Applied Intelligence (2022) 52:954–966
https://doi.org/10.1007/s10489-021-02363-w

Personalized recommendation system based on knowledge embedding and historical behavior

Bei Hui[1] · Lizong Zhang[1] · Xue Zhou[2] · Xiao Wen[3] · Yuhui Nian[2]

Accepted: 16 March 2021 / Published online: 14 May 2021

Abstract
Collaborative filtering (CF) usually suffers from limited performance in recommendation systems due to the sparsity of user–item interactions and cold start problems. To address these issues, auxiliary information from knowledge graphs, such as social networks and item properties, is typically used to boost performance. The current recommended algorithms based on knowledge graphs fail to utilize rich semantic associations. In this paper, we regard knowledge graphs as heterogeneous networks to add auxiliary information, propose a recommendation system with unified embeddings of behavior and knowledge features, and mine user preferences from their historical behavior and knowledge graphs to provide more accurate and diverse recommendations to the users. Our proposed ReBKC shows a significant improvement on three datasets compared to state-of-the-art methods. These results verify the effectiveness of learning short-term and long-term user preferences from their historical behavior and by integrating knowledge graphs to deeply identify user preferences.

Keywords Collaborative filtering · Recommendation system · Knowledge graph · Historical behavior

1 Introduction

The explosive growth of online content and services makes it difficult for users to make clear choices on various items such as news, movies, music, restaurants, and books. The recommendation systems aim to satisfy personalized user interests by providing specific products, thus addressing the problem of information overload. Traditional recommendation algorithms consider the historical behaviors of users and make recommendations based on their potential common preferences, and these algorithms have achieved impressive success [8]. However, interactions between users and items are typically very sparse. Using only historical behavior to make recommendations can lead to overfitting, and when a new user joins, the lack of relevant information can lead to cold start problems. To address these limitations, researchers have added auxiliary information to CF, such as social networks, user/item attributes, pictures, and contextual information. Among various types of auxiliary information, knowledge graphs contain rich semantic associations between entities, including more useful facts and item-related links. Knowledge graphs provide entities connected by different relations. As a semantic network connecting users and various items, knowledge graphs provide rich auxiliary information for recommendation systems and enhance the accuracy of recommendations and user satisfaction. Knowledge graphs are widely used in information retrieval, question-and-answer, text classification and recommendation systems.

Inspired by the successful application of KGs in various tasks, researchers have also tried to use KGs to improve the performance of recommendation systems. KGs can improve recommendation performance from three aspects: a) KGs introduce semantic correlations between items, which can identify potential relations between items and improve recommendation accuracy; b) KGs are composed of various types of relationships, which helps to reasonably explore user interests and increase the diversity of recommended items; c) KGs combine user historical behavior with

✉ Lizong Zhang
l.zhang@uestc.edu.cn

[1] Trusted Cloud Computing and Big Data Key Laboratory of Sichuan Province, University of Electronic Science and Technology of China, Chengdu, 611731, China

[2] School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu, 611731, China

[3] School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, 611731, China

recommendation records, making recommendation systems interpretable.

State-of-the-art methods of KG-aware recommendations are broadly classified as either embedding-based models or path-based models. Embedding-based models [17–19, 26] take the attributes of users and items as inputs to the recommendation algorithms; use knowledge graph embedding techniques to preprocess entities to embed emotional networks, social networks, and personal knowledge networks for recommendations; and then embed the learned entities into the recommendation framework. This method uses knowledge graphs as item attributes, using the one-hop neighbor information in the knowledge graph of the corresponding entity. For example, a deep knowledge-aware network (DKN) [18] treats entity embedding and word embedding as different channels and designs convolutional neural networks to combine them for news recommendations. Collaborative knowledge base embedding (CKE) [26] combines the CF module with the knowledge embedding, text embedding, and image embedding of items to form a unified Bayesian framework. Signature heterogeneous information network embedding (SHINE) [17] uses a deep autoencoder. Embedding-based methods show high flexibility in KG-aware recommendation systems [19]. However, the knowledge graph embedding algorithm used in these methods is more suitable for link prediction and other in-graph applications, not recommendation systems. An entity embedding algorithm that has learned by embedding-based methods is less intuitive and effective in representing relationships between items. This kind of method utilizes information of the entity's one-hop neighbors in the knowledge graph, but it fails to introduce multihop knowledge information, so the semantic network information of the knowledge graph cannot be effectively used.

Path-based methods [25, 27] explore various connection modes between items in KGs and provide additional guidance for recommendations. These methods treat knowledge graphs as heterogeneous information networks and then construct metapaths or metagraphs between items to mine the potential relations between items. However, this approach requires strong expert experience to manually design those paths, thus limiting the spread of knowledge. For example, personalized entity recommendations (PERs) [25] and metagraph-based recommendations [27] treat KGs as heterogeneous information networks (HINs) and extract latent features based on metapaths/metagraphs to represent connections between users and items along with different types of relation paths/graphs. Path-based methods use KGs in a more natural and intuitive manner, but they rely heavily on manually designed metapaths, which is difficult to optimize in practice. In addition, it is impossible to design handcrafted metapaths in certain scenarios, such as news where entities and relationships are not in the same domain. These methods limit the spread of knowledge to a certain extent and fail to consider the user's attention and preference for different knowledge entities.

Entities and relations in knowledge graphs form a semantic network. In the real world, users have different preferences for each entity in a network. The above two methods attempt to use knowledge information, but they fail to mine different user preferences for each knowledge entity. The current recommendation algorithms based on knowledge graphs fail to learn user preference for each knowledge entity. Deng et al. [4] proposed a subspace clustering model to aggregate neighborhood information. Graph attention networks [16] aggregate neighborhood features and enable assigning different weights to different nodes in a neighborhood. Inspired by a graph attention network, which pays different attention to different neighboring nodes, we use the graph attention network model to learn user preferences for knowledge entities.

Therefore, embedding-based models fail to effectively use semantic network information, path-based models limit the spread of knowledge because they require expert knowledge, and both methods fail to explore user preferences for knowledge. As shown in Fig. 1, the KGs for the movie scenario contain more comprehensive auxiliary information, such as the background knowledge of the item and the relations between them. In this paper, we propose a personalized recommendation system based on knowledge embedding and historical behavior. Our model uses a graph attention network to dynamically learn user preferences for knowledge, which when combined with their historical preferences, deeply mines the user preferences and improves the recommendation accuracy.

Our contributions are as follows:

- Through the research and analysis of related technologies, we find that current methods represent triplets separately, causing triplet information to be lost during the model learning process. The recommendation system adopts one-hop features and manually sets multiple features, which limits the spread of knowledge.
- A personalized recommendation system based on knowledge embedding and historical behavior is proposed. The self-attention mechanism is used to mine short-term or long-term user preferences from the historical behavior of each user's log, and the historical preferences are combined with knowledge graphs to further mine user preferences. This approach addresses current recommendation systems based on knowledge graphs that fail to efficiently use the semantic network information of the knowledge graph.
- The effectiveness of our recommendation system, which combines historical behaviors and knowledge

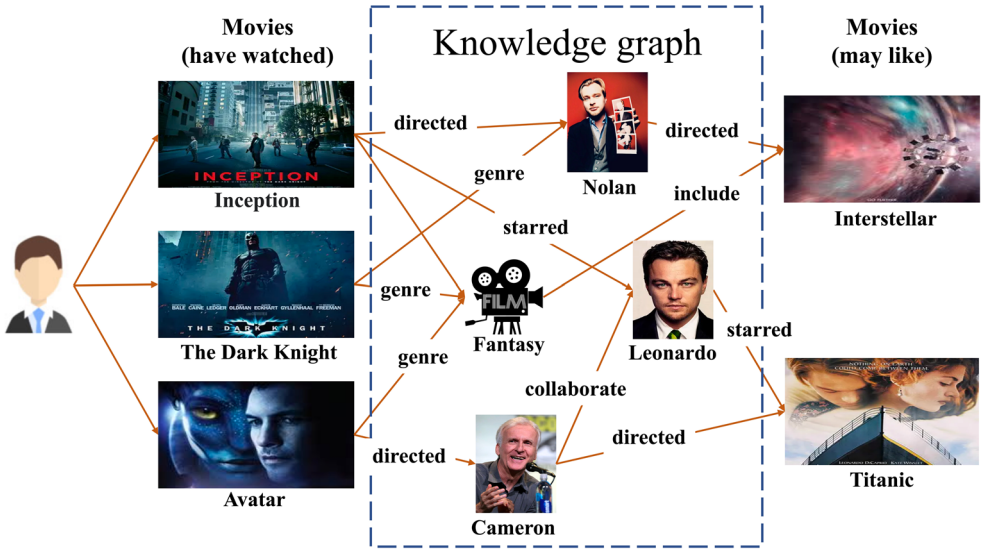

Fig. 1 Movie recommendation based on the knowledge graph

graphs, is verified through experiments. A click-through rate (CTR) prediction experiment proves that it is feasible to use historical behaviors and knowledge graphs to mine user preferences.

2 Related work

Recommendation systems based on knowledge graphs are a research topic of interest that has attracted considerable attention in recent years. Researchers have found that knowledge graphs have multisource heterogeneous association information between entities and can filter data on the Internet. After filtering, the association information can make the features of the user items more fine-grained. The connections between the user and the user, the user and the item, and the item and the item are calculated more accurately, which provides interpretability, diversity, and accuracy to the recommendation system. Recommendation models based on knowledge graphs can be classified into two types: path-based and embedding-based recommendation methods.

Path-based recommendation methods [27] treat knowledge graphs as heterogeneous information networks and then construct metapath or metagraph features between items. Briefly, a metapath is a specific path connecting two entities, such as "$actor- > movie- > director- > movie- > actor$". This metapath can connect two actors, so it can be regarded as a way to mine potential relationships between actors. Some methods use path similarity to calculate user preferences for different items, learning a preference matrix under the path and combining with a traditional collaborative filtering model to propose a personalized recommendation system. The knowledge graph is a relational semantic network of entities. Researchers use the connected information of paths in a knowledge graph to calculate the similarity between items. Researchers [25, 26] have explored the various links between items in KGs and provided additional guidance for recommendations. For example, personalized entity recommendation (PER) [25] and path-based recommendation [26] treat KGs as heterogeneous information networks (HINs) and extract metapath-based latent features to represent the connections between users and items along with different types of paths. Cheekula et al. [2] proposed a content-based recommendation system that uses an extended activation algorithm on DBpedia to identify personalized entities. The advantage of this method is that it fully and intuitively utilizes the network structure information of the knowledge graph. The disadvantage is that it requires strong expert knowledge to manually design the metapath or metagraph, which makes it difficult to achieve the best performance in practice. In addition, this type of method cannot be applied in scenarios where the entity does not belong to the same field because it is impossible to predefine metapaths or metagraphs for such a scenario.

Embedding-based models use a knowledge graph embedding algorithm to preprocess a knowledge graph and merge the learned entity embedding into the recommendation system. For example, a deep knowledge-aware network (DKN) [18] treats entity embedding and word embedding as different channels and then designs a convolutional neural network to combine them for news recommendations. Passant et al. [12] proposed a music recommendation system by calculating the semantic distance contained in a knowledge graph. Esposito et al. [5] proposed a hybrid query expansion approach based on lexical resources and word embeddings for question-and-answer systems. Yang et al. [24] proposed a graph-based cross-heterogeneous domain recommendation system. For each domain, a bipartite graph

is used to represent the relationship between its entities and features, and an effective propagation algorithm is designed to obtain similarities between entities from different fields. Researchers [3, 14] have introduced the concept of similarity based on metapaths, where metapaths are paths that consist of a series of relationships defined between different entities, thus defining a new similarity measure method under the metapath framework, called PathSim, to find peer objects in the network. Yu et al. [25] introduced metapaths based on latent features to represent connectivity between users and items on different types of paths and utilize the relationship heterogeneity in an information network to provide high-quality personalized recommendations. Collaborative knowledge base embedding (CKE) [26] combines the CF module with knowledge embedding, text embedding, and image embedding to form a unified Bayesian framework. The signature heterogeneous information network (SHINE) [15] designs a deep autoencoder to embed emotional networks, social networks, and personal knowledge networks for celebrity recommendations. Embedding-based methods show high flexibility in KG-aware recommendation systems, but the knowledge graph embedding algorithm used in these methods is usually more suitable for graphic applications such as link predictions rather than recommendations. It fails to represent the relationships between items.

Regardless of embedding-based recommendation algorithms or path-based recommendation algorithms, knowledge semantic features must be learned before using knowledge graphs. Common semantic feature extraction algorithms include word2vec, which entails knowledge representation learning in knowledge graphs.

Knowledge representation learning aims to project the semantic information of knowledge into a low-dimensional vector space to obtain dense low-dimensional real-valued vectors. Knowledge representation learning is mainly used to determine entities and relations in knowledge graphs. Modeling entities and relations in a low-dimensional dense vector space and learning semantic information through training is a core technology that runs through the entire knowledge graph construction process. The vector dimension of knowledge representation learning is low, which helps to improve the calculation efficiency while making full use of the semantic information between items.

TransE [17] considers the relation as the translation from the head entity to the tail entity, that is, the relation vector is used as the head entity to tail entity translation. The head entity vector and the relation vector are added to be approximately equal to the vector of the tail entity. Then, the translation is adjusted to be as equal as possible through training. In this way, not only can the structure information of the triplet be learned, but the training parameters are minimized, and the model is relatively simpler. To a certain extent, this makes up for the shortcomings of the traditional method, where training is complicated and not easily expanded. However, this can also become too simple, which makes it difficult for TransE to model complex relations such as 1-N, N-1 and N-N and other types of relations. To compensate for this defect, researchers have analyzed and proposed improved models. TransH [21] introduced hyperspace for the relations, TransR [26] projected entities and relationships into different spaces, and TransA [25] used different dynamic matrices for the head entity and tail entity rather than the same projection matrix. Researchers [22] also combined the entity description information to propose the DKRL model, introducing Freebase to provide the description information. The continuous bag of words model (CBOW) and the deep convolutional neural network model (CNN) have been used to represent the description of entity pairs. Xie et al. [23] proposed a model to combine word2vec and TransE to represent knowledge. It mainly uses Baidu Encyclopedia and Wikipedia to extract information to form triplets and then joins these triplets in the process of training word2vec, thus making the head entity vector closer to the tail entity vector and then fusing the text and knowledge graph. PTransE [9] considers the multistep relation paths that represent the semantic relations between entities. IntentGC [28] collects abundant relations from common user behaviors and item information to leverage both explicit preferences and heterogeneous relationships using graph convolutional networks. PHGR [6] proposed a new graph neural network that contains different graph convolutions specifically for the recommendation scenario, which can effectively excavate and combine heterogeneous information among user graphs, item graphs, and interaction graphs.

In this paper, we adopt the concept of sequential learning, which brings clear interpretability to the recommendation system. Combined with the graph attention mechanism, it deeply mines user preferences. In the KG-aware recommendation system, we mine user preferences by learning the entity information associated with the central entity. In practice, users have different degrees of preference for different entities, and users may only like one of them or like all but have different degrees of preference. The graph attention mechanism focuses on the importance of different entities, so we adapt the graph attention mechanism. A novel knowledge representation learning model is designed based on the translation model. The translation model and the representation learning model combined with multisource information learn the triplet structure information in different ways to obtain the knowledge representation. However, these methods treat triplets separately and ignore the structural information implicit in the neighborhood field, leading to the loss of structural information. To solve this problem, we propose

a novel knowledge representation learning method that combines all the relevant triplets and learns the knowledge representation from the triplets to improve the quality of knowledge representation.

In this paper, we propose a personalized recommendation system based on knowledge embedding and historical behavior, which considers user behaviors with different attention to knowledge entities, combined with user historical preferences, to offer accurate and diverse recommendations to users. To obtain high-quality knowledge representations and provide rich information for recommendations, this paper provides a novel knowledge representation learning algorithm.

3 Method

In this section, we introduce our personalized recommendation system based on knowledge embedding and historical behavior in detail, showing how to effectively apply the knowledge graph to recommendation systems. To address the issues with the current knowledge representation learning methods, we propose a novel knowledge representation learning model to represent knowledge semantic features, thus providing high-quality knowledge information to recommendations. We first introduce the overall architecture of the ReBKC and give an example to illustrate how our model generates recommendations in specific scenarios. Then, we introduce the learning process for user historical preferences, which is the first part of the ReBKC. Moreover, we use knowledge graph embedding to learn the entities and relations more effectively and propose our knowledge discovery based on the learned knowledge graph to mine the items of user interest. Finally, we calculate the predicted probability of candidate items based on the knowledge discovery.

3.1 Overall architecture

Our personalized recommendation system (ReBKC) based on knowledge embedding and historical behavior attempts to mine user preferences from knowledge graphs and use this as a basis for user recommendations. The model consists of four parts: historical preference learning, knowledge graph embedding, knowledge discovery, and predicted probability. The overall framework is shown in Fig. 2. Knowledge graph embedding learns representations of entities and relations, and historical preference learning

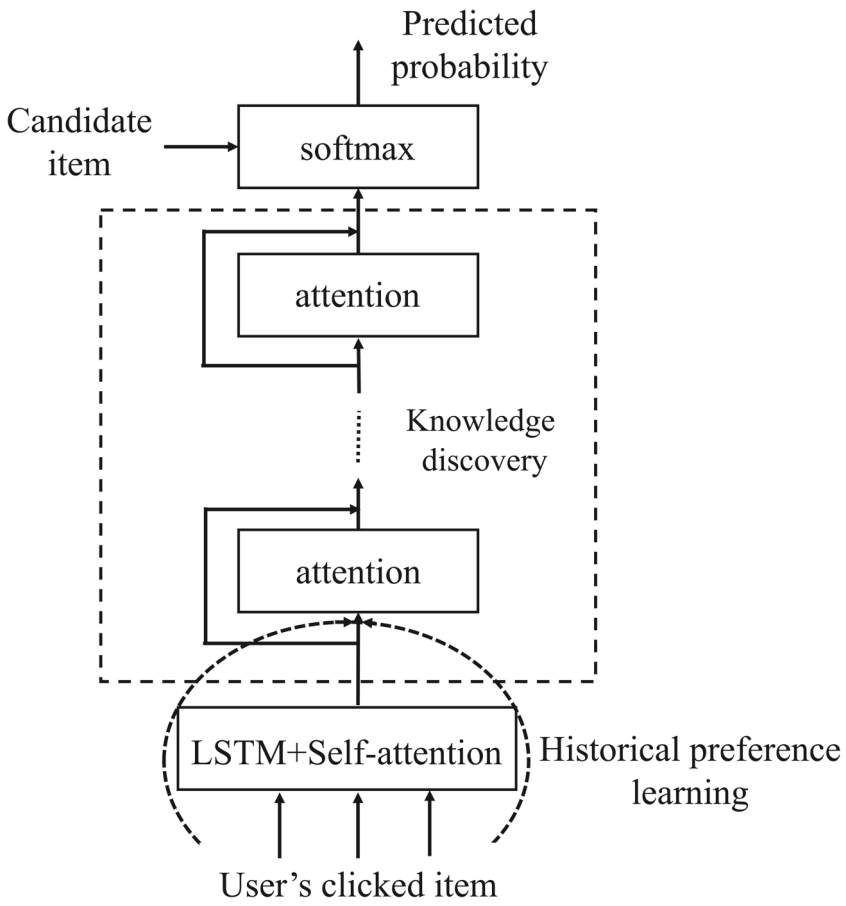

Fig. 2 The overall framework of our personalized recommendation system

mines user preferences from user browsing histories. The knowledge discovery uses the semantic network information of knowledge graphs to further mine the user preferences on the basis of historical preference. The predicted probability uses the information learned by knowledge graph embedding to generate candidate recommendation results. The ReBKC combines the user historical information with the knowledge graph to help the recommendation system using the attention mechanism. In this paper, our personalized recommendation system based on knowledge embedding and historical behavior mainly consists of historical preference learning and knowledge discovery. Before knowledge discovery is implemented, we use knowledge representation learning to represent knowledge semantic features to obtain knowledge embeddings.

For example, as shown in Fig. 1, in the movie recommendation scene, we first learn the user's historical preferences, including Inception, The Dark Knight, and Avatar. Then, we learn the entities and relations of the knowledge graph and find possible related entities such as Nolan, Fantasy, Leonardo, and Cameron based on the learned knowledge graph and the user's historical preferences. Finally, we predict the movies that users may be interested in based on the learning results, such as Titanic and Interstellar.

3.2 Historical preference learning

The historical preference learning module is the first stage of the entire system, and its purpose is to learn short-term or long-term user preferences. In the recommendation system, users have usually entered considerable click information from which we can obtain short-term or long-term user preferences. The inputs from this stage include the short-term or long-term user preferences, which are the user's click historical matrix H, and LSTM combined with a self-attention mechanism can be used to learn the user preferences and output the user's preference embedding m. At present, the common methods use recurrent neural networks such as RNNs and LSTM [13], but these methods are not suitable for the current field, where users pay different levels of attention to each historical click item. According to previous studies, we found that attention mechanisms can learn different weights for each click item, such as the user preferences for each item. We use LSTM combined with the self-attention mechanism to find user preferences through the user's click behavior.

The input in this section is the user's click historical matrix $H \in R^{(M\times d)}$, where M represents M click items, and d is the dimension of sentence embedding of historical behaviors learned through the LSTM+Attention module. Different from the general attention mechanism, the query, key, and value in the self-attention mechanism are all from the same source, and the value, query, and key are obtained by the nonlinear conversion of historical behavior matrix H.

$$Q = ReLU(HW_Q) \tag{1}$$

$$K = ReLU(HW_K) \tag{2}$$

where $W_Q \in R^{d\times d} = W_K \in R^{d\times d}$ are weight matrices of the query and key, respectively, and ReLU is an activation function. The weight matrix S is calculated as follows:

$$S = softmax\left(\frac{QK^T}{\sqrt{d}}\right) \tag{3}$$

The output is a similar matrix of M historical behaviors. Finally, the similarity matrix and historical matrix are multiplied to obtain the output of self-attention.

$$\alpha = SH \tag{4}$$

$\alpha \in R^{L\times d}$ can be regarded as the user preferences. To learn a single attention value, we perform a mean calculation on the self-attention results. Based on LSTM, we can learn the user preferences, which are sequential information.

$$m = \frac{1}{L}\sum_{l=1}^{L}\alpha_l \tag{5}$$

m is the user's preference embedding.

3.3 Knowledge graph embedding

Before mining user preferences for each knowledge entity, we first use a knowledge representation learning algorithm to represent entities and relations. The inputs of the knowledge graph embedding stage are the entity matrix H and the relation matrix G, and the output is the updated entity and relation embeddings. The knowledge representation quality determines whether the knowledge graphs can bring high-quality information into the recommendation system. We investigate and analyze the advantages and disadvantages of existing methods and propose a novel knowledge representation learning algorithm based on the translation model.

In knowledge graphs, entities have multiple semantics, and each triple actually contains the semantics of the entity in the triple. However, knowledge representation learning not only learns the semantics of the entity in one triple [1, 7, 10, 21] but also learns the semantics in all triples. Therefore, the triples should not be processed separately. In this paper, we handle triples at the same time to learn their entity representations.

The module takes two embedding matrices as inputs. The entity embedding matrix is represented by $H \in R^{N_e\times T}$, the i-th row represents the embedding of the entity e_i, N_e represents the number of target entities, and T is the

feature dimension of the entities. Similarly, the relational embedding matrix is represented by the matrix $G \in R^{N_r \times T}$. The embedding matrices of the output layer are $H' \in R^{N_e \times T'}$ and $G' \in R^{N_r \times T'}$, which are new representations of entities and relationships. The approximate calculation process of this module is shown in Fig. 3. In order toTo obtain a new embedding of entity e_i, our module first learns the representation of each triple associated with e_i. These embeddings are learned by performing a linear transformation over the concatenation of entities and relations into a special triplet $t^i_{jk} = (e_i, r_k, e_j)$, as shown in Equation 6.

$$c^i_{jk} = W_1[h_j||g_k||h_j] \tag{6}$$

c^i_{jk} is the embedding representation of triplet t^i_{jk}. h_j and g_k are embedding representations of entity e_j and the relationship r_k. W_1 is a projection matric. f^i_{jk} is learned to represent the importance of each triplet t^i_{jk}. The weighted matrix W_2 is used to perform the parameterized linear transformation, and then the LeakyReLU function is used to obtain the absolute attention value of the triplet.

$$f^i_{jk} = f\left(c^i_{jk}\right) = W_2 c^i_{jk} \tag{7}$$

To obtain the relative attention values, we use SoftMax to calculate f^i_{jk} as shown in Equation 8.

$$a^i_{jk} = softmax\left(f^i_{jk}\right) = \frac{\exp\left(f^i_{jk}\right)}{\sum_{j \in N^i_e, k \in N^i_g} \exp\left(f^i_{jk}\right)} \tag{8}$$

a^i_{jk} represents the absolute attention value of triplet c^i_{jk} associated with entity e_i, $j \in N^i_e$ is the head entity and tail entity set, and $k \in N^i_g$ is the relation set associated with e_i. The new representation of the entity e_i is the weighted sum represented by related triplets as shown in Eq. 9:

$$h'_i = \sum_{j=1}^{m} a^i_{jk} c^i_{jk} \tag{9}$$

where m represents the number of triples associated with entity e_i, and h'_i is the new representation of entity e_i.To stabilize the learning process and introduce more neighbor information, we adapt multihead attention; that is, multiple independent attention mechanisms calculate the output embedding and then connect them.

$$h'_i = \Big\Vert_{k=1}^{K} \sum_{j=1}^{m} a^i_{jk} c^i_{jk} \tag{10}$$

The weight matrix $W^R \in R^{T \times T'}$ is used to perform a linear transformation on the relation embedding matrix G, where T' is the dimension of the output relation.

$$G' = G \cdot W^R \tag{11}$$

In the last layer, we use an average method to obtain the final representation embedding:

$$h'_i = \sigma\left(\frac{1}{K} \sum_{k=1}^{K} \sum_{j=1}^{m} a^i_{jk} c^i_{jk}\right) \tag{12}$$

The module incorporates the semantic information of the entity in each triplet, and no information is lost because of the order.

3.4 Knowledge discovery

The knowledge semantic features are obtained through knowledge representation learning, and the knowledge discovery module will deeply mine user preferences. The inputs of the knowledge discoverdiscovery part stage is are the user's preference embedding m and the click historical matrix H, and the output is the preference embedding A'', which combined multi-hopcombines multihop information of from historical click itemitems. The knowledge graph is a semantic network, a directed graph whose nodes and edges represent entities and relations between entities. In reality, users have different preferences for entities in knowledge graphs, and existing methods have failed to effectively use

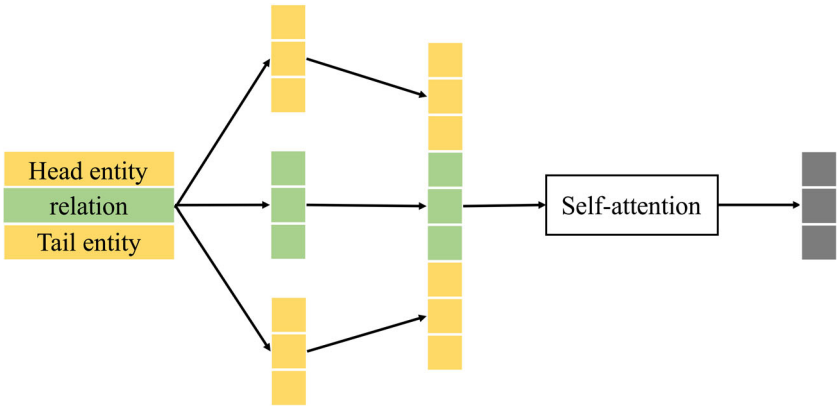

Fig. 3 The representation learning model

utilize this ideaconcept. Therefore, this section combines knowledge graphs with the information learned by the historical preference learning module to further explore user'suser preferences. We adapt the graph attention network to learn structure information. The knowledge in the knowledge graph has been trained into embeddings through representation learning, and this section takes the input and output of the historical preference learning module as input, that is, the historical behavior matrix $H \in R^{M\times d}$ and user'suser preference embedding $m \in R^d$. In order to To represent the user's hierarchical preferences based on the knowledge graph, this section recursively defines k-hop entity sets.

$$E_u^k \in \left\{ t \middle| (h,r,t) \in G \; and \; h \in E_u^{k-1} \right\}, \; k = 1, 2, \cdots, T \tag{13}$$

E_u^0 represents the entity in the click history, as the seed set which is also the first-hop entity.

$$S_u^k = \{(h,r,t) | (h,r,t) \in G \; and \; h \in E_u^{k-1}\}, \; k = 1, 2, \cdots, T \tag{14}$$

S_u^k are triplets associated towith the entities in E_u^k. There are many entities in the user's historical behavior, and these entities form a huge large semantic network with the entities associated with them. In this section, we limit the semantic network to 4-hop hops.

The entities in the semantic network are directly or indirectly related to the head entity, but having a relation does not mean that users will be interested in these entities. This section uses graph attention networks to learn the semantic network. For entities in each hop, we use the attention mechanism to learn entities that the user prefers. As shown in Eq. 15:

$$p_i = softmax\left(m^T r_i h_i\right) = \frac{\exp\left(m^T r_i h_i\right)}{\sum_{(h,r,t)\in S_u^k} \exp\left(m^T r h\right)} \tag{15}$$

$r_i \in R^{d\times d}$ and $h_i \in R^{d\times d}$ respectively represent relations and tail entities, respectively, and p_i is the weight indicating the user's interest in the entity h_i under the relation r_i. we We believe that relations are important. People usually recognize entities through relations, which is in line with the law of people's understanding of things. Adding relations is to better learns the user's interests in entities. For a 1-hop neighbor, the calculation process is shown in Eq. 16:

$$p_i = softmax\left(m^T r_i h_i\right) = \frac{\exp\left(m^T h_i\right)}{\sum_{h\in E_u^0} \exp\left(m^T h\right)} \tag{16}$$

After obtaining the relevant probabilities, this section multiplies the entity with it and obtains the embedding hop_k by linear addition.

$$hop_k = \sum_{(h_i,r_i,t_i)\in S_u^k} p_i h_i \tag{17}$$

The module performs this process iteratively on triplet set S_u^k, where $k = 1, 2, \cdots, T$, and T=4. Therefore, the user's preferences will spread to k-hops away from his the click history, and it is can be observed that a user u havehas several different several preference sequences: $hop_1, hop_2, \cdots, hop_T$. By merging all the embeddings to represent use'suse preference embeddings, we adopt a structure whichthat is similar to restful, while retaining the historical embeddings of the historical preference learning module. We believe that historical information is important in the recommendation system:

$$A^1 = W^1 A^1 + m \tag{18}$$

$$A'' = W^{k-1} A^{k-1} + A^k \tag{19}$$

where W^{k-1} is the weight matrix of the k-1 hop, and A^k is the embedding obtained after linear transformation on hop_k. After fusion, we use the embedding A'' as the output of this module.

3.5 Predicted probability

As the last stage, the input of predicted probability is the candidate item embedding $V = v_1, v_2, \cdots, v_n$ and the user preference vector A'', and the output is the click probability of the candidate item. Our ReBKC recommendation algorithm is shown in Fig. 4.

$$P_i = \sigma\left(A''^T v_i\right) \tag{20}$$

where $\sigma(x) = \frac{1}{1+\exp(-x)}$ is the sigmoid function.

4 Experiments

In this section, we test and evaluate the performance of the model in three practical scenarios: movies, books, and news recommendations. The experimental results are given in Section 4.3 and we analyze our model.

4.1 Datasets

We utilize the following three datasets: MovieLens-1M, Book-Crossing, and Bing-News. MovieLens-1M and Book-Crossing have a small number of entities, but the entity names are complex. In contrast, Bing-News has more

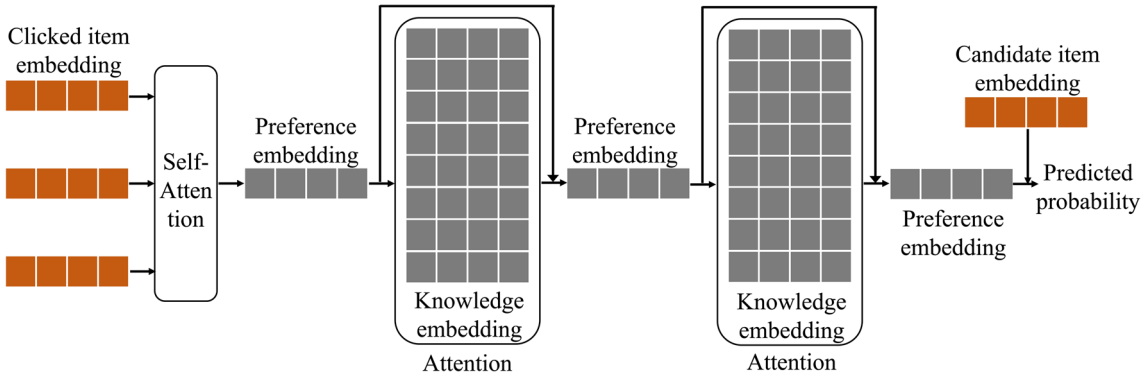

Fig. 4 Our ReBKC recommendation algorithm

information than MovieLens-1M and Book-Crossing. It contains more entities and useful information, which also introduces more noise, but the entity names in Bing News are relatively simple. Using these three types of datasets for testing can not only reflect the learning ability of the knowledge graph learning module proposed in our paper but also prove that our KG-aware recommendation system can be applied in many real scenarios.

MovieLens-1M is a widely used benchmark dataset in movie recommendation. It contains approximately 1 million rating data points on the MovieLens website.

Book-Crossing contains 1,149,780 rating data points from the Book-Crossin community.

Bing-News contains 1025192 click news items collected from the server logs of Bing News from October 16 to August 8, 2016. Each piece of news consists of a headline and a piece of content.

Because MovieLens-1M and Book-Crossing are not labeled, we label each item as 1, indicating that the user has clicked the item. In addition, the item is labeled as 0 for not clicked. This section uses Microsoft Satori to construct a knowledge graph for each dataset.

4.2 Experiment setup

In this section, click-through rate (CTR) prediction is used to verify our system, and the trained model is applied to each interaction in the test set to predict click probabilities. Compared with state-of-the-art models, our KG-aware recommendation system is effective. Accuracy and AUC were used to evaluate the effectiveness of CTR prediction.

In the ReBKC, we set hop H=2 for MovieLens-1M and Book-Crossing and H= 3 for Bing-News. Experimental results show that larger H values do not improve performance, but they increase the amount of calculation. In MovieLens-1M, the embedding dimension d of items and the knowledge graph is 16, and the learning rate η is 0.01. The dimension d of Book-Crossing is 4, and the learning rate η is 0.001; the dimension d of Bing-News is 32, and the learning rate η is 0.005. The final optimal parameters are determined by AUC on validation sets. For each dataset, the ratio of the training set, validation set, and test set is 6:2:2. Each experiment was repeated 10 times, and the average value was used for performance analysis.

To verify our novel KG-aware recommendation system, we chose the following comparison models:

1. CKE [26] combines CF, structure, text, and images in a unified recommendation framework. We implement CKE, as CF adds a structural knowledge module.
2. SHINE [17] designs a deep autoencoder to embed semantic networks, social networks, and personal knowledge networks for recommendations. An autoencoder is used for item interaction and item description information to predict click probability.
3. DKN [18] regards entity embedding and word embedding as multiple channels and combines them together in CNN for CTR prediction. In our paper, we use the movie name, book name, and news title as the text input for DKN.
4. PER [25] regards a KG as an HIN and extracts the characteristics based on the metapath to represent the connection between users and items.
5. LibFM [20] is a feature-based factorization model widely used in CTR scenarios. The user ID, item ID, and average entity embeddings obtained from TransR are connected as the input of LibFM.
6. Wide&Deep [15] is a recommendation model that combines a wide linear channel and a deep neural network model. Similar to LibFM, users, items, and entity embeddings are used to provide breadth and depth.

4.3 Results

The experimental results of click-through rate prediction are shown in Table 1 and Fig. 5, 6 and 7. This experiment demonstrates that our recommendation system

Table 1 Experimental results of AUC and accuracy on CTR prediction

Method	MovieLens-1M		Book-Crossing		Bing-News	
	AUC	ACC	AUC	ACC	AUC	ACC
CKE	0.796	0.739	0.674	0.635	0.560	0.157
SHINE	0.778	0.778	0.668	0.631	0.554	0.537
DKE	0.655	0.589	0.621	0.598	0.661	0.604
PER	0.712	0.667	0.623	0.588	–	–
LibFM	0.892	0.812	0.685	0.639	0.644	0.588
Wide&Deep	0.903	0.822	0.711	0.623	0.654	0.595
PHGR	0.929	–	0.735	–	–	–
ReBKC	**0.931**	**0.846**	**0.735**	**0.663**	**0.694**	**0.622**

Bold data signify the best score

offers significant improvement by comparing it with other recommendation models based on knowledge graphs. The model is effective not only for simple fields such as books and movies but also for news, which has a large number of entities, and entity names are simple. By testing in different fields, the performance of our model can be demonstrated.

The performance of CKE is relatively poor. CKE combines CF with structural knowledge, textual knowledge, and visual knowledge in a unified framework, but only structural knowledge is used here. In multiple comparison models, SHINE performs better in movie and book recommendations than in news. This is because there are more entities in the news, more negative noise signals are introduced, and it is too complicated to take one-hop triplets as input. In contrast to other models, the DKN model performs best in news recommendations and slightly worse in movie and book recommendations. This is because the DKN model was designed to focus on the complexity of the news and extract the text and knowledge graph entities through a convolutional neural network. The names of movies and books are usually very short and ambiguous, making it difficult to extract useful information. However, news generally exists in the form of texts that can provide useful information for the model.

PER does not perform well in movie and book recommendations because user-defined metapaths are usually not the best approach. In addition, because the types of entities and relations in news are too complicated to define the metapath in advance, it cannot be applied to news recommendations. As two general recommendation tools, LibFM and Wide&Deep have achieved satisfactory performance because these two models have deeply extracted the features of users, items, and knowledge entities in different ways.

ReBKC performs well on MovieLens-1M and Book-Crossing. The performance on the Bing-News dataset is not as good as that on the first two datasets, but it

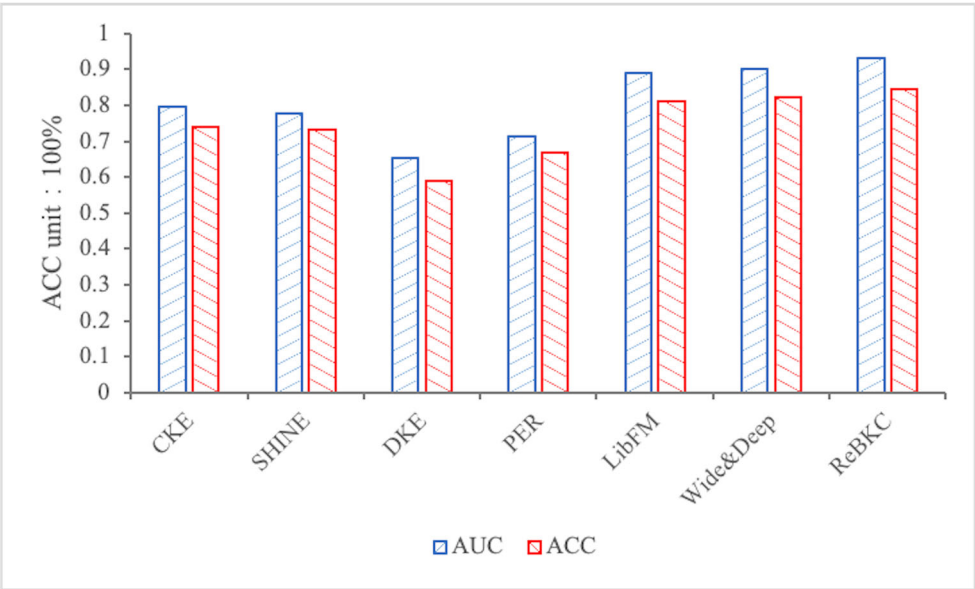

Fig. 5 AUC and ACC(100%) results on MovieLens-1M

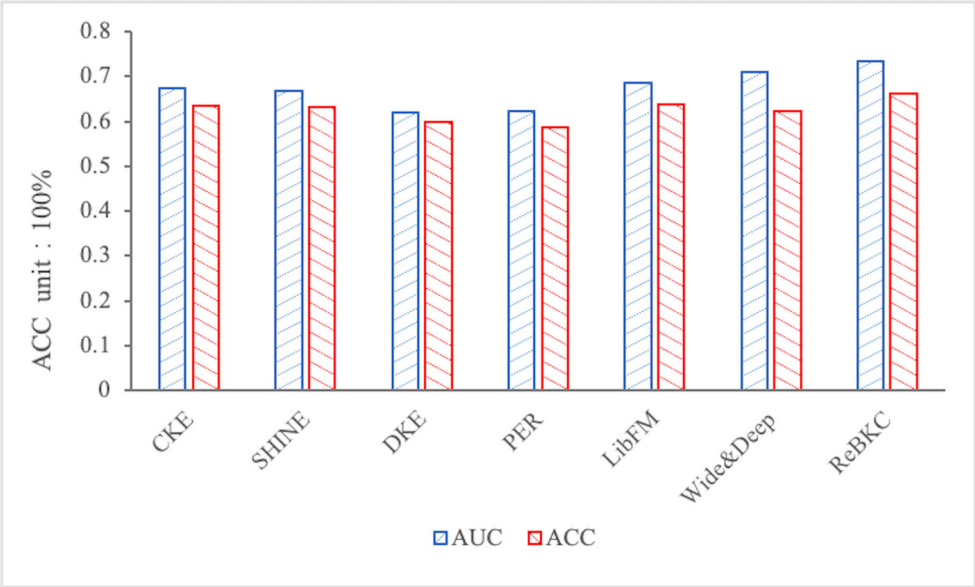

Fig. 6 AUC and ACC (100%) results on Book-Crossing

is also better than other comparable models and even better than the DKN model specifically designed for news. This is because we learned the short-term or long-term user preferences from the historical behaviors and integrated them into the knowledge graph to further mine user preferences for different knowledge, reducing the useless information and better learning user preferences. The reason why the performance on the Bing-News dataset is lower than that on the other two datasets may be that the news is more complicated. There are many entities in the news, and the continuous appearance of new entities brings noise to the model. On the whole, ReBKC can mine short-term or long-term user preferences from historical behaviors and combine a knowledge graph to further mine user preferences by using the attention mechanism, filter information that users are not interested in, and fully discover user preferences, thus improving the recommendation ability.

To verify our concept that the attention mechanism can learn user preferences for knowledge, which can effectively improve recommendation performance, we evaluate our model by an ablation study. In this section, the ablation model without an attention mechanism does not filter user preferences for knowledge but simply integrates the knowledge. The experimental results are shown in Table 2. There is a clear gap between the performance of the ablation model without the attention mechanism and the ReBKC. The ablation model performs similarly to CKE and SHINE but worse than LibFM and Wide&Deep. This shows that after removing the attention mechanism, the

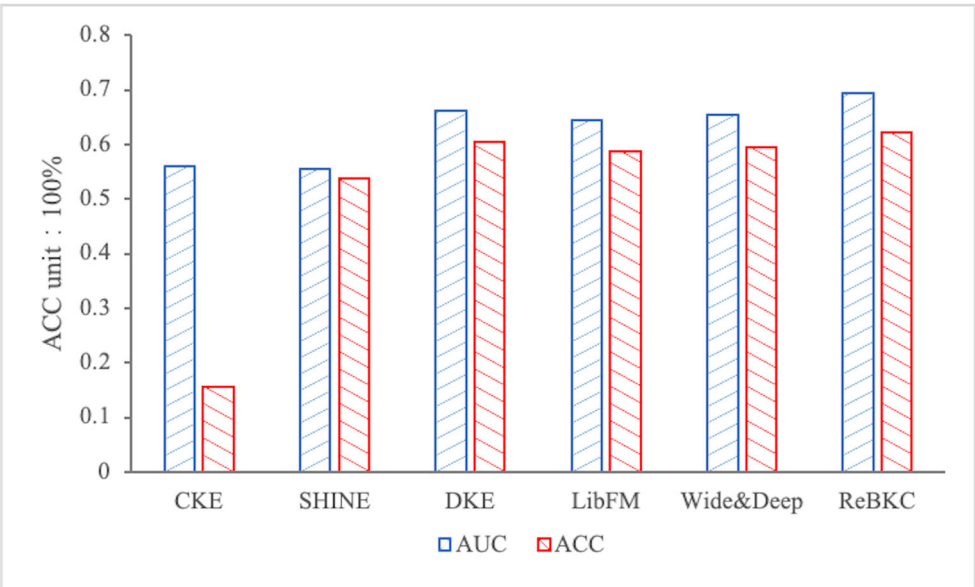

Fig. 7 AUC and ACC(100%) results on Bing-News

Table 2 Ablation study on ReBKC

Method	MovieLens-1M		Book-Crossing		Bing-News	
	AUC	ACC	AUC	ACC	AUC	ACC
ReBKC-Attention[a]	0.743	0.697	0.665	0.621	0.566	0.534
ReBKC+Attention[b]	**0.931**	**0.846**	**0.735**	**0.663**	**0.694**	**0.622**

[a]ReBKC-Attention represents learning user's preferences without the graph attention network

[b]ReBKC+Attention is our proposed model in this paper, which represents learning user's preferences for knowledge through graph attention network

Bold data signify the best score

model does not filter the knowledge features in the KG, which introduces too much information that users are not interested in, affecting the recommendation ability. Especially on the Bing-News dataset, there are many entities in the news, leading to considerable noise when using the knowledge graph semantic network. After using the attention mechanism, the knowledge is filtered according to the user's historical preferences, reducing the knowledge that users are not interested in, thus improving the recommendation performance.

At the same time, the experimental results of the maximum hop H are shown in Table 3. The experiment changes the maximum hop H to evaluate the model performance proposed in our paper. From the results, when H is 1, that is, only the direct relations of the knowledge graph are used, the performance is poor because the advantages of the knowledge graph are not utilized. When H is larger than or equal to 4, the advantage of the knowledge graph is exerted, but the associations between entities may be weak, leading to poor performance. The best performance is when H is 2 and 3. The trade-off between long-distance-dependent positive signals and noisy-negative signals makes H too small, it is difficult to explore the correlation and dependence between long-distance entities, and a too-large H brings much more noise than useful information.

Table 3 ACC results on different hop numbers

Datasets	MovieLens-1M	Book-Crossing	Bing-News
Hop number 1[a]	0.828	0.661	0.619
Hop number 2	0.837	0.659	0.619
Hop number 3	0.846	0.663	0.622
Hop number 4	0.824	0.542	0.608

[a]Hop number N represents a local network of N-hop entities centered on a certain entity

5 Conclusion and future work

Our KG-aware recommendation model uses the self-attention mechanism to mine short-term or long-term user preferences from each user's historical behavior and further combines the historical preferences with the knowledge graph to further mine user preferences, thus addressing the failure of the current KG-aware recommendation model to efficiently use the semantic network information of the knowledge graph.

In our model, we first learn the embeddings of entities and relations. However, the embeddings are learned to restore the triple in the knowledge graph, not to specifically learn for recommendation tasks. In the future, we intend to study how to deeply combine the knowledge graph and the recommendation system. At the same time, we will use an enhanced model [11] that has achieved great success in the game field in the recommendation system to enhance the recommendation effect.

Acknowledgements This work was supported by the National Key R&D Program of China (No. 2018YFC0807500), by National Natural Science Foundation of China (No. U19A2059), and by Ministry of Science and Technology of Sichuan Province Program (No. 2018GZDZX0048,20ZDYF0343).

Declarations

Conflict of Interests All authors declare that: (i) no support, financial or otherwise, has been received from any organization that may have an interest in the submitted work ; and (ii) there are no other relationships or activities that could appear to have influenced the submitted work.

References

1. Bordes A, Usunier N, Garcia-Duran A, Weston J, Yakhnenko O (2013) Translating embeddings for modeling multi-relational data. In: Advances in neural information processing systems, pp 2787–2795

2. Cheekula SK, Kapanipathi P, Doran D, Jain P, Sheth A (2015) Entity recommendations using hierarchical knowledge bases. In: KNOW@LOD
3. De Campos LM, Fernández-Luna JM, Huete JF, Rueda-Morales MA (2010) Combining content-based and collaborative recommendations: a hybrid approach based on bayesian networks. International Journal of Approximate Reasoning 51(7):785--799
4. Deng T, Ye D, Ma R, Fujita H, Xiong L (2020) Low-rank local tangent space embedding for subspace clustering. Information Sciences 508:1–21. https://doi.org/10.1016/j.ins.2019.08.060. http://www.sciencedirect.com/science/article/pii/S0020025519308096
5. Esposito M, Damiano E, Minutolo A, De Pietro G, Fujita H (2020) Hybrid query expansion using lexical resources and word embeddings for sentence retrieval in question answering. Information Sciences 514:88–105. https://doi.org/10.1016/j.ins.2019.12.002. http://www.sciencedirect.com/science/article/pii/S0020025519311107
6. Fu Y, Wan J, Zhao H, Jiang W, Pu S (2020) Preference-aware heterogeneous graph neural networks for recommendation. In: 2020 IEEE 32nd international conference on tools with artificial intelligence (ICTAI), pp 41–46. https://doi.org/10.1109/ICTAI50040.2020.00017
7. Ji G, He S, Xu L, Liu K, Zhao J (2015) Knowledge graph embedding via dynamic mapping matrix. In: Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: Long papers), pp 687–696
8. Koren Y, Bell R, Volinsky C (2009) Matrix factorization techniques for recommender systems. Computer 42(8):30–37
9. Lin Y, Liu Z, Luan H, Sun M, Rao S, Liu S (2015) Modeling relation paths for representation learning of knowledge bases. arXiv:150600379
10. Lin Y, Liu Z, Sun M, Liu Y, Zhu X (2015) Learning entity and relation embeddings for knowledge graph completion. In: Twenty-ninth AAAI conference on artificial intelligence
11. Mnih V, Kavukcuoglu K, Silver D, Graves A, Antonoglou I, Wierstra D, Riedmiller M (2013) Playing atari with deep reinforcement learning. arXiv:13125602
12. Passant A (2010) dbrec—music recommendations using dbpedia. In: International semantic web conference. Springer, pp 209–224
13. Pota M, Marulli F, Esposito M, De Pietro G, Fujita H (2019) Multilingual pos tagging by a composite deep architecture based on character-level features and on-the-fly enriched word embeddings. Knowledge-Based Systems 164:309–323. https://doi.org/10.1016/j.knosys.2018.11.003. http://www.sciencedirect.com/science/article/pii/S0950705118305392
14. Sun Y, Han J, Yan X, Yu PS, Wu T (2011) Pathsim: meta path-based top-k similarity search in heterogeneous information networks. Proceedings of the VLDB Endowment 4(11):992–1003
15. Tan YK, Xu X, Liu Y (2016) Improved recurrent neural networks for session-based recommendations. In: Proceedings of the 1st workshop on deep learning for recommender systems, pp 17–22
16. Veličković P, Cucurull G, Casanova A, Romero A, Liò P, Bengio Y (2018) Graph attention networks. In: 6th International conference on learning representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings, OpenReview.net, https://openreview.net/forum?id=rJXMpikCZ
17. Wang H, Zhang F, Hou M, Xie X, Guo M, Liu Q (2018) Shine: signed heterogeneous information network embedding for sentiment link prediction. In: Proceedings of the eleventh ACM international conference on web search and data mining, pp 592–600
18. Wang H, Zhang F, Xie X, Guo M (2018) Dkn: deep knowledge-aware network for news recommendation. In: Proceedings of the 2018 world wide web conference, pp 1835–1844
19. Wang Q, Mao Z, Wang B, Guo L (2017) Knowledge graph embedding: a survey of approaches and applications. IEEE Trans Knowl Data Eng 29(12):2724–2743
20. Wang X, Wang Y (2014) Improving content-based and hybrid music recommendation using deep learning. In: Proceedings of the 22nd ACM international conference on multimedia, pp 627–636
21. Wang Z, Zhang J, Feng J, Chen Z (2014) Knowledge graph embedding by translating on hyperplanes. In: Proceedings of the twenty-eighth AAAI conference on artificial intelligence, AAAI Press, AAAI'14, pp 1112–1119
22. Xie R, Liu Z, Jia J, Luan H, Sun M (2016) Representation learning of knowledge graphs with entity descriptions. In: Thirtieth AAAI conference on artificial intelligence
23. Xie R, Liu Z, Sun M (2016) Representation learning of knowledge graphs with hierarchical types. In: IJCAI, pp 2965–2971
24. Yang D, He J, Qin H, Xiao Y, Wang W (2015) A graph-based recommendation across heterogeneous domains. In: Proceedings of the 24th ACM international on conference on information and knowledge management, pp 463–472
25. Yu X, Ren X, Sun Y, Gu Q, Sturt B, Khandelwal U, Norick B, Han J (2014) 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
26. Zhang F, Yuan NJ, Lian D, Xie X, Ma WY (2016) 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
27. Zhao H, Yao Q, Li J, Song Y, Lee DL (2017) Meta-graph based recommendation fusion over heterogeneous information networks. In: Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining, pp 635–644
28. Zhao J, Zhou Z, Guan Z, Zhao W, Ning W, Qiu G, He X, 2019 Intentgc: a scalable graph convolution framework fusing heterogeneous information for recommendation. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery and data mining, pp 2347–2357