FolkRank++: An Optimization of FolkRank Tag Recommendation Algorithm Integrating User and Item Information

Jianli Zhao[1*], Qinzhi Zhang[1], Qiuxia Sun[2], Huan Huo[3], Yu Xiao[1], and Maoguo Gong[1,4*]
[1] College of Computer Science & Engineering
Shandong University of Science and Technology, Qingdao, China
[e-mail: jlzhao@sdust.edu.cn, 779239367@qq.com, 540593814@qq.com]
[2] College of Mathematics and Systems Science
Shandong University of Science and Technology, Qingdao, China
[e-mail: qiuxiasun@sdust.edu.cn]
[3] Faculty of Engineering and Information
Technology University of Technology, Sydney, Australian
[e-mail: Huan.Huo@uts.edu.au]
[4] Key Laboratory of Intelligent Perception and Image Understanding
Ministry of Education, Xidian University, Xi'an, China
[e-mail: mggong@mail.xidian.edu.cn]
*Corresponding author : Jianli Zhao, Maoguo Gong

Received November 25, 2019; revised October 22, 2020; accepted December 19, 2020; published January 31, 2021

Abstract

The graph-based tag recommendation algorithm FolkRank can effectively utilize the relationships between three entities, namely users, items and tags, and achieve better tag recommendation performance. However, FolkRank does not consider the internal relationships of user-user, item-item and tag-tag. This leads to the failure of FolkRank to effectively map the tagging behavior which contains user neighbors and item neighbors to a tripartite graph. For item-item relationships, we can dig out items that are very similar to the target item, even though the target item may not have a strong connection to these similar items in the user-item-tag graph of FolkRank. Hence this paper proposes an improved FolkRank algorithm named FolkRank++, which fully considers the user-user and item-item internal relationships in tag recommendation by adding the correlation information between users or items. Based on the traditional FolkRank algorithm, an initial weight is also given to target user and target item's neighbors to supply the user-user and item-item relationships. The above work is mainly completed from two aspects: (1) Finding items similar to target item according to the attribute information, and obtaining similar users of the target user according to the history behavior of the user tagging items. (2) Calculating the weighted degree of items and users to evaluate their importance, then assigning initial weights to similar items and users. Experimental results show that this method has better recommendation performance.

This paper is supported by the National Key R&D Plan (No. 2018YFC0831002, No.2017YFC0804406), Humanity and Social Science Fund of the Ministry of Education (No. 18YJAZH136, No. 17YJCZH262), National Natural Science Foundation (No. 62072288).

http://doi.org/10.3837/tiis.2021.01.001 **ISSN : 1976-7277**

Keywords: Tag Recommendation, Initial Weight, Personalized Recommendation, FalkRank

1. Introduction

With the rapid development of network technology, the Internet has become the main platform for the manufacture, dissemination and processing of information data, and the data scale has expanded dramatically. Redundant information affects users to find the information they need, makes information selection difficult, which is called information overload [1]. In order to alleviate the information overload, the recommendation system came into being. The recommendation system can model users' interests by analyzing their historical behaviors without specific requirements provided by users, so as to actively recommend information that may meet their interests and needs. With the rapid development of Web 2.0 technology, tags have become an important information for network users to share and store network resources. The tag represents the users' preferences and the attributes of the items [2]. If a reasonable tag is recommended to users, they will be able to manage the items based on the tag. It is helpful for users to find and manage the information they are interested in among the numerous network resources by providing reasonable tags for users. In fact, users are often unable to find appropriate tags when tagging items and recommending tags to them can reduce such trouble and improve the efficiency of users' online browsing. As a result, users have improved user experience with the system. Therefore, it is crucial to look for potential interest tags for users.

Hotho et al. [3] proposed a graph-based FolkRank tag recommendation algorithm, which can achieve a better tag recommendation performance. The FolkRank tag recommendation algorithm effectively obtains the relationships between users, items and tags (user-item, user-tag, item-tag), which can be reflected by a tripartite graph. However, the FolkRank algorithm ignores the relationship between user-user, item-item, which is reflected in the adjacency matrix diagonal element as zero. This results in a problem: FolkRank is completely unable to use the tagging behavior of users who are extremely similar to the target user. This will have a non-negligible impact on the recommendation accuracy in the recommendation system, and it needs to be solved urgently.

In order to solve this limitation, we propose an improved tag recommendation algorithm by supplying the relationship between user-user and item-item on the FolkRank algorithm. The main idea of FolkRank tag recommendation algorithm is giving an initial weight to the target user and target item. And the weight between target users or items and their neighbors, if directly imitating the initialization method of FolkRank, there will be a lack of correlation between users or items, which has become another motivation of our work. Based on the traditional FolkRank algorithm, we give these neighbors a more accurate initial weight based on the similarity calculation and propose the FolkRank++ algorithm.

For this paper, the main contributions are as follows:

(1) In order to supply the relationship of user-user and item-item that FolkRank algorithm lacks, we accurately get the neighbors of target users and target items. Items that have exactly the same attribute information as the target item can be considered as similar

items, at the same time, similar users are obtained according to the historical tagging behavior.

(2) In order to more accurately describe the relationship of user-user, item-item, we give the neighbors of the target user and target item an initial weight. We first calculate the weighted degree of items to evaluate the importance of items and give initial weights to similar items according to the importance, then assign initial weights to the similar users according to their historical tagging behavior.

The rest of this paper is structured as follows: Firstly, related research work is discussed in Section 2. Section 3 describes the proposed algorithm in detail. Then the experiment is analysed in Section 4. Section 5 summarizes the work of this paper.

2. Related Work

With the vigorous development of machine learning, more and more algorithms are playing their role in big data processing [4-6]. Among them, the recommendation system has achieved great success in information filtering [7-9] and tag recommendation can be used to describe the user's preference and item's characteristics more efficiently. In the tag system, the user can freely add a description to the online item according to personal cognition, generally in the form of a phrase or keyword data. In today's multimedia era when the amount of network data explodes rapidly, tag can effectively filter data and help users find useful information. At present, the existing tag prediction methods can be summarized as collaborative filtering method, graph-based method and content-based method:

(1) Collaborative filtering is a common technique in recommendation systems. The traditional collaborative filtering method has a ternary relationship among users, items and tags. Only when the ternary relationship is reduced to a lower dimensional space can it be directly applied. Jäschke et al. [10] proposed a tag recommendation algorithm based on popularity. (2) The basic idea of the graph-based approach is to construct graphs with users, items, and tags as vertices based on the user's tagging behavior, and build edges (Liu et al.) [11], this method does not need to consider the content of the items and the semantic information of the tags. (3) Content-based methods typically use the content of the items and employ machine learning algorithms to recommend tags. Feng and Wang [12] proposed an optimization framework to learn the optimal feature weights by maximizing the average area under the tag recommender curve.

Based on the above three baselines, Jäschke et al. [13] compared several methods, including improved PageRank, FolkRank, CFS and sorting algorithm based on collaborative filtering, found evidence that graph-based algorithm FolkRank provides the best performance. In view of the problem that the graph-based tag recommendation algorithm cannot effectively represent the extra information of the data set, and is not sensitive to spatiotemporal information, many scholars have improved the FolkRank algorithm. Among them, Xiance et al. [14] proposed an improved algorithm of the FolkRank algorithm, which combines the Pop-Tag algorithm and the FolkRank algorithm. Gemmell et al. [15] and Zhang et al. [16] proposed a hybrid recommendation model, in which the tag ranking is calculated by the weighted sum of the FolkRank score and the item-based collaborative filtering score. Landia et al. [17] proposed Content-FolkRank to solve the cold start problem by combining FolkRank with text Content. Yamasaki et al. [18] proposed the recommendation algorithm of FolkPopularityRank, which can recommend tags that can improve social popularity. Riaz et al. [19] proposed a TimeFolkRank tag recommendation algorithm, which takes into account the time information of tags and assigns a greater weight to newer tags.

The graph-based tag recommendation algorithm FolkRank has always achieved a high tag recommendation accuracy because it can make better use of the valid information in the user-item-tag graph, but there are still some problems with this method. For example, the FolkRank tag recommendation algorithm does not effectively utilize the information of the user-user, item-item, tag-tag, which is reflected in the adjacency matrix diagonal element as zero. The FolkRank++ algorithm can make better use of the relationship of user-user and item-item, and improve the accuracy of tag recommendation. Therefore, this research is meaningful.

3. FolkRank++ Tag Recommendation Algorithm

In this section, we introduce FolkRank tag recommendation algorithm and then introduce the main idea for the optimization of FolkRank tag recommendation algorithm. First, finding the neighbors of the target user and item separately; next, the initial weights are given to the neighbor users and items respectively.

3.1 Related Parameter Definition

The tagging system includes users, items, and tags, and the user's tagging behavior reflects the relationship between them. The behavior of the user tagging the item can be seen as an undirected tripartite graph: $G_{UIT} = (U \cup I \cup T, E)$, the U, I, T represents the set of users, items and tags, respectively. E represents the ternary relationship among users, items and tags, i.e $E \subseteq U \times I \times T$.

For a given user and item pair (u, i), the correlation between the tag and user u and item i needs to be calculated. A higher correlation means that the user u is more likely to use this tag for the item i, then the most relevant tag can be recommended to the user. **Table 1** summarizes some of the mathematical implications of the tag recommendation graph model algorithm.

Table 1. Mathematical definitions in the label recommendation graph model

Definition	Description
E	A ternary relationship of users, items, and tags
A	Graph model adjacency matrix
$\tilde{A}$	The column random matrix of the adjacency matrix
$\vec{p}$	Preference vector
G_{UIT}	The undirected tripartite graph
M_{UT}, M_{UI} , M_{TI}	User-tag matrix, user-item matrix and tag-item matrix
d	A damping coefficient
$\vec{w}_0, \vec{w}_1$	Global sorting vector, personalized sorting vector
q_i, N_{q_i}	The probability that the user u uses the tag t_i, the number of times the user u uses the tag t_i
$N(v, i)$	The number of times the neighboring user v has tagged the target item i or the neighbors of the target item i
$ID(i), OD(i)$	Degree of entry, degree of out in the tripartite graph
α, β	Threshold to control whether similar users or items are given weights

3.2 FolkRank Tag Recommendation Algorithm

The main idea of the page sorting algorithm PageRank [20] is that if there are many web pages connected to a web page, and if the web pages themselves are important, then the web page is important. The FolkRank algorithm improves the PageRank tag recommendation algorithm. Hotho et al. [3] adopted Google's search ranking principle in folksonomy and proposed the FolkRank tag recommendation algorithm. In this subsection, we will briefly introduce the principle of the algorithm and explain how it can be used to construct tag recommendations.

G_{UIT} is used to represent the undirected tripartite graph, i.e., $G_{UIT} = (U \cup I \cup T, E)$. The nodes in the tripartite graph can be divided into three disjoint sets, namely user set U, item set I and tag set T. Graph G_{UIT} can be expressed as an adjacency weighted matrix A, as shown in the below:

$$A = \begin{Bmatrix} 0_{UU} & M_{UT} & M_{UI} \\ M_{TU} & 0_{TT} & M_{TI} \\ M_{IU} & M_{IT} & 0_{II} \end{Bmatrix} \tag{1}$$

Where M_{UT} stands for user-tag matrix, M_{UI} stands for user-item matrix, M_{TI} stands for tag-item matrix. Taking the M_{UT} matrix as an example, the element at the (1,1) position of the matrix represents the number of times $user_1$ uses tag_1. 0_{UU}、0_{TT}、0_{II} represents the zero matrix of $|U| \times |U|, |T| \times |T|, |I| \times |I|$, because the user-user, item-item, tag-tag is not directly connected in the undirected tripartite graph. Other than that, M_{TU}、M_{IU}、M_{IT} is the transposed matrix of M_{UT}、M_{UI}、M_{TI}. In general, a vector of sorting $\vec{w}$ (column vector) is defined by the following equation:

$$\vec{w} = d\tilde{A}\vec{w} + (1-d)\vec{p} \tag{2}$$

Where $d \in (0,1)$ is a damping coefficient, $\tilde{A}$ is a column random matrix of the adjacency matrix A. $\vec{p}$ is an initial sorting vector, the dimension of $\vec{p}$ is $|U| + |I| + |T|$. After iterating, we can get the final $\vec{w}$.

FolkRank adds a personalized preference vector calculation. Given a user $u \in U$ and a given item $i \in I$, the FolkRank sorting vector calculation equation is as follows:

$$\vec{f} = \vec{w}_1 - \vec{w}_0 \tag{3}$$

For the (u,i) pair to be recommended, $\vec{w}_1$ is the personalized sorting vector obtained by the non-uniform vector $\vec{p}$, where the corresponding element of (u,i) sets a larger initial weight $(\vec{p}(u) = 1 + |U|, \vec{p}(i) = 1 + |I|)$, where $\vec{p}(u)$ represents the weight corresponding to the target user u in the vector $\vec{p}$, and $|U|$ represents the number of users, $\vec{p}(i)$ represents the weight corresponding to the target item i in the vector $\vec{p}$, and $|I|$ represents the number of items, and the remaining elements are set to 1. After giving the target user and item pairs a better weight by spreading in the user-item-tag graph, the tags associated with them will also receive a relatively large weight, which helps to obtain the tags associated with them. $\vec{w}_0$ is a global sorting vector, which is obtained by the uniform vector $\vec{p} = \vec{1}$, and $\vec{f}$ is the FolkRank sorting vector. For the user-item pair (u,i) to be recommended, $\vec{f}$ is the sorting vector of all the tags in the tag set.

Each entity in $\vec{f}$ represents the weight value of a certain tag in the recommendation prediction process. By sorting these weight values from large to small, we select the Top-N tags to recommend to the user-item pair (u, i) and achieve the purpose of tag recommendation.

3.3 FolkRank++ Tag Recommendation Algorithm

The FolkRank++ algorithm can supply the relationship of user-user and item-item based on the FolkRank algorithm. The main idea of the algorithm is similar to the traditional FolkRank tag recommendation algorithm: An initial weight is given to the target user and target item. This paper also gives the neighbors of the target item and user an initial weight to utilize the correlation information between them.

Due to target items may not have a strong connection with their similar items in the graph, giving these similar items an initial weight can effectively capture the commonly used tag set of them, which complements the missing item-item relationship in the graph, and the same is true for users.

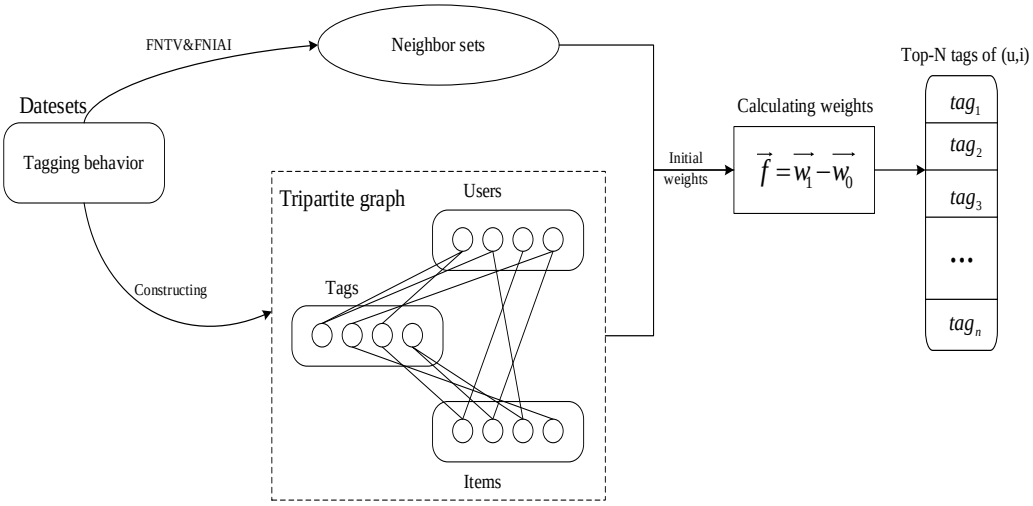

Fig. 1. The process of FolkRank++

Specifically, supposing there are 3 users, 4 items, 3 tags. We currently need to recommend tags for $(user_1, item_1)$. According to (2) and (3), the traditional FolkRank algorithm sets the non-uniform vector $\vec{p}$ as follows: $\vec{p}(u) = 1 + |U|, \vec{p}(i) = 1 + |I|$, and the remaining elements are set to 1, so $\vec{p} = (u_1, u_2, u_3, i_1, i_2, i_3, i_4, t_1, t_2, t_3) = (4,1,1,5,1,1,1,1,1,1)$. The FolkRank++ algorithm also assigns an initial weight to the neighbors of $user_1$ and $item_1$, respectively. For example, the neighbor of $user_1$ is $user_2$, and the neighbor of $item_1$ is $item_3$, so $\vec{p} = (u_1, u_2, u_3, i_1, i_2, i_3, i_4, t_1, t_2, t_3) = (4,1.5,1,5,1,2.5,1,1,1,1)$. The structure and process of FolkRank++ are shown in **Fig. 1**. Next, we will introduce how to find neighbors and how to give neighbors initial weights.

3.3.1 Get the Neighbors of Target User and Item

In order to accurately find the neighbors of target user and item, this subsection mainly introduces two methods to find the neighbors: One is to find neighbors based on tag vector (FNTV); The other is to find neighbors based on item attribute information (FNIAI).

The cosine similarity calculation method based on the tag vector is more biased to find the nearest neighbor by the historical behavior of the user's tagging, and the method of finding the nearest neighbor based on the attribute information of the item is considered from the attribute information dimension of the item unilaterally. Take the movie as an example. When the attributes of two movies are horror movies, they are considered to be similar. The two methods have different emphases, so we choose between them in different situations.

(1) Find neighbors based on tag vector (FNTV)

This method finds neighbors by calculating the cosine similarity based on the tag vector. It is assumed that there are N tags, and each user u has an N-dimensional vector $u = (q_1, q_2, \ldots, q_N)$, where the element q_i is represented as follows:

$$q_i = \frac{N_{q_i}}{N_{q_1}+N_{q_2},\ldots,+N_{q_N}} \tag{4}$$

In the above equation, N_{q_i} represents the number of times the user u uses the tag t_i, q_i represents the probability that the user u uses the tag t_i. The cosine similarity of users u and v is defined as follows:

$$sim(u,v) = \frac{\sum_{i=1}^{N} u_{q_i} \times v_{q_i}}{\sqrt{\sum_{i=1}^{N}(u_{q_i})^2} \times \sqrt{\sum_{i=1}^{N}(v_{q_i})^2}} \tag{5}$$

Similarly, the cosine similarity of items i and j is defined as follows:

$$sim(i,j) = \frac{\sum_{i=1}^{N} i_{q_i} \times j_{q_i}}{\sqrt{\sum_{i=1}^{N}(i_{q_i})^2} \times \sqrt{\sum_{i=1}^{N}(j_{q_i})^2}} \tag{6}$$

(2) Find neighbors based on item attribute information (FNIAI)

The procedure of using the item attribute information to find the neighbors are mainly divided into the following two steps: First, the items with similar attribute information of the target item are defined as similar items; Further, similar users of the target user are obtained based on the user's historical behavior.

First, finding the similar items of the target item. In this case, the attribute information of the item (movie) in the MovieLens dataset is analyzed. The attribute information of each item is represented by an 18-dimensional vector (Horror, Adventure, Comedy, Action, …, Animation, Children, Comedy, Fantasy). The dimension of a vector is the number of item (movie) types. It is important to note that the attribute information here is not a set of tags to be recommended. For an item, if there is the attribute, the vector element is 1, otherwise it is 0. The target user is u_1, and the target movie is i_1. We assume that the attribute information vector $\vec{i}_1$ of item i_1 is (Horror, Comedy, Action), the attribute information vector $\vec{i}_2$ of item i_2 is (Horror, Comedy, Action). The item i_2 has exactly the same attribute information as the target item i_1, attribute information vector $\vec{i}_1 = \vec{i}_2 = (1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0)$. The cosine similarity is used to obtain the similarity between the item i_1 and the item i_2, here the similarity between the item i_2 and the target item i_1 is 1.0.

Further, similar users of the target user are obtained based on the user's historical behavior. We hold the opinion that the users who have tagged the target item or the neighbor items of the target item have a similar degree to the target user. Supposing that all the records for tagging the target item $i_1$or the neighbor item i_2 of the target item are shown in **Table 2** below:

Table 2. Tag Recording

User ID	Item ID	Tag ID
u_1	i_1	t_1
u_2	i_1	t_2
u_2	i_2	t_2
u_2	i_2	t_3
u_3	i_2	t_3
u_4	i_2	t_4

User u_2 has tagged item i_1 and i_2 three times in total, and user u_3 and user u_4 have tagged item i_2 once. Then the user u_2、u_3、u_4 can be considered as the neighbors of the target user, and the similarity calculation will be introduced in the next subsection.

3.3.2 Assigning Initial Weights to Neighbors

This subsection describes how to assign an initial weight to the neighbors. In the FolkRank algorithm, the initial weight of the target user and the target item is set to a large value. The value is generally set to the number of users or the number of items. FolkRank++ algorithm is similar to the FolkRank algorithm, it gives the neighbors an initial weight that complements the user-user, item-item relationship.

There are two methods to assign the initial weight: The first method is to assign the initial weight according to the cosine similarity mentioned in the previous subsection; The second method is to calculate the weighted degree of item nodes as the index of item importance, the initial weight is given to similar items according to the importance degree, then the initial weight is given to similar users according to the user's historical tagging behavior. The two methods are introduced below.

(1) Assign the initial weight according to the cosine similarity

For the first method (FNTV) proposed in section 3.3.1, we use the following strategy to give the neighbor an initial weight.

In the FolkRank algorithm, the initial weight of the user-item pair to be recommended is: $\vec{p}(u) = 1 + |U|, \vec{p}(i) = 1 + |I|$, and the initial weight of the neighbor is assigned:

$$\vec{p}(v) = sim(u, v) * \vec{p}(u) \tag{7}$$

$$\vec{p}(j) = sim(i, j) * \vec{p}(i) \tag{8}$$

Where v is the neighbor of the target user u , j is the neighbor of the target item i , $sim(u, v)$ is obtained by the (5), representing the similarity between user u and user v, $sim(i, j)$ is obtained by the (6), representing the similarity between item i and item j, and the neighbor whose similarity is greater than a certain threshold α is selected to be given the initial weight. Assume that $user_2$ and $user_3$ have similarity with target $user_1$ of 0.9 and 0.6, respectively, and the threshold value α is 0.8, then $user_2$ will eventually become the neighbor of $user_1$ and obtain an initial weight.

For the second method (FNIAI) proposed in section 3.3.1, we use the following strategy to give the neighbor an initial weight.

For the method of obtaining the neighbors of the target item by using the attribute information, assign initial weights to similar items j according to (8).

Where j is the neighbor of the target item, $sim(i,j)$ represents the cosine similarity between item i and item j based on attribute information, and the neighbor whose similarity is greater than a certain threshold β is selected to be given the initial weight.

Similarly, it can be seen from **Table 2** that user u_2 has tagged item i_1 and i_2 three times in total, and user u_3 and user u_4 only tag item i_2 once, so the initial weight of the neighbors of the target user is given as follows:

$$\vec{p}(v) = \frac{N(v,i)}{\sum_{v \in U(j)} N(v,i)} \vec{p}(u) \tag{9}$$

Where $N(v,i)$ is the number of times the neighboring user v has tagged the target item i or the neighbors of the target item i. $U(j)$ is the neighbor user set, and represents a collection of users that have tagged the target item i or the neighbor item j of the target item i.

(2) Assign the initial weight according to the weighting degree of the item

The second method calculates the weighting degree of the item nodes and performs normalization operation as the item importance index, and assigns the initial weight to the similar items according to the item importance index. The equation for calculating the importance of the item node in the figure is as follows:

$$T(i) = ID(i) + OD(i) \tag{10}$$

Where $ID(i)$ represents the degree of entry and $OD(i)$ represents the degree of out. Since the graph is undirected, the degree of entry and the degree of out cannot be calculated, the sum of the weights of the corresponding edges between the nodes connected to the item node is taken as the importance of the item node. For example, the weight of the corresponding edge of the item node i and the user node u in the graph represents the number of times the user u tags the item i, then we normalize the importance degree of all item nodes and use it as the basis to assign weight to each similar item. Assuming that the node importance of similar item i_1 is 0.3, the weight should be given as: $0.3 * (1 + |I|)$.

For the first method (FNTV) proposed in section 3.3.1, we use the following strategy to give the neighbor an initial weight.

First, we give an initial weight to similar items using (10).

Then, the initial weight is further given to the similar users according to (7): $\vec{p}(v) = sim(u,v) * \vec{p}(u)$.

For the second method (FNIAI) proposed in section 3.3.1, we use the following strategy to give the neighbor an initial weight.

First, we give an initial weight to similar items using (10).

Then, the initial weight is further given to the similar users according to the user's historical behavior. The method is the same as that described in (9), and is not introduced here.

Since it is based on the work performed by FolkRank, the iteration time complexity of FolkRank++ is the same as its order of magnitude, but this work spends extra overhead on establishing the relationship between user-user, item-item and tag-tag, and assigns the initial weight of each neighbors in the tripartite graph according to the cosine similarity, which consumes a certain amount of time cost to complete the accuracy improvement.

The flow of the FolkRank++ tag recommendation algorithm is shown in **Table 3**.

Table 3. FolkRank++ algorithm Flow

Algorithm 1: FolkRank++ tag recommendation algorithm

Input: The graph information of the training file, i.e., $G_{UIT} = (V, E)$ where $V = T \cup I \cup U$ and $E = \{\{u,t\},\{t,i\},\{u,i\}|\{i,t,u\}\} \in Y\}$, the adjacency matrix A, the given item i and the given user u.
Output: The ranking of all tags
//Initialize
1. for each $t \in T, i \in I, u \in U$ do
2. $w_0[t] = w_1[t] = w_0[i] = w_1[i] = w_0[u] = w_1[u] = 1$
3. $p[t] = p[i] = p[u] = 1$
4. end
5. adjust parameters
//Iteration for w_0
6. repeat
7. $w_0 = dAw_0 + (1-d)p$
8. until convergence
//Initialize
9. $p[i] = 1 + |I|$
10. $p[u] = 1 + |U|$
11. According to section 3.3.1, find the nearest neighbor of user u and item i.
12. According to section 3.3.2, give the nearest neighbor an initial weight.
//Iteration for w_1
13. repeat
14. $w_1 = dAw_1 + (1-d)p$
15. until convergence
16. $f = w_1 - w_0$

4. Experimental Evaluation

In this section, experiments were conducted on two real-world data sets to evaluate the effectiveness of the proposed tag recommendation algorithm.

4.1 Datasets and Experimental Method

Two data sets were used to verify the proposed method. The data sets are respectively the MovieLens data set (HetRec-MovieLens) published by HetRec [21] in 2011 and MovieLens-10M data set. The two data sets contain both the attribute information of items and the user's tagging behavior. The HetRec-MovieLens data set contains 2,113 users, 5,908 items, 9,079 tags, and 47,957 tagged records; The MovieLens-10M data set contains 4009 users, 7,601 items and 95,580 tagged records.

We preprocess the data set: First, we preprocess the tags in the MovieLens-10M data set, excluded invalid tags and combined similar tags. Then, because the tag data is too sparse, p-core =5 is adopted to conduct preprocessing for the two data sets, that is, to retain the data of users, items and tags that have appeared for more than 5 times. The specific statistical information of the data set after data preprocessing is shown in **Table 4**.

In order to reduce the contingency of the experimental results, we divide the entire data set into ten equal parts according to the principle of random division. Each time eight parts are randomly selected as the training set, the rest as the test set, and the above step will be repeated five times. That means each experiment is repeated five times, and different experimental results are produced each time. We take the average of the five experimental results, and use this as the final result. First, the model parameters are optimized and adjusted on the training

set, and we find the best value 0.85 of the damping coefficient d. Then we get recommendations on the test set and evaluate the methods based on Precision, Recall, and F1. Finally, the FolkRank++ tag recommendation algorithm is compared with other baseline methods on two different public data sets to verify the effectiveness of the algorithm.

Table 4. Statistics of two data sets

Datasets	HetRec-MovieLens	ML-10M
Number of users	456	786
Number of items	1973	2403
Number of tags	1222	1657
Tag records	27026	39504

4.2 Baseline Models

- Pop [10]: A popularity-based tag recommendation algorithm.
- PITF [22] (paired interactive tensor decomposition): a tag recommendation algorithm based on tensor decomposition.
- FolkRank [23]: FolkRank algorithm has the expansion of adaptive PageRank algorithm. The core idea of this algorithm is to use user preference vector to sort tags.
- TimeFolkRank [19]: based on the algorithm of FolkRank tag recommendation, the time information of tags is considered. The newer the tag is, the more weight it will have in recommendation.

We choose the above algorithms for comparison because these algorithms are either basic algorithms or FolkRank-related algorithms, which can increase the comparability and credibility of our experimental results and prove the effectiveness of our method.

4.3 Evaluation Metrics

The recommendation algorithm is trained on the training set, and the prediction is made on the test set. Evaluation metrics selection are Precision, Recall and F1.

$$Pr\,e\,cision = \frac{1}{N}\sum_{(u,r)\in test}\frac{|\{recommend\ tags\}| \cap |\{actual\ annotation\ tags\}|}{|\{recommend\ tags\}|}$$

$$Re\,c\,all = \frac{1}{N}\sum_{(u,r)\in test}\frac{|\{recommend\ tags\}| \cap |\{actual\ annotation\ tags\}|}{|\{actual\ annotation\ tags\}|}$$

$$F1 = \frac{2 \times Pr\,e\,cision \times Re\,c\,all}{Pr\,e\,cision + Re\,c\,all}$$

4.4 Results

First, we use the first method proposed in section 3.3.1 to obtain neighbors, and use the first method proposed in section 3.3.2 to give the neighbors an initial weight. The cosine similarity is used to obtain the nearest neighbors of the user and the item, and the nearest user and the item are given an initial weight by the similarity value. The experimental selection recommendation list is Top-5, and the abscissa is the similarity threshold α. That is, when the similarity is greater than the threshold α, it is considered to be a neighbor, and the ordinate is the F1 value.

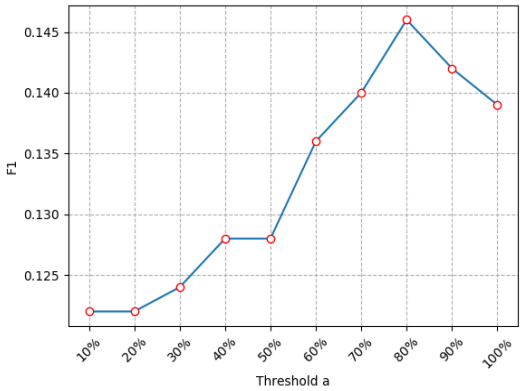

Fig. 2. Experiment results on HetRec-MovieLens

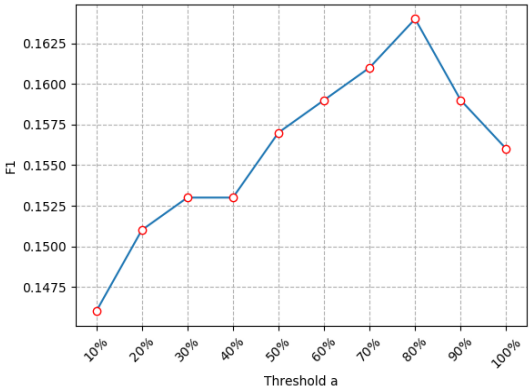

Fig. 3. Experiment results on MovieLens-10M

The experiment was carried out on the HetRec-MovieLens and MovieLens-10M data sets. It can be seen from the figures that the F1 value obtains an optimal value when the threshold is 80%, which is 0.146 and 0.164, respectively.

Second, we use the second method proposed in section 3.3.1 to obtain neighbors, and use the first method proposed in section 3.3.2 to give the neighbors an initial weight. Using the attribute information of the target item to obtain its neighbor items, the target user lacks the attribute information, so the neighbor users of the target user are obtained according to the method of part (2) of section 3.3.1; The initial weight of the neighbor item is given according to the attribute similarity of the item, and the initial user is given an initial weight according to (9). The experimental selection recommendation list is Top-5, and the abscissa is the similarity threshold β. That is, when the similarity is greater than the threshold β, it becomes the neighbor items of the target item, and the ordinate is the F1 value.

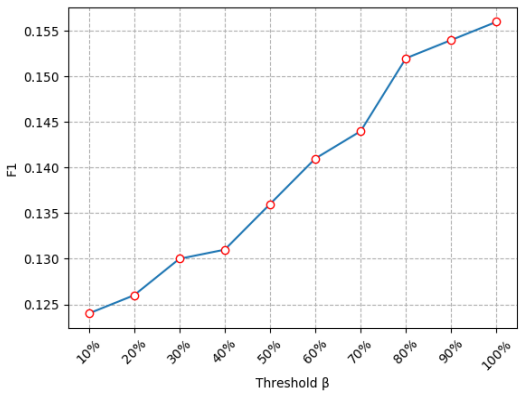

Fig. 4. Experiment results on HetRec-MovieLens

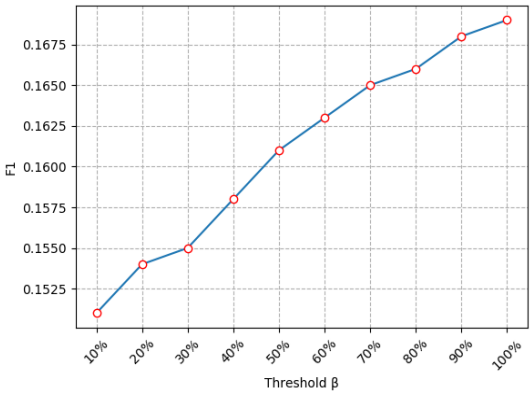

Fig. 5. Experiment results on MovieLens-10M

As can be seen from **Fig. 4** and **Fig. 5**, when the similarity threshold value in the two data sets ranges from 10% to 100%, F1 shows an upward trend. When the threshold value is 100%, the F1 value of the algorithm is optimal, and the F1 optimal value is 0.156 and 0.169 respectively. That is, when the items with the same attribute information of the target item are used as the neighbor, the algorithm has the best effect and the result is obviously better than the experiment (1).

Third, Taking the optimal result in the experiment (2), when the attribute information similarity threshold β is taken as 100%, that is, it is a neighbor items only when it is identical to the target item attribute information. We use the second method proposed in section 3.3.1 to obtain neighbors, and use the second method proposed in section 3.3.2 to give the neighbors an initial weight. We give the neighboring item an initial weight according to the importance of the item node mentioned in part (2) of section 3.3.2 and give the neighboring users an initial weight according to (9). The abscissa is the length of the tag recommendation list, and the ordinate is the F1 value.

Fig. 6 and **Fig. 7** show that when the recommended list is top 5, the importance of the item node is used to give the initial weight to the nearest neighbor items, and the F1 values on the data sets of Hetrec-MovieLens and MovieLens-10M are respectively 0.164 and 0.172, which are both improved compared with the experimental results in experiment (2) of 0.156 and 0.169.

It can be seen from the figures that our method can obtain effective results in a variety of situations, which can indicate that FolkRank++ has convergence and can obtain experimental results stably.

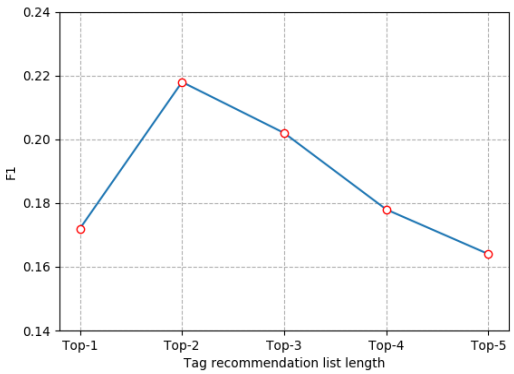

Fig. 6. Eexperiment results on HetRec-MovieLens

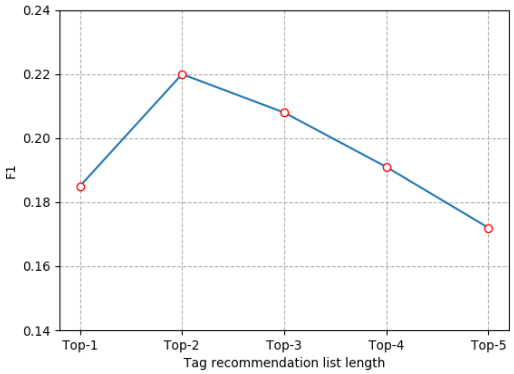

Fig. 7. Eexperiment results on MovieLens-10M

In experiment (3), the optimal algorithm proposed in this paper is determined and compared with other algorithms. The idea of the optimal algorithm: First, through the attribute category information of the items, the item that is exactly the same as the attribute information of the target item can be obtained as a similar item, at the same time, the target users' similar users are further obtained according to the user's historical tagging behavior; Second, by calculating the weighted degree of item nodes, as the index of items importance, the initial weight of similar items is given according to the degree of importance, the initial weights are further given to the similar users according to the frequency of the user's historical tagging behavior.

Table 5. Performance comparison of method in Precision

Dataset/Method		P@1	P@2	P@3	P@4	P@5
HetRec-MovieLens	*Pop*	*0.1899*	*0.1488*	*0.1168*	*0.0978*	*0.0814*
	FolkRank	*0.1905*	*0.1582*	*0.1212*	*0.0996*	*0.0832*
	PITF	*0.2058*	*0.1668*	*0.1288*	*0.1069*	*0.0923*
	TimeFolkRank	*0.2298*	*0.1740*	*0.1368*	*0.1098*	*0.0959*
	FolkRank++	***0.2596***	***0.1802***	***0.1422***	***0.1168***	***0.1038***
MovieLens-10M	*Pop*	*0.1982*	*0.1619*	*0.1298*	*0.1082*	*0.0924*
	FolkRank	*0.1992*	*0.1642*	*0.1356*	*0.1112*	*0.0936*
	PITF	*0.2158*	*0.1758*	*0.1382*	*0.1158*	*0.1006*
	TimeFolkRank	*0.2383*	*0.1799*	*0.1442*	*0.1196*	*0.1038*
	FolkRank++	***0.2673***	***0.1904***	***0.1472***	***0.1251***	***0.1078***

Table 6. Performance comparison of method in Recall

Dataset/Method		R@1	R@2	R@3	R@4	R@5
HetRec-MovieLens	*Pop*	*0.1121*	*0.2277*	*0.2868*	*0.3127*	*0.3353*
	FolkRank	*0.1183*	*0.2170*	*0.2736*	*0.3207*	*0.3577*
	PITF	*0.1204*	*0.2289*	*0.2778*	*0.3179*	*0.3491*
	TimeFolkRank	*0.1227*	*0.2407*	*0.3004*	*0.3400*	*0.3663*
	FolkRank++	***0.1286***	***0.2758***	***0.3485***	***0.3739***	***0.3904***
MovieLens-10M	*Pop*	*0.1286*	*0.2548*	*0.3159*	*0.3657*	*0.3982*
	FolkRank	*0.1365*	*0.2525*	*0.3172*	*0.3607*	*0.3904*
	PITF	*0.1348*	*0.2373*	*0.3198*	*0.3666*	*0.4029*
	TimeFolkRank	*0.1383*	*0.2580*	*0.3371*	***0.4082***	***0.4269***
	FolkRank++	***0.1414***	***0.2605***	***0.3543***	*0.4036*	*0.4253*

Table 7. Performance comparison of method in F1

Dataset/Method		F@1	F@2	F@3	F@4	F@5
HetRec-MovieLens	*Pop*	*0.1412*	*0.1801*	*0.1662*	*0.1494*	*0.1315*
	FolkRank	*0.1468*	*0.1831*	*0.1682*	*0.1524*	*0.1352*
	PITF	*0.1525*	*0.1931*	*0.1762*	*0.165*	*0.1482*
	TimeFolkRank	*0.1622*	*0.2024*	*0.1887*	*0.1662*	*0.1541*
	FolkRank++	***0.1724***	***0.2185***	***0.2024***	***0.1781***	***0.1642***
MovieLens-10M	*Pop*	*0.1561*	*0.1985*	*0.1844*	*0.1671*	*0.1505*
	FolkRank	*0.1627*	*0.1991*	*0.1925*	*0.1704*	*0.1511*
	PITF	*0.1667*	*0.2024*	*0.1935*	*0.1762*	*0.1616*
	TimeFolkRank	*0.1754*	*0.2125*	*0.2023*	*0.1858*	*0.1674*
	FolkRank++	***0.1852***	***0.2244***	***0.2085***	***0.1912***	***0.1721***

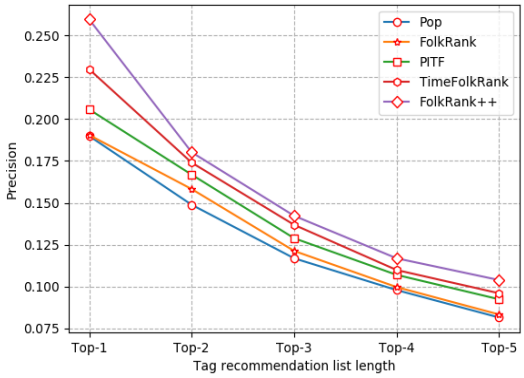

Fig. 8. Precision on HetRec-MovieLens

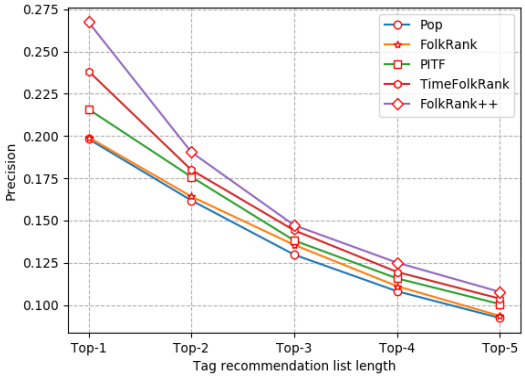

Fig. 9. Precision on MovieLens-10M

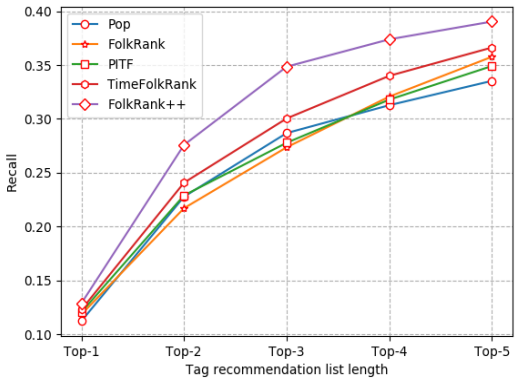

Fig. 10. Recall on HetRec-MovieLens

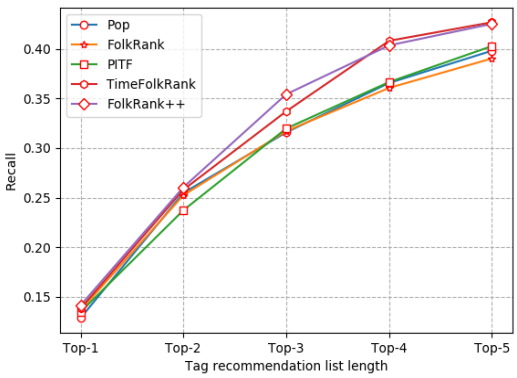

Fig. 11. Recall on MovieLens-10M

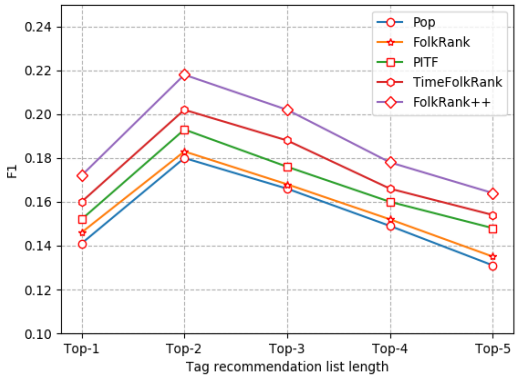

Fig. 12. F1 on HetRec-MovieLens

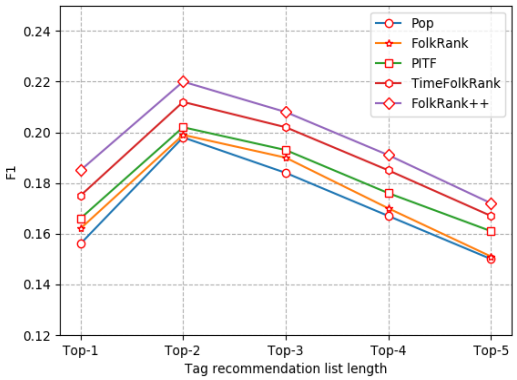

Fig. 13. F1 on MovieLens-10M

The experimental results verify the effectiveness of our proposed algorithm. The experiments are carried out on the data sets of Hetrec-MovieLens and MovieLens-10M, and it can be clearly seen from the F1 value and Precision that our proposed method is significantly superior to the experimental results of the other four comparison algorithms. In the Recall rate, only Top-4 and Top-5 results are not as good as TimeFolkRank algorithm.

The main reason for the good experimental results is that FolkRank++ algorithm can better obtain the internal relationship between user-user and item-item. For item-item relationships, we can use some methods to dig out some items that are very similar to the target items, but the target item may not have a strong connection to these similar items in the user-item-tag graph of FolkRank. Adding this part of the information is useful for tag recommendation, the same is true for user-user relationships.

5. Conclusions and Future Work

This paper considers that the FolkRank tag recommendation algorithm cannot use the user-user, item-item relationship very well. In order to solve this problem, this paper proposes an improved FolkRank tag recommendation algorithm and names it FolkRank++. This paper first seeks the neighbors of the target user and target item, then, similar to the FolkRank algorithm, the neighbors are also given an initial weight to participate in the diffusion of the algorithm. This method is intended to supply the relationship between the user-user and the item-item.

The advantage of giving the initial weight to the neighbor is that the neighbor can be fully utilized to obtain correlation information between neighbors and the target user or the target item which cannot be reflected in the graph. This method optimizes the initialization assignment of training in the tripartite graph. Experimental results show that this method has better recommendation performance than the state-of-the-art.

Many extensions can be done with this work. In the next step, the user and the item's additional information can be used to cluster users and items by clustering algorithm, so as to further obtain similar users and items, optimizing the complexity of the algorithm and improving the experimental metrics is also a future research direction [24]. We can also consider comparing with the methods of Graph Convolution [25] and knowledge graph [26] to further measure the proposed method and combine our work with deep learning methods [27, 28].

References

[1] S. L. Jones and R. Kelly, "Dealing with Information Overload in Multifaceted Personal Informatics Systems," *Human-Computer Interaction*, vol. 33, no. 1, pp. 1-48, 2017. Article (CorssRef Link)

[2] L. Xiang, "Recommended System Practice," Beijing People's Posts and Telecommunications Press, pp. 96-97, 2012.

[3] R. Jäschke, L. Marinho, A. Hotho, L. Schmidt-Thieme, and G. Stumme, "Tag Recommendations in Folksonomies," in *Proc. of European Conference on Principles & Practice of Knowledge Discovery in Databases*, vol. 4702, 2007. Article (CrossRef Link)

[4] Y. Chen, W. Xu, J. Zuo, and K. Yang, "The fire recognition algorithm using dynamic feature fusion and IV-SVM classifier," *Cluster Computing*, vol. 22, pp. 7665-7675, 2019. Article (CrossRef Link)

[5] Y. Chen, J. Tao, L. Liu, R. Xia, J. Xie, Q. Zhang, and K. Yang, "Research of improving semantic image segmentation based on a feature fusion model," *Journal of Ambient Intelligence and Humanized Computing*, 2020. Article (CrossRef Link)

[6] J. Zhao, X. Gao, X. Wang, C. Li, M. Song, and Q. Sun, "An Efficient Radio Map Updating Algorithm based on K-Means and Gaussian Process Regression," *Journal of Navigation*, vol. 71, no. 5, pp. 1055-1068, 2018. Article (CrossRef Link)

[7] J. Zhao, X. Geng, J. Zhou, Q. Sun, Y. Xiao, Z. Zhang, and Z. Fu, "Attribute mapping and autoencoder neural network based matrix factorization initialization for recommendation systems," *Knowledge-Based Systems*, vol. 166, pp. 132-139, 2019. Article (CrossRef Link)

[8] J. Zhao, W. Wang, Z. Zhang, Q. Sun, H. Hue, L. Qu, and S. Zheng, "TrustTF: A tensor factorization model using user trust and implicit feedback for context-aware recommender systems," *Knowledge-Based Systems*, vol. 209, 2020. Article (CrossRef Link)
[9] S. Ji, W. Yang, S. Guo, D. K. W. Chiu, C. Zhang, and X. Yuan, "Asymmetric response aggregation heuristics for rating prediction and recommendation," *Applied Intelligence*, vol. 50, pp. 1416-1436, 2020. Article (CrossRef Link)
[10] R. Jäschke, L. Marinho, A. Hothe, L. Schmidt-Thieme, and G. Stumme, "Tag recommendations in social bookmarking systems," *AI Communications*, vol. 21, no. 4, pp. 231-247, 2008. Article (CrossRef Link)
[11] Z. Liu, C. Shi, and M. Sun, "Folkdiffusion: A graph-based tag suggestion method for folksonomies," in *Proc. of Asia Information Retrieval Symposium*, vol. 6458, pp. 231-240, 2010. Article (CrossRef Link)
[12] W. Feng and J. Wang, "Incorporating heterogeneous information for personalized tag recommendation in social tagging systems," in *Proc. of the 18th ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1276-1284, 2012. Article (CrossRef Link)
[13] R. Jäschke, L. Marinho, A. Hothe, L. Schmidt-Thieme, and G. Stumme, "Tag recommendations in Folksonomies," in *Proc. of European Conference on Principles of Data Mining and Knowledge Discovery*, vol. 4702, pp. 506-514, 2007. Article (CrossRef Link)
[14] S. Xiance, L. Zhiyuan, L. Peng, J. Qixia, and M. Sun, "Content-based and graph-based tag suggestion," in *Proc. of the 2009 international conference on ECML PKDD Discovery Challenge*, vol. 497, no. 18, pp. 243-260, 2009. Article (CrossRef Link)
[15] J. Gemmell, T. Schimoler, M. Ramezani, L. Christiansen, and B. Mobasher, "Improving folkrank with item-based collaborative filtering," in *Proc. of the Workshop on Recommender Systems*, 2009. Article (CrossRef Link)
[16] Y. Zhang, N. Zhang, and J. Tang, "A collaborative filtering tag recommendation system based on graph," in *Proc. of the 2009 International Conference on ECML PKDD Discovery Challenge*, vol. 497, pp. 297-306, 2009. Article (CrossRef Link)
[17] N. Landia, S. S. Anand, A. Hotho, R. Jäschke, S. Doerfel, and F. Mitzlaff, "Extending folkrank with content data," in *Proc. of the 4th ACM RecSys Workshop on Recommender systems and the social web*, pp. 1-8, 2012. Article (CrossRef Link)
[18] T. Yamasaki, J. Hu, S. Sano, and K. Aizawa, "Folkpopularityrank: Tag recommendation for enhancing social popularity using text tags in content sharing services," in *Proc. of the 26th International Joint Conference on Artificial Intelligence*, pp. 3231-3237, 2017. Article (CrossRef Link)
[19] F. Riaz, R. Abbasi, and Z. Mahmood, "Adding Temporal Dimension in Social Network by Using Link Analysis," in *Proc. of 2017 International Conference on Frontiers of Information Technology*, pp. 223-228. 2017. Article (CrossRef Link)
[20] S. Brin and L. Page, "The anatomy of a large-scale hyper textual Web search engine," *Computer Networks and ISDN Systems*, vol. 30, no. 1-7, pp. 107-117, 1998. Article (CrossRef Link)
[21] I. Cantador, P. Brusilovsky, and T. Kuflik, "Second workshop on information heterogeneity and fusion in recommender systems(HetRec2011)," in *Proc. of the 5th ACM conference on Recommender systems*, pp. 387-388, 2011. Article (CrossRef Link)
[22] S. Rendle and L. Schmidt-Thieme, "Pairwise interaction tensor factorization for personalized tag recommendation," in *Proc. of the third ACM International Conference on Web Search and Data Mining*, pp. 81-90. Article (CrossRef Link)
[23] K. Liu, B. Fang, and W. Zhang, "Exploring social relations for personalized tag recommendation in social tagging systems," *IEICE Transactions on Information and Systems*, vol. 94, no. 3, pp. 542-551, 2011. Article (CrossRef Link)
[24] L. V. Tran, Y. Tay, S. Zhang, G. Cong and X. Li, "HyperML: A Boosting Metric Learning Approach in Hyperbolic Space for Recommender Systems," in *Proc. of the 13th International Conference on Web Search and Data Mining*, pp. 609-617, 2020. Article (CrossRef Link)
[25] R. Berg, T. N. Kipf, and M. Welling, "Graph convolutional matrix completion," *arXiv preprint arXiv:1706.02263*, 2017. Article (CrossRef Link)

[26] Y. Cao, X. Wang, X. He, Z. Hu, and T. S. Chua, "Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences," in *Proc. of the World Wide Web Conference*, pp. 151-161, 2019. Article (CrossRef Link)
[27] Y. Chen, J. Xiong, W. Xu, and J. Zuo, "A novel online incremental and decremental learning algorithm based on variable support vector machine," *Cluster Computing*, vol. 22, pp. 7435-7445, 2019. Article (CrossRef Link)
[28] Y. Luo, J. Qin, X. Xiang, Y. Tan, Q. Liu, and L. Xiang, "Coverless real-time image information hiding based on image block matching and dense convolutional network," *Journal of Real-Time Image Processing*, vol. 17, pp. 125-135, 2010. Article (CrossRef Link)

Jianli Zhao received his PhD degree in 2006 from the Department of Computer Application Technology, Northeastern University, China. In 2011 and 2019, he was promoted as an Associate Professor and a Full Professor in the College of Computer Science and Engineering, Shandong University of Science and Technology. His Major Research Direction includes pervasive computing, personalized recommendation, indoor location.

Qinzhi Zhang received his B.E degree in College of Computer Science and Engineering from Shandong University of Science and Technology, Qingdao, China, in 2018.

Qiuxia Sun received her PhD degree of System Theory in 2011 from Qingdao University, China. Now, she served an associate professor in the College of Mathematics and Systems Science, Shan-dong University of Science and Technology, Major Research Direction in Big Data Analysis.

Huan Huo is a senior lecturer at University of Technology Sydney. She received her Ph.D. in Computer Software and Theory from Northeastern University in 2007, then she worked as an associate professor in Optical-Electrical and Computer Engineering School at University of Shanghai for Science and Technology. Her research includes cloud data management technology, data stream query optimization, XML data management technology, etc.

Yu Xiao received his B.E degree in College of Computer Science and Engineering from Shandong University of Science and Technology, Qingdao, China, in 2017.

Maoguo Gong received the B.S. and Ph.D. degrees in electronic science and technology from Xidian University, Xi'an, China, in 2003 and 2009, respectively. Since 2006, he has been a Teacher with Xidian University. In 2008 and 2010, he was promoted as an Associate Professor and a Full Professor, respectively, both with exceptive admission. His current research interests include computational intelligence with applications to optimization, learning, data mining, and image understanding.