Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 2382892, 9 pages
https://doi.org/10.1155/2021/2382892

Research Article

Recommendation Model Based on Semantic Features and a Knowledge Graph

Yudong Liu and Wen Chen

School of Computer Science, South China Normal University, Guangzhou 510631, China

Correspondence should be addressed to Yudong Liu; liuyudong@scnu.edu.cn

Received 1 June 2021; Revised 5 July 2021; Accepted 9 July 2021; Published 20 July 2021

Academic Editor: Zhihan Lv

In the field of information science, how to help users quickly and accurately find the information they need from a tremendous amount of short texts has become an urgent problem. The recommendation model is an important way to find such information. However, existing recommendation models have some limitations in case of short text recommendation. To address these issues, this paper proposes a recommendation model based on semantic features and a knowledge graph. More specifically, we first select DBpedia as a knowledge graph to extend short text features of items and get the semantic features of the items based on the extended text. And then, we calculate the item vector and further obtain the semantic similarity degrees of the users. Finally, based on the semantic features of the items and the semantic similarity of the users, we apply the collaborative filtering technology to calculate prediction rating. A series of experiments are conducted, demonstrating the effectiveness of our model in the evaluation metrics of mean absolute error (MAE) and root mean square error (RMSE) compared with those of some recommendation algorithms. The optimal MAE for the model proposed in this paper is 0.6723, and RMSE is 0.8442. The promising results show that the recommendation effect of the model on the movie field is significantly better than those of these existing algorithms.

1. Introduction

With the rapid development of the Internet, smart terminals. and digital resources, recommendation systems are becoming more and more important as a tool for users to obtain the information they need in the network. Moreover, since more and more information have characteristics as short text with few features and incomplete information (e.g., microblog and short messages, movie introductions, and product reviews), accurate recommendation of items based on short texts is a hot topic on the recommendation system.

Unfortunately, the existing recommendation systems have some drawbacks for short text recommendation. For example, the traditional collaborative filtering technology mainly considers user ratings and ignores important semantic factors; LDA-based recommendation is effective in processing texts with complete information, but it has drawbacks in processing short texts; neural network-based recommendation requires a lot of computing resources, and its performance is highly related to the quality of the corpus.

To solve the above problems, we propose a recommendation model based on semantic features and extension using a knowledge graph for the current tremendous amount of items with short text characteristics. We first extend the short texts of items based on a knowledge graph and combine the semantic features of the items according to user ratings, making the recommendation result more semantically accurate.

The main contributions of this paper are summarized as follows:

(i) We propose an improved method based on a knowledge graph of short text feature extension; it can extend the semantic information of the item. We first select DBpedia as a knowledge graph, then use DBpedia Spotlight to identify the named entity in the short texts of items, and then obtain the extension words

according to the resource pages in DBpedia of these identified entities

(ii) We propose a recommendation model based on semantic features and a knowledge graph. We first extend the short texts of items based on a knowledge graph and then combine the user semantic similarity and the item semantic similarity according to the semantic features and further calculate prediction ratings of items

The remainder of this paper is organized as follows. Section 2 summarizes the related work on the recommendation system. Section 3 presents the proposed method of short text feature extension based on DBpedia. Section 4 presents the proposed recommendation model. In Section 5, experiments are conducted to verify the effectiveness of the proposed model. Finally, Section 6 presents the conclusions of this paper and the future work.

2. Related Research

The research of the recommendation system mainly includes the following categories: (1) the traditional collaborative filtering recommendation algorithms, (2) the recommendation algorithms based on deep learning, and (3) the recommendation algorithms based on the content of the item.

First, traditional collaborative filtering recommendation algorithms include user-based collaborative filtering algorithms [1–5] and item-based collaborative filtering algorithms [6–9]. The idea of the user-based collaborative filtering algorithm is when a user needs to be recommended, we would first find other users who are similar to him and then recommend those users' preferred items to him. The item-based collaborative filtering algorithm is to find items that are similar to the user's preferred items for recommendation. Because these two collaborative filtering recommendation algorithms have some deficiencies, such as items with high popularity and cold start, scholars have proposed several recommendation algorithms based on hybrid strategies [10–12]. However, traditional recommendation algorithms mainly consider aspects such as ratings and user features and often ignore important semantic features.

Secondly, with the successful application of deep learning in the fields of computer vision and natural language processing, many scholars have introduced deep learning into the recommendation system, using deep feature extraction to improve recommendation performance.

Georgiev proposed to use RBM to extract potential features of user preferences or item ratings in the recommendation field [13]. In addition, DBN [14] is also used to help extract hidden and useful features from the audio content for content-based and mixed music recommendation. Deep learning can effectively extract the features of user items and other contents through deep structure mining features and enhance the recommendation capability. However, the implementation of a deep learning model requires a lot of computing resources and its recommendation effect is closely related to the corpus, which brings the challenge of constructing an ideal corpus. Moreover, due to the lack of information for items with short text characteristics, the effect of directly using deep learning methods is not ideal.

Thirdly, some scholars consider recommendation based on the content of the item. This is the continuation and development of collaborative filtering technology. For example, An et al. proposed content-based personalized recommendation of popular microtopics [15]. Some researchers introduced the LDA model; Zhao et al. proposed a Twitter-LDA model suitable for short texts and gave the topic construction idea of the "single microblog-single topic" [16]. Ben-Lhachemi et al. proposed to use the semantic embedding representation of tweets to help users select relevant tweet topics for their posts in real time and then capture the semantic similarity or relevance between tweets, so as to realize the recommendation of tweet topics [17]. However, due to the small number of words in the short texts of items, the topics of the short text are poor and the problem of topic incompleteness occurs in the process of topic modeling; therefore, the effect of using the topic to extend the feature of the short text is not ideal either.

Therefore, for the items with short text characteristics, since the short text contains little or even the lack of information, directly using the above methods will have problems such as unsatisfactory recommendation effect or excessively complicated implementation. Obviously, if we want to obtain more accurate recommendation results, we must first effectively extend the items with short text characteristics.

3. The Short Text Feature Extension Method Based on DBpedia

There are two main existing text feature extension algorithms applied to short texts: one is based on external documents and search engines [18, 19] and the other is based on the external knowledge base [20–22]. The first method has a higher correlation between the original feature words and the extended features after processing, but it is difficult to implement and time-consuming. At the same time, it depends on the quality of the results returned by the search engine to a certain extent. The second extension method can alleviate the data sparse problem of short texts, but the extension effect depends on the quality of the external knowledge base, and the amount of calculation is large and time-consuming.

Therefore, considering that short texts of item often contain entities and entities have rich meanings, it is a simple and efficient choice to extend from entities. The DBpedia knowledge graph is chosen as the external knowledge source. On the basis of the Li extension method [23], we propose an improved short text extension method to extend the semantic information of the short texts of items and then integrate the semantic features of the item when recommending.

The extension process of short texts of items based on the DBpedia is shown in Figure 1, and it mainly includes the following tasks: first, we use DBpedia Spotlight [24] to identify entities of the short texts of items and express them as entities of DBpedia (there is noise and need to be filtered), so as to get the source feature entity set. Secondly, based on DBpedia to

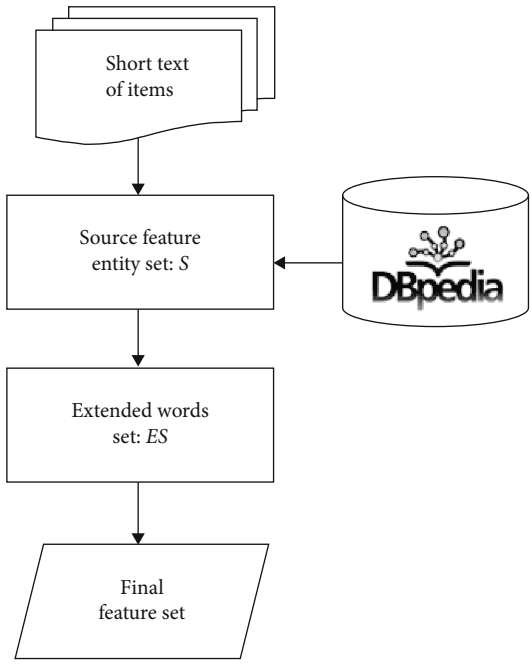

Figure 1: Extension process of short texts of items.

further extend the previously entities, according to entity resource pages of DBpedia, put the values of the Type attribute for these feature entities as the extended entities of the short texts; then, we obtain the extended words set of the short texts.

One of the tasks is as described in Algorithm 1, first, using DBpedia Spotlight, an open-source named the entity recognition system, to identify the named entity in short texts and then generate a set of entity candidates and further disambiguate the candidate entities. Among them, by calculating the labeling probability of named entities in Wikipedia [25], the candidates that are lower than a certain threshold will be deleted. In the disambiguation processing, the method proposed by Han and Sun [26] is used to realize the disambiguation of the entities.

Algorithm 1: Obtain the feature set of items.**Input**: short texts of items

Output: S - source feature entity set of items

1:S=Ø

2: Use DBpedia Spotlight to annotate named entity words in short texts, and calculate their tagging probability values, and get a set (W) of named entity words according to a given threshold. And $W = \{w_1, w_2, \cdots, w_n\}$.

3: for $i = 1$ to n

4: According to the Wikipedia page of w_i, we get the candidate entity set $(E(wi))$ for w_i. And $E(wi) = \{e_{i1}, e_{i2}, \cdots, e_{ik}\}$.

5: Calculate the probability of each candidate entity in $E(w_i)$ being marked as an entity under the current context conditions, and we select the entity with the largest probability to add to S.

6: end for

7: return $S = \{e_1, e_2, \cdots, e_m\}$

In Algorithm 1, we use DBpedia Spotlight to annotate the named entity in the short texts. The threshold of the tagging probability can be set according to your own needs. We refer to the literature [25] and set it to 0.45.

Another task is as described in Algorithm 2, by the entity resource pages in DBpedia; we use the Type attribute value of the entity in the resource page as candidate extension words for the short text; then, the union of the source feature set and the extended words set is taken as the final features set.

Flisar and Podgorelec [27] and others studied the Type, Topic, and Category attributes of the DBpedia knowledge graph entity resource page and believed that the information of the Type attribute is the most effective for the text extension of the entity. In order to avoid introducing ambiguous information, we only consider the Type attribute. This information is obtained from the Infobox of the Wikipedia page while constructing DBpedia. It is closely related to the entity and can achieve high-quality semantic extension.

When choosing an extended entity, we generally decide the choice by comparing the semantic similarity between the entities. There are many ways to measure the semantic similarity of entities, for example, the cosine similarity of the embedding vector of the entity word can be used but the accuracy of the embedding vector depends on the quality of the corpus and training. Therefore, considering that the entity is derived from the Type attribute of source feature entities and these entities are at different levels of the taxonomy structure of the knowledge graph, it is very suitable for passing through the distance of their semantic path and depth to measure the similarity between them. We adopt a simple and efficient method [28], and the semantic similarity is calculated according to equation (1) as follows:

$$\text{Sim\_Dist}(A, B) = \frac{1 - \log\left(\text{Dist}(A, B)\right)}{\log\left(2 * \text{max\_depth}\right)}, \quad (1)$$

where $\text{Dist}(A, B)$ represents the number of edges of the shortest path between entity A and entity B and max_depth represents the maximum depth of the taxonomy structure.

Algorithm 2: Calculate the extended word set of source features.**Input**: $S = \{e_1, e_2, \cdots, e_m\}$ - source feature entity set of items

Output: ES- extended words set

1: $ES = \emptyset$

2: if $|S| = 0$ then output Ø

3: for $j = 1$ to m

4: According to the Type attribute of the resource page in DBpedia, the extended word set of e_j is T_j. And $T_j = \{e_{j1}, e_{j2}, \cdots, e_{jk}{}'\}$.

5: Calculate $Sim\_Dist(A, B)$ between e_j and e_{jl} $(l = 1, \cdots, k')$ according to formula (1), and determine

Input: short texts of items
Output: S - source feature entity set of items
1:S=Ø
2: Use DBpedia Spotlight to annotate named entity words in short texts, and calculate their tagging probability values, and get a set (W) of named entity words according to a given threshold. And $W = \{w_1, w_2, \cdots, w_n\}$.
3: for $i = 1$ to n
4: According to the Wikipedia page of w_i, we get the candidate entity set $(E(wi))$ for w_i. And $E(wi) = \{e_{i1}, e_{i2}, \cdots, e_{ik}\}$.
5: Calculate the probability of each candidate entity in $E(w_i)$ being marked as an entity under the current context conditions, and we select the entity with the largest probability to add to S.
6: end for
7: return $S = \{e_1, e_2, \cdots, e_m\}$

Algorithm 1: Obtain the feature set of items.

Input: $S = \{e_1, e_2, \cdots, e_m\}$ - source feature entity set of items
Output: ES- extended words set
1: $ES = \emptyset$
2: if $|S| = 0$ then output Ø
3: for $j = 1$ to m
4: According to the Type attribute of the resource page in DBpedia, the extended word set of e_j is T_j. And $T_j = \{e_{j1}, e_{j2}, \cdots, e_{jk}{}'\}$.
5: Calculate $Sim\_Dist(A, B)$ between e_j and e_{jl} $(l = 1, \cdots, k^{'})$ according to formula (1), and determine the extended word set $(T_j^{'})$ for e_j according to a given threshold, and $ES = ES \cup T_j^{'}$.
6: end for
7: return ES

Algorithm 2: Calculate the extended word set of source features.

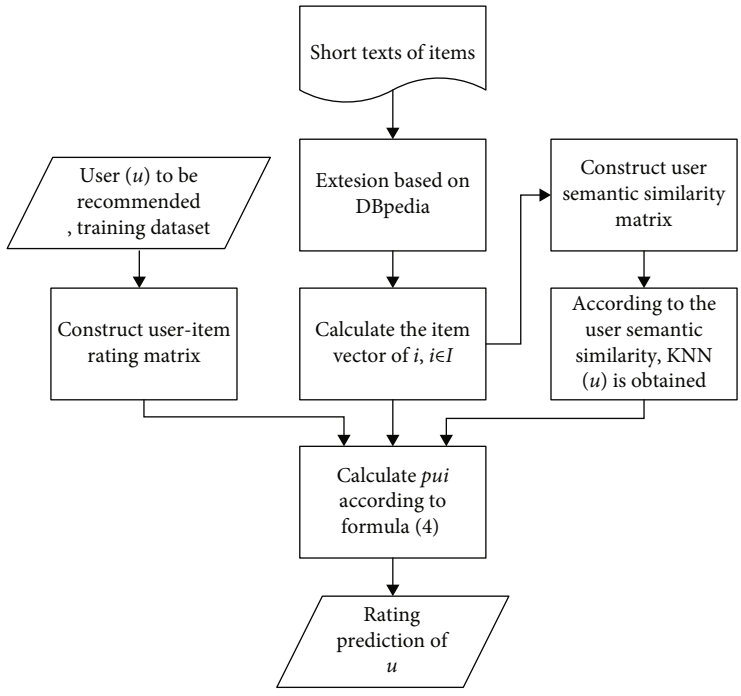

Figure 2: Flow chart of the recommended model.

Input: The user (u) to be recommended，short texts of items;
Output: The prediction rating of the item (i) by the user (u) .
1: For each item $i(i \in I)$, extend short texts of the item based on the DBpedia, get extension feature set $W^{(i)}$.
2: Calculate the item vector of i according to $W^{(i)}$.
3: Construct the user-item rating matrix for dataset.
4: Construct user semantic similarity matrix.
5: Select the set $KNN(u)$ composed of the first K users most similar to u.
6: According to formula (5), calculate prediction rating of i by u (p_{ui}).

Algorithm 3: Recommendation algorithm based on semantic features and DBpedia.

Table 1: Several typical weight settings.

	α	β
Setting 1	0.8	0.2
Setting 2	0.7	0.3
Setting 3	0.6	0.4
Setting 4	0.5	0.5
Setting 5	0.4	0.6
Setting 6	0.3	0.7
Setting 7	0.2	0.8

the extended word set ($T_j^{'}$) for e_j according to a given threshold, and $ES = ES \cup T_j^{'}$.

6: end for

7: return ES

4. Recommendation Model Based on Semantic Features and DBpedia

From Sections 1and 2, we know that most of the current recommendation algorithms ignore important semantic factors. Therefore, we consider semantic features in the recommendation model proposed in this paper. As more and more items (such as microblog, short messages, movie introductions, and product reviews) have short text characteristics, their short text processing cannot simply be processed by traditional text processing methods and usually need to be extended first. But how to extend is also a difficult problem. In this paper, we use the distance of the semantic path and the depth to measure similarity between entities and give an improved method of item short text extension based on DBpedia and then propose a recommendation model based on semantic features (see Figure 2).

4.1. Construct User-Item Rating Matrix. According to the user's rating of the item, a user-item rating matrix can be constructed.

Definition 1. Assume that in the dataset, the user set is $U = \{u_1, u_2, \cdots, u_m\}$ (m users) and the item set is $I = \{I_1, I_2, \cdots, I_n\}$ (n items); then, the user-item rating matrix $\mathbf{R}$ is defined as a matrix with m rows and n columns, each row represents a user, and the columns correspond to different items and the corresponding elements represent the ratings given by the user for the item, namely,

$$R = \begin{bmatrix} r_{11} & \cdots & r_{1n} \\ \vdots & \ddots & \vdots \\ r_{m1} & \cdots & r_{mn} \end{bmatrix}. \quad (2)$$

4.2. User Semantic Similarity Calculation. The short texts of items are extended based on DBpedia after data cleaning, word splitting, stopping word removal, stemming, and other preprocessing; then, we use the word2vec [29] toolkit to deal with the extension word set, so as to get an embedding representing the given dimension of each word. And further, we calculate the vector of the short texts for each item; for simplicity, the average of word embedding is used to represent the vector of the item [30].

Definition 2. Given item q, if the word set of its short text after the above extension is $T = \{w_1, w_2, \cdots, w_i, \cdots, w_k\}$, for $\forall w_i \in T$, assuming that the corresponding word embedding of a given dimension obtained by word2vec processing is $\vec{w_i}$ ($i = 1, \cdots, k$), then, the item vector of q is defined as follows:

$$\vec{q} = \text{average}\left(\vec{w_1}, \cdots, \vec{w_i}, \cdots, \vec{w_K}\right). \quad (3)$$

Definition 3. Assume that the first s items whose rating of u are greater than a given value are selected as $q_1, q_2, \cdots, q_s$ and the corresponding vector of q_i is $\vec{q_i}$ ($i = 1, \cdots, s$), we define the representation vector of u as follows:

$$\vec{u} = \text{average}\left(\vec{q_1}, \cdots, \vec{q_i}, \cdots, \vec{q_s}\right). \quad (4)$$

Definition 4. Assume that the representation vectors of u and v are $\vec{u}$ and $\vec{v}$ (u and v are users), respectively; we define the semantic similarity between u and v as follows: Sim_Sem(u, v) = cos ($\vec{u}$, $\vec{v}$).

4.3. Predict the Ratings of Items. In the model proposed in this paper, because some users may tend to give high or low ratings to all items when rating, we provide the relative difference of ratings in the prediction to make the results more reasonable. Therefore, we use a strategy based on the average of user ratings to calculate prediction ratings of items.

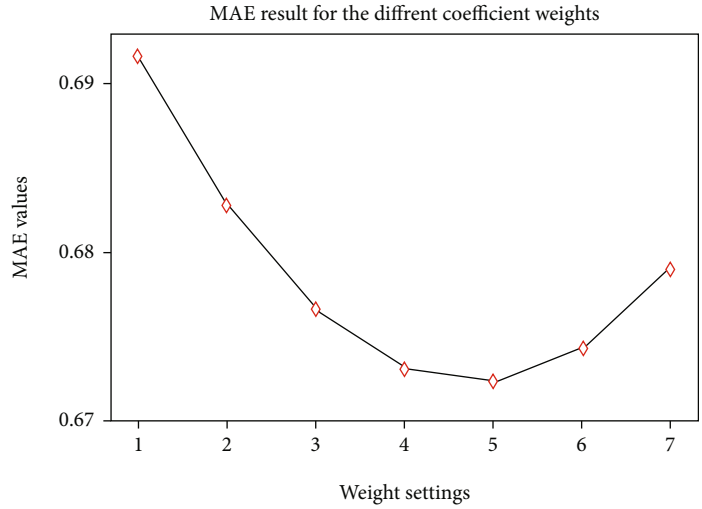

Figure 3: MAE results for the different coefficient weights.

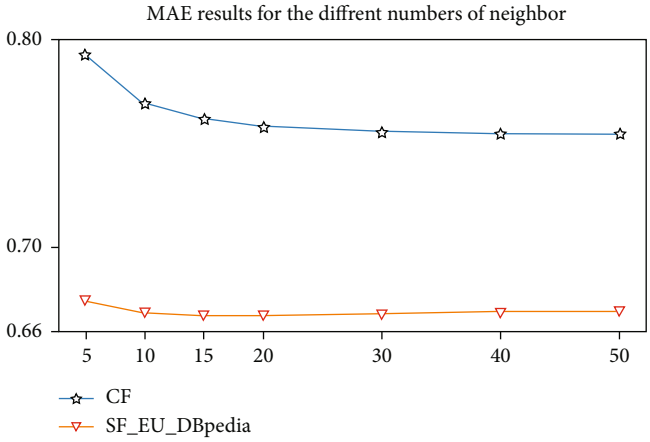

Figure 4: MAE results for the different numbers of the nearest neighbor.

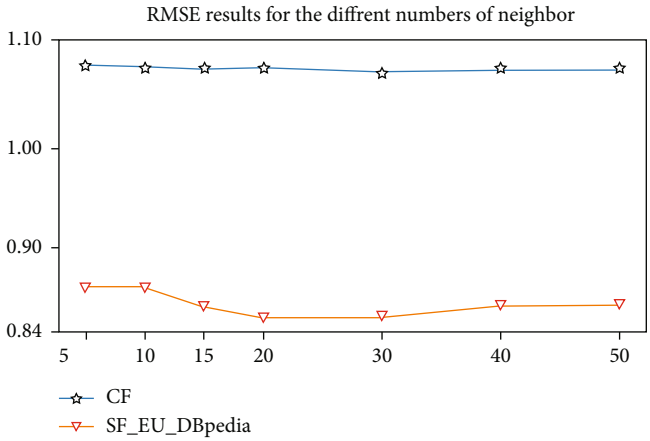

Figure 5: RMSE results for the different numbers of the nearest neighbor.

Specifically, according to user-item rating matrix and the semantic features of the short texts of items, we integrate the u's semantic similarity to calculate the user u's prediction rating p_{ui} for item i. The formula is as follows:

$$p_{ui} = \alpha * p'_{ui} + \beta * p''_{ui}, \tag{5}$$

where α and β are the weight coefficients and $\alpha + \beta = 1$. The formulas of p'_{ui} and p''_{ui} are defined as follows:

$$p'_{ui} = \bar{r_u} + \frac{\sum_{v \in \mathrm{KNN}(u)} \mathrm{Sim\_Sem}(u, v) * (r_{vi} - \bar{r_v})}{\sum_{v \in \mathrm{KNN}(u)} \mathrm{Sim\_Sem}(u, v)}, \tag{6}$$

$$p''_{ui} = \frac{\sum_{j \in \mathrm{KNN}(i)} \cos\left(\vec{i}, \vec{j}\right) * r_{uj}}{\sum_{j \in \mathrm{KNN}(i)} \cos\left(\vec{i}, \vec{j}\right)}, \tag{7}$$

where in equation (6), $\mathrm{Sim\_Sem}(u, v)$ is semantic similarity and $\mathrm{KNN}(u)$ represents the set of the K-nearest neighbor of u (calculated by semantic similarity) and $\bar{r_u}$ represents the average of the ratings generated by u and r_{vi} represents the rating of i for v (v is a neighbor of u, and $\bar{r_v}$ represents the average of the ratings generated by v.

In equation (7), $\mathrm{KNN}(i)$ represents the set of the K-nearest neighbor of i (calculated by semantic similarity) and $\vec{i}$ and $\vec{j}$ are the vectors corresponding to **i** and **j**, respectively.

Algorithm 3 gives the specific process of the proposed model.

Algorithm 3: Recommendation algorithm based on semantic features and DBpedia.**Input**: The user (u) to be recommended, short texts of items;

Output: The prediction rating of the item (i) by the user (u) .

1: For each item $i(i \in I)$, extend short texts of the item based on the DBpedia, get extension feature set $W^{(i)}$.

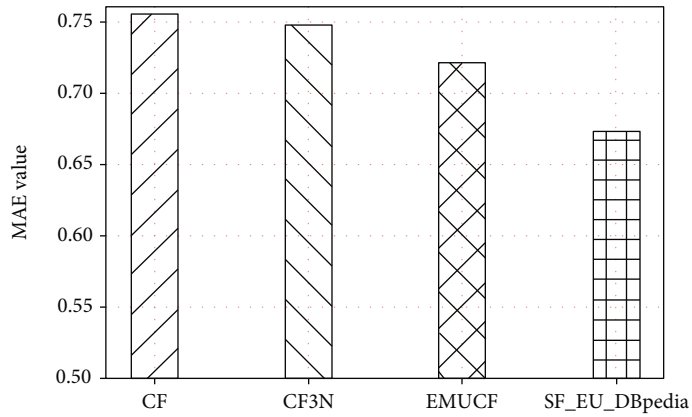

Figure 6: Optimal MAE results for the different models.

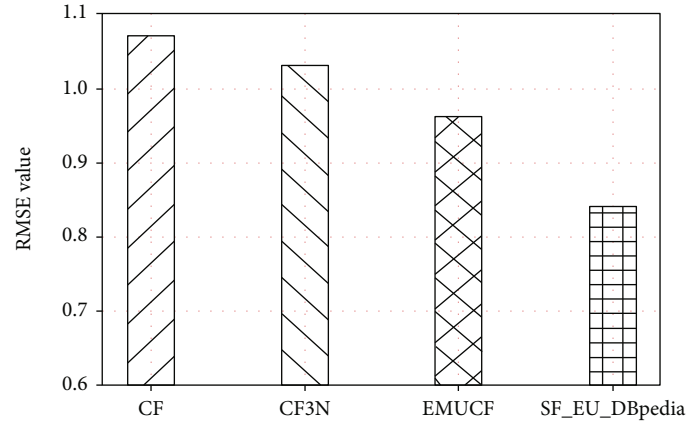

Figure 7: Optimal RMSE results for the different models.

2: Calculate the item vector of i according to $W^{(i)}$.

3: Construct the user-item rating matrix for dataset.

4: Construct user semantic similarity matrix.

5: Select the set $KNN(u)$ composed of the first K users most similar to u.

6: According to formula (5), calculate prediction rating of i by u (p_{ui}).

5. Experiment and Result Analysis

The experiment in this paper uses the MovieLens1M dataset published by UCI for experimentation. MovieLens was developed by the GroupLens project team of the University of Minnesota. It contains 6040 users' ratings of 3952 movies, with a total of 1000209 ratings. Each user will rate at least 20 movies (the rating value is an integer between 0–5 and 0 means that the user did not rate the movie) and also provides auxiliary information such as the user's occupation, movie category, and movie duration. The sparsity of this dataset is 95.80%.

The movie data (movies.dat) in the dataset includes fields such as MovieID, Title, and Genres, which represent the movie ID, movie name, and movie style, respectively. In this experiment, data in the two fields of Title and Genres of the movie data is considered to be the short texts of the corresponding movie item. In addition, the user data (user.dat) is randomly divided into 90% training set and 10% test set according to the user ID.

We use genism as the word embedding calculation toolkit in the experiment, which is an open-source third-party python toolkit. It supports a variety of model algorithms including word2vec and streaming training and provides API for common tasks such as similarity calculation and information retrieval.

The hardware environment used in the experiment settings is as follows: CPU is Intel® Core™ i7-8700, and the memory is 16 G Double Data Rate Fourth SDRAM and equipped with 4 TB hard drive and 128 G Solid State Disk. The software environment is a Windows10 operating system, Python3.8, and Gensim development platform.

5.1. Evaluation Metrics. Since the MovieLens1M dataset contains the user's rating for each item, so we can train and learn the user's prediction rating in experiment.

We use mean absolute error (MAE) and root mean square error (RMSE) as evaluation metrics of the experiment, which are widely used in recommendation systems. These metrics can measure the error between the user's actual

rating and the prediction rating. When the MAE and RMSE are smaller, the error between the prediction rating and the actual rating is smaller and the prediction rating accuracy of the algorithm is higher. For u (u is a user) to be recommended, let T be the set of items in the dataset where users have rating behavior and $|T|$ is the specific number of T. For $\forall i \in T$, let r_{ui} denote u's actual rating for i and p_{ui} is the prediction rating obtained by the model proposed in this paper. MAE and RMSE are defined as follows:

$$\begin{aligned} \mathrm{MAE} &= \frac{\sum_{i \in T} |r_{ui} - p_{ui}|}{|T|}, \\ \mathrm{RMSE} &= \sqrt{\frac{1}{|T|} \sum_{i \in T} (r_{ui} - p_{ui})^2}. \end{aligned} \tag{8}$$

5.2. Coefficient Determination. The recommendation model based on semantic features and extension using DBpedia includes weight coefficients α and β which appear in equation (5), and the value of the parameters will affect the quality of the recommendation result.

In equation (5), α and β are weight coefficients and $\alpha + \beta = 1$.

The prediction ratings of items combine the users' semantic features. Therefore, we select several typical weight combinations (as listed in Table 1) and set the number of the nearest neighbor to 20 for experiment and, then, calculate the MAE values of different coefficient weights. The results are shown in Figure 3. It can be found that the recommendation effect of the model proposed in this paper is the best when the weights are set to the fifth case (that is, $\alpha = 0.4$, $\beta = 0.6$).

5.3. Analysis of Results. The experiment is mainly divided into two parts:

(i) Analyze the recommendation effect of the traditional collaborative filtering technology and the model proposed in this paper on the MovieLens1M dataset and consider different numbers of the nearest neighbor.

(ii) Compare the recommendation effect of the model in this paper and some other models. Considering that some of these models are not tested based on the number of the nearest neighbor, the best experimental data of various models are selected.

Experiment 1. We compared the recommendation effect of the traditional collaborative filtering recommendation technology (CF) and the model proposed in this paper (SF_EU_DBpedia) at different nearest neighbor numbers (including $K = 5$, 10, 15, 20, 30, 40, and 50). We use experiments to investigate the impact of different recommendation models and different numbers of the nearest neighbor on MAE and RMSE, and the results are shown in Figures 4 and 5 as follows:

As shown in Figure 4, the MAE values for the method proposed in this paper (SF_EU_DBpedia) are always much better than the traditional collaborative filtering recommendation technology (CF) at different numbers of the nearest neighbor and so are the RMSE in Figure 5.

Experiment 2. In the experiment, we choose the traditional collaborative filtering recommendation technology (CF), collaborative filtering recommendation integrating the user-centric natural nearest neighbor (CF3N) [5], enhanced multistage user-based collaborative filtering through nonlinear similarity (EMUCF) [12], and deep neural network-based recommendation algorithm (DNN) [31] in comparison with the method proposed in this paper (SF_EU_DBpedia), since the EMUCF and DNN do not perform experiments based on the numbers of the nearest neighbor; for comparison, we select the best experimental results of various algorithms. The experimental results are shown in Figures 6 and 7 as follows:

In Figure 6, the optimal MAE obtained by the EMUCF is about 0.7211 and the method proposed in this paper is 0.6723. In Figure 7, the optimal RMSE obtained by the DNN is about 0.9631 and that by the method proposed in this paper is 0.8442. The experimental results show that the method proposed in this paper is superior to the abovementioned algorithms on MAE and RMSE.

6. Conclusion

We proposes a recommendation model based on semantic features and a knowledge graph and first select DBpedia as a knowledge graph to extend short text features of items and then integrate semantic features to calculate the prediction rating of the user to be recommended. Experiment results are shown that the proposed model in this paper works well.

As future works, we are planning to calculate the semantic vector that characterizes users for various items of different categories, so as to further improve the general applicability of the model. In addition, we can consider more factors for selecting nearest neighbors, such as user semantic similarity and user characteristics; thus, we can choose suitable factors to achieve more accurate personalized recommendation according to categories of items.

Data Availability

The data included in this paper are available without any restriction.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

We wish to express our appreciation to the reviewers for their helpful suggestions which greatly improved the presentation of this paper. This work was supported by the project of the Guangzhou Science and Technology Bureau, China, under Grant no. 202007040006.

References

[1] M. Balabanović and Y. Shoham, "Fab," *Communications of the ACM*, vol. 40, no. 3, pp. 66–72, 1997.

[2] J. Sandvig, B. Mobasher, and R. Burke, "Robustness of collaborative recommendation based on association rule mining," in *Proceedings of the 2007 ACM conference on Recommender systems - RecSys '07*, pp. 105–112, Minneapolis, MN, US, 2007.

[3] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 6, pp. 734–749, 2005.

[4] V. Moscato, A. Picariello, and A. M. Rinaldi, "Towards a user based recommendation strategy for digital ecosystems," *Knowledge Based Systems*, vol. 37, no. 2, pp. 165–175, 2013.

[5] Y. Wang, X. Wang, and W. Tang, "Collaborative filtering recommendation integrating user-centric natural nearest neighbor," *Computer Engineering and Applications*, vol. 54, no. 7, pp. 77–83, 2018.

[6] Y. Huang, X. Gao, and S. Gu, "UARR: a novel similarity measure for collaborative filtering recommendation," *Cybernetics and Information Technologies*, vol. 13, pp. 122–130, 2013.

[7] K. Choi, D. Yoo, G. Kim, and Y. Suh, "A hybrid online-product recommendation system: combining implicit rating-based collaborative filtering and sequential pattern analysis," *Electronic Commerce Research and Applications*, vol. 11, no. 4, pp. 309–317, 2012.

[8] V. Zheng, Y. Zheng, X. Xie, and Q. Yang, "Towards mobile intelligence: learning from GPS history data for collaborative recommendation," *Artificial Intelligence*, vol. 184-185, no. 2, pp. 17–37, 2012.

[9] Qi Liu, Enhong Chen, Hui Xiong, C. H. Q. Ding, and Jian Chen, "Enhancing collaborative filtering by user interest expansion via personalized ranking," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 1, pp. 218–233, 2012.

[10] J. Zhang, Q. Peng, S. Sun, and C. Liu, "Collaborative filtering recommendation algorithm based on user preference derived from item domain features," *Physica A: Statistical Mechanics and its Applications*, vol. 396, no. 2, pp. 66–76, 2014.

[11] Y. Li, C. Zhai, and Y. Chen, "Exploiting rich user information for one-class collaborative filtering," *Knowledge and Information Systems*, vol. 38, no. 2, pp. 277–301, 2014.

[12] A. Jain, S. Nagar, P. Singh, and J. Dhar, "_EMUCF_ : Enhanced multistage user-based collaborative filtering through non- linear similarity for recommendation systems," *Expert Systems with Applications*, vol. 161, article 113724, 2020.

[13] K. Georgiev and P. Nakov, "A non-IID framework for collaborative filtering with restricted Boltzmann machines," in *International conference on machine learning*, pp. 1148–1156, Atlanta, Georgia, USA, 2013.

[14] X. Wang and Y. Wang, "Improving content-based and hybrid music recommendation using deep learning," in *Proceedings of the 22nd ACM international conference on Multimedia*, pp. 627–636, San Francisco, California, USA, 2014.

[15] Y. An, B. Li, R. Yang, and L. Hu, "Content-based personalized recommendation on popular micro-topic," *Journal of Intelligence*, vol. 2, pp. 155–160, 2014.

[16] W. X. Zhao, J. Jiang, J. Weng et al., "Comparing Twitter and traditional media using topic models," in *Advances in Information Retrieval*, pp. 338–349, Springer, Berlin Heidelberg, 2011.

[17] N. Ben-Lhachemi and E. Nfaoui, "Using tweets embeddings for hashtag recommendation in Twitter," *Procedia Computer Science*, vol. 127, pp. 7–15, 2018.

[18] Y. Guo, *Feature Expansion Method for Short Text Classification*, Harbin Institute of Technology, 2013.

[19] L. Xiangdong, C. Huan, D. Cong, and H. Li, "Short-text classification based on HowNet and domain keyword set extension," *New Technology of Library and Information Service*, vol. 31, no. 2, pp. 31–38, 2015.

[20] Y. Ning, X. Fan, and Y. Wu, "Short text classification based on domain word ontology," *Computer Science*, vol. 36, no. 3, pp. 142–145, 2009.

[21] J. Flisar and V. Podgorelec, "Improving short text classification using information from DBpedia ontology," *Fundamenta Informaticae*, vol. 172, no. 3, pp. 261–297, 2020.

[22] C. Bizer, J. Lehmann, G. Kobilarov et al., "DBpedia - a crystallization point for the Web of Data," *Journal of Web Semantics*, vol. 7, no. 3, pp. 154–165, 2009.

[23] C. Li, *Research on Short Text Classification Based on Knowledge Graph and Deep Learning*, South China Normal University, 2020.

[24] P. N. Mendes, M. Jakob, A. García-Silva, and C. Bizer, "DBpedia spotlight: shedding light on the web of documents," in *Proceedings the 7th International Conference on Semantic Systems, I-SEMANTICS 2011*, pp. 1–8, Graz, Austria, 2011.

[25] Y. Fu, X. Wang, Z. Feng, and X. Qiang, "Named entity recognition optimization on DBpedia spotlight," *Journal of Frontiers of Computer Science and Technology*, vol. 11, no. 7, pp. 1044–1055, 2017.

[26] X. Han and L. Sun, "A generative entity-mention model for linking entities with knowledge base," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pp. 945–954, Portland, Oregon, 2011.

[27] J. Flisar and V. Podgorelec, "Document enrichment using DBPedia ontology for short text classification," in *Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics*, pp. 1–9, Novi Sad, Serbia, 2018.

[28] V. N. Garla and C. Brandt, "Semantic similarity in the biomedical domain: an evaluation across knowledge sources," *BMC Bioinformatics*, vol. 13, no. 1, pp. 261–274, 2012.

[29] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013, https://arxiv.org/abs/1301.3781.

[30] M. Tang, L. Zhu, and X. Zou, "Document vector representation based on Word2Vec," *Computer Science*, vol. 43, no. 6, pp. 214–217, 2016.

[31] J. Bi, Y. Liu, and Z. Fan, "A deep neural networks based recommendation algorithm using user and item basic data," *International Journal of Machine Learning and Cybernetics*, vol. 11, no. 4, pp. 763–777, 2020.