world datasets demonstrates that our approach can efficiently alleviate the sparsity problem and improve the quality of item recommendation compared to several benchmark methods.

The remainder of this article is organized as follows. Section 2 briefly reviews prior work on tag-based recommendation and random-walk-based recommendation. In Section 3, we present the proposed random walk model. In Section 4, an empirical evaluation is presented to compare our approach with other recommendation methods. Section 5 highlights our research contributions and describes future directions.

2. RELATED WORK

Three streams of work are closely related to this article: hybrid recommender systems, tag-based recommendation and random-walk-based recommendation.

2.1. Hybrid Recommender Systems

Recommender systems are usually classified into the three categories, namely content-based recommender systems, collaborative filtering, and hybrid recommender systems [Adomavicius and Tuzhilin 2005; Balabanović and Shoham 1997]. Content-based recommender systems use the textual features of users and items for recommendations [Si and Jin 2003; Su and Khoshgoftaar 2009], while collaborative filtering only uses the user-item interaction information (either explicit or implicit) such as ratings, purchases, and browsing history to make predictions. Hybrid recommender systems combine both content-based recommendation and collaborative filtering to make predictions.

Hybrid recommender systems can usually be further classified into four classes [Adomavicius and Tuzhilin 2005]. One class of hybrid systems implements content-based and collaborative methods separately and then combines their predictions using linear addition [Bellogin et al. 2011; Claypool et al. 1999], voting [Pazzani 1999], switching [Burke 2002], or cascading [Ghazanfar and Prugel-Bennett 2010]. Another class incorporates content-based characteristics into collaborative models [Balabanović and Shoham 1997; Good et al. 1999; Vipul 2012]. A third class adds collaborative characteristics to content-based models [Soboroff and Nicholas 1999]. The fourth class builds a general unifying model that incorporates different recommendation methods (usually content-based and CFs) [Basu et al. 1998; Gunawardana and Meek 2009; Popescul et al. 2001; Wang et al. 2006]. For example, Wang et al. [2006] proposed a generative probabilistic framework that can unify user-based and item-based CF approaches by similarity fusion.

Most of this work deal with rating data in which numerical feedbacks of users on items are available. However, in the context of social tagging, the "feedbacks" of users on items are presented in the form of tags and numerical feedbacks are absent. To accommodate the special nature of tagging data, we proposed a random-walk-based recommendation model for tag-aware item recommendation, which uses the content information such as tag and user-item interaction information, and applies the basic ideas of user-based and item-based CF approaches in a coherent way.

2.2. Tag-Based Recommendation

There have been a number of studies on tag-based (or tag-aware) recommendation in the literature. One way is to use tag information to compute user or item similarity. This idea can be easily incorporated into existing similarity-based CF algorithms which recommend items similar users have purchased or items similar to those the active user has already purchased [Tso-Sutter et al. 2008; Zeng and Li 2008; Zhao et al. 2008; Zheng and Li 2011]. For example, Zeng and Li [2008] proposed two variants of the standard user-based and item-based methods by calculating user and item similarities based on TF-IDF weighted tag vectors. Tso-Sutter et al. [2008] extended item vector for user profile and user vector for item profile with tags. They then applied a linear interpolation method to fuse the resulting user-based and item-based methods.

Except the above heuristic methods, several model-based algorithms for tag-based recommendation have been proposed for tag-based CF recommendation. Zhen et al. [2009] employed user similarities in the tag space to regularize the probabilistic matrix factorization procedure. Wetzker et al. [2009] presented a probabilistic Latent Semantic Analysis (PLSA) model capturing both the item-user and item-tag co-occurrence information for recommendation. Zhang et al. [2010] proposed a recommendation algorithm based on an integrated diffusion on user-item-tag tripartite graphs. Peng et al. [2010b] presented a joint item-tag recommendation framework, which explicitly pointed out the topical interests of users in the recommended items and made full use of all available interactions among users, items, and tags. In addition, a framework named Collaborative Filtering with Unlabeled Items (CFUI) [Peng et al. 2010a] was proposed to deal with the sparsity problem by making effective use of unlabeled items.

A recent book [Marinho et al. 2012] summarizes the state of the art of recommendation techniques for social tagging systems. This book introduces the recent advanced technologies (e.g., tensor factorization, relational classifier, and exploring the content of resources and social relations, etc.) used in the tag recommendation of social tagging systems. Some of these advanced technologies, such as tensor factorization, can also be used for item recommendation in social tagging systems; the research problem explored in this article. For instance, Nanopoulos et al. [2010] exploited the HOSVD model combined with music similarity based on audio features to leverage the latent ternary structure of social tagging systems for personalized music recommendation. Guy et al. [2010] proposed a personalized item recommendation algorithm based on people and tags with an enterprise social media application suite that included blogs, bookmarks, communities, wikis, and shared files. The content of resources [Jeon et al. 2011; Li et al. 2008] and social relations [Jiang et al. 2010; Liu et al. 2010] mentioned in the Marinho et al. book also can be used for the computation of item and user similarities in our proposed approach. The proposed approach is based on random walks applied to the associations among the user-item, user-tag, and item-tag bipartite graphs. It is different from tensor factorization applied to the 3rd-order tensor representation of social tagging systems in Nanopoulos et al. [2010]. Similarly, the social relation (e.g., friendship, organizational relation etc.) in Guy et al. [2010] is not used in the proposed paper, but it could be incorporated into our model by combining it with user similarity (a possible future direction).

The sparsity problem limits the performance of recommender systems both in the conventional user-item setting and in the context of social tagging systems. However, there has been limited research dealing with the sparsity problem in the context of social tagging. Compared with these methods mentioned previously, our method leverages the transitive associations that are ignored in these methods to deal with the sparsity problem.

2.3. Random-Walk-Based Recommendation

Random walk on graph is an effective way to compute the similarity between nodes [Fouss et al. 2007] and explore transitive associations between nodes [Gori and Pucci 2007; Jamali and Ester 2009; Yildirim and Krishnamoorthy 2008]. Random walk models have been used in recommender systems in several different ways. Fouss et al. [2007] presented a computing method on random-walk-based similarity between nodes of a graph with application to collaborative recommendation. Gori and Pucci [2007] presented a biased PageRank-like scoring algorithm named ItemRank, which can be used to rank products according to expected user preferences. Yildirim and Krishnamoorthy [2008] proposed an item-oriented recommendation algorithm that used random walk to calculate item similarity matrix.

The experiments in Huang et al. [2004] and Yildirim and Krishnamoorthy [2008] empirically showed that transitive associations are a valuable source of information worthy of being explored to deal with the sparsity problem. In Huang et al. [2004], they model the user-item interaction in bipartite graphs. One set of nodes represents users, and the other set of nodes represents items. The links connecting nodes between these two sets represent the transactions of users. Then they treat collaborative filtering as associative retrieval on the user-item bipartite graph, and apply several spreading activation algorithms to generate transitive associations between users and items. Although our work also explores the transitive associations between nodes, we do not only use the transitive associations between users and items. Instead, we apply random walk model to explore the transitive associations among users, items, and tags.

In the context of social tagging, there are some researches using random walk model to explore transitive associations among nodes [Bogers 2010; Hotho et al. 2006; Konstas et al. 2009]. Hotho et al. [2006] proposed a PageRank-like search and ranking algorithm for folksonomies. In their study, only one graph consisting of users, items, and tags was built; they then presented a new algorithm, called FolkRank, that takes into account the folksonomy structure for ranking search requests. Bogers [2010] presented ContextWalk, a recommendation algorithm that can include different types of contextual information. It models the browsing process of a user on a movie database website by taking random walks over the contextual graph consisting of users, items, tags, genres, and actors.

Our approach differs from prior work in several ways. First, while many studies [Fouss et al. 2006; Yildirim and Krishnamoorthy 2008] did not consider the effect of tags on recommendation quality, our method exploits social tagging information in the proposed random walk model. Second, the graph structures utilized by Bogers [2010] and Hotho et al. [2006] are different from the item graph and user graph used in this article; users, items, and tags are all represented as nodes in a single, larger graph in their paper. Consequently, despite using the same data sets, the size and structure (i.e., quantity and types of nodes) used in their study were considerably different from those employed in our article. This is an important distinction since larger graphs can dramatically increase run times (e.g., those associated with matrix multiplication of the transition probability matrix when computing random walks), thereby making certain algorithms computationally infeasible on larger data sets. Third, Gori and Pucci [2007] and Yildirim and Krishnamoorthy [2008] only constructed an item graph and didn't consider the effect of tags on recommendation quality. Our method not only employs random walk, but also incorporates tag information into the building process of

Table I. Notations

Notation	Description
U_i	The ith user in the user set U
I_j	The jth item in the item set I
T_k	The kth tag in the tag set T
UI_{ij}	The element in the user-item matrix UI, if user i saves item j, it equals one, and otherwise zero
UT_{ik}	The element in matrix user-tag UT, it equals the frequency of tag k used by user i
IT_{jk}	The element IT_{jk} in item-tag matrix IT, it equals the frequency of tag k assigned to item j.
M_{norm}	A stochastic matrix generated by normalizing each row of the matrix M to be of unit length
$M_{i\cdot}$	The ith row vector of the matrix M
$M_{\cdot j}$	The jth column vector of the matrix M
S^{item}	Item similarity matrix
S^{user}	User similarity matrix
UI_{final}	Item ranking matrix of each user

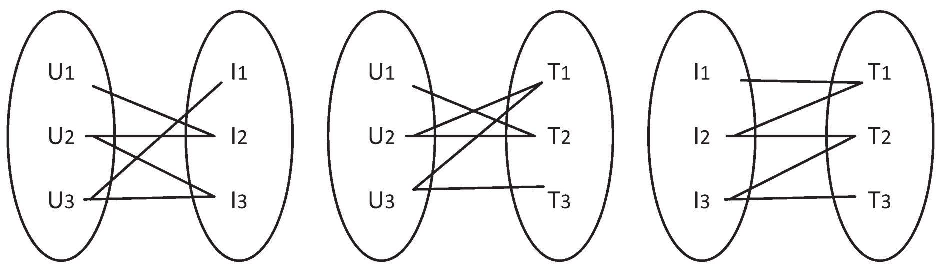


Fig. 1. The user-item, user-tag, and item-tag bipartite graphs.

an item graph and a user graph using a probabilistic method. Details regarding our method are provided in the following section.

3. RANDOM-WALK-BASED RECOMMENDATION MODEL

In this section, we first provide an overview of our approach, focusing on how to exploit the transitive associations to alleviate the data sparsity problem of collaborative filtering. Then, we present the details of the random walk model for item recommendation.

3.1. Model Overview

A social tagging system consists of three main components: users, tags, and items. In this study, we represent a social tagging system using three bipartite graphs depicted in Figure 1, since such bipartite graphs can explicitly represent the user-item, user-tag, and item-tag relations. Table I lists all notations used in this article:

In order to provide readers with a good sense of item recommendations in real-world social tagging systems, we use the well-known bookmark site CiteULike as an example. In Figure 2, users (e.g., zhuzi) can use their preferred words (termed as tags in the social tagging system) such as academia, career etc. to annotate the papers or URLs (termed as items) that they are interested in, such as the paper titled Future impact: Predicting scientific success. When users' tagging histories are collected and analyzed by the recommender system, the recommender system can predict which papers or

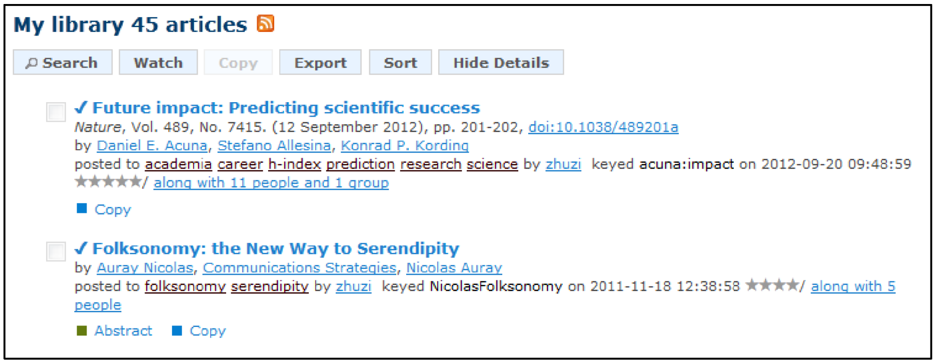


Fig. 2. A snapshot from CiteULike that illustrates a user's tagging behavior in a social tagging system.

Fig. 3. A snapshot from CiteULike that illustrates the item recommendations generated in a social tagging system.

URLs users may like. Figure 3 presents the papers recommended by the recommender system of CiteULike.

In this study, we treat the problem of item recommendation as a link prediction task aiming to predict the strengths of the unknown associations between users and items. We propose to deal with the sparsity problem in social tagging by applying random walks on ternary interaction graphs to explore transitive associations between users and items. Taking advantage of these transitive associations can allow more accurate measurement of the relevance between two entities (e.g., user-item, user-user, and item-item). Furthermore, we design a PageRank-like algorithm to explore these transitive associations by spreading users' preferences on an item similarity graph and items' influences on a user similarity graph. The proposed algorithm can result in a personalized item rank for each user, and then the top-N item recommendation can be generated by sorting the items in descending order of ranking scores.

Transitive associations can be explored to alleviate the sparsity problem in item recommendation. In social tagging systems, users tend to annotate a small number of items with a few tags, consequently the quantity of direct user-item, item-tag, and user-tag interactions is sparse. Here, direct interaction means there is no intermediate entity between one entity (e.g., user, item, or tag) to another entity. However, one entity can reach another entity through other entities, whose path is termed as a transitive association in this article. Specifically, if we take into account these transitive associations when measuring the relevance between two entities, transitive associations as the hidden interaction information can add more information in the measurement of the relevance between any two entities. Therefore, transitive associations can make the relevance measure between any two entities more accurate. In the context of item recommendation for social tagging, inter-item and inter-user similarities can be measured more accurately, which leads to the alleviation of the sparsity problem and the improvement of recommendation performance.

The intuition on how transitive associations can alleviate sparsity can be explained by the following example. In Figure 1, we need to compute the strength of the connections between U_1 and I_1 before we can recommend I_1 to U_1. Since U_1 already has an edge with I_2, first, we can find some of the associations between I_1 and I_2 in the item-tag (e.g., $I_1 - T_1 - I_2$ and user-item graphs (e.g., $I_1 - U_3 - I_3 - U_2 - I_2$). The path $I_1 - U_3 - I_3 - U_2 - I_2$ is one of the transitive associations between I_1 and I_2. Then we can get some of the connections between U_1 and I_1 by connecting $U_1 - I_2$ with $I_1 - T_1 - I_2$ (or $I_1 - U_3 - I_3 - U_2 - I_2$), resulting in: $I_1 - I_2 - T_1 - I_1$ or $U_1 - I_2 - U_2 - I_3 - U_3 - I_1$. Such associations, which are often ignored in many recommendation models, allow the representation of otherwise hidden relations among users, items, and tags. In this example, these transitive associations can enhance the accuracy of the relevance measure between U_1 and I_1, thereby alleviating the problem of diminished recommendation quality attributable to data sparsity.

Random walk on graph is an effective way to explore transitive associations between nodes [Gori and Pucci 2007; Yildirim and Krishnamoorthy 2008] and compute the similarity between nodes [Fouss et al. 2007]. A random walk over a graph is a stochastic process in which the initial state is known and the next state is governed by a transition probability matrix that indicates the likelihood of jumping from node i to node j in the graph [Bogers 2010]. According to the definition of the transition probability matrix, one-step transition probability matrix indicates that the probability from one node to another node without any intermediate node. Moreover, multi-step transition probability matrix indicates the probability from one node to another node through other intermediate nodes. Therefore, the strength of transitive associations between any two nodes can be measured by random walk on a graph. The intuition about how the transitive associations are captured by random walk on a graph can be explained by the following example. In a directed graph consisting of four nodes (e.g., A, B, C, and D), the weight of a link between two nodes indicates the transition probability from one node to the other. Suppose node A has no direct link to node D, but has a link to node C. Also suppose node A has a link to node B; node B has a link to *C; and* node C has a link to D. This easy graph only has these four links. Then there are two directed transitive associations starting from node A to node D that are $A \rightarrow C \rightarrow D$ and $A \rightarrow B \rightarrow C \rightarrow D$. A random walker starting from node A can reach node D after three steps random walk with a probability that equals the product of the weights of the links in the path $A \rightarrow B \rightarrow C \rightarrow D$.

The proposed random-walk-based recommendation model has two underlying assumptions.

— One is that each user will choose new items similar to the ones they have chosen in the past.
— The other is that users will choose new items that were previously selected by similar users (i.e., ones with other common items).

The first assumption is based on standard content-based recommendations while the second assumption is based on collaborative filtering recommender systems. In response to the first assumption, we construct an item graph, in which each edge between two item nodes is weighted by their similarity. Then, the item similarity matrix is treated as the transition probability matrix of the random walk on the item graph and the saved item nodes of a user, indicating the user's preference, are used as the starting nodes of a random walker on the item graph. After the user wanders on the item graph according to the transition probability, the user can reach the item nodes that connected to the initial item nodes with the transitive associations. This means that the transitive associations between the user and other items can be captured by random walks and the user's preference has spread on the item graph. Subsequently, random walks on the item graph will generate a vector for each item that signifies users' preference degrees for the items.

In response to the second assumption, a user graph is constructed in a manner similar to how the item graph is built. Then, the user similarity matrix is treated as the transition probability matrix of the random walk on the user graph, and the initial users of each item before random walk are used as the starting nodes of the random walk on the user graph of this item. After the item wanders on the user graph according to the transition probability, the item can reach the user nodes that connected to the initial user nodes with the transitive associations. This means that the transitive associations between the item and other users can be captured by random walks and the item's influence to users has spread on the user graph. Subsequently, random walks on the user graph will generate a vector in the space of the user that can predict the probabilities of the different users' choices for that item.

In our recommendation algorithm, we use linear interpolation to combine the ranking scores of items for each user, which result from the random walk on the item graph and the random walk on the user graph. Finally, the proposed algorithm can result in a personalized item rank for each user, and then the top-N item recommendation can be generated by sorting the items in descending order of ranking scores.

3.2. Random-Walk-Based Item Recommendation

3.2.1. Tag-Based Item Recommendation Algorithm. The proposed item recommendation algorithm is similar to personalized PageRank [Page et al. 1999] in the sense that both of them employ random walk to rank nodes of a graph. PageRank uses the Markov chain to model the process of the random walk on the web graph consisting of a large number of pages as nodes. It assumes that a random surfer will randomly jump to another page j from the current page i with transition probability p(j|i), which is determined by the structure of web hyperlinks, and forms the transition probability matrix $\mathbf{P}$. After a long run, the stationary probability of staying at some page f reflects the authority of the page f. The formulation for PageRank can be described as follows.

$$\pi(t) = \alpha \cdot \pi(t-1) \cdot P + (1-\alpha) \cdot \nu, \tag{1}$$

where π is the row vector of the ranking score of nodes, $\mathbf{P}$ is the transitional probability matrix in which every row sums up to 1. α is a tunable decay factor that is between 0 and 1. The row vector $\mathbf{v}$ also sums up to 1 and has non-negative entries, and it can be used to bias PageRank to be topic sensitive or personalized. The PageRank algorithm will result in a global ranking of the authority of nodes. As to recommendation algorithm, we need a personalized ranking of items for each user.

The procedure of random-walk-based item recommendation algorithm is depicted in Table II. The computation of item similarity matrix S^{item} and user similarity matrix S^{user} is discussed in the Section 3.2.2. The equation at line 6 of Table II reflects the random walk on the item graph, and the equation at line 10 reflects the random walk on the user graph. UI^{item} and UI^{user} are the item-centric and user-centric predicted ranking matrices, respectively. The parameters η and λ are the damping factors that are between 0 and 1. Larger values of these two parameters increase the importance of the transitive associations captured by the multi-step random walks. However,

Table II. The Procedure of Random-Walk-Based Item Recommendation

Algorithm: random-walk-based item recommendation
1: **Input:** UI, S^{item}, S^{user}, q // q is the number of iterations
2: **Output**: UI_{final}
3: $UI^{item}(0) = UI^{user}(0) = \overline{UI} = UI_{norm}$ // $\overline{UI}$ is a temporary variable
4: **for** t ← *0* to *(q-1)* **do**
5: **for** i ← *0* to *(m-1)* **do //** m is the number of users
6: $UI^{item}_{i\cdot}(t+1) = \eta \cdot UI^{item}_{i\cdot}(t) \cdot S^{item} + (1-\eta) \cdot \overline{UI}_{i\cdot}$
7: **end**
8: **for** j ← *0* to *(n-1)* **do //** n is the number of items
10: $UI^{user}_{\cdot j}(t+1) = \lambda \cdot S^{user} \cdot UI^{user}_{\cdot j}(t) + (1-\lambda) \cdot \overline{UI}_{\cdot j}$
11: **end**
12: **end**
13: $UI_{final} = \mu \cdot UI^{item}(q) + (1-\mu) \cdot UI^{user}(q)$

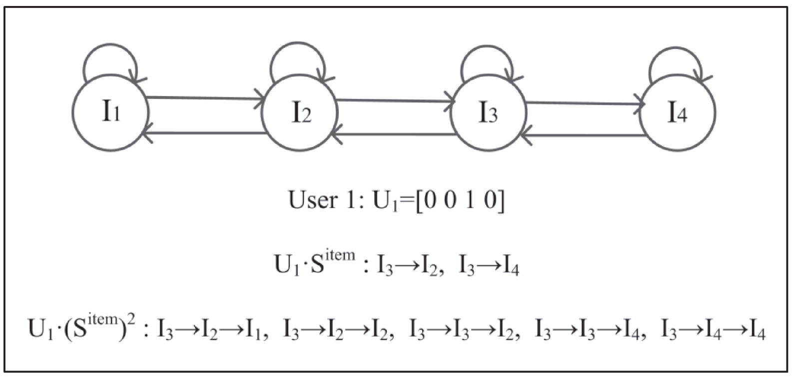


Fig. 4. An illustration of random walks on the item graph.

lengthier multi-step transitive associations may be not helpful for the recommendation performance of the proposed method. Every row vector of $\overline{UI}$ at line 6 and 10 is the preference vector of the corresponding user. When UI^{item}(t) and UI^{user}(t) reach the acceptable optimal performance, we can get the final predicted user-item score matrix UI_{final} by fusing them with the use of the linear combination. With respect to the number of iterations, we will discuss its impact on the recommendation performance in Section 4.3.

When we extend the equation at line 6, we can get Eq. (2). From the following equation, we can see that the transitive associations between items are represented by the power of item similarity matrix $(\eta S^{item})^k$. It means that the transitive probability of k steps random walks on the item graph can be obtained from $(\eta S^{item})^k$. We illustrate random walks on the item graph with Figure 4. As shown in Figure 4, the items are connected by the arrowed lines whose weights are the transitive probabilities. U_1 is represented by the first row of the user-item matrix UI, that is $U_1 = [0, 0, 1, 0]$. Starting from I_3, U_1 (as a random walker) can reach I_2 and I_4 directly after a one-step random walk, and get to I_1 with the paths $I_3 \rightarrow I_2 \rightarrow I_1$ after two-step random walks. Additionally, U_1 can reach I_2 with the paths $I_3 \rightarrow I_2 \rightarrow I_2$ and $I_3 \rightarrow I_3 \rightarrow I_2$, and reach I_4 with the paths $I_3 \rightarrow I_3 \rightarrow I_4$ and $I_3 \rightarrow I_4 \rightarrow I_4$ after two-step random walks. Note that a random walker can remain in its current position with a certain probability. In this illustration, we ignore the paths for which the starting point and end point are I_3, since we don't need to predict the ranking score of I_3. Likewise, when we extend the equation at line 10, we can get the following Eq. (3). From the following equation, we can see that the transitive associations between users are represented by the power of user similarity matrix $(\lambda S^{user})^k$:

$$UI^{item}_{i\cdot}(t+1) = UI^{item}_{i\cdot}(0) \cdot (\eta S^{item})^{t+1} + (1-\eta)\overline{UI}_{i\cdot} \sum_{k=0}^{t} (\eta S^{item})^k \tag{2}$$

$$UI^{user}_{\cdot j}(t+1) = (\lambda S^{user})^{t+1} \cdot UI^{user}_{\cdot j}(0) + (1-\lambda) \left(\sum_{k=0}^{t} (\lambda S^{user})^k \right) \overline{UI}_{\cdot j}. \tag{3}$$

If the iteration is infinite, we can get the following two equations that are the matrix notation of two equations.

$$UI^{item} = (1-\eta)\overline{UI} \sum_{k=0}^{\infty} (\eta S^{item})^k = (1-\eta)\overline{UI}(1-\eta S^{item})^{-1} \tag{4}$$

$$UI^{user} = (1-\lambda) \left(\sum_{k=0}^{\infty} (\lambda S^{user})^k \right) \overline{UI} = (1-\lambda)(1-\lambda S^{user})^{-1}\overline{UI}. \tag{5}$$

Interestingly, the two equations $\sum_{k=0}^{\infty} (\eta S^{item})^k = (1-\eta S^{item})^{-1}$ and $\sum_{k=0}^{\infty} (\lambda S^{user})^k = (1-\lambda S^{user})^{-1}$ are the von Neumann diffusion kernels [Fouss et al. 2006] of item graph and user graph. In Eqs. (4) and (5), all the transitive connections between items and users are captured. However, if we directly make use of the above two equations, we need to compute the inverse of the matrix. From the Eqs. (2) and (3), we can see that item-based CF and user-based CF are the specific cases of Eq. (2) and Eq. (3) respectively, when the number of iterations t equals to zero. After we get the ranking matrix UI_{final}, we select top-N ranked items that have not been saved for each user, by sorting the ranking scores in the descending order.

3.2.2. Item and User Similarities. In this section, we discuss how to compute the item and user similarity matrices. In the proposed method, the similarity computation is an important step, since different similarity computation methods may result in varying recommendation performance. Unlike the numeric rating data found in traditional recommender systems (e.g., scaling from 1 to 5 or 10), the elements in the user-item matrix are binary. The commonly used similarity methods such as Pearson Correlation Coefficient and adjusted cosine similarity [Sarwar et al. 2001] fail, because both the numerator and the denominator in the formulas equal zero. Therefore, we present a probability-based method for similarity computation, and also briefly introduce the cosine similarity in the previous literature for comparison. These two methods are described as follows.

Probability-Based Similarity. As proposed by Deshpande and Karypis [2004], each row of the binary user-item matrix is normalized to be of unit length in the computation of item similarity. Consequently, customers that have purchased more items will tend to contribute less to the overall cosine similarity between items. This gives emphasis to the purchasing decisions of the customers that have bought fewer items. Inspired by the idea of normalization, we propose a probability-based similarity method for deriving the item similarity method, and we incorporate IT and UI into the calculation of item similarity. Because IT contains the content information of items and UI contains the user-item interaction information, we expect that the integration of the two will be more effective for computing the similarity between items. The resulting formulation for probability-based item similarity is as follows:

$$S^{item} = \alpha \cdot IT_{norm} \cdot (IT^T)_{norm} + (1-\alpha) \cdot (UI^T)_{norm} \cdot UI_{norm},$$

where α is a tunable parameter that is between 0 and 1. It can control the weight of IT and UI in the computation of item similarity. The '·' denotes the dot-product operation. Since the calculation rationale for both parts of the equation (i.e., the IT part and the UI part) is identical, we simply illustrate using $IT_{norm} \cdot (IT^T)_{norm}$ as an example. We can get the probability from one item to all the tags from the row vectors of IT_{norm} and the probability from one tag to all the items from the row vectors of $(IT^T)_{norm}$. Then, the similarity between item i and item j can be computed as the dot-product of the ith row vector of IT_{norm} and the jth column vector of $(IT^T)_{norm}$, which represents the probability that item i jumps to item j through all of the tags. Likewise, we can compute the user similarity matrix as follows. S^{item} and S^{user} can be directly used as the transition probability matrix of the item and user graphs respectively, because twice normalization operations in the similarity computation make them become stochastic matrices.

$$S^{user} = \beta \cdot UT_{norm} \cdot (UT^T)_{norm} + (1-\beta) \cdot UI_{norm} \cdot (UI^T)_{norm}.$$

Cosine Similarity. Cosine similarity is a commonly used way of computing similarity between two items or users in recommender system. We first represent each item or user as a vector, and then treat the cosine value between the two vectors as the similarity value. Formally,

$$sim(i,j) = cos(\overline{v_i}, \overline{v_j}) = \frac{\overline{v_i} \cdot \overline{v_j}}{\|\overline{v_i}\|_2 \|\overline{v_j}\|_2},$$

where "·" denotes the vector dot-product operation. With the use of this formula, we can get the similarity between item i and item j.

$$sim(I_i, I_j) = \alpha \cdot cos(IT_{i\cdot}, IT_{j\cdot}) + (1-\alpha) \cdot cos(UI_{\cdot i}, UI_{\cdot j})$$

Then, we can get the item similarity matrix $\widetilde{S}^{item}$ whose element $\widetilde{S}^{item}_{ij}$ equals $sim(I_i, I_j)$. Afterwards, we need to normalize each row of $\widetilde{S}^{item}$ to be of unit length, and then the normalized item similarity matrix S^{item} can be used as the transition probability matrix of the item graph. Likewise, we can get the similarity between user i and user j as well as the normalized user similarity matrix S^{user}

$$sim(U_i, U_j) = \beta \cdot cos(UT_{i\cdot}, UT_{j\cdot}) + (1-\beta) \cdot cos(UI_{i\cdot}, UI_{j\cdot}).$$

The user-item, item-tag, and user-tag matrices are usually very sparse. However, the item and user similarity matrices become less sparse due to the matrix multiplication and addition in the computation. Moreover, during each iteration of the proposed algorithm, the item-centric ranking matrix UI^{item} is multiplied by the item similarity matrix S^{item}; UI^{item} then become less sparse after the matrix multiplication. The user-centric ranking matrix UI^{user} is similar to UI^{item}. With respect to the time complexity of the generation of item recommendation, we can select the k largest elements of each row of the item and user similarity matrices and set the reminder elements smaller than the k largest elements to zeros.

3.2.3. Computational Complexity. The time complexity of the proposed method contains two parts. One is the time complexity for the computation of item and user similarity matrices as well as the selection of the most similar items and users. The other is the time complexity for the random-walk-based recommendation. The upper bound on the time complexity for the computation of the item similarity matrix is $O(mn^2 + ln^2)$, while the upper bound on the time complexity for computing the user similarity matrix is $O(nm^2 + lm^2)$. Since the user-item, item-tag, and user-tag matrices are very sparse in real-world applications, we can reduce the computational complexity by using sparse data structures to store the sparse matrices and to calculate the multiplication of the sparse matrices. Fortunately, this part can be computed offline. The time complexity of the random-walk-based recommendation is $O(qmkn)$. Because the number of iterations q and the number of the most similar items (users) k is too small in comparison with the number of users m and the number of items n, the time complexity of the online part is *O(mn)*.

4. EMPIRICAL EVALUATION

In this section, we evaluated the proposed method by using three tagging datasets from real-world social tagging systems and conducted different experiments to address the following questions: (1) How effective is the proposed random-walk-based algorithm under sparse data, compared with other benchmark methods? (2) Which is more effective in the computation of similarity, tagging information or user-item interaction information? (3) Is probability-based similarity more effective than cosine similarity in our recommendation model? (4) How do the parameters impact the performance of the proposed algorithm?

4.1. Dataset

Three different datasets are used to test our approach. The first dataset is the BibSonomy dataset[1] that is widely used in the tagging domain (the 2009-07-01 snapshot is used in this article). The BibSonomy dataset includes bookmarks for both general web resources and bibliographies, of which only the part for general web resources was used in our experiment. The second dataset is a snapshot of the CiteULike database[2] that is downloaded on 1/21/2010. The transactions in 2009 were collected and contained 3,390,000 transactions from 27,160 users on 926,721 bibliographies with 247,452 tags. The third dataset was crawled from Delicious on which users can post their favorite URLs and share them with their friends. The collected dataset contains bookmarking data of 5,000 users dated from 6/1/2008 to 12/31/2008. We identified these 5,000 users by using a breadth-first approach to traverse the Delicious user network, starting from a small set of randomly selected seed users. This datasets includes 3,622,279 transactions from 5,000 users on 653,690 bookmarks with 203,983 tags.

During data preprocessing, take the small Delicious dataset for example, we iteratively removed users that had saved less than 15 items and items that had been saved by less than 15 users (termed as unqualified items) until the percentage of unqualified items were less than 2% for each (filtered) dataset. Table III contains the specific thresholds for the other two datasets. In addition, the Snowball stemmer[3] (Porter 2) was used to stem each tag by eliminating the effect of word variations. For computational efficiency, in each testbed, we only considered tags that had been used more than 10 times in the filtered training set. If a <user, item> co-occurrence did not involve any frequent tags, we set the tag entry as null but did not remove it. This was the key difference between our preprocessing method and the approach undertaken with the k-core pruning strategy [Jäschke et al. 2008]. This difference enabled us to

[1] http://www.kde.cs.uni-kassel.de/bibsonomy/dumps
[2] http://www.citeulike.org/faq/data.adp
[3] http://snowball.tartarus.org/

Table III. Dataset Description

Dataset	BibSonomy	CiteULike	Delicious (small)	Delicious (large)
Number of users: m	125	338	548	1097
Number of items: n	388	392	1080	1872
Number of selected/total tags: l	78/2311	52/2822	379/12067	526/9608
Number of total transactions: p	4383	6031	28591	44599
Data density: $p/(mn)$ (%)	9.04	4.55	4.83	2.17
Avg. number of items per user	35.06	17.84	52.17	40.66
Avg. number of users per item	11.30	15.39	26.47	23.82
Number of items per user	>=10	>=5	>=15	>=10
Number of users per item	>=8	>=10	>=15	>=10
Frequency of selected tag	>=10	>=10	>=10	>=10

process transactions without assigned tags. Table III summarizes the statistics for the cleaned datasets.

4.2. Evaluation Metrics

To evaluate the quality of the proposed algorithm, we randomly selected a certain percentage associated with the saved items of each user to form the training dataset, and withheld the remainder as test data. During the training phase, the model was built based on the training data collected from all users. During the prediction phase, we recommended N items to each user and then compared them with bookmarks in the test set. To make sure that the experiment results were not sensitive to the partition of each dataset, we performed 10 runs for each experiment, each time using a different random split. The results reported in the rest of the article are the average of the 10 trials.

The evaluation metrics in this paper are ones commonly employed in prior recommender system research [Herlocker et al. 2004], and include precision, recall, F-measure, and rankscore [Breese et al. 1998]. For each user, precision equals that the number of correct item recommendations divided by the number of all N item recommendations, where correct recommendations refer to those items appearing in the target user's test set. Recall equals the number of correct item recommendations divided by the number of test items. F-measure is a composite measure of the harmonic mean between precision and recall. We adopted the F1 measure in our experiment in order to pay equal attention to precision and recall.

Rankscore measures the ranking quality of a ranked list as compared to the ideal item list. The rankscore measure assumes that each successive item in a list is less likely to be viewed by the user with an exponential decay. In this metric, the expected utility of a ranked list of item recommendations for user i is defined as $R_i = \sum_j \left(\frac{q_j}{\frac{j-1}{2h-1}}\right)$, where j indicates the index of an item in the predicted ranked list, and q_j equals value 1 if the jth item is actually saved by the active user, and otherwise 0. The parameter h is the viewing half-life (the rank of the item on the list such that there is a 50% chance the user will save that item), which was set to 10 in our experiments. The final recommendation utility score of user i is $100 \cdot \frac{R_i}{R_i^{max}}$, where R_i^{max} equals $\sum_j \left(\frac{1}{\frac{j-1}{2h-1}}\right)$ and is the maximum achievable utility if all the item recommendations of user i had been at the top of the ranked list.

Table IV. Formula of Evaluate Metrics

Metric	Formula
Precision	$N_{\mathrm{hit}}/N_{\mathrm{rec}}$
Recall	$N_{\mathrm{hit}}/N_{\mathrm{test}}$
F-measure	$2 \cdot \frac{precision \cdot recall}{precision + recall}$
Rankscore	$100 \sum_j \left(\frac{q_j}{\frac{j-1}{2h-1}} \right) / \sum_j \left(\frac{1}{\frac{j-1}{2h-1}} \right)$

Table V. Experimental Result on BibSonomy

	BibSonomy			
Algorithm	Precision	Recall	F-measure	Rankscore
RAND	6.82	1.19	2.02	6.81
UB	13.90	2.84	4.72	14.20
IB	10.90	2.38	3.90	11.08
FUS	18.88	4.15	6.81	18.96
PLSA	15.94	3.27	5.43	16.24
TagiCoFi	15.17	3.03	5.04	15.19
RW-IT	18.93	3.94	6.52	19.41
RW-UT	16.66	3.27	5.46	17.01
RW-UI	16.70	3.52	5.81	17.02
RW	**19.97**	**4.26**	**7.01**	**20.55**

Formal definitions of these four metrics are summarized in Table IV, where N_{hit} indicates the number of correct recommendations, N_{rec} indicates the number of recommendations, and N_{test} indicates the number of items in the active user's test set. Note that all of them are used for each user, and the final value in each trial is the average across all the users.

4.3. Results

We compared the proposed approach with six other approaches. The RAND algorithm generated random recommendations for every user. The classical user-based (UB) [Breese et al. 1998; Resnick et al. 1994] and item-based (IB) [Sarwar et al. 2001] methods were implemented as baselines. Since there are no rating data in social tagging systems involved in this article, UB and IB are not exactly the same as the original algorithms. In our implementation of UB, the ranking score of the active user for the target item equals the sum of the cosine similarity scores with him/her of the active user's neighbors. In our implementation of IB, the ranking score of the active user for the target item equals the sum of the cosine similarity scores with the target item of the items most similar with the target item. The other three methods were tag-based recommendation methods: the fusion (FUS) [Tso-Sutter et al. 2008], PLSA [Wetzker et al. 2009], and the TagiCoFi [Zhen et al. 2009] methods. The RW variants of the RW method. RW-IT only used the IT matrix in the computation of item and user similarities. Similarly, RW-UT only used the UT matrix, and RW-UI only utilized the UI matrix. However, RW usually used all of the IT, UT, and UI matrices.

To investigate the capability of the proposed approach under sparse data, for each user we randomly select only 20% of the bookmarks for the training set and withheld the remaining 80% of the data for testing on the 10 random data splits. Note that when we tuned a given parameter for the proposed method and the baseline methods, the other parameters were fixed. As we observed that the relative performances of these implemented algorithms are generally consistent across different evaluation

Table VI. Experimental Result on CiteULike

	CiteULike			
Algorithm	Precision	Recall	F-measure	Rankscore
RAND	3.91	1.32	1.97	3.88
UB	11.01	4.63	6.52	11.40
IB	7.67	3.43	4.47	7.80
FUS	12.67	5.28	7.45	12.80
PLSA	12.97	5.19	7.40	13.32
TagiCoFi	8.05	3.11	4.46	8.21
RW-IT	14.61	6.05	8.55	14.96
RW-UT	10.71	4.29	6.12	11.04
RW-UI	10.70	4.56	6.39	11.10
RW	**15.18**	**6.32**	**8.91**	**15.55**

Table VII. Experimental Result on Delicious (small)

	Delicious (small)			
Algorithm	Precision	Recall	F-measure	Rankscore
RAND	3.88	0.46	0.82	3.88
UB	25.76	3.56	6.25	26.36
IB	13.66	1.98	3.45	13.65
FUS	29.82	4.37	7.62	30.27
PLSA	29.61	4.44	7.72	30.40
TagiCoFi	12.99	1.29	2.35	13.33
RW-IT	32.56	4.92	8.55	33.17
RW-UT	25.10	3.65	6.37	25.79
RW-UI	27.99	3.94	6.91	28.53
RW	**32.69**	**4.93**	**8.57**	**33.29**

metrics, we used precision as performance metric when tuning the parameters. Due to computational constraints associated with traversing the entire parameter space in order to attain optimal parameter settings, the reported results in these tables are not optimal. Moreover, when we tuned different parameters for any algorithm, this algorithm was then run again on the same data. Tables V, VI, VII, and VIII summarize the experimental results of top 5 recommendations on the four different real-world datasets. Note that except for rankscore, all values in these tables are showed in percentage.

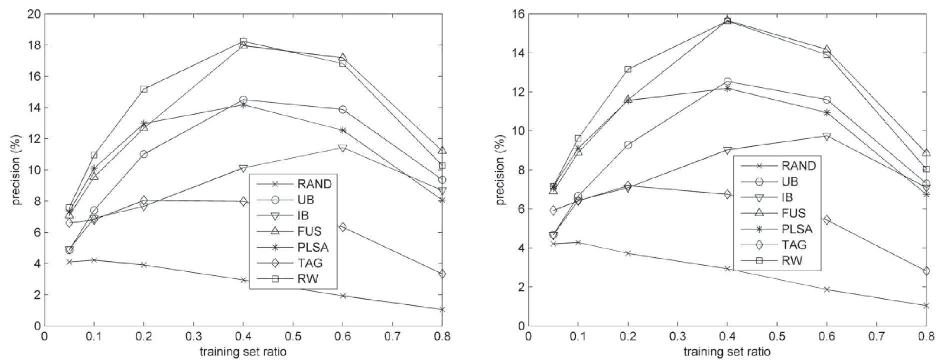
As shown in Tables V, VI, VII, and VIII, RW-IT outperformed RW-UI, demonstrating that the tagging information was more effective than the transactional information in the computation of item similarity. However, the difference between the results for RW-UT and RW-UI were not significant, implying that the tagging information didn't outperform the transaction information in the computation of user similarity. The combination of tag information with the transitive associations among users, tags, and items enabled RW to outperform all comparison methods on all of evaluation conditions. According to an ANOVA test, RW was significantly better than the other algorithms including UB, IB, PLSA, FUS, and TagiCoFi, with $p < 0.001$ on all evaluation metrics for all four datasets except for FUS on the BibSonomy and Delicious (large) datasets. It is also important to note that the results of RW-IT and RW were similar. This indicates that the item-tag interaction information was more important than user-item and user-tag interaction information in the proposed random-walk-based model.

Table VIII. Experimental Result on Delicious (large)

	Delicious (large)			
Algorithm	Precision	Recall	F-measure	Rankscore
RAND	1.78	0.27	0.47	1.77
UB	4.51	0.67	1.17	4.58
IB	2.74	0.43	0.74	2.73
FUS	**7.47**	1.22	2.10	7.49
PLSA	5.78	0.91	1.57	5.83
TagiCoFi	3.28	0.51	0.88	3.27
RW-IT	7.26	1.19	2.05	7.33
RW-UT	5.98	0.98	1.69	6.11
RW-UI	4.04	0.62	1.07	4.07
RW	7.43	**1.24**	**2.12**	**7.56**

Another interesting observation was that UB was significantly better than IB in these four datasets. We believe that this is related to the characteristics of the datasets, in which the average number of items per user was more than the average number of users per item. As a result, it was more accurate to form user neighbors than item neighbors. To investigate the performance of our approach at different density levels, we also conducted an experiment on the CiteULike dataset. We changed the ratio of the training set to the whole dataset and obtained different density levels, as done by Yildirim and Krishnamoorthy [2008]. In other words, a training set ratio of 0.1 meant that 10% of the entire dataset was used for training. For each training set ratio, ANOVA tests were run across the 10 trials, and the *p*-values were used to compare the statistical significance of performance differences between methods. Figure 5 summarizes the experimental results. Since the relative difference between the methods in the above experiment on the values of these four evaluation metrics was consistent, we only tested the performance difference using the precision metric. Additionally, we omitted the experiment results for the top 20 recommendations, since they were very similar to the results reported for the top 10 recommendations. As shown in Figure 5, RW significantly outperformed all comparison methods when using between 10% and 35% of the data for training. RW also outperformed most methods when using larger quantities of training data. For other training set ratios (e.g., 0.4 and 0.6), RW significantly outperformed all comparison methods (with $p < 0.001$), with the exception of FUS. When the training set ratio was 0.8, FUS was significantly better than RW, and RW was significantly better than other methods with $p < 0.007$. Interestingly, when the training set ratio was less than 0.1, the performance difference among RW, FUS and PLSA as well as the performance difference between UB and IB were not significant. We suspect that this was due to the small proportion of the overall dataset used for training; at this setting the training data was simply too sparse to extract meaningful recommendation patterns. Overall, the findings suggest that leveraging tag information and the transitive associations among users, tags, and items can be very beneficial, particularly in situations involving highly sparse data.

As discussed in Section 3.2.2, the computation of item and user similarities is a critical component of the proposed RW method. To understand the impact of probability-based similarity versus cosine similarity for the recommendation performance, we evaluated them on three datasets. Table IX summarizes the recommendation precisions of these two similarity methods. The methods with names beginning with "c" use cosine similarity. As to the bold values in Table IX, we can observe that the precision values using cosine similarity are significantly smaller than the



(a) top 5 recommendations (b) top 10 recommendations

Fig. 5. Experimental results at different density levels.

Table IX. Precisions of Two Similarity Methods

Algorithm	CiteULike	BibSonomy	Delicious (small)
RW-IT	14.61	**18.93**	**32.56**
cRW-IT	13.99	**16.59**	**22.60**
RW-UT	**10.71**	16.66	**25.10**
cRW-UT	**9.06**	16.98	**23.53**
RW-UI	**10.70**	**16.70**	**27.99**
cRW-UI	**8.20**	**11.70**	**17.69**
RW	15.18	19.97	**32.69**
cRW	14.55	19.47	**29.27**

corresponding precision values using probability-based similarity (e.g., $16.59 < 18.93$), with $p < 0.02$.

4.4. Sensitivity of Parameters

Critical factors to the success of RW are the weighting factors (e.g., $\alpha, \beta, \lambda, \eta$, and μ), the number of iterations q and the tag threshold s. Here, tag threshold is the same as the frequency of selected tag in Table III. The functions of the other factors in our model are discussed in Section 3.2. This subsection aims to investigate how these parameters impact the performance of the proposed algorithm. As the relative performances of implemented algorithms are generally consistent across different evaluation metrics, we tuned the parameters based on precision here. Note that when we were tuning a given parameter, the other parameters were fixed. The following parameter tuning experiments were conducted on these four datasets.

Weighting Factor α. As shown in Figure 6(a), the precision when α surpassed 0.5 was higher than the precision where α was less than 0.5. This indicates that the item-tag information was more important than the user-item information in the computation of item similarity in our model.

Weighting Factor β and λ. As can be seen in Figures 6(a) and 7(b), all the former three performance curves were flat, suggesting that the variations in performance were miniscule. This was due to the fact that when we tuned the weighting factor β and λ, the weighting factor μ balanced the impact of the user graph and the item graph equaled 0.9, 0.7, and 0.9 for the CiteULike, Bibsonomy, and Delicious datasets, respectively. Consequently, the contribution of the user graph to the final performance

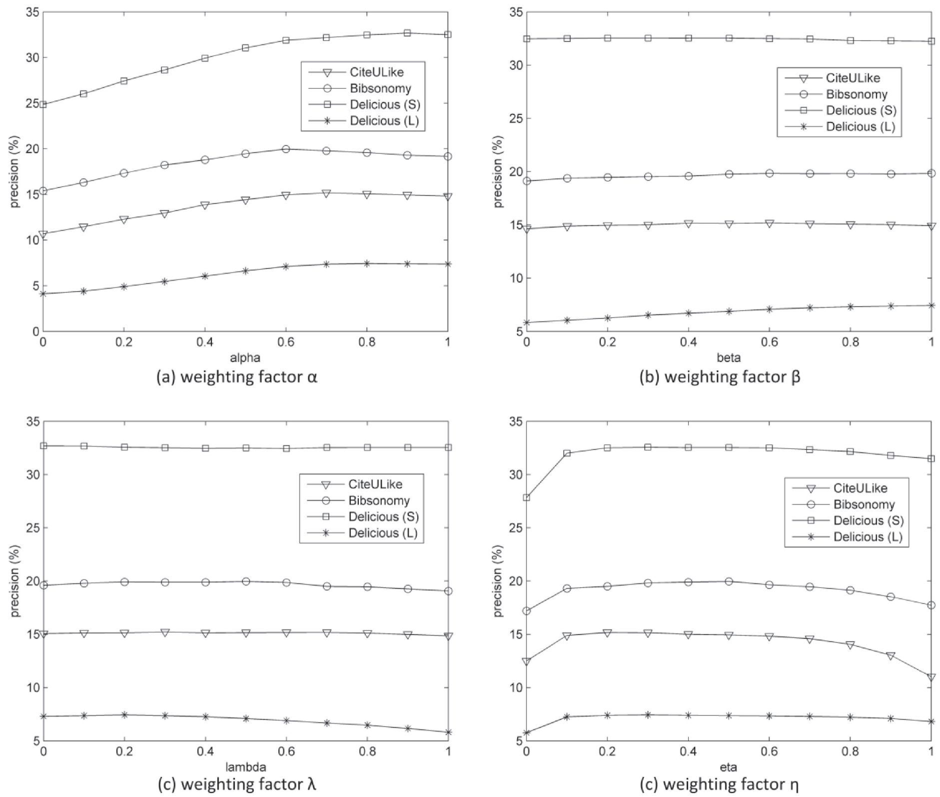


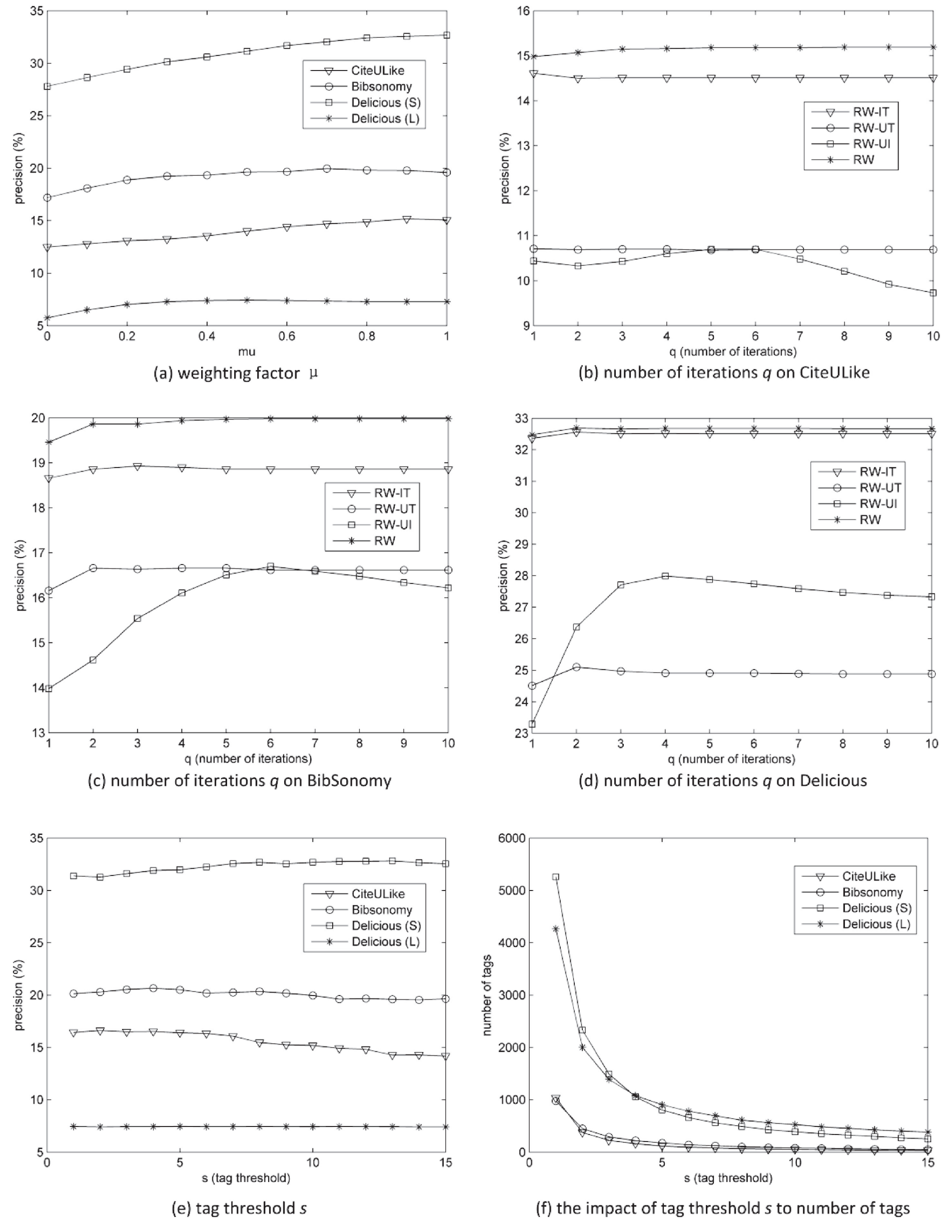
Fig. 6. Sensitivity analysis of parameters $\alpha, \beta, \lambda, \eta$.

was fairly small irrespective of how the user graph related parameters β and λ were changed.

Weighting Factor η. In Figure 6(d), we can observe that the precision when η equaled 0 was lower than the precision when η was between 0 and 0.5. The reason was that the item-tag information was unused when η equaled 0. However, when η surpassed 0.5, the performance began to decline with subsequent increases in η. The results imply that while transitive associations play a critical role in the overall performance, excessive usage of these associations can diminish the accuracy of item recommendations.

Weighting Factor μ. As shown in Figure 7(a), the precision when μ surpassed 0.5 was higher than the precision when μ was less than 0.5. This indicates that the contribution of the item graph to the final performance was greater than that of the user graph.

Number of Iterations q. Figure 7(b)$\sim$(d) correspond to the experimental results on the CiteULike, Bibsonomy and Delicious (small) datasets in turn. It is not significant for the variations of the performance of RW-IT and RW-UT with the increase of the number of iterations. However, it is obvious for the impact of the number of iterations to the performance of RW-UI, and RW reaches the highest value with q ranging from 4 to 7. This could be due to the fact that the IT and UT matrices were denser than the UI matrix. For example, the proportion of nonzero elements in the UI, IT, and



(a) weighting factor μ
(b) number of iterations q on CiteULike
(c) number of iterations q on BibSonomy
(d) number of iterations q on Delicious
(e) tag threshold s
(f) the impact of tag threshold s to number of tags

Fig. 7. Sensitivity analysis of parameters μ, q, and s.

UT matrices in the CiteULike dataset were 0.010, 0.034, and 0.033, respectively. In addition, when q surpassed 6, the performance began to decline with subsequent increases in the value of q. This finding implies that the lengthier multi-step transitive associations may not be helpful for the recommendation performance. As we discussed in Section 3.1, transitive associations can make the inter-item and inter-user similarity measures more accurate, which facilitates the alleviation of sparsity. However, considering that lengthier transitive associations are weighted lower, such associations provide limited improvement to the inter-item and inter-user similarity measures. Based on the results, while shorter transitive associations are beneficial, lengthier multi-step transitive associations do not appear to improve recommendation performance.

Tag Threshold s. As shown in Figure 7(e), the impact of the tag threshold on performance was small. However, Figure 7(f) shows that the number of tags dramatically declined as the tag threshold increased. This implies that selecting tag threshold that reduces the number of tags can save computational resources, including time and space, without having a significant adverse effect on precision.

5. CONCLUSIONS

In this study, we proposed a novel random-walk-based recommendation model for social tagging systems. This approach can effectively improve the recommendation performance and alleviate the data sparsity problem by leveraging the transitive associations among the transaction records available as <user,tag,item> tuples. Furthermore, an empirical evaluation on three real-world datasets showed that our approach outperformed existing methods under sparse data, largely due to its ability to better capture the transitive associations between users, items, and tags. Additional experiments showed that the probability-based similarity mechanism proposed in this study outperformed the cosine similarity method commonly adopted in prior work. Through sensitivity analyses, we found that user-item, user-tag, and item-tag interaction information had different kinds of impact on recommendation performance.

Social tagging has become a useful and popular method for organizing and sharing information in social media applications. Improving tag-based recommendation can alleviate the information overload problem.

In future research, we plan to incorporate social network information into our model and evaluate their impact on recommendation performance, since social network provides a valuable resource about the connections between users. We also plan to apply other methods (e.g., associative retrieval techniques, network analysis approach etc.) [Huang et al. 2004; Wei and Ram 2012] to explore the transitive associations among <user,tag,item> transaction data and study the impact of data characteristics on recommendation performance in social tagging systems [Adomavicius and Zhang 2012]. Another research direction is to explore alternatives for implementing the random-walk-based model in a Big Data environment. In addition, we hope to evaluate the top-N recommendation results of our model against practically-relevant metrics such as novelty, diversity, and serendipity [Herlocker et al. 2004], which have begun to draw considerable attention in the fields of recommender systems and information retrieval.

REFERENCES

Adomavicius, G. and Tuzhilin, A. 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. Knowl. Data Eng. 17*, 6, 734–749.

Adomavicius, G. and Zhang, J. 2012. Impact of data characteristics on recommender systems performance. *ACM Trans. Manage. Inf. Syst. 3*, 1, 1–17.

Balabanović, M. and Shoham, Y. 1997. Fab: Content-based, collaborative recommendation. *Commun. ACM 40*, 3, 66–72.

Basu, C., Hirsh, H., and Cohen, W. 1998. Recommendation as classification: Using social and content-based information in recommendation. In *Proceedings of the National Conference on Artificial Intelligence*. 714–720.

Bellogin, A., Castells, P., and Cantador, I. 2011. Self-adjusting hybrid recommenders based on social network analysis. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1147–1148.

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

Breese, J. S., Heckerman, D., and Kadie, C. 1998. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*. 43–52.

Burke, R. 2002. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction 12*, 4, 331–370.

Cai, Y., Zhang, M., Luo, D., Ding, C., and Chakravarthy, S. 2011. Low-order tensor decompositions for social tagging recommendation. In *Proceedings of the 4th ACM International Conference on Web Search and Data Mining*. 695–704.

Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D., and Sartin, M. 1999. Combining content-based and collaborative filters in an online newspaper. In *Proceedings of ACM SIGIR Workshop on Recommender Systems*.

Deshpande, M. and Karypis, G. 2004. Item-based top-n recommendation algorithms. *ACM Trans. Inf. Syst. 22*, 1, 143–177.

Fouss, F., Yen, L., Pirotte, A., and Saerens, M. 2006. An experimental investigation of graph kernels on a collaborative recommendation task. In *Proceedings of the IEEE International Conference on Data Mining*. 863–868.

Fouss, F., Pirotte, A., Renders, J. M., and Saerens, M. 2007. Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *IEEE Trans. Knowl. Data Eng. 19*, 3, 355–369.

Ghazanfar, M. A. and Prugel-Bennett, A. 2010. A scalable, accurate hybrid recommender system. In *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*. 94–98.

Goldberg, D., Nichols, D., Oki, B. M., and Terry, D. 1992. Using collaborative filtering to weave an information tapestry. *Commun. ACM 35*, 12, 61–70.

Good, N., Schafer, J. B., Konstan, J. A., Borchers, A., Sarwar, B., Herlocker, J., and Riedl, J. 1999. Combining collaborative filtering with personal agents for better recommendations. In *Proceedings of the National Conference on Artificial Intelligence*. 439–446.

Gori, M. and Pucci, A. 2007. ItemRank: A random-walk based scoring algorithm for recommender engines. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*. 2766–2771.

Guan, Z., Wang, C., Bu, J., Chen, C., Yang, K., Cai, D., and He, X. 2010. Document recommendation in social tagging services. In *Proceedings of the 19th International Conference on World Wide Web*. 391–400.

Gunawardana, A. and Meek, C. 2009. A unified approach to building hybrid recommender systems. In *Proceedings of the 3rd ACM Conference on Recommender Systems*. 117–124.

Guy, I., Zwerdling, N., Ronen, I., Carmel, D., and Uziel, E. 2010. Social media recommendation based on people and tags. In *Proceedings of the 33rd International ACM SIGIR conference on Research and Development in Information Retrieval*. 194–201.

Herlocker, J. L., Konstan, J. A., Terveen, L. G., and Riedl, J. T. 2004. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst. 22*, 1, 5–53.

Hofmann, T. 2003. Collaborative filtering via gaussian probabilistic latent semantic analysis. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*. 259–266.

Hofmann, T. 2004. Latent semantic models for collaborative filtering. *ACM Trans. Inf. Syst. 22*, 1, 89–115.

Hotho, A., J Schke, R., Schmitz, C., and Stumme, G. 2006. Information retrieval in folksonomies: Search and ranking. In *Proceedings of the 3rd European Conference on the Semantic Web: Research and Applications*. 411–426.

Huang, Z., Chen, H., and Zeng, D. 2004. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Trans. Inf. Syst. 22*, 1, 116–142.

Jäschke, R., Marinho, L., Hotho, A., Schmidt-Thieme, L., and Stumme, G 2008. Tag recommendations in social bookmarking systems. *AI Commun. 21*, 4, 231–247.

Jamali, M. and Ester, M. 2009. 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*. 397–406.

Jeon, W., Cho, S., Cha, J., and Byun, H. 2011. Tag suggestion using visual content and social tag. In *Proceedings of the 5th International Conference on Ubiquitous Information Management and Communication*. 1–5.

Jiang, B., Ling, Y., and Wang, J. 2010. Tag Recommendation Based on Social Comment Network. *J. Digital Content Technol. Appli. 4*, 8, 110–117.

Jin, R., Chai, J. Y., and Si, L. 2004. An automatic weighting scheme for collaborative filtering. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. 337–344.

Konstas, I., Stathopoulos, V., and Jose, J. M. 2009. On social networks and collaborative recommendation. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 195–202.

Li, X., Snoek, C. G. M., and Worring, M. 2008. Learning tag relevance by neighbor voting for social image retrieval. In *Proceedings of the 1st ACM International Conference on Multimedia Information Retrieval*. 180–187.

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

Liu, K., Fang, B., and Zhang, W. 2010. Speak the same language with your friends: Augmenting tag recommenders with social relations. In *Proceedings of the 21st ACM Conference on Hypertext and Hypermedia*. 45–50.

Nanopoulos, A., Rafailidis, D., Symeonidis, P., and Manolopoulos, Y. 2010. MusicBox: Personalized music recommendation based on cubic analysis of social tags. *Trans. Audio, Speech and Lang. Proc. 18*, 2, 407–412.

Ma, H., King, I., and Lyu, M. R. 2007. Effective missing data prediction for collaborative filtering. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. 39–46.

Ma, H., Zhou, T. C., Lyu, M. R., and King, I. 2011. Improving recommender systems by incorporating social contextual information. *ACM Trans. Inf. Syst. 29*, 2, 9.

Marinho, L. B., Hotho, A., J Schke, R., Nanopoulos, A., Rendle, S., Schmidt-Thieme, L., Stumme, G., and Symeonidis, P. 2012. *Recommender Systems for Social Tagging Systems.* Springer, New York.

Page, L., Brin, S., Motwani, R., and Winograd, T. 1999. The PageRank citation ranking: Bringing order to the web.

Pazzani, M. J. 1999. A framework for collaborative, content-based and demographic filtering. *Artif. Intell. Rev. 13*, 5, 393–408.

Peng, J., Zeng, D., Liu, B., and Zhao, H. 2010a. CFUI: Collaborative filtering with unlabeled items. In *Proceedings of the 20th Workshop on Information Technologies and Systems.*

Peng, J., Zeng, D. D., Zhao, H. and Wang, F. 2010b. Collaborative filtering in social tagging systems based on joint item-tag recommendations. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*. 809–818.

Peng, J., Zeng, D. D., and Huang, Z. 2011. Latent subject-centered modeling of collaborative tagging: An application in social search. *ACM Trans. Manage. Inf. Syst. 2*, 3, 1–23.

Popescul, A., Pennock, D. M., and Lawrence, S. 2001. Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. In *Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence*. 437–444.

Rendle, S., Balby Marinho, L., Nanopoulos, A., and Schmidt-Thieme, L. 2009. Learning optimal ranking with tensor factorization for tag recommendation. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 727–736.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J. 1994. GroupLens: An open architecture for collaborative filtering of netnews. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work*. 175–186.

Salakhutdinov, R. and Mnih, A. 2008. Probabilistic matrix factorization. *Adv. Neural Inf. Proc. Syst. 20*. 1257–1264.

Sarwar, B., Karypis, G., Konstan, J., and Reidl, J. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web*. 285–295.

Si, L. and Jin, R. 2003. Flexible mixture model for collaborative filtering. In *Proceedings of the 20th International Conference on Machine Learning (ICML'03).*

Soboroff, I. and Nicholas, C. 1999. Combining content and collaboration in text filtering. In *Proceedings of the International Joint Conference on Artificial Intelligence Workshop: Machine Learning for Information Filtering*.

Su, X. and Khoshgoftaar, T. M. 2009. A survey of collaborative filtering techniques. *Adv. Artif. Intell. 2*, 2.

Symeonidis, P., Nanopoulos, A., and Manolopoulos, Y. 2010. A unified framework for providing recommendations in social tagging systems based on ternary semantic analysis. *IEEE Trans. Knowl. Data Eng. 22*, 2, 179–192.

Tso-Sutter, K. H. L., Marinho, L. B., and Schmidt-Thieme, L. 2008. Tag-aware recommender systems by fusion of collaborative filtering algorithms. In *Proceedings of the ACM Symposium on Applied Computing*. 1995–1999.

Vipul, V. 2012. Hybrid recommender systems: Content-boosted collaborative filtering for improved recommendations. In *Proceedings of the International Conference on Communication Systems and Network Technologies*. 649–653.

Wang, J., De Vries, A. P., and Reinders, M. J. T. 2006. Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. 501–508.

Wei, W. and Ram, S. 2012. Using a network analysis approach for organizing social bookmarking tags and enabling web content discovery. *ACM Trans. Manage. Inf. Syst. 3*, 3, 1–16.

Wetzker, R., Umbrath, W., and Said, A. 2009. A hybrid approach to item recommendation in folksonomies. In *Proceedings of the Workshop on Exploiting Semantic Annotations in Information Retrieval (WSDM'09)*. 25–29.

Yildirim, H. and Krishnamoorthy, M. S. 2008. A random walk method for alleviating the sparsity problem in collaborative filtering. In *Proceedings of the ACM Conference on Recommender Systems*. 131–138.

Zeng, D. and Li, H. 2008. How useful are tags?—An empirical analysis of collaborative tagging for Web page recommendation. In *Proceedings of the IEEE ISI PAISI, PACCF, and SOCO International Workshops on Intelligence and Security Informatics*. 320–330.

Zhang, Z. K., Zhou, T., and Zhang, Y. C. 2010. Personalized recommendation via integrated diffusion on user-item-tag tripartite graphs. *Physica A: Stat. Mech. Its Appl. 389*, 1, 179–186.

Zhao, S., Du, N., Nauerz, A., Zhang, X., Yuan, Q., and Fu, R. 2008. Improved recommendation based on collaborative tagging behaviors. In *Proceedings of the 13th International Conference on Intelligent User Interfaces*, 413–416.

Zhen, Y., Li, W. J., and Yeung, D. Y. 2009. TagiCoFi: Tag informed collaborative filtering. In *Proceedings of the 3rd ACM Conference on Recommender Systems*. 69–76.

Zheng, N. and Li, Q. 2011. A recommender system based on tag and time information for social tagging systems. *Expert Syst. Appl. 38*, 4, 4575–4587.

Received April 2012; revised November 2012, March 2013; accepted May 2013