Hindawi
Mathematical Problems in Engineering
Volume 2019, Article ID 9605393, 1 page
https://doi.org/10.1155/2019/9605393

Retraction

Retracted: Collaborative Filtering Recommendation Algorithm Based on Knowledge Graph

Mathematical Problems in Engineering

Received 8 December 2018; Accepted 8 December 2018; Published 29 January 2019

At the request of the authors, the article titled "Collaborative Filtering Recommendation Algorithm Based on Knowledge Graph" [1] has been retracted. The article was found to contain a substantial amount of material, without citation, from an earlier article in Chinese by other authors; [2] its details are as follows:

Xiyu Wu, Qimai Chen, Hai Liu, and Chaobo He, "Collaborative filtering recommendation algorithm based on representation learning of knowledge graph," Computer Engineering, 2018, 44(2): 226-232,263.

References

[1] R. Mu and X. Zeng, "Collaborative filtering recommendation algorithm based on knowledge graph," *Mathematical Problems in Engineering*, vol. 2018, Article ID 9617410, 11 pages, 2018.

[2] W. Xiyu, C. Qimai, H. Liu, and C. He, "Collaborative filtering recommendation algorithm based on representation learning of knowledge graph," *Journal of Electrical and Computer Engineering*, vol. 44, no. 2, pp. 226–232, 2018, http://www.ecice06.com/EN/Y/V44/I2/226.

Hindawi
Mathematical Problems in Engineering
Volume 2018, Article ID 9617410, 11 pages
https://doi.org/10.1155/2018/9617410

Research Article

Collaborative Filtering Recommendation Algorithm Based on Knowledge Graph

Ruihui Mu[1,2] and Xiaoqin Zeng[1]

[1]*College of Computer and Information, Hohai University, Nanjing 210098, China*
[2]*College of Computer and Information Engineering, Xinxiang University, Xinxiang 453003, China*

Correspondence should be addressed to Ruihui Mu; muruihui@126.com

Received 29 April 2018; Revised 29 June 2018; Accepted 18 July 2018; Published 31 July 2018

Academic Editor: Francesco Lolli

To solve the problem that collaborative filtering algorithm only uses the user-item rating matrix and does not consider semantic information, we proposed a novel collaborative filtering recommendation algorithm based on knowledge graph. Using the knowledge graph representation learning method, this method embeds the existing semantic data into a low-dimensional vector space. It integrates the semantic information of items into the collaborative filtering recommendation by calculating the semantic similarity between items. The shortcoming of collaborative filtering algorithm which does not consider the semantic information of items is overcome, and therefore the effect of collaborative filtering recommendation is improved on the semantic level. Experimental results show that the proposed algorithm can get higher values on precision, recall, and F-measure for collaborative filtering recommendation.

1. Introduction

Due to information explosion, huge number of items are present over web which makes it difficult for user to find appropriate item from available set of options. Recommender system (RS) overcomes the problem of information overload and suggests items that interest a user. It has gained a lot of popularity in past decades and huge amount of work has been done in this field.

Collaborative Filtering (CF) is the most popular and widely used approach for RS which tries to analyze the user's interest over the target item on the basis of views expressed by other like-minded users. It has been successfully applied in the industrial fields such as e-commerce [1], online learning [2], and news media [3]. It has also attracted attention from a large number of scholars in the academic. Recommendation is according to the similarity between the target user and other users. The recommender systems let users give ratings about a set of items such as movies, songs, and hotels; thus, we can make recommendation to each user when enough information is stored on the system. In other words, we can compute the similarity between two users or items by considering the corated items, which are commonly rated by both users. In traditional CF, items are recommended based on rating information of similar users on the items and using some well-known similarity measurements, such as the Pearson correlation coefficient, cosine similarity, and adjusted cosine measure.

In [4], they integrated the Facebook user's personalized data into CF and improved the accuracy of recommendation by merging multi-domain data sets. In [5], they proposed a collaborative filtering model that combines singularity and diffusion processes to solve the problem of information overload by using the rating context information. In [6], they proposed content-based CF algorithm and improved the performance of news recommendation. In the traditional CF recommender systems, they use the information such as user implicit feedback data (such as purchase records) and user explicit feedback data (such as rates) to make predictions or recommendations.

However, these methods mainly used the external information, such as user-item rating matrix, and did not fully consider the internal semantic information of the items itself.

With the development of knowledge graph technology, the industry has accumulated a large number of open semantic data, such as Freebase [7] and Dbpedia [8]; it contains a vast of information. How to use these open semantic data to improve the performance of recommender system is a hot topic. In [9], they attempted to use the structural features of the knowledge graph to fuse the ontology into the CF algorithm. Existing research shows that the knowledge graph representation learning method can embed the knowledge graph into a low-dimensional semantic space and then use continuous numerical vectors to reflect the structural features of the knowledge graph. This method can efficiently calculate the semantic connections between entities. In [10], they attempted to combine the knowledge graph representation learning algorithm with implicit feedback based collaborative filtering, convert the original data into a preference sequence for parameter learning, and enhance the performance of the recommendation, but they did not consider the semantic information of the items itself.

The purpose of our work is to take a step further in addressing the cold start and data sparsity problems by proposing a novel approach to integrate the semantic information of the items itself with the CF recommendation algorithm.

In this paper, we address the above issues by introducing a novel similarity model based on knowledge graph representation learning. The proposed similarity model can select different neighbors which have higher similarity with the target user for each different target item and take the semantic similarity of the items into account.

The proposed model uses the knowledge graph representation learning-TransE algorithm to obtain the semantic information of the item, calculates the semantic similarity between the items, integrates the semantic information of the item into the recommendation process, and produce the recommended list for users.

The rest of this paper is organized as follows. Section 2 gives a brief overview of related literatures on item-based CF and KG representation learning. In Section 3, we describe the details of the proposed algorithm. In Section 4, we present an experimental evaluation comparing our models with existing work. Finally, Section 5 concludes our work and outlines potential future work.

2. Related Work

2.1. Item-Based Collaborative Filtering Recommendation Algorithm. Item-based CF recommendation algorithm [11] is a nearest neighbor recommendation algorithm; it is based on the assumption that users tend to like similar items. In the recommender system, the algorithm recommends the most similar items to the user by calculating the similarity between the items. The recommendation process is as follows: suppose there is a list of m users $U = (U_1, U_2, \ldots, U_m)$ and a list of n items $I = (I_1, I_2, \ldots, I_n)$ and the user-item rating matrix $R_{m\times n}$ is

$$\boldsymbol{R}_{m\times n} = \begin{bmatrix} R_{11} & R_{12} & \cdots & R_{1j} & \cdots & R_{1n} \\ R_{21} & R_{22} & \cdots & R_{2j} & \cdots & R_{2n} \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ R_{i1} & R_{i2} & \cdots & R_{ij} & \cdots & R_{in} \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ R_{m1} & R_{m2} & \cdots & R_{mj} & \cdots & R_{mn} \end{bmatrix} \quad (1)$$

where R_{ij} denotes the rate of the user i on the item j and represents the user i's preference of the item j. The similarity between two items can be measured by treating each item as a vector and computing the cosine of the angle formed by the vectors.

Suppose A and B are vectors and n is their dimension; vector cosine similarity between A and B is defined as

$$\begin{aligned} sim_{\cos}(\boldsymbol{A}, \boldsymbol{B}) = \cos(\theta) &= \frac{\boldsymbol{A}\cdot\boldsymbol{B}}{\|\boldsymbol{A}\| \times \|\boldsymbol{B}\|} \\ &= \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n}(A_i^2)} \times \sqrt{\sum_{i=1}^{n}(B_i^2)}} \end{aligned} \quad (2)$$

$\boldsymbol{sim_{\cos}}$ **(*A, B*)** is bigger, and the vectors A and B are more similar. When $\boldsymbol{sim_{\cos}}$ **(*A, B*)** =0, it means the two vectors are completely dissimilar and $\boldsymbol{sim_{\cos}}$ **(*A, B*)** =1 means they are completely similar.

The ratings are relative sparse in the user-item rating matrix. In [12], they proposed the importance of similarity weights to avoid data sparsity. The method uses the idea of shrinking in the Bayesians. When only a small number of ratings are used for similarity calculation, reduce the similarity weight between them. Therefore, the improved similarity weight between items is

$$w'_{ij} = \frac{|U_{ij}|}{|U_{ij}| + \beta} \times w_{ij} \quad (3)$$

where w'_{ij} is similarity weight after contraction. w_{ij} (i.e., $\boldsymbol{sim_{\cos}}$ **(*A, B*)**) denotes the similarity of the items, computed by (2). $|U_{ij}|$ represents the number of users who corated item i and j, and β is a contraction coefficient. When $|U_{ij}|$ is small, β plays contraction effect. When $|U_{ij}| >> \beta$, the effect is not significant.

According (2) and (3), the item-based CF recommendation algorithm converts the user-item rating matrix into item-item similarity matrix and uses Top-k recommendation algorithm to produce k most similar items. It has two steps: (1) prefiltering neighbors from the item-item similarity matrix. Prefiltering neighbors is to filter the neighbors of the similarity weight larger than the threshold. (2) Find k most similar items after computing the similarities and then get the Top-k most similar items for recommendation.

2.2. Knowledge Graph Representation Learning. The Word2Vec model is proposed in [13], the model embeds words into K-dimensional space. The method is to use the analogy to find the similarity between words. In [14], they applied the method to the knowledge graph and proposed the TransE learning algorithm. An energy-based model is for learning low-dimensional embeddings of entities. In TransE, relationships are represented as translations in the embedding space: if (h, r, t) holds, then the embedding of the tail entity t should be close to the embedding of the head entity h plus some vector that depends on the relationship.

Given a training set S of triplets(h, r, t)composed of two entities h, $t \in E$ (the set of entities) and a relationship $r \in L$ (the set of relationships), the model learns vector embeddings of the entities and the relationships. The basic idea behind the model is that the functional relation induced by the 'r - labeled edges corresponds to a translation of the embeddings; i.e., we want $h + r \approx t$ when (h, r, t) holds (t should be a nearest neighbor of $h + r$, while $h + r$ should be far away from t otherwise).

All embeddings for entities and relationships are first initialized. At each main iteration of the algorithm, the embedding vectors of the entities are first normalized. Then, a small set of triplets is sampled from the training set, and will serve as the training triplets of the minibatch. For each such triplet, we sample a single corrupted triplet. The parameters are then updated by taking a gradient step with constant learning rate. The algorithm is stopped based on its performance on a validation set.

The basic idea of this algorithm is to embed the entities and relations of the knowledge graph into a low-dimensional vector space and then convert the entities and relations into vector representations. Therefore, the similar entities in the KG are also similar representations in the vector space. That is, semantic similar entities are also similar in vector space.

Based on the above ideas, we can embed a triplet with rich semantic information into a K-dimensional semantical space and generate corresponding vectors and realize the knowledge graph learning representation.

Through the KG representation learning, we can calculate the semantic similarity of two entities, and the knowledge graph can be conveniently used in other learning tasks. There are two methods to implement the KG representation learning: (1) tensor decomposition-based methods and (2) translation-based methods. The former includes NTN [15–19] and RESCAL [19–22]. The latter includes TransE [23–25], TransH [26–29], and TransR/CtransR [30–32]. The relationships are very large and very sparse in the large-scale knowledge bases; the tensor decomposition-based methods are ineffective, so it is more appropriate to select the translation-based methods [33–36].

The purpose of our work is to take a step further in addressing the cold start and sparsity problems by proposing a novel approach to combine the knowledge graph in CF.

3. Proposed Method

In this section, we describe the proposed collaborative filtering recommendation algorithm based on knowledge graph (KG). First, we introduce the framework of the algorithm, followed by the TransE algorithm based KG representation learning. Next, we present the semantic similarity measure. Then, the details of fusion semantic neighbors are provided. Finally, we give the details of the algorithm description.

3.1. Algorithm Framework. We propose a novel CF recommendation algorithm based on KG representation learning. Its basic idea is using the KG representation learning-TransE algorithm to obtain the semantic information of the item, calculating the semantic similarity between the items, integrating the semantic information of the item into the recommendation process, and producing the recommended list for users.

In the traditional CF algorithms, they only use the external ratings information, such as user-item rating matrix, and the proposed algorithm adds internal knowledge (knowledge graph) into CF and will get better results and improve the recommended performance.

We use the KG representation learning algorithm to embed entities into a low-dimensional vector space. Then calculate the semantic similarity between the entities and generate the semantic similarity matrix. Use Top-k algorithm to get the k nearest semantic neighbors of the items. At the same time, we use the user-item rating matrix to calculate the similarity between items and get the item-item similarity matrix and then use Top-k algorithm to get the k nearest neighbors of the items. Finally, adjust the fusion ratio, integrate the nearest semantic neighbors of the items with the CF nearest neighbors of the items, and produce the recommended list.

The rich semantic information can alleviate the cold start problem in the recommender system. Figure 1 shows the flowchart of proposed algorithm (TransE-CF).

3.2. TransE Algorithm Based KG Representation Learning. The knowledge graph is a knowledge network and can be represented by a directed graph (or digraph) formed by the triples and the mutual links between the triples. The triplets carry the semantic information of the entity itself. Nodes represent the entities and edges represent the relationships between the entities.

Taking the movie field as an example, the movie entity mainly includes the main features of it, such as actors, types, and directors. These features characterized the movies. Using the movie features, one can obtain a triplet of a movie knowledge graph. As shown in Figure 2.

In Figure 2, the movie entity and the actor entity are linked to form a triplet (M, starring_actor, A) and the knowledge graph is formed by the mutual linkage between the triplets. Figure 3 shows the structure of some selected movies and their attributes in the Freebase. In Figure 3, it can be seen that two nodes are closer in the knowledge graph and they are more similar in semantics. Therefore, in the movies recommendation, we can use the user's rating information of the movie and the movie's own semantic information. In the CF recommender system, if the ratings of the two items are similar, they are determined as neighbors. Similarly, if the movies are semantically similar in the knowledge graph, they can be judged as neighbors intuitively.

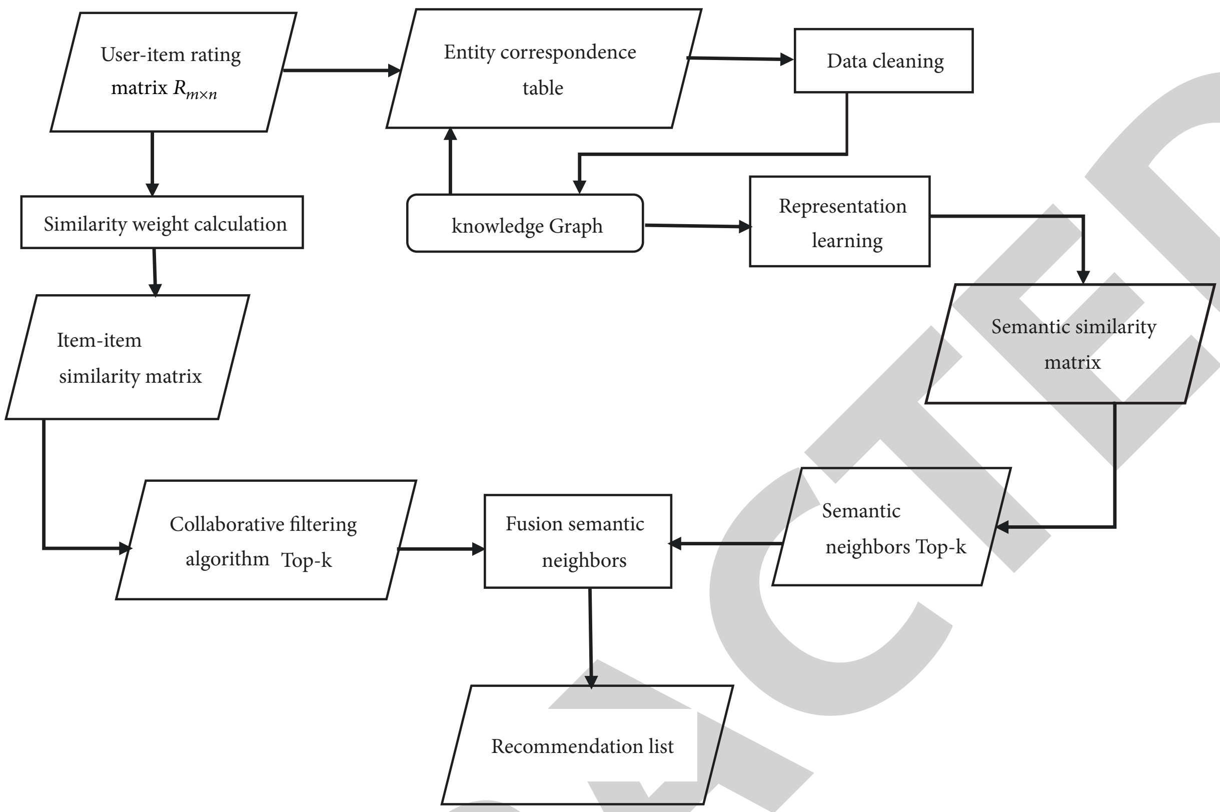
User-item rating matrix $R_{m\times n}$
Entity correspondence table
Data cleaning
Similarity weight calculation
knowledge Graph
Representation learning
Item-item similarity matrix
Semantic similarity matrix
Collaborative filtering algorithm Top-k
Fusion semantic neighbors
Semantic neighbors Top-k
Recommendation list

Figure 1: TransE-CF flowchart.

Movie
Starring_actor
Actor

Figure 2: Triples of knowledge graph.

Based on the above ideas, we use the TransE algorithm to integrate the items' semantic information with the user-item rating matrix for recommendation. According to the definition of the TransE algorithm, triplet (*h, r,* and *t*) in the knowledge graph $\boldsymbol{S}$ can be trained using the loss function, defined as

$$L = \sum_{(h,r,t)\in S} \sum_{(h',r,t')\in S'_{(h,r,t)}} \left[\gamma + \|h + r - t\| - \left\|h' + r - t'\right\|\right]_+ \quad (4)$$

$$S'_{(h,r,t)} = \left\{\left(h', r, t\right) \mid h' \in E\right\} \cup \left\{\left(h, r, t'\right) \mid t' \in E\right\} \quad (5)$$

where (h', r, t') is the corrupted triple as negative training sample. The negative sample is the head entity or tail entity of the original correct triplet by randomly replacing it with other entities. γ is margin hyperparameter, denotes the size of the gap, and generally sets γ=1. $\| \bullet \|$ is either the L_1 or the L_2-norm. $[\cdot]_+$ is the hinge loss function, as shown in

$$[x]_+ = \begin{cases} x, & x > 0 \\ 0, & x \le 0 \end{cases} \quad (6)$$

where $[x]_+$ denotes the positive part of x.

We minimize the loss function L over the training set, and the optimization is carried out by stochastic gradient descent (in minibatch mode), over the possible

h, *r,* and *t*, with the additional constraints that the L_2-norm of the embeddings of the entities, are 1 (no regularization or norm constraints are given to the label embeddings *r*).

Throughout the training process, TransE uses the idea of maximum separation, pulling distances between positive and negative sample vectors to train and learn. The stochastic gradient descent is continuously iteratively degraded to optimize the loss function. The final training results are (1) similar semantic entities in the KG have closer distances in the low-dimensional space; (2) the head node's vector plus the relational vector is basically equal to the tail node's vector. The specific training process is described in [14].

3.3. Semantic Similarity Measure. TransE algorithm transforms entities into low-dimensional real-valued vectors. In the different KG representation learning methods, the meanings of the vectors are different and the similarity measures

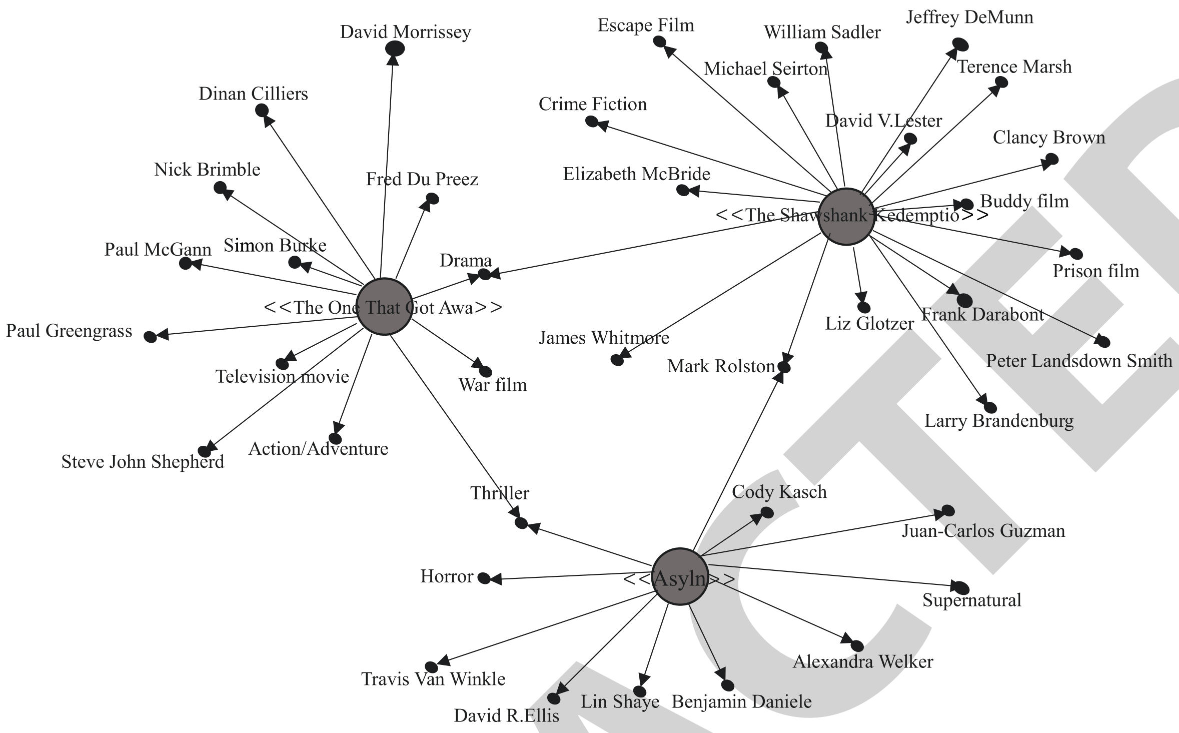

Figure 3: Movie knowledge graph.

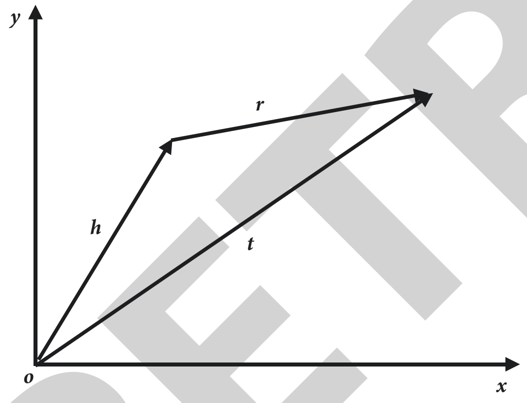

Figure 4: TransE model.

are different too. The representation of the vectors in the TransE algorithm is defined as

$$\|h+r\| \approx t \tag{7}$$

where h and t are the head entity and the tail entity respectively and r is the relationship between the entities. The norm $\|\bullet\|$ can be selected as the L_2 norm. t should be a nearest neighbor of $h+r$. The closer the entities in the KG are, the more similar the vectors are, as shown in Figure 4.

Since the TransE algorithm chooses the Euclidean distance to calculate the loss function, we use the same norm Euclidean distance to measure the similarity between the entities, where the selected Euclidean distance range is $[0, \infty)$ and the similarity measure range is $[0, 1]$. For the two entity vectors A and B, through mathematical transformation, the two value domains are matched one by one by. The final semantic similarity measure is defined as

$$sim_{\text{sem}}(\boldsymbol{A},\boldsymbol{B}) = 1 - \frac{\|\boldsymbol{A}-\boldsymbol{B}\|}{\|\boldsymbol{A}-\boldsymbol{B}\|+1} = \frac{1}{\|\boldsymbol{A}-\boldsymbol{B}\|+1} \tag{8}$$

In (8), when the value of sim_{sem} (A, B) is closer to 1, the entity vectors A and B are more similar; that is, the relationship is closer in the KG. Conversely, if the value of sim_{sem} (A, B) is smaller, the more alienation relationship between A and B, and the lower semantic similarity between A and B.

We use the semantic similarity for recommendation; the process is similar to the item-based CF recommendation. We use (8) to get the item-item semantic similarity matrix, as shown in Table 1.

In Table 1, a_{ij} is the semantic similarity between the item i and the item j, which is obtained according to (8). The similarity matrix element satisfies the condition $a_{ij} = a_{ji}$. The ith row is the semantic nearest neighbors of the item i. Fuse the semantic neighbors with the item-based CF neighbors and then get the recommended list.

3.4. Fusion Semantic Neighbors. We suppose I is the set of items that the CF algorithm will recommend to users. Next, sort items $i \in I$ in decreasing order of the similarity and then get a corresponding sequence L. Similarly, T is the set of semantic neighbors and also sorted in decreasing order and then get a corresponding sequence K. The elements j in K satisfy $\{j \mid j \in T, j \notin I\}$. Suppose l is the length of the sequence L; the fusion ratio is p: q. The number of

Table 1: Item-tem semantic similarity matrix.

Item	$Item_1$	$Item_2$	...	$Item_j$	...	$Item_n$
$Item_1$	1	a_{12}	...	a_{1j}	...	a_{1n}
$Item_2$	a_{21}	1	...	a_{2j}	...	a_{2n}
⋮	⋮	⋮		⋮		⋮
$Item_i$	a_{i1}	a_{i2}	...	a_{ij}	...	a_{in}
⋮	⋮	⋮		⋮		⋮
$Item_n$	a_{n1}	a_{n2}	...	a_{nj}	...	1

substitutions is $n = \lceil l \times (p/(p+q)) \rceil$, where $\lceil \cdot \rceil$ is roundup function. Get the following fusion process.

Algorithm 1 (fusion semantics neighbor algorithm).

Input

Collaborative Filtering Neighbor Set I, Semantic Neighbor Set T

Output

Recommendation Set C

```
(1) Set $L = \text{list}(I)$, K = list($T$)
(2) For $L_i \in \{L_{l-n}, L_{l-n+1}, \ldots, L_l\}$ do
(3)         For $K_j \in \{K_1, K_2, \ldots, K_n\}$ do
(4)             $L_i = K_j$
(5)         End do
(6) End do
(7) Set $C = \text{set}(L)$.
```

Through Algorithm 1, we get the final recommendation set C.

3.5. Algorithm Description. The algorithm flowchart is shown in Figure 1; the algorithm description is as follows:

Algorithm 2 (transE-CF algorithm).

Input

User-item rating matrix $R_{m\times n}$, knowledge graph KG

Output

Collaborative filtering recommendation based on knowledge graph representation learning

(1) According to (2), calculate the similarity weight w_{ij} (i.e., $\boldsymbol{sim}_{\mathbf{cos}}$ ***(A, B)***); next, use (3) to calculate similarity weight after contraction w'_{ij}; then generate the item-item similarity matrix, and use the Top-k recommendation algorithm to produce the k most similar items according to the similarities.

(2) Match the items with the entities in the KG and get the entity correspondence table.

(3) Transform the entities and relationships in the KG into the vector set E and the vector set R, respectively.

(4) According to the result of step (3), use (8) to calculate the semantic similarity sim_{sem} *(A, B)* and generate the semantic similarity matrix of the items and then use Top-k recommendation algorithm to produce the k most semantic similar neighbors.

(5) According to Algorithm 1 and the result of step (2), select the fusion ratio and replace the recommended items generated in step (1) with the semantic similar neighbors generated in step (4). Finally, produce the recommended list and recommend to users.

4. Experimental Results

In this section, we detail our experiments and results. First, the datasets and evaluation metrics are introduced, followed by the experimental settings. Finally, the results and analysis of experiments are presented.

4.1. Data Sets. We use Amazon's movie dataset in our experiments. It is provided by the University of California, San Diego. The dataset includes Amazon's products ratings and their metadata. Each data record mainly describes the ratings of the movie by different users. The ratings scale from 1 to 5, which corresponds to the user's review on the movie: 1 represents the least liked and 5 represents the most liked, at the same time, using metadata to map the movie ID in each record to the movie entity represented by the name of the movie in the knowledge graph.

We select the movie ontologies in Freebase 2012-11-09 as knowledge graph dataset. The movie ontology data mainly includes ontology objects such as movies, directors, and actors. In order to fuse multiple ontology objects provided by Freebase, we replace Freebase's own ID with the entity's name and extract the entities and relationships. At the same time, in order to reduce the impact of noise data, we selected the movies released after 1979 in Freebase to match the Amazon's ratings and filtered out the relationships with less than 3 occurrences and finally got a total of 20 semantic relationships of knowledge graph such as "genre" and "starring_actor" on.

Table 2: Confusion matrix.

System	Users like	Users dislike
Recommended	TP	FP
No recommended	FN	TN

In order to better match Amazon rating data with the Freebase, we further processed the movie name. Because knowledge graph used unsupervised or semisupervised relation extraction methods [37–43], the extraction of movie names in Freebase is limited by this processing method. The problem is that the item names and the ontology entity names cannot be completely matched. For example, some movies use Roman numerals 'II', while others use Arabic numerals '2'. In another example, the movie 'Here Comes the Groom (1934)' and the movie 'Here Comes the Groom' are essentially the same movie but, due to the different version numbers, the Amazon movie name cannot completely match the Freebase movie name.

To address above problems, we use the distance editing and string matching rule to preclean the data of the movie name such as version number deletion. Through the edit distance algorithm, the string literal similarity recognition of the original data is corrected and a good cleaning result is obtained. After the final processing, experimental datasets include 672 910 users' data, 41 255 movies data, and 20 semantic relationships. We divide the datasets 80% as the training set and 20% as the testing set. The training set is used to construct user-item rating matrix. The testing set is used to test the performance of the algorithm.

4.2. Evaluation Metrics. In order to evaluate the recommended results, we use three metrics to measure them: Precision, Recall, and F-measure. These three metrics can be derived from the confusion matrix, as shown in Table 2.

According to the confusion matrix, there are formulas as follows:

Recall:

$$Recall = \frac{TP}{TP + FN} \tag{9}$$

Precision:

$$Precision = \frac{TP}{TP + FP} \tag{10}$$

F-measure:

$$F - \text{measure} = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{11}$$

Recall denotes the proportion of the user's actual liked items in the recommended list. Precision denotes the recommended level of the recommender system. F-measure is a weighted average of Precision and Recall and evenly reflects the performance of recommendation.

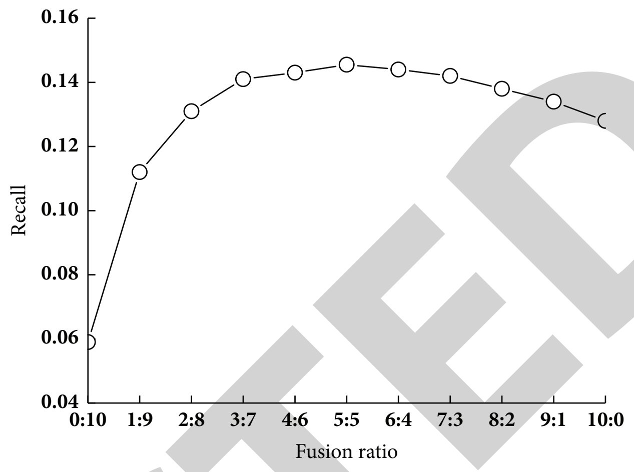

Figure 5: Recall at k=100.

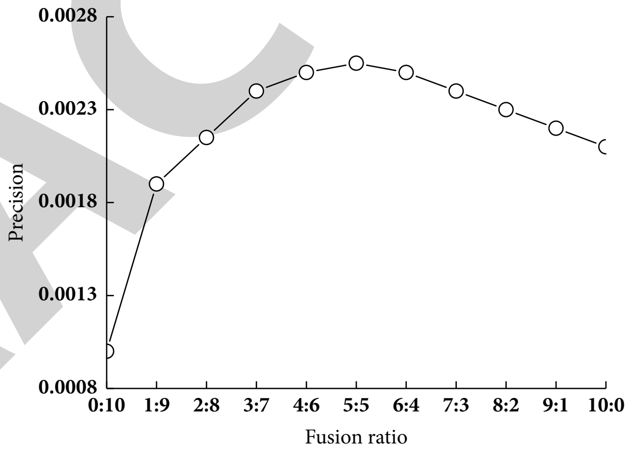

Figure 6: Precision at k=100.

4.3. Experimental Settings. The experimental hardware environment is Intel(R) Core (TM) i7-6700 with 8 GB of memory; the experimental software environment is Python 3.5.

4.4. Results and Analysis

4.4.1. Comparisons of Different Fusion Ratios. In the proposed algorithm, we adjust the fusion ration as the parameter. The fusion ratio takes different values; recommended results are also different. We set Top-k $k = 100$, representation learning embedding dimension is 200, and the fusion ratio scale from (0:10) to (10:0) represents from fully using the semantics for recommendation to fully using collaborative filtering for recommendation. Figures 5–7 are Recall curve, Precision curve, and F-measure curve, respectively. For each set of experiments, iterate 10 times and take the average value.

From Figures 5–7, it can be seen that with the increase of the fusion ratio, Recall, Precision, and F-measure all increased and reached the peak at a fusion ratio of 5:5. For $k = 100$, the best fusion ratio is 5:5.

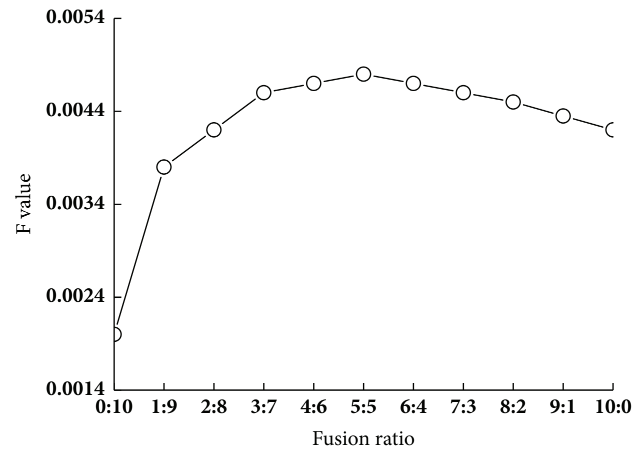

Figure 7: F-measure at k=100.

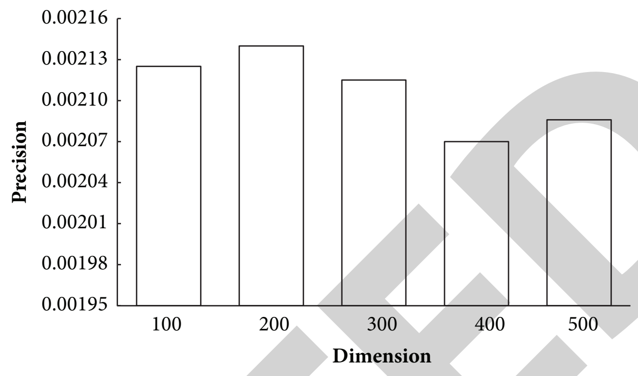

Figure 9: Precision in different dimensions.

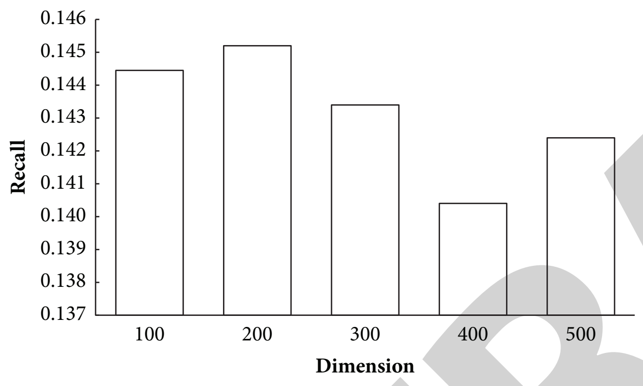

Figure 8: Recall in different dimensions.

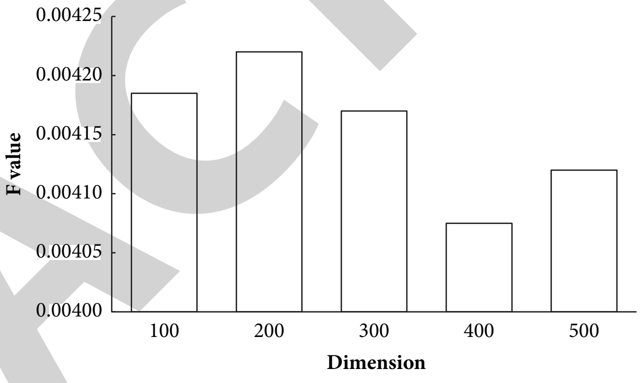

Figure 10: F-measure in different dimensions.

4.4.2. Comparisons of Different Embedding Dimensions. The knowledge graph representation learning is to embed the knowledge graph into a low-dimensional space; different dimensions have different effects. In the proposed algorithm, different dimensions have different recommended performance. In the TransE learning algorithm, initialize the triplet (h, r, and t) needed to set the embeddings dimension k. We select experimental dimension from 100 to 500. For each set of experiments, iterate 10 times and take the average value.

Figures 8–10 show different recommended performance in different dimensions. It can be seen that when embedding dimension is 200, recommended performance is the best.

4.4.3. Comparisons of Different Algorithms. We select the fusion ratio with the highest F-measure as a representative and compare with other collaborative filtering recommendation algorithms. We select the embedding dimension is 200, k-nearest neighbor is 60, 80, 100, and 120 respectively. For each set of experiments, iterate 10 times and take the average value.

From Figures 11–13, it can be seen that the collaborative filtering recommendation algorithm based on knowledge graph is superior to the other collaborative filtering recommendation algorithm. The use of semantic information can improve the recommended performance to a certain extent and also get higher values on Recall, Precision, and F-measure, respectively. It can compensate the lack of semantic in the item-based collaborative filtering algorithms.

5. Conclusion and Future Work

In this paper, we proposed a similarity model based on knowledge graph representation learning. The proposed similarity model can select different neighbors which have higher similarity with the target user for each different target item and take the semantic similarity into account.

First, we used the knowledge graph representation learning-TransE algorithm to obtain the semantic information of the items; next, calculated the semantic similarity between the items, then integrated the semantic information of the items into the recommendation process, and finally produced the recommendation list for users.

Experimental results showed that our method obtained significantly better prediction accuracy in dealing with both general situation and cold start situation, compared with state-of-the-art CF methods.

In future work, we plan to apply the algorithm to the content outside the movies for recommendation and further optimize the performance of recommendation.

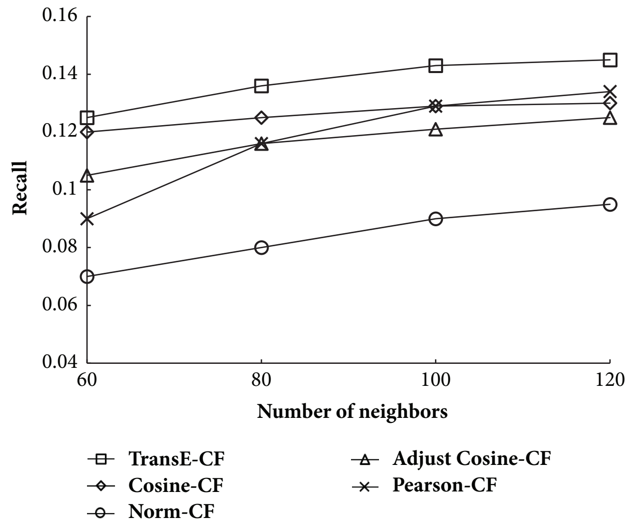

FIGURE 11: Recall with different ***k*** values.

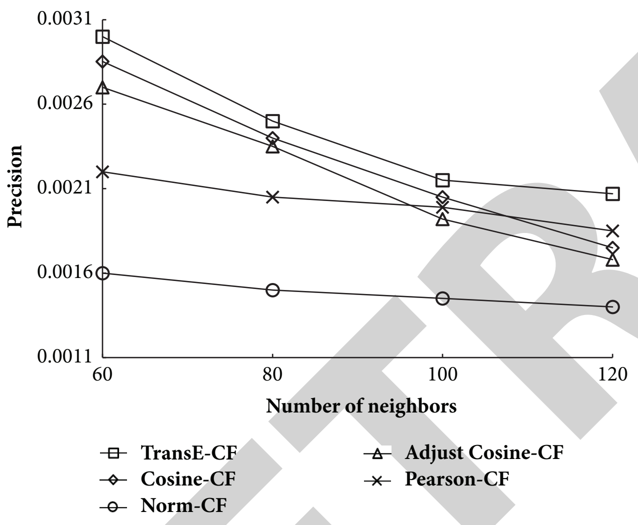

FIGURE 12: Precision with different ***k*** values.

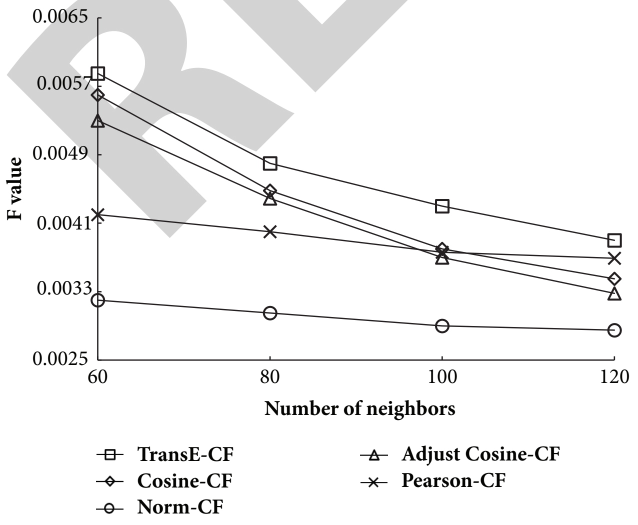

FIGURE 13: F-measure with different ***k*** values.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was partially supported by the National Key Research and Development Plan Key Projects of China under Grant no. 2017YFC0405800, the National Natural Science Foundation of China Grant (Grant nos. 60971088, 60571048, 61432008, and 61375121), and the Natural Science Foundation of the Colleges and Universities in Jiangsu Province of China, no. 17KJB520022.

References

[1] B. Song and J. Gao, “The enhancement and application of collaborative filtering in e-learning system,” in *Proceedings of the International Conference in Swarm Intelligence*, pp. 188–195, Springer, Berlin, Germany, 2014.

[2] F. Garcin, K. Zhou, B. Faltings, and V. Schickel, “Personalized news recommendation based on collaborative filtering,” in *Proceedings of the International Joint Conferences on Web Intelligence and Intelligent Agent Technology*, pp. 164–173, IEEE Press, Washington D.C., USA, 2012.

[3] I. Fernández-Tobías and I. Cantador, “Personality-aware collaborative filtering: an empirical study in multiple domains with facebook data,” in *Proceedings of the International Conference on Electronic Commerce and Web Technologies*, pp. 125–137, Springer, Berlin, Germany, 2014.

[4] J. Lehmann, G. Kobilarov et al., “A crystallization point for the web of data,” *Web Semantics Science Services & Agents on the World Wide Web*, vol. 7, no. 3, pp. 154–165, 2009.

[5] Z. Zhang, L. Gong, and J. Xie, “Ontology-based collaborative filtering recommendation algorithm,” in *Proceedings of the International Conference on Brain Inspired Cognitive Systems*, pp. 172–181, Springer, Berlin, Germany, 2013.

[6] Z. Fuzheng, Y. Lian Defu et al., “Collaborative knowledge base embedding for recommender systems,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 353–362, ACM Press, NewYork, USA, 2016.

[7] B. Usuniern, D. Garcia et al., “Translating embeddings for modeling multi relational data,” in *Proceedings of the International Conference on Neural Information Processing Systems*, pp. 2787–2795, ACM Press, New York, USA, 2013.

[8] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, “Item-based collaborative filtering recommendation algorithms,” in *Proceedings of the International Conference on World Wide Web*, pp. 285–295, ACM Press, New York, USA, 2001.

[9] R. Bell, Y. Koren, and C. Volinsky, “Modeling relationships at multiple scales to improve accuracy of large recommender systems,” in *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 95–104, ACM Press, New York, USA, 2007.

[10] I. Sutskever, T. Mikolov et al., "Distributed representations of words and phrases and their compositionality," in *Proceedings of the International Conference on Neural Information Processing Systems*, pp. 3111–3119, ACM Press, New York, USA, 2013.

[11] M. Nickelm, V. Tresp, and H. P. Kriegelh, "A three-way model for collective learning on multi relational data," in *Proceedings of the 28th International Conference on Machine Learning*, pp. 809–816, Springer, Berlin,Germany, 2011.

[12] B. Tan, X. Shen, and C. Zhai, "Mining long-term search history to improve search accuracy," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 888–895, ACM Press, Philadelphia, PA, USA, 2006.

[13] D. Vallet, I. Cantador, and J. M. Jose, "Personalizing web search with folksonomy-based user and document profiles," in *Proceedings of the 32nd European conference on Advances in Information Retrieval*, pp. 531-438, ACM Press, Milton Keynes, UK, 2010.

[14] B. Antoine, N. Usunier et al., "Translating embeddings for modeling multi relational data," in *Proceedings of the NIPS*, pp. 926–934, ACM Press, New York, USA, 2013.

[15] D. Godoy and A. Corbellini, "Folksonomy-based recommender systems: a state-of-the-art review," *International Journal of Intelligent Systems*, vol. 31, no. 4, pp. 314–346, 2016.

[16] F. Abel, M. Baldoni, C. Baroglio et al., "Leveraging search and content exploration by exploiting context in folksonomy systems," *New Review of Hypermedia and Multimedia*, vol. 16, no. 1-2, pp. 133–140, 2010.

[17] H. Al-khalifa, H. Davis et al., "Measuring the semantic value of folksonomies," in *Proceedings of the Conference on Innovations in Information Technology*, pp. 89–95, IEEE Press, Dubai, United Arab Emirates, 2006.

[18] H. Xie, Q. Li, and Y. Cai, "Community-aware resource profile for personalized search in folksonomy," *Journal of Computer Science and Technology*, vol. 27, no. 3, pp. 611–622, 2012.

[19] Y. Cai, Q. Li, H. Xie, and H. Min, "Exploring personalized searches using tag-based user profiles and resource profiles in folksonomy," *Neural Networks*, vol. 58, pp. 188–190, 2014.

[20] M. G. Noll and C. Meinel, "Web search personalization via social bookmarking and tagging," in *Proceedings of the 6th International the Semantic Web and 2nd Asian Conference on Asian Semantic Web Conference*, pp. 257–266, ACM Press, Berlin, Heidelberg, Germany, 2007.

[21] Y. Cai, Q. Li, H. Xie, L. Yu et al., "Personalized resource search by tag-based user profile and resource profile," in *Proceedings of the International Conference on Web Information Systems Engineering*, pp. 266–276, ACM Press, Hong Kong, China, 2010.

[22] S. E. Robertson and S. Walker, "Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval," in *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 320–330, ACM press, Dublin, Ireland, 1994.

[23] S. Choy and A. Lui, "Web information retrieval in collaborative tagging systems," in *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, pp. 262–369, IEEE Press, Hong Kong, China, 2006.

[24] P. J. Morrison, "Tagging and searching: search retrieval efficiency of folksonomies on the world wide web," *Information Processing and Management*, vol. 44, pp. 1352–1363, 2008.

[25] C. Biancalana and A. Micarelli, "Social tagging in query expansion: A new way for personalized web search," in *Proceedings of the International Conference on Computational Science and Engineering*, pp. 1060–1065, IEEE Press, Vancouver, Canada, 2009.

[26] S. Xu, S. Bao, B. Fei, Z. Su, and Y. Yu, "Exploring folksonomy for personalized search," in *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 265–272, ACM Press, Singapore, 2008.

[27] Q. Du, H. Xie, Y. Cai et al., "Folksonomy-based personalized search by hybrid user profiles in multiple levels," *Neurocomputing*, vol. 204, pp. 142–152, 2016.

[28] S. Jin, H. Lin, and S. Su, "Query expansion based on folksonomy tag co-occurrence analysis," in *Proceedings of the 2009 IEEE International Conference on Granular Computing (GRC)*, pp. 223–231, Nanchang, China, 2009.

[29] H. Kim, M. Rawashdeh, A. Alghamdi, and A. El Saddik, "Folksonomy-based personalized search and ranking in social media services," *Information Systems*, vol. 37, no. 1, pp. 61–76, 2012.

[30] F. Font, J. Serrà, and X. Serra, "Folksonomy-based tag recommendation for collaborative tagging systems," *International Journal on Semantic Web and Information Systems*, vol. 9, no. 2, pp. 65–73, 2013.

[31] W.-T. Hsieh, J. Stu, Y.-L. Chen, and S.-C. T. Chou, "A collaborative desktop tagging system for group knowledge management based on concept space," *Expert Systems with Applications*, vol. 36, no. 5, pp. 8363-8273, 2009.

[32] Z. Zhu, X. You, P. C. L. Chen et al., "An adaptive hybrid pattern for noise-robust texture analysis," *Pattern Recognition*, vol. 48, no. 8, pp. 2592–2608, 2015.

[33] L. Wang, J. Tao, R. Ranjan et al., "G-Hadoop: mapReduce across distributed data centers for data-intensive computing," *Future Generation Computer Systems*, vol. 29, no. 3, pp. 739–750, 2013.

[34] J. Ma, G. Li, M. Zhong et al., "Latent genre aware micro-video recommendation on social media," in *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 13*, pp. 549–556, New York, NY, USA, 2013.

[35] Y.-J. VizSearch, "A collaborative Web searching environment," *Computers & Education*, vol. 45, no. 4, pp. 503–510, 2005.

[36] B. Smyth, "A community-based approach to personalizing web search," *The Computer Journal*, vol. 40, no. 8, pp. 130–138, 2007.

[37] D. Zhou, J. Weston, and A. Gretton, "Ranking on data manifolds," *Advances in Neural Information Processing Systems*, vol. 15, pp. 78–88, 2005.

[38] S.-M. Choi, S.-K. Ko, and Y.-S. Han, "A movie recommendation algorithm based on genre correlations," *Expert Systems with Applications*, vol. 39, no. 9, pp. 8079–8085, 2012.

[39] Q. Zheng and H. H. Ip, "Customizable surprising recommendation based on the tradeoff between genre difference and genre similarity," in *Proceedings of the 14th ICWL Conference on Web-Based Learning*, pp. 272–283, Springer Press, Hong Kong, China, 2016.

[40] S. Frémal and F. Lecron, "Weighting strategies for a recommender system using item clustering based on genres," *Expert Systems with Applications*, vol. 77, pp. 222–230, 2017.

[41] D. Anand and B. Mampilli, "Folksonomy-based fuzzy user profile for improved recommendations," *Expert Syst. Appl*, vol. 41, pp. 2334–2445, 2014.

[42] H. Kumar, S. Lee, and H.-G. Kim, "Exploiting social bookmarking services to build clustered user interest profile for

personalized search," *Information Sciences*, vol. 281, pp. 299–317, 2014.

[43] Y. Yang and S. Hu, "Exploring reviews and ratings on reviews for personalized search," in *Proceedings of the 14th ICWL Conference on Web-Based Learning*, pp. 252–263, Springer Press, Hong Kong, China, 2015.