Recommendation of Knowledge Graph Convolutional Networks Based on Multilayer BiLSTM and Self-Attention

Yao Qiu, Yajie Liu, Ying Tong, and Xuyu Xiang

School of Computer and Information Engineering, Central South University of Forestry and Technology, Changsha, Hunan 410004, China

Correspondence should be addressed to Yao Qiu; 20191100317@csuft.edu.cn

Received 14 June 2022; Accepted 22 July 2022; Published 7 September 2022

Academic Editor: Shadi Aljawarneh

Copyright © 2022 Yao Qiu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

To solve the problems of cold start, sparse data, and poor recommendation performance in collaborative filtering recommendation, an end-to-end framework algorithm based on BiLSTM and BAGCN was proposed. In order to discover the higher-order structural information in the knowledge graph, stacked BiLSTM is used to extract the features of embedded entities and relationships, respectively, and the depth dependence features of user-item interaction matrix are mined. The neighborhood representation of each entity is then calculated by sampling adjacent entities of a fixed size. Then, the self-attention mechanism is used to learn the semantic association between entities and neighboring entities to obtain the final neighborhood information. Aggregators are used to combine neighborhood information and bias information when computing node representations. By extending the sampling of adjacent entities to multihop simulation of higher-order adjacent information, users’ potential long-distance interests can be captured. Compared with the baseline model, the superiority of this method is verified.

1. Introduction

With the explosive growth of Internet information, users are faced with the problem of information overload [1], and traditional search engines have been unable to meet users’ retrieval needs, so the recommendation system have emerged as the times require.

Traditional methods, such as collaborative filtering (CF) [2], exploit the entire user-item interaction matrix to mine users’ interest, which suffer from cold start and data sparsity problems. The matrix factorization (MF) model [3] added the concept of latent vector to the existing matrix, strengthening the model’s ability to deal with sparse data. Koren [4] put forward SVD, which transforms both items and users into the same hidden factor space. The space tries to explain the rating by describing the items and users by the factors that are automatically inferred from user feedback. He et al. [5] proposed the matrix factorization model NCF based on the neural network structure, which takes into account both the user’s explicit rating and implicit feedback on the item. Followed by this, Guo et al. [6] proposed DeepFM to explicitly describe user preferences for different factors. The above methods model each pair of user-item interaction data as an independent data instance and do not consider the association between them, so attribute-based collaboration information cannot be extracted from user behavior.

The graph structure can well describe the degree of association between data, and knowledge graph (KG) is a natural graph data that contains a lot of heterogeneous information [7]. The essence of KG is a large-scale semantic network, which contains rich semantic features among items. It can help to discover users’ potential interest as auxiliary information for recommender systems. At the same time, the data with semantic correlation can make the recommendation results interpretable while learning.

The current mainstream knowledge graph-based recommendation can roughly be divided into two categories: path-based methods and embedding-based and joint-based methods. Embedding-based methods all use the knowledge graph embedding (KGE) method [8] to map entity vectors or
relation vectors to a low-dimensional vector space, such as CKE proposed by Zhang et al. [9] and DKN proposed by Wang et al. [10]. However, this method ignores the connectivity of information in KG and lacks interpretability.

Path-based methods use the user-item graph to calculate the path similarity of users or items by predefining meta-paths and use semantic connectivity in KG for recommendations. The Hete-MF, proposed by Yu et al. [11], utilized $L$ different metapaths to find the similarity between items on each path. Luo et al. [12] proposed Hete-CF, which uses the similarity between users and items together as a regularization to find items of interest for users. The PER algorithm treats KG as a heterogeneous information network and extracts latent features from metapaths to represent the connectivity in different relational paths between users and items. However, the type and number of metapaths of such methods need to be manually defined, and the performance is easily affected [13].

The joint method combines the above two methods, which not only makes full use of the semantic information in KG but also inherits the interpretability of the path-based method. For example, RippleNet [14] used the embedding-based propagation idea to model users by analogizing the propagation process of user preferences to ripple diffusion. Subsequently, Wang et al. [15] proposed the KGCM model, which characterized nodes by mining the association attributes between entities on the KG, capturing the correlation between items and aggregating the information of neighbor nodes. However, this scheme is greatly affected by sparse data, and the representation of nodes is not accurate enough, so the performance of model prediction needs to be further improved.

Therefore, in order to solve the above problems, this paper proposes a recommendation algorithm for knowledge graph convolutional networks based on multilayer BiLSTM and self-attention mechanism. Long and short-term neural network is a kind of recurrent neural network, while bidirectional short and long-term memory network is divided into two independent long and short-term neural network. First, we apply the stacked multilayer BiLSTM to perform feature extraction on the initial entity and relationship vectors and then combine the learned entity feature vector with the initial vector to obtain the entity’s own vector, that is, the 0-order representation. Finally, we realize high-order information representation through information transfer. The influence of data sparseness on the model is slowed down to a certain extent by adding its own information. Meanwhile, the self-attention mechanism can further learn the relationship between entities and adjacent entities, making the representation of nodes more accurate and effectively improving the performance of the recommendation system.

2. The Recommendation Algorithm Based on Multilayer BiLSTM and Self-Attention Mechanism

The overall framework of our method is shown in Figure 1, which is divided into a knowledge graph embedding layer, a neighborhood information calculation layer, and a
2.1. Problem Description. In a recommendation scenario, a user set and an item set are given, which is denoted as \( U = \{ u_1, u_2, \ldots, u_m \} \) and \( I = \{ i_1, i_2, \ldots, i_n \} \). The interaction matrix of users and items is \( Y \in \mathbb{R}^{m \times n} \), interaction data is defined as \((u, y_{ui}, i)\), and \( m \) and \( n \) are the number of users and items, respectively. \( y_{ui} = 1 \) indicates that there is interaction between the user and the item, which is regarded as a positive example of user-item interaction; otherwise, it indicates that no interaction has occurred, which is regarded as a negative example. Given a knowledge graph \( G \) and a user-item interaction matrix \( Y \), we need to predict the probability that a user \( u \) will interact with an item \( i \) without interacting before. The learning objective function can be expressed as \( \bar{y}_{ui} = P(u, i; \theta) \), where \( \bar{y}_{ui} \) represents the probability that the user will interact with the item and \( \theta \) represents all model parameters of the function.

2.2. BAGCN Layer. The BAGCN algorithm is widely used in accurately capturing the higher-order structural proximity between entities in knowledge graphs and the semantic association of relations between entities and adjacent entities. The self-attention mechanism is actually a kind of network attention mechanism that is actually a kind of network configuration \([16, 17]\).

2.2.1. Knowledge Graph Embedding. Knowledge graph embedding is an effective way to convert entities and relations into vector representations, which can transfer high-dimensional sparse features into low-dimensional feature vectors, to obtain a more convenient form for model input. The user, \( U = \{ u_1, u_2, \ldots, u_m \} \), obtains the corresponding vector representation by querying the user embedding matrix \( I \in \mathbb{R}^{m \times s} \), where \( s \) is the dimension of the embedding vector. Given a candidate pair of user \( u \) and item (entity) \( v \), the entity vector and the relation vector get the corresponding initial embeddings \( \bar{v}^{(0)} \) and \( \bar{r} \) in the entity embedding matrix and the relation embedding matrix through embedding lookup.

In this paper, we apply the multilayer BiLSTM to learn features for entity and relation embeddings. The single-layer BiLSTM has achieved good results in simple prediction tasks, but the potential interest of users is often deeply hidden in the user-item interaction matrix. Therefore, this paper proposes to use the stacked multilayer time series network BiLSTM to extract the deep dependency features in the user and item interaction matrix to mine the potential interest of users. A single-layer BiLSTM \([18]\) is composed of a forward LSTM and a backward LSTM, the former calculates the hidden layer state \( h(h_1, h_2, \ldots, h_t) \) and the latter calculates the hidden layer state \( h(h_1, h_2, \ldots, h_t) \), which can be spliced to obtain the final hidden layer state. The multilayer BiLSTM network is composed of the forward multilayer LSTM and backward multilayer LSTM, and the input of the \( N \)-th layer is the output of the \( N-1 \)-th layer. The model structure can be seen in Figure 2.

First, take \( \bar{v}^{(0)} \) and \( \bar{r} \) as the input of the multilayer BiLSTM model for feature extraction and then splice the extracted entity embedding vector with the initial embedding vector \( \bar{v}^{(0)} \) to obtain the 0th-order information \( v^{(0)} \) of the entity, that is, the representation vector of the entity itself. Its process can be formulated as:

\[
\begin{align*}
X_n &= B(X_{n-1}), \\
X_{n-1} &\leftarrow \bigcup <X_{n-1}^c, X_{n-1}^m>, \\
R_n &= B(X_{n-1}) \cup B(X_{n-2}), \\
X_{n-1}^c &\leftarrow \bigcup <X_{n-1}^{c1}, X_{n-1}^{c2}, \ldots, X_{n-1}^m>, \\
v^{(0)} &= \bar{v}^{(0)} + X_n,
\end{align*}
\]

where \( X_n \) and \( R_n \) are the embedded representations of entities and relations obtained after feature extraction, respectively. \( C_1, C_2, \ldots, C_m \) represents hidden layer nodes, and the number of internal hidden layer nodes is determined by the dimensions of entities and relationships. \( \bigcup \) indicates the feature stacking symbol.

Since the input of BiLSTM contains the output of forward and backward LSTM, we sum and average the forward and backward features to effectively utilize the forward and backward feature information, which can be formulated as:

\[
X_{n-1}^{c1} = \frac{X_{n-1}^{\text{Forward}} + X_{n-1}^{\text{Backward-Output}}}{H},
\]

where \( H \) is the number of hidden layer nodes and \( N \) is the total number of cells. \( X_{n-1}^{\text{Forward}} \) represents the forward LSTM output vector and \( X_{n-1}^{\text{Backward-Output}} \) represents the backward LSTM output vector.

2.2.2. Neighborhood Information Calculation. The aggregation calculation of neighborhood information needs to integrate the information of neighboring nodes. Specifically, \( N(v) \) represents the set of neighbor nodes directly connected to entity \( v \), \( r_{x,p} \) indicates the relationship between \( e_x \) and \( e_p \), and the neighborhood of entity \( v \) is represented as \( v^{(p)}_{N(v)} \). When calculating the information representation of the neighborhood, the inner product function \( z: \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R} \) is used to calculate the weight matrix between the user vector and the adjacency vector.

\[
A = z(u, r),
\]

where \( d \) represents the dimension and \( A \) represents the impact of different relationships on users by adding user information. In order to characterize the neighborhood structure of entity \( v \), first calculate the linear combination of neighborhood as:

\[
v^\mu_{N(v)} = \sum_{e \in P(v)} \bar{A} e,
\]

where \( \bar{A} \) is the normalized matrix of \( A \).

\[
\bar{A} = \frac{\exp(A)}{\sum_{e \in P(v)} \exp(A)},
\]
where \( e \) is the vector representation of the neighbor entity.

### 2.2.3. Self-Attention

Attention mechanism [19, 20] was initially applied in the field of machine translation, and now, it has been widely applied in image processing, recommendation system, and other aspects. This method draws on the mechanism of human visual selective attention, and its core purpose is to screen out the important content of the current task from a large amount of information. The self-attention mechanism proposed in this paper consists of a gated recurrent unit (GRU) module and a self-attention module. The calculated neighborhood representation \( \tilde{v}_P(v) \) is taken as the input of self-attention, and the aggregation features of lower-dimensional neighborhood information are extracted through the GRU module. Input the obtained feature vector into the self-attention module to further learn the semantic association between entities and adjacent entities and more accurately calculate the representation of neighborhood information. The specific implementation process is as follows:

Firstly, \( \tilde{v}_P(v) \) is sent to the GRU module as the input feature to extract the low-dimensional feature, and the output feature \( I \) is obtained through the learning of function \( h \).

\[
I = h(\tilde{v}_P(v) | u, r).
\]

(6)

Then, the function body Atten is created, which is consistent with the implementation process of ordinary self-attention mechanism. The feature vector \( I \) processed by the GRU module is used as the input of the self-attention module, and the specific operation is shown in the following formula:

\[
\begin{align*}
\nu &= \tanh(I \bullet W^a + b), \\
\nu' &= \nu \bullet u, \\
\alpha &= \text{soft max}(\nu r), \\
O &= I \ast \alpha,
\end{align*}
\]

(7)

where \( \bullet \) is the tensor-dot operation, \( W^a \) is the trainable transformation matrix, \( b \) and \( u \) are vectors for training, \( \alpha \) is the normalized weight, and \( O \) is the final output feature of self-attention.

Finally, a full connection layer is used to obtain the final neighborhood information representation \( \tilde{v}^\mu_P(v) \), and the specific calculation process is as follows:

\[
\tilde{v}^\mu_P(v) = FC(O),
\]

(8)

where FC is the operation function of the full connection layer.

### 2.2.4. Information Aggregation

The last step of the BAGCN layer is information aggregation. The model aggregates the node’s own information \( v^{(0)} \) and all its neighborhood information \( \tilde{v}^\mu_P(v) \) as the final representation of the node through the aggregator, takes the sum of the two vectors, and then performs nonlinear transformation on them. Here, \(|J(v)| = K\), \( K \) is a configurable constant, representing the

![Figure 2: BiLSTM structure with multiple layers stacked.](image-url)
number of neighbors sampled. Technically, if $|J(v)| < K$, this paper samples with a put back; otherwise, it randomly samples fixed $K$ neighbor nodes. In the real knowledge graph, there are great differences in the number of neighbor nodes of different entities. In this paper, the complete set of neighbor nodes is not included in the calculation, but a fixed size neighborhood set is sampled for each entity to ensure the same calculation mode and calculation efficiency of each batch. Its calculation is as follows:

$$\text{agg} = \sigma(W \cdot (v(0) + \bar{y}_{J(v)}) + b),$$

(9)

where $W$ and $b$ are transformation weights and bias terms, respectively, and $\sigma$ is a nonlinear function.

The initial embedding representation of an entity is zero-order, and it is propagated to its neighbors to obtain first-order representation. The process is repeated from first-order representation to higher-order representation. The high-order representation can tap deeper potential interest of users.

### 2.3. Prediction Layer.

The prediction layer predicts the probability of interaction item $i$ of user $u$ by inputting a function $f: R^d \times R^d \rightarrow R$, as shown in the following formula:

$$\bar{y}_{ui} = f(u, i).$$

(10)

### 2.4. Loss Function.

The cross-entropy loss function is used to train the model in this paper, and the specific formula is as follows:

$$\text{loss} = \sum_{u \in U} \sum_{i: y_{ui} = 1} J(y_{ui}, \bar{y}_{ui}) - \sum_{n=1}^{T_{u}} E_{i_n} \sim Q(v_n) J(y_{ui_n}, \bar{y}_{ui_n}) + \lambda \|F\|_2^2,$$

(11)

where $J$ is the cross-entropy loss, $Q$ is the negative sampling distribution, and $T_{u}$ is the negative sample number of user $u$. In this paper, $T_{u} = |\{v: y_{uv} = 1\}|$, $Q$ follows uniform distribution and the last term is $L2$ regularization.

### 3. Experimental Results and Analysis

#### 3.1. Datasets.

We evaluate the proposed BAGCN model by recommending movies, books, and music on three datasets, i.e., MovieLens-20M, Book-Crossing, and Last.FM, respectively. The triple information of constructing the knowledge graph of each dataset comes from Microsoft Satori, and a subset of triples is selected in the entire knowledge graph, whose confidence is greater than 0.9. Table 1 records the basic information of the three datasets.

| Table 1: Basic data set information. |
|------------------------------------|
| **User-item’ interactive information** | **MovieLens-20M** | **Book-crossing** | **Last.FM** |
| Users | 138159 | 19676 | 1872 |
| Items | 16954 | 20003 | 3846 |
| Interactions | 13501622 | 172576 | 42346 |
| **KG information** | | | |
| Entities | 102569 | 25787 | 9366 |
| Relations | 32 | 18 | 60 |
| KG triples | 499474 | 60787 | 15516 |

Table 2: Hyperparameter settings.

| Hyperparameter | MovieLens-20M | Book-crossing | Last.FM |
|----------------|--------------|---------------|--------|
| $K$ | 4 | 8 | 16 |
| $d$ | 8 | 16 | 8 |
| $\lambda$ | $10^{-7}$ | $2 \times 10^{-5}$ | $10^{-4}$ |
| $\eta$ | $2 \times 10^{-2}$ | $2 \times 10^{-4}$ | $5 \times 10^{-4}$ |
| $H$ | 1 | 1 | 1 |
| Batch_size | 65536 | 256 | 128 |

Table 3: Model performance on MovieLens-20M.

| Model | ACC | AUC | F1 | Pre@1 | Recall@1 |
|-------|-----|-----|----|-------|----------|
| SVD [4] | — | 0.9630 | 0.9190 | — | — |
| CKE [9] | — | 0.9240 | 0.8710 | — | — |
| PER [13] | — | 0.8320 | 0.7880 | — | — |
| KGCN [15] | 0.9235 | 0.9720 | 0.9242 | 0.9165 | 0.9321 |
| BAGCN | 0.9335 | 0.9780 | 0.9343 | 0.9228 | 0.9461 |

Table 4: Model performance on Book-Crossing.

| Model | ACC | AUC | F1 | Pre@1 | Recall@1 |
|-------|-----|-----|----|-------|----------|
| SVD [4] | — | 0.6720 | 0.6350 | — | — |
| CKE [9] | — | 0.6770 | 0.6110 | — | — |
| PER [13] | — | 0.6170 | 0.5620 | — | — |
| KGCN [15] | 0.6010 | 0.6471 | 0.6159 | 0.5975 | 0.6403 |
| BAGCN | 0.6268 | 0.6791 | 0.6425 | 0.6203 | 0.6682 |

Table 5: Model performance on Last.FM

| Model | ACC | AUC | F1 | Pre@1 | Recall@1 |
|-------|-----|-----|----|-------|----------|
| SVD [4] | — | 0.7690 | 0.6960 | — | — |
| CKE [9] | — | 0.7440 | 0.6730 | — | — |
| PER [13] | — | 0.6330 | 0.5960 | — | — |
| KGCN [15] | 0.7093 | 0.7848 | 0.7107 | 0.7089 | 0.7154 |
| BAGCN | 0.7196 | 0.7913 | 0.7188 | 0.7227 | 0.7175 |

These three datasets belong to explicit feedback. In order to mine user preference information, we convert explicit feedback into implicit feedback to learn model recommendation in this paper. In implicit feedback, the user with a positive interaction with an item can be counted as 1, and a sampled negative sample set for each user is counted as 0,
which means the sample set has not been observed. The rating range of the MovieLens-20M dataset is 1–5, and the rating range of Book-Crossing is 1–10. Meanwhile, the rating threshold of the MovieLens-20M dataset is set to 4, which means a rating greater than 4 is regarded as a positive interaction. The two datasets, Book-Crossing and Last.FM, are sparser than MovieLens-20M, so we do not set a threshold.

3.2. Comparative Models Introduction. SVD [4]: a classic CF-based model. It analyzes the user’s liking of each factor and the degree to which the item contains each factor based on the existing data.

CKE [9]: integrate various auxiliary information such as structured, textual, and visual knowledge with CF in a unified recommendation framework for joint training.

PER [13]: viewing the knowledge graph as a heterogeneous information network, it represents the connectivity between users and items by extracting latent features of meta-paths.

KGCN [15]: a representative of a hybrid recommendation training model. It implements recommendation by integrating the characteristics of knowledge

---

**Figure 3:** (a) The performance of pre on MovieLens-20M. (b) The performance of pre on Book-Crossing. (c) The performance of pre on Last.FM.
graphs and graph convolutional neural networks and aggregating neighbor information and bias to obtain item representations [21–24].

3.3. Experimental Setup. In BAGCN, we set the functions $z$ and $f$ that are the inner products, $\sigma$ is the activation function ReLU of the nonlast layer aggregator, and tanh is the activation function of the last layer aggregator. Table 2 provides other hyperparameter settings of this model. $K$ denotes the number of neighbors of each sampled entity, $d$ denotes the dimension of the embedding, $H$ denotes the number of iterations, $\eta$ denotes the learning rate, and $\lambda$ denotes the regularization term coefficient. The hyperparameters are sized by optimizing AUC on the validation set, and the ratio of training set, evaluation set, and test set is set to 6 : 2 : 2. We evaluate the BAGCN algorithm in two experimental scenarios: (1) in CTR prediction, the trained model is used to predict each interaction in the test set, and AUC, $F_1$, and ACC are used to evaluate CTR prediction. (2) In top-N recommendation, the trained model is designed to select $N$ items with the highest predicted click probability for each user in the test set for recommendation. By default, $N = 1$. All trainable parameters are optimized using the Adam algorithm.

PER uses artificially designed paths as features, SVD sets MovieLens-20M, and Book-Crossing $d = 8$, $\eta = 0.5$. For Last.FM, $d = 8$, $\eta = 0.1$. The dimension $d$ of the three datasets of the CKE model is set to 64, 128, and 64, respectively, the training weight for the KG part is 0.1, and the learning rate is set the same as SVD. The KGCN model is set to $K = 4$, $d = 32$, $\eta = 0.02$, $H = 2$, and $\lambda = 10^{-7}$ in the MovieLens-20M dataset, and the parameters of Book-Crossing are set to $K = 8$, $d = 64$, $\eta = 0.0002$, $H = 1$, and $\lambda = 2 \times 10^{-5}$, respectively. In Last.FM,
$K = 8, d = 16, \eta = 0.0005, H = 1, \lambda = 10^{-4}$, and $H$ represents the number of aggregation iterations. The number of training rounds is set to 60, and the number of training rounds for MovieLens-20M is set to 30 for the Last.FM and Book-Crossing datasets.

3.4. Experimental Results Analysis. The CTR prediction and top-N recommendation results in different datasets are displayed in Tables 3–5, Figures 3, and 4. Overall, the BAGCN model outperforms the baselines on all three datasets tested, including ACC, AUC, F1, precision, and recall. Comprehensively comparing the performance of various indicators in the three datasets, our method has improved the accuracy rate (ACC), while AUC has increased. The multilayer bidirectional recurrent neural network BiLSTM can effectively utilize the forward and backward feature information of entities and relationships. At the same time, adding an attention mechanism layer can perform biased aggregation of neighborhood information to make the representation of nodes more accurate. By observing the experimental results, the following conclusions can be drawn:

1. The performance of BAGCN is better than that of all the compared baseline models. Compared with the KGNN model in the three datasets of movies, books, and music, ACC is improved by 1.00%, 2.58%, and 1.03%, and the pre is improved by 0.63%, 2.28%, and 1.38%.

2. PER performs the worst due to its heavy reliance on manually designed meta-paths, and the optimal path is difficult to define in reality.

3. In the three data sets, Book-Crossing and Last.FM are more sparse than MovieLens-20M data, but the improvement of each index is more obvious, indicating that the BAGCN model proposed in this paper can alleviate data sparsity to a certain extent sexual problems.

3.5. Hyperparameter Optimization. Different hyperparameters will affect the effect of the model. In order to find the best value of the model on the validation dataset, it is necessary to optimize the super parameters. This section optimizes the number of sampling neighbors $K$ and the embedding dimension $d$.

3.5.1. The Influence of Different $K$ on the Model. The use efficiency of the knowledge graph is studied by changing the number of sampling neighbors $K$. It can be observed from Table 6 that, for the datasets MovieLens-20M, Last.FM, and Book-Crossing, the model has the best performance when $K = 4$, 16, and 8, respectively. Because the dataset of MovieLens-20M is denser than the other two, only fewer neighbors need to be sampled. If $K$ is too small, the model cannot include enough neighbor information to represent nodes; but if $K$ is too large, noise will be introduced, which will affect the recommendation effect of the model.

3.5.2. The Influence of Different $d$ on the Model. In addition, the influence of different embedding dimensions $d$, which is also the number of hidden nodes of BiLSTM, on the model is studied. It can be observed from Table 7 that increasing the size of $d$ at first can improve the recommendation effect of the model, but as the value of $d$ continues to increase, the performance of the model becomes worse instead. This is because a larger $d$ will cause the model to be overfitted, which will affect the effect of the model.

4. Conclusion

Aiming at the problems of cold start, data sparsity, and poor performance of existing recommendation algorithms, this paper proposed the BAGCN algorithm. The algorithm extracts the features of the embedding of entities and relationships in the knowledge map, learns the semantic information in the knowledge map through the self-attention mechanism, aggregates the neighborhood information in a biased way, and then extends to multihop simulation of high-order structure information to tap the potential interest of users. Through experiments on several datasets, the BAGCN model is proved to be superior to the baseline model in various performance evaluation indexes of film, book, and music recommendation. In this paper, negative samples are sampled in a uniform way, and high-quality negative samples are as important for model learning as positive samples. In addition, the user's demographic information and the recommendation model will be considered as auxiliary information to improve the performance of user integration.

Data Availability

The data used to support the findings of this study are available in the following URL: https://github.com/hhwang55/KGCN/tree/master/data.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding this work.
Acknowledgments

This study was supported by the National Natural Science Foundation, Research on Cryptographic Image Retrieval in Cloud Environment Based on Visual Semantics (61772561).

References

[1] H. Xiong and Z. Liu, “A situation information integrated personalized travel package recommendation approach based on TD-LDA model,” in Proceedings of the 2015 International Conference on Behavioral, Economic and Socio-cultural Computing (BESC), pp. 32–37, Nanjing, China, October 2015.

[2] L. N. H. Nam, “Towards comprehensive approaches for the rating prediction phase in memory-based collaborative filtering recommender systems,” Information Sciences, vol. 589, pp. 878–910, 2022.

[3] Y. Koren, R. Bell, and C. Volinsky, “Matrix factorization techniques for recommender systems,” Computer, vol. 42, no. 8, pp. 30–37, 2009.

[4] Y. Koren, “Factorization meets the neighborhood: a multi-faceted collaborative filtering model,” in Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 426–434, Las Vegas, NV, U.S.A, August 2008.

[5] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T. S. Chua, “Neural collaborative filtering,” in Proceedings of the International World Wide Web Conferences Steering Committee, pp. 173–182, Australia, April 2017.

[6] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, “DeepFM: a factorization-machine based neural network for CTR prediction,” in Proceedings of the 26th International Joint Conference on Artificial Intelligence, pp. 1725–1731, Australia, August 2017.

[7] K. Bollacker, R. Cook, and P. Tufts, Freebase: A Shared Database of Structured General Human Knowledge, pp. 1962–1963, AAAI, Menlo Park, CA, U.S.A, 2007.

[8] H. Wang, F. Zhang, M. Zhao, W. Li, X. Xie, and M. Guo, “Multi-task feature learning for knowledge graph enhanced recommendation,” in Proceedings of the world wide web conference, pp. 2000–2010, San Francisco, CA, U.S.A, May 2019.

[9] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W. Y. Ma, “Collaborative knowledge base embedding for recommender systems,” in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 333–362, San Francisco, California, U.S.A, August 2016.

[10] H. Wang, F. Zhang, X. Xie, and M. Guo, “DKN: deep knowledge-aware network for news recommendation,” in Proceedings of the 2018 World wide Web Conference, pp. 1835–1844, France, April 2018.

[11] X. Yu, X. Ren, Q. Gu, Y. Sun, and J. Han, Collaborative Filtering with Entity Similarity Regularization in Heterogeneous Information Networks JICAI HINA, vol. 27, 2013.

[12] C. Luo, W. Pang, and Z. Wang, “Hete-CF: social-based collaborative filtering recommendation using heterogeneous relations,” in Proceedings of the 2014 IEEE International Conference on Data Mining, pp. 917–922, Shenzhen, China, January 2015.

[13] Y. Xiao, R. Xiang, Y. Sun et al., “Personalized entity recommendation: a heterogeneous information network approach,” in Proceedings of the 7th ACM International Conference on Web Search and Data Mining, pp. 283–292, New York, NY, U.S.A, February 2014.

[14] H. Wang, F. Zhang, J. Wang et al., “RippleNet: propagating user preferences on the knowledge graph for recommender systems,” in Proceedings of the 27th ACM International Conference on Information and Knowledge Management, pp. 417–426, Italy, October 2018.

[15] H. Wang, M. Zhao, X. Xie, W. Li, and M. Guo, “Knowledge graph convolutional networks for recommender systems,” in Proceedings of the World Wide Web Conference, pp. 3307–3313, San Francisco, CA, U.S.A, May 2019.

[16] S. Lu, X. Jia, N. Zhang, and Y. Lu, “Application of clinical database in cancer research in the era of big data,” Infectious diseases, vol. 33, no. 04, pp. 301–311, 2020.

[17] S. Chen, L. Tai, and Y. Peng, “Dynamic load matching method for automatic parameter tuning,” Computer Applications, vol. 41, no. 3, pp. 657–661, 201.

[18] F. Yu, Q. Liu, S. Wu, L. Wang, and T. Tan, “A dynamic recurrent model for next basket recommendation,” in Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 729–732, Italy, July 2016.

[19] F. Xue, X. He, X. Wang, J. Xu, K. Liu, and R. Hong, “Deep item-based collaborative filtering for top-N recommendation,” ACM Transactions on Information Systems, vol. 37, no. 3, pp. 1–25, 2019.

[20] Y. Cheng and J. Pan, “A study on resource security under legal protection of e-commerce data sovereignty based on blockchain network,” Security and Communication Networks, vol. 2022, p. 7, Article ID 3713891, 2022.

[21] A. Radwan, M. F. Dominguez, and J. Rodriguez, “Mobile caching-enabled small-cells for delay-tolerant e-Health apps,” in Proceedings of the 2017 IEEE International Conference on Communications Workshops (ICC Workshops), pp. 103–108, IEEE, France, May 2017.

[22] F. B. Saghezchi, A. Radwan, J. Rodriguez, and T. Dagiuklas, “Coalition formation game toward green mobile terminals in heterogeneous wireless networks,” IEEE Wireless Communications, vol. 20, no. 5, pp. 85–91, 2013.

[23] Di Wu, Y. Lei, M. He, C. Zhang, and Li Ji, “Deep reinforcement learning-based path control and optimization for unmanned ships,” Wireless Communications and Mobile Computing, vol. 2022, pp. 1–8, Article ID 7135043, 2022.

[24] F. M. Abd Algalil and S. P. Zambare, “Effects of temperature on TD-LDA model,” in Proceedings of the 2015 International Conference on Communication, Control, and Computing Technologies, pp. 1–4, San Francisco, CA, U.S.A, May 2015.