Computers, Materials & Continua
DOI: 10.32604/cmc.2022.030150
Article

Tech Science Press

KGSR-GG: A Noval Scheme for Dynamic Recommendation

Jun-Ping Yao[1], Kai-Yuan Cheng[1,*], Meng-Meng Ge[2], Xiao-Jun Li[1] and Yi-Jing Wang[1]

[1]Xi'an Research Institute of High-Tech, Xi'an, Shaanxi, 710025, China
[2]Nanyang Technological University, Singapore, 639798, Singapore
*Corresponding Author: Kai-Yuan Cheng. Email: ckycky_com@163.com
Received: 19 March 2022; Accepted: 08 June 2022

Abstract: Recommendation algorithms regard user-item interaction as a sequence to capture the user's short-term preferences, but conventional algorithms cannot capture information of constantly-changing user interest in complex contexts. In these years, combining the knowledge graph with sequential recommendation has gained momentum. The advantages of knowledge graph-based recommendation systems are that more semantic associations can improve the accuracy of recommendations, rich association facts can increase the diversity of recommendations, and complex relational paths can hence the interpretability of recommendations. But the information in the knowledge graph, such as entities and relations, often fails to be fully utilized and high-order connectivity is unattainable in graph modelling in knowledge graph-based sequential recommender systems. To address the above problems, a knowledge graph-based sequential recommendation algorithm that combines the gated recurrent unit and the graph neural network (KGSR-GG) is proposed in the present work. Specifically, entity disambiguation in the knowledge graph is performed on the preprocessing layer; on the embedding layer, the TransR embedding technique is employed to process the user information, item information and the entities and relations in the knowledge graph; on the aggregation layer, the information is aggregated by graph convolutional neural networks and residual connections; and at last, on the sequence layer, a bi-directional gated recurrent unit (Bi-GRU) is utilized to model the user's sequential preferences. The research results showed that this new algorithm performed better than existing sequential recommendation algorithms on the MovieLens-1M and Book-Crossing datasets, as measured by five evaluation indicators.

Keywords: Sequential recommendation; knowledge graph; graph neural network; gated recurrent unit

1 Introduction

Most of the famous sequential recommendation algorithms nowadays are based on the Markov chains and recurrent neural networks (RNNs). The Markov chain-based methods assume that the

user's next behavior relies on the preceding behavior or behaviors, and these methods have achieved good performance in short-term recommendation. Cai et al. [1] proposed a socially-aware personalized Markov chains model, which showed excellent performance in addressing the cold-start problem on sparse datasets, but could not capture frequently-changing dynamic information in complex contexts. Meanwhile, under the independency assumption of Markov chains, the preceding interactive and independent combinations of behaviors would reduce the performance of recommendation. The RNN-based recommendation methods code the user-item interactions into hidden vectors, and predict the user's next behavior according to the status of these vectors, which facilitate the storage and updating of information status. For instance, Xu et al. [2] put forward a recurrent convolutional neural network (RCNN), which used the RNN to capture long-term dependence relations and the recurrent module of the CNN to extract hidden short-term sequential relations. Nonetheless, the RNN often underperforms in explicit extraction of complex shifts of user tastes or more fine-grained user preferences from interactive sequences; it often fails to model the user time information and contextual information, and entails training by massive high-density data to reach the expected recommendation effect.

In these years, the knowledge graph (KG) has been introduced as assistant information to sequential recommender models [3]. The major KG modelling methods nowadays include the path-based method and the graph embedding-based method. The former decomposes the KG into linear paths, but it needs to define many meta-paths, and hence is not applicable to tasks that involve lots of KGs. The graph embedding-based method, however, falls short of inductive learning capacity; when a new node occurs, the model will have to learn the feature representations of the whole graph, which are not correlated to the downstream tasks, and consequently, the downstream task result cannot be used to optimize feature representations of the graph. Moreover, both methods cannot establish high-order connectivity of the graphs. Thus, many researchers have tried to optimize the combination of knowledge graphs with sequential recommender models. Huang et al. [4] put forward a knowledge-enhanced sequential recommendation model, which for the first time combined sequential recommender systems with external memory networks based on a large-scale knowledge base. Wang et al. [5] modeled session sequences into session graphs, and combined the graph neural network (GNN) and key-value memory networks to fulfill recommendation.Sun et al. [6] combined the user-item bipartite graph and the knowledge graph into a unified graph, and employed a graph attention network for propagation among nodes.

In response to the problems of existing sequential recommendation methods, such as the inability to capture fine-grained dynamic user preferences under complex conditions and the inability to establish higher-order connections of graphs, this paper proposes a knowledge graph sequential recommendation algorithm that fuses gated recursive units and graph neural networks. Under the premise of capturing the dynamic information of the recommendation system in real time, the accuracy of the recommendation results is improved by fusing the user-item history interaction information and the features of the knowledge graph. The main contributions are as follows.

(1) A graph convolutional neural network (GCNN) framework for aggregating user information and item information end-to-end learning features is constructed. And the residual connectivity mechanism is introduced at the input and output of the graph convolutional network, and the Bi-GRU network is able to fuse item sequences and knowledge graph entity sequences for short-term interest modeling, which solves the problem of higher-order connectivity of the graph that cannot be established in the application of the knowledge graph to serialized recommendation algorithms.

(2) To address the problem that higher-order connectivity cannot be established in the knowledge graph applied to the sequential recommendation algorithm, this paper proposes a sequential recommendation algorithm that incorporates higher-order semantic relationships in the knowledge graph of item attributes, and effectively improves the accuracy of the recommendation results by introducing rich speech information.
(3) The proposed algorithm is compared with several baseline methods on MovieLens-1M and Book-Crossing datasets, and the comparison experiments show that the proposed algorithm has improved the recall, mean reciprocal rank (MRR), normalized discounted cumulative gain (NDCG), hit rate (HR) and precision of recommendation.

2 Related Works

2.1 Sequential Recommendation

Conventional recommender systems include demographics-based recommendation, content-based recommendation, association rules-based recommendation, collaborative filtering-based recommendation, knowledge-based recommendation and hybrid recommendation [7]. The major shortcomings of these systems are as follows. First, algorithms in these systems rely mainly on static historical interactive data and cannot capture dynamically-changing information, which reduces the accuracy of recommendation; secondly, conventional recommendation algorithms regard user behaviors as independent events and overlook the impacts of behavior sequences on recommendation; thirdly, conventional systems consider only long-term user taste, but not the short-term user preferences at different time intervals. Sequential recommendation algorithms that consider both short-term and long-term user preferences have overcome these shortcomings and hence gained momentum in these years.

Conventional recommender systems like the content-based systems and collaborative filtering systems rely on static user-goods interactive data to generate recommendations [8]. In real-world scenarios, the user interest, and the goods popularity changes from time to time, and to capture the changing information is important to improve the performance of recommender systems. Sequential recommender systems, which regard user-goods interactive information as dynamic sequences, model the sequence dependency and hence can achieve higher recommendation accuracy than conventional systems. Specifically, in sequential recommendation algorithms, user behaviors are represented as a sequential decision-making process, i.e., the sequence of the user's previous and current behaviors will affect their future behavior sequences [9,10]. Meanwhile, the sequential recommendation algorithms can model the information about the changes in the parameters and types of the goods as well as their popularity to provide more accurate, customized and dynamic recommendations.

2.2 Knowledge Graph-Based Recommender System

The knowledge graph, as assistant information to recommender systems, boasts many advantages. First, the KG modelled by the goods parameters and social networks introduces more semantic associations, which will allow the system to mine user interests and improve the accuracy of recommendations. Zhu et al. [11] put forward a knowledge graph-enhanced neural collaborative

recommendation framework, which made use the rich relations in the graph to improve the accuracy of recommendation. The rich relations between KG entities facilitate the expansion of the inter-item relations and increase the diversity of recommendations. For instance, Sang et al. [12] designed a learning path recommendation model based on a multidimensional knowledge graph framework, which provided diverse and customized learning path recommendations for e-learners. Last, the inter-entity relation paths in KGs can connect users with the recommendations, which hence increases interpretability of the recommendations. Xie et al. [13] put forth an explainable recommendation framework based on the KG and multi-objective optimization, in which the path between the target user and the recommended item is used as an explanation basis.

3 Knowledge Graph-Based Sequential Recommendation Combining Gated Recurrent Unit and Graph Neural Network (KGSR-GG)

3.1 Problem Description and Definitions

In a knowledge graph, both the knowledge framework and the entity data are described by structured 3-tuples to store inter-entity relations in the real world. The KG is expressed as $G = (\varepsilon, \Re, S)$, where $\varepsilon = \{e_1, e_2, \ldots, e_{|\varepsilon|}\}$ stands for the entity set consisting of $|\varepsilon|$ entities; $\Re = \{r_1, r_2, \ldots, r_{|R|}\}$ stands for the relation set, comprised of $|R|$ relations; and $S \subseteq \varepsilon \times \Re \times \varepsilon$ is the 3-tuple set in the KG. The 3-tuple describes a fact in a given domain, consisting of a head entity, a tail entity and the relation between the two entities.

In the KGSR-GG algorithm, the task is defined as follows: a knowledge graph (G) for the historical interactive sequence $S(u)$ of a user u and the item is given to predict the user u's behavioral sequence for the item in the next time point, i.e., the user u's potential interest in the interactive item v. A prediction function is constructed here:

$$\hat{y}_{uv} = f(u, v|S(u), G, \Theta) \tag{1}$$

where $\hat{y}_{uv}$ refers to the probability of the user u to visit the item v at the next time point, Θ is the correlation parameter of f.

3.2 Implementation of the KGSR-SS Algorithm

Fig. 1 shows the architecture of the algorithm, which consists of four layers—a pre-processing layer, an embedding layer, an aggregation layer, and a sequence layer. In sequential recommendation, the KG serves as assistant information to the recommender system to model user interest and generate recommendations. The pre-processing layer pre-treat the partial source data to improve the quality of the KG; the embedding layer embeds the user information, item information, and the KG entities and relations into a unified low-dimension vector space to generate user and item embeddings; the aggregation layer fuses the information between the central node and the neighboring nodes in the user vectors and item vectors to capture more latent vectors of the user. On the sequence layer, the item sequences and the entity sequences are processed to model the user's sequential preferences, and the loss function is calculated.

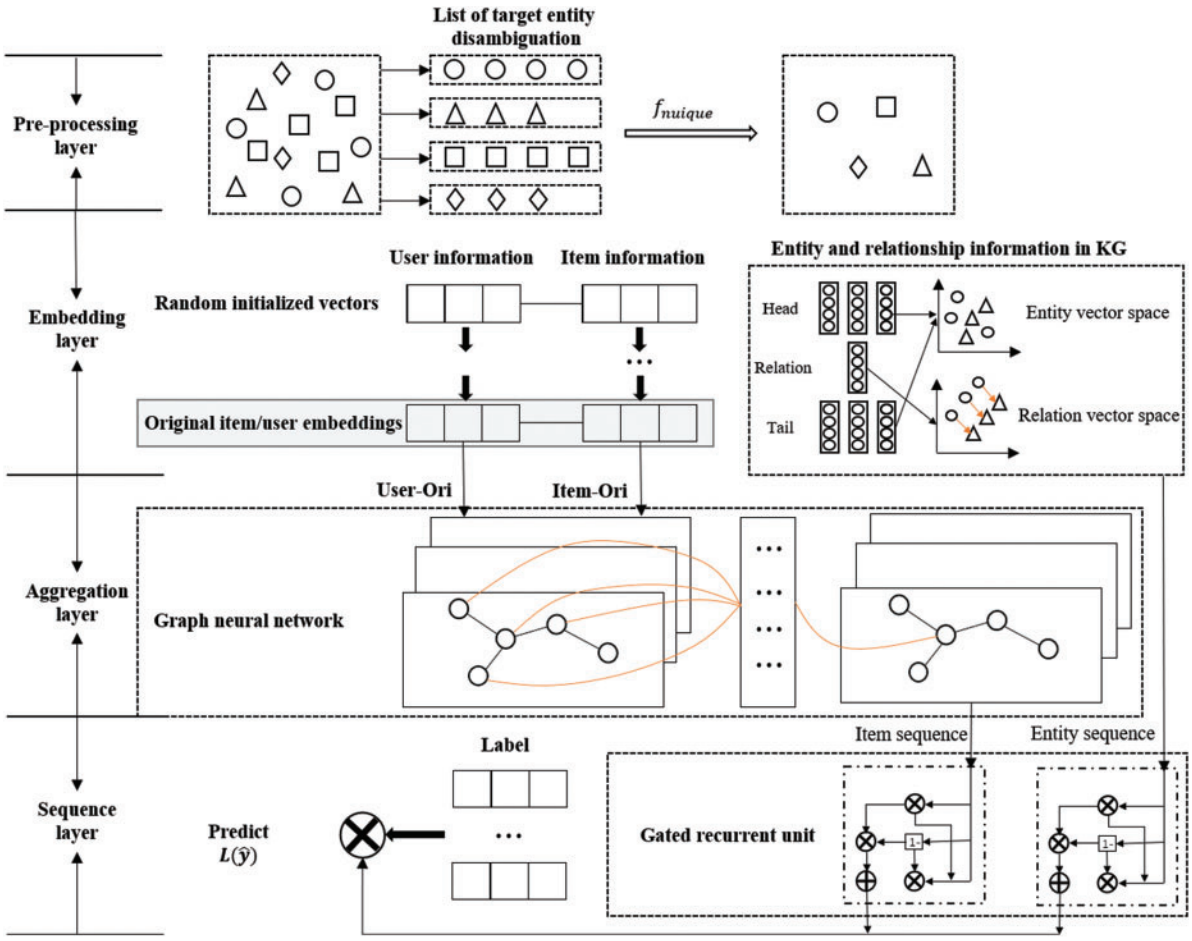

Figure 1: Architecture of the KGSR-GG algorithm

3.2.1 The Pre-Processing Layer

The main goal of the pre-processing layer is to cleanse the partial source data, perform entity disambiguation, and ensure the quality of the KG in the algorithm. Existing KG-based recommender systems rely on the user's social network and attributes of items to generate simple KGs. Hence, the conventional entity disambiguation method is adopted in the present work. In the present work, duplicate entity names N were clustered into the same target list E, and a function for deleting repetitive entities was designed as the entity disambiguation function δ:

$$\delta = f_{unique}() \tag{2}$$

3.2.2 The Embedding Layer

The major goal of the embedding layer is to embed information about users and items as well as entities and relations in the KG from a high-dimensional sparse feature vector space into a low-dimensional dense feature vector space, while maintaining the inherent structure and semantic information, and reducing the overhead of data storage and computation in the recommendation process. The TransR embedding method was employed here to process the entities and relations in the

KG, and the general embedding method was used to process user information and item information. TransD, though more simplistic and faster than TransR, has lower knowledge representation accuracy. Thus, TransR was selected in the present study to generate embeddings for information in the KG.

$$\varepsilon_r = M_r \varepsilon \tag{3}$$

$$s_r = M_r s \tag{4}$$

The score function used in the TransR method is

$$f_r(\varepsilon, s) = ||\varepsilon_r + r - s_r||_{L_1/L_2} \tag{5}$$

Matrix factorization is an embedding method that factorizes the user embedding matrix and the item embedding matrix; the obtained simplified matrices maintain the basic attributes of the original matrices, incorporate the "implicit vectors", and have stronger capacity in addressing sparse matrices. Given a user matrix $U \in R^{m \times k}$, an item matrix $V \in R^{k \times n}$, and a cooccurrence matrix $Y \in R^{m \times n}$, the predicted score of the user u to the item v is:

$$r_{uv} = q_v^T p_u \tag{6}$$

where p_u is the row vector of the user u in the user matrix U, q_v is the column vector of the item v in the item matrix V. The object function of matrix factorization is to reduce the difference between the original score r_{uv} and $q_v^T p_u$, and store the original information in the co-occurrence matrix to the maximum extent. The object function of the regularizes is introduced:

$$U \in R^{m \times k}, V \in R^{k \times n} \min_{q^*, p^*} \sum_{(u,v) \in K} \left(r_{uv} - q_v^T p_u\right)^2 + \psi \left(||q_v||^2 + ||p_u||^2\right) \tag{7}$$

where k is the dimension of the implicit vector, which determines the expression capacity of matrix factorization, ψ is the regularization parameter, and a larger parameter indicates stronger constraints of regularization, which can alleviate overfitting.

3.2.3 The Graph Neural Network-Based Aggregation Layer

The aggregation layer processes unstructured graph data generated by the embedding layer, such as the user vectors and item vectors. It works on the end-to-end learning structure in the graph, improves the algorithm's learning capacity and hence optimizes the recommendation accuracy. As Fig. 2 shows, the core function of the aggregation layer is to define the convolutional calculation on the graph data based on the features of the neighboring nodes. Specifically, the convolution of the central node and the neighboring nodes in the CNN was used to represent the aggregation between neighboring nodes and update the status of the nodes. For two given graph signals x_1 and x_2, the graph convolution calculation is defined as

$$x_1 * x_2 = H_{\tilde{x}_1} x_2 \tag{8}$$

where the graph displacement operator $H_{\tilde{x}1} = Vdiag(\tilde{x}_T)V^T$, $V \in R^{N \times N}$ is an orthogonal matrix, and $\tilde{x}_1$ is the Fourier coefficient of x_1.

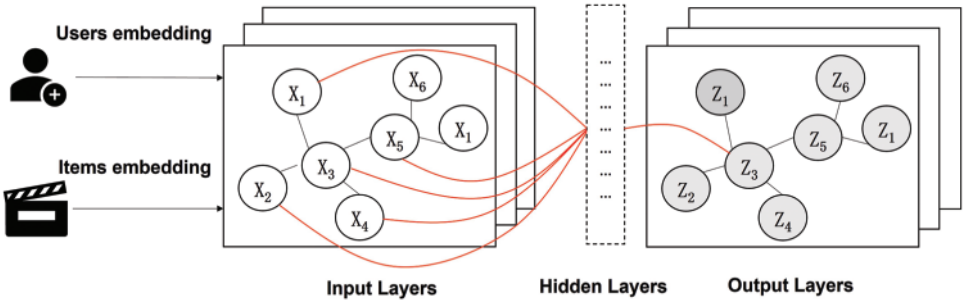

Figure 2: Architecture of the graph convolutional neural network on the aggregation layer

The graph convolutional neural network (GCNN) on the aggregation layer is a message transmission network, and the message transmission can be divided into two steps: message delivery and status updating, which are expressed by the functions of message function M and the updating function U, respectively:

$$c_{vw} = \left(d\left(v\right) d\left(w\right)^{1/2} A_{vw}\right) \tag{9}$$

$$M_t\left(\varepsilon_v^t, \varepsilon_w^t\right) = c_{vw}\varepsilon_w^t \tag{10}$$

$$U_v^t\left(\varepsilon_v^t, m_v^{t+1}\right) = \mathrm{Re}LU\left(W^t m_v^{t+1}\right) \tag{11}$$

Specifically, the message transmission rule in the multi-layer graph convolutional network is:

$$f\left(H^{(l)}, A\right) = \sigma\left(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right) \tag{12}$$

where l is the l-thlayer in the graph convolutional network; $\hat{A} = A + I$, where A is the feature adjacency matrix, and I is the unit matrix; $\hat{D} = \sum j\hat{A}_{ij}$ is the diagonal matrix of the matrix $\hat{A}$; $H^{(l)}$ is the feature of the feature of the convolutional neural network on the l-th layer; $W^{(l)}$ is the weight matrix of the convolutional neural network on the l-th layer; σ is the nonlinear activation function.

A single entry in the dataset exists in the form of a user and its corresponding preferred item. For example, "User A prefers cell phones", where more than one item is preferred by a user, is stored in the dataset in multiple single entry format.

In the embedding layer, we encode the items in the dataset according to the user identity to generate the corresponding item embedding vector. The vector representation of the "user-item" sequence is input to the graph neural network pair by pair, and all item features of interest to the user are aggregated to the user node. At this point, the aggregation layer outputs the sequence of items of interest to the user and sends it to the next layer for processing.

Meanwhile, the residual connection mechanism was introduced to the input layer and the output layer of the GCNN to fuse the features of the items, improve the model's feature learning capacity, improve the accuracy of the algorithm, and alleviate the problem of vanishing gradient and exploding gradient during network training. We assume that the GCNN is a single-layer network, and the principle of residual connection is:

$$Z_{\mathrm{concat}}^{l+1} = \sigma\left(\hat{\mathrm{A}}\mathrm{H}^l\mathrm{W}^l\right) + \mathrm{H}^l \tag{13}$$

where $Z_{\mathrm{concat}}^{l+1}$ is the final fused feature vector by the function *concat* ().

3.2.4 The Sequence Layer

Recurrent neural networks are often employed in recommender systems to learn the user's short-term interest from their recent interactive sequences. Knowledge graphs establish connections between items in terms of interactions between nodes. Other items related to the items of interest to the user in the dataset are pointed out by the knowledge graph. Enhancing the recommendation system model with knowledge of new item interactions from the knowledge graph helps to find potential points of interest for the user. The set of items that may be of interest to this user is obtained by learning the sequence of items of interest to the user and the sequence of relationships between items in the knowledge graph. The basic unit of the RNN on the sequence layer uses a Bi-GRU to process the item sequence vectors generated on the aggregation layer and the entity sequence vectors generated by the TransR. The computing equation of the reset gate and the update gate is:

$$r_t = \sigma\left(W_r x_t + U_r h_{t-1}\right) \tag{14}$$

$$Z_t = \sigma\left(W_z x_t + U_z h_{t-1}\right) \tag{15}$$

where x_t is the input vector at the time step t, and h_{t-1} is the stored information at the previous time step $t-1$. $W_r \in \mathbb{R}^{n\times m}$, $W_z \in \mathbb{R}^{n\times m}$, $U_r \in \mathbb{R}^{n\times m}$, and $U_z \in \mathbb{R}^{n\times m}$ are all weight parameters of the network. The calculation equations for the candidate hidden layer and the output hidden layer are:

$$\tilde{h}_t = \tanh\left(W_h x_t + U\left(r_t \odot h_{t-1}\right)\right) \tag{16}$$

$$h_t = \left(1 - z_t\right) \odot h_{t-1} + z_t \odot \tilde{h}_t \tag{17}$$

where $\odot$ is the Hadamard product; r_t and z_t represent the outputs of the reset gate and the update gate at the time point t; h_t and $\tilde{h}_t$ represent the status and the candidate status at the time point t.

Given the complexity of the user's short-term interest, in our algorithm, the GRU output layer is connected to a bi-directional fully-connected layer to serve as a classifier of the sequences. The fully-connected layer comprised of multiple neurons maps the learnt user short-term sequence features into the sample label space to transform and classify the sequence features. After these aforementioned procedures, the algorithm output sequences-based recommendations. At last, the cross entropy is employed to calculate the loss of the recommendations:

$$L\left(\hat{y}\right) = -\frac{1}{n}\sum_{i-1}^{n}\left(y_i \ln \hat{y}_i + \left(1 - y_i\right)\ln\left(1 - \hat{y}_i\right)\right) \tag{18}$$

where n is the number of samples, y_i is the true value, and $\hat{y}_i$ is the output value of the algorithm.

3.3 Workflow of the KGSR-GG Algorithm

Tab. 1 shows the implementation steps of the proposed KGSR-GG algorithm. As per the mathematical theories for the algorithm detailed in the previous section, the KGSR-GG algorithm is further described here.

Table 1: Procedures of the KGSR-GG algorithm

Input	"User-item" interactive information; knowledge graph G
Output	Prediction function $\hat{y}_{uv} = f(u, v \mid S(u), G, \Theta)$
1	Initialization: $v_{data}, u, NERD(vindataset, G)$
2	fulfill KG embeddings: $v_G \leftarrow TransR(G)$ Eqs.(4)–(6)
3	For i in u:
4	complete information aggregation: $v^u_{data} \leftarrow GCN(u, \mathrm{v}_{data})$ Eq. (13)
5	graph Convolution Output: $Z^{l+1}_{concat} \leftarrow \mathrm{Res}\left(v^u_{data}\right)$ Eq. (14)
6	processing output sequences: $\tilde{y}_{uv} \leftarrow \mathrm{BiGRU}\left(Z^{l+1}_{concat}, \mathrm{v}_G\right)$ Eqs. (15)–(18)
7	regularized prediction sequence: $\hat{y}_{uv} \leftarrow \mathrm{LSR}\left(\tilde{y}_{uv}\right)$
8	return $\hat{y}_{uv}$
9	Comparing $\hat{y}_{uv}$ and y_{uv}

The algorithm updates the model by giving the input "user-item" historical interaction information $S(u)$ and the linkage information G of the items in the knowledge graph, and expects the output prediction $\hat{y}_{uv}$ and compares it with the real label. First, the user vector u and item vector v are randomly initialized and named entities are linked from the knowledge graph, and the graph items corresponding to the items v_{data} in the dataset are found after disambiguation. These items are transformed into the graph item embedding vector v_G (line 2) by TransR. For each user, the following steps are performed separately: this user vector and the vector of items he is interested in are jointly input to the GCN for learning to obtain the user representation vector v^u_{data} (line 4) that aggregates information about all items preferred by the user. The aggregation of item information is enhanced by connecting the graph input layer with the output layer using the residual mechanism to obtain Z^{l+1}_{concat} (line 5). At this point, the output sequence Z^{l+1}_{concat} of the graph neural network and the item sequence v_G of the knowledge graph are input to the Bi-GRU for knowledge enhancement to obtain the initial prediction vector $\tilde{y}_{uv}$ (line 6), and the final prediction vector $\hat{y}_{uv}$ (line 7) is obtained by regularization. Finally, the model parameters are updated by comparing with the real labels (line 9).

4 Experiment and Analysis

To verify our proposed KGSR-GG algorithm, the dataset, data processing steps and evaluation indicators are described first; then, the proposed algorithm is compared with other advanced baseline methods; at last, the performance of our proposed algorithm is evaluated by performance indicators.

4.1 Experiment Datasets

The movie rating dataset MovieLens-1M and the book rating dataset Book-Crossing were employed to train and evaluate the algorithms in the experiment. The datasets involve data of different sizes, data sparsity degrees and application realms, and are publicly accessible. The specifics of the datasets are as follows. The details of the two data sets are shown in Tab. 2.

The KGs in the two datasets were constructed based on the attributes of the items, which were compared with the movie names and book names in the datasets to identify relations and hence match

the items with entities. Then, with the obtained entity set that was related to the datasets, the item IDs were matched with the head entity and tail entity in the three-tuples of the KGs.

Table 2: Specifics of the two datasets used in our experiments

Dataset	MovieLens-1M	Book-crossing
Number of users	6040	278858
Number of items	3952	271379
Number of interaction logs	1000209	69873
Number of KG entities	79347	77903
Number of KG relations	49	25
Number of KG three-tuples	385924	151500
Number of items connected to KG	3655	14967

4.2 Baseline Methods

To verify the overall performance of the KGSR-GG algorithm, it was compared with several baseline algorithms. The baseline methods used in the present work are as follows.

FPMC [14]: The factorizing personalized Markov chains (FPMC) system introduces the Markov chains to customized transition matrix and the matrix factorization model, and merges the sequence and customization for recommendation;

HRM [15]: in the hierarchical representation model (HRM), a layered structure that merges the sequential behaviors of the user's general taste is put forward to model the complex interactions between nonlinear factors.

GRU4Rec [16]: this model generates session-based recommendations with RNNs.

TransRec [17]: this model is a Markov chain model, but the major distinction is that the model introduces a novel "transition space" into the space vectors and is explainable.

Caser [18]: this model ranks the timestamps of the user-interactive items, embeds the items into an image, models the network using a CNN, and optimizes the network by the minimum cross entropy.

SASRec [19]: in this model, the self-attention mechanism is used to model the user sequential information, the obtained valuable information and all item embeddings are used to generate the inner product, and items are sequenced by the size of correlation to provide sequential forecasts.

4.3 Experiment Setting

4.3.1 Experiment Environment

To verify the recommendation performance of the proposed KGSR-GG algorithm, the Pytortch machine-learning architecture was implemented on the Ubuntu operating system, PyCharm2020, AMD 3700x CPU, GTX 1080Ti GPU, 64GB memory; and the experiment results were analyzed in Python 3.7.

4.3.2 Evaluation Indicators

After the dataset was filtered, the two datasets were divided into a training set and a test set evenly, and repetitive optimization experiments were performed to reach the optimal result. Five indicators were employed for evaluation:

(1) *recall@K*: the recall is the ratio of the items in the test set to the items in the recommended list, which is a measure of the precision ratio of the recommender system. The recall is defined as:

$$recall@K = \frac{\sum_{u \in U} |R(u) \cap T(u)|}{\sum_{u \in U} |T(u)|} \tag{19}$$

where $R(u)$ is the recommendation list generated for the user u; $T(u)$ is the list of items the user interacts with in the test set. K is the length of the recommendation list, i.e., the K items with the highest predicted recommendation score.

(2) *MRR@K*: the mean reciprocal rank (MRR) is the mean of the reciprocal rankings of the expected items, which measures the system's performance based on the ranking of the correctly recommended items in the recommendation list. The mathematical expression of *MRR* is:

$$MRR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{rank_i} \tag{20}$$

where $|Q|$ is the number of users, $rank_i$ is the ranking of the first item in the ground-truth result in the recommendation list for the *i-th* user.

(3) *NDCG@K*: Normalized discounted cumulative gain (NDCG) measures the accuracy of the ranking by the ranking of the item in the list. NDCG is defined as:

$$NDCG@N = \frac{1}{IDCG_i} \sum_{i=1}^{N} \frac{2^{rel_i} - 1}{\log_2 (i+1)} \tag{21}$$

(4) *HR@K*: the hit rate (HR) is a popular measure of the recall rate in recommendation systems and indicates "accuracy" of the forecasts. Its mathematical expression is:

$$HR@K = \frac{\sum_{i=1}^{N} hits@K}{N} \tag{22}$$

where N is the number of users, *hits* is the number of items in the test set that occur in the list of recommendations.

(5)*Precision@K*: precision, or precision ratio, is a popular measure of precision of sequential recommendation systems. The mathematical expression of this indicator is as follows.

$$Precision@K = \frac{\sum_{u \in U} |R(u) \cap T(u)|}{\sum_{u \in U} |R(u)|} \tag{23}$$

where $R(u)$ means the list of user behaviors on the training set, and $T(u)$ is the list of user behaviors on the test set.

4.3.3 Experiment Parameter Setting

For the training hyper parameters, the checkpoint_dir parameter was used to store the model, the epochs for each training were set at 150, and the batch size of samples captured in each training was set at 1024, and the learning rate was set at 0.0005. For the evaluation hyper parameters, the batch size of samples captured by each training was set at 2048. In algorithm implementation, the embedding

size of the embedding layer was set at 64, and the hidden size of the hidden layer was set at 128, the dropout rate of the model was set at 0.3, and the learning rate at 0.001; the cross entropy loss function was used as the loss function.

4.4 Experiment Result and Analysis

On the selected two datasets, five evaluation indicators *recall*@10, *MRR*@10,*NDCG*@10, *HR*@10 and *Precision*@10 were employed to evaluate the performance of our KGSR-GG algorithm and other baseline algorithms. The models achieved different results on the same datasets under the same experiment conditions. Tabs. 3 and 4 present the results.

Table 3: Experiment results of different recommender systems on the MovieLens-1M dataset

Model	MovieLens-1M				
	recall@10	*MRR*@10	*NDCG*@10	*HR*@10	*Precision*@10
FPMC	0.1621	0.0569	0.0814	0.1621	0.0162
HRM	0.1656	0.0484	0.0755	0.1656	0.0166
GRU4Rec	0.2295	0.0957	0.1270	0.2295	0.0229
TransRec	0.1185	0.0309	0.0511	0.1185	0.0119
Caser	0.2096	0.0838	0.1131	0.2096	0.0210
SASRec	0.2224	0.0842	0.1162	0.2224	0.0222
KGSR-GG	0.2502	0.1087	0.1394	0.2505	0.0251

Table 4: Experiment results of different recommender systems on the Book-Crossing dataset

Model	Book-crossing				
	recall@10	*MRR*@10	*NDCG*@10	*HR*@10	*Precision*@10
FPMC	0.0667	0.0310	0.0393	0.0667	0.0067
HRM	0.0669	0.0401	0.0464	0.0669	0.0067
GRU4Rec	0.0706	0.0422	0.0489	0.0706	0.0071
TransRec	0.0727	0.0428	0.0498	0.0727	0.0073
Caser	0.0706	0.0421	0.0488	0.0706	0.0071
SASRec	0.0730	0.0423	0.0494	0.0730	0.0073
KGSR-GG	0.0776	0.0435	0.0515	0.0776	0.0078

Knowledge-enhancing item associations help discover more potential preferred items for users. As Fig. 3 shows, when the length of the recommendation list was 10, the KGSR-GG algorithm achieved the highest values of all these five indicators on the two datasets. On the MovieLens-1M dataset, the KGSR-GG algorithm achieved *recall*@10, *MRR*@10, *NDCG*@10, *HR*@10 and *Precision*@10 that were 2.07, 1.3, 1.24, 2.1, and 0.22 percent higher than those achieved by the GRU4Rec algorithm that ranked the second among all algorithms; on the Book-Crossing dataset, it achieved the *recall*@10,

MRR@10, *NDCG*@10, *HR*@10 and *Precision*@10 that were 0.46, 0.12, 0.21, 0.46 and 0.05 percent higher than those achieved by SASRec that ranked the second among all algorithms. After analysis, the item interaction information in the knowledge graph can link some possible new items for the list of items of interest to the user, which helps the model to process new data in the test set in a more informed way and discover the potential preferences of the user.

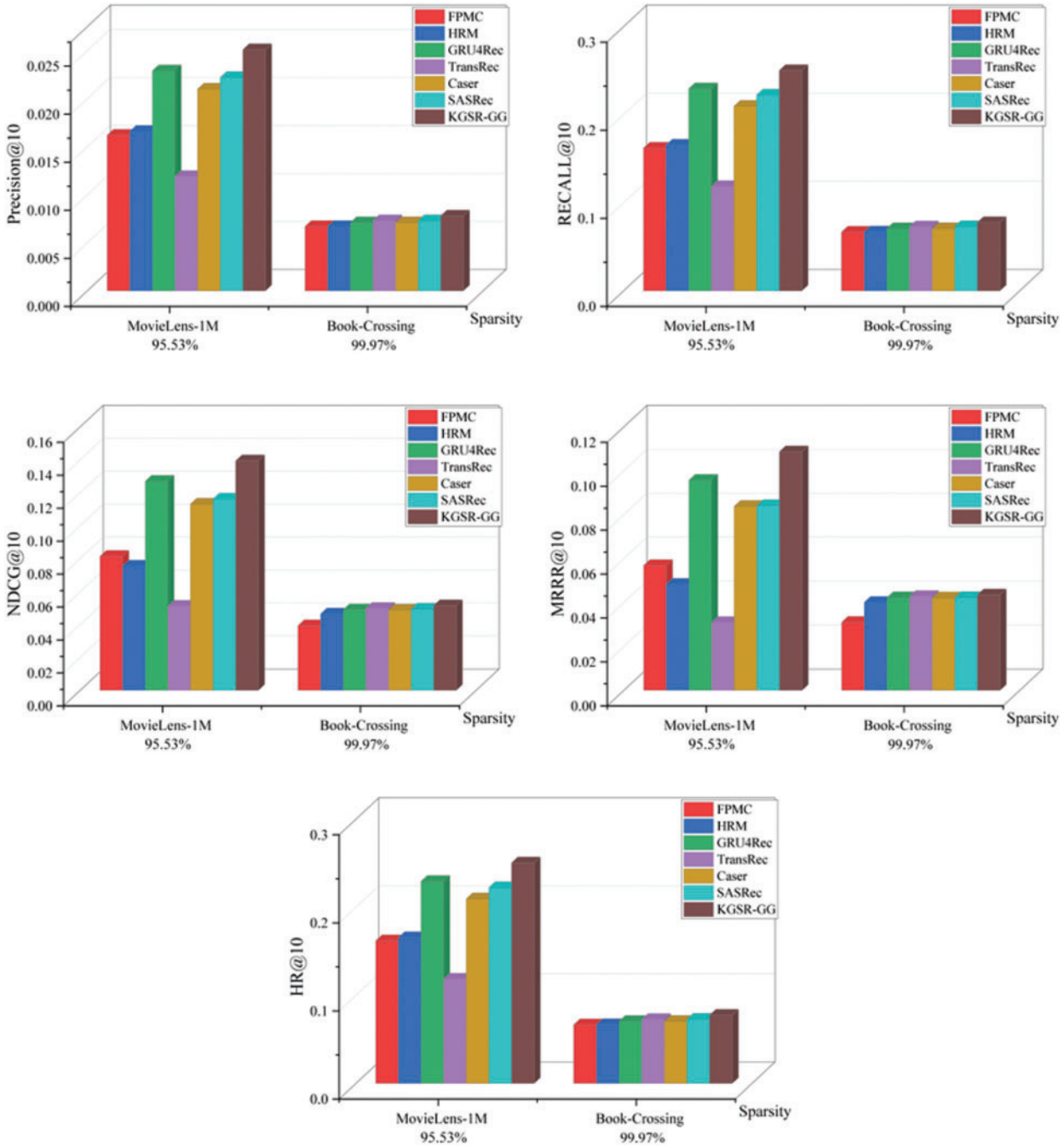

Figure 3: Comparison of performance between KGSR-GG and baseline models

The combination of graph neural network and Bi-GRU can extract the features of items of interest to users in the sequence more effectively. In Tabs. 3 and 4, the KGSR-GG model outperforms the GRU4Rec model, which also uses GRU modeling. GRU4Rec feeds the items clicked by the user into multiple GRUs, and finally converts them into ratings for the next sequence of users selecting all items through the forward network FFN. Compared with GRU4Rec, the graph neural network module of KGSR-GG can aggregate the item features of interest to users more effectively, while Bi-GRU can learn the degree of association between item and users better.

KGSR-GG performs better in sparser data sets. As shown in Figs. 3 and 4, the sparsity degree of the data has some effect on the experiment result. The KGSR-GG algorithm performed better on the dataset of lower data sparsity, the reason of which is that the dataset of lower data sparsity has incomplete KGs and interactive data, and these problems will reduce the algorithm's accuracy.

Other factors, aside from the data sparsity, which would affect the experiment result were explored. A contrast experiment was performed on the KGSR-GG algorithm. The length of the recommendation list K in the two datasets was set at 10, 20, 40, 60, and 100, to construct corresponding training sets and test sets. Fig. 4 shows the experiment results of the algorithm under different values of K. As the figure. shows, within a given range, a longer length of the recommendation list corresponded to a higher recall, MRR, NDCG and hit rate, but a lower precision. In sum, setting K at 10 could well reflect the performance of the KGSR-GG algorithm.

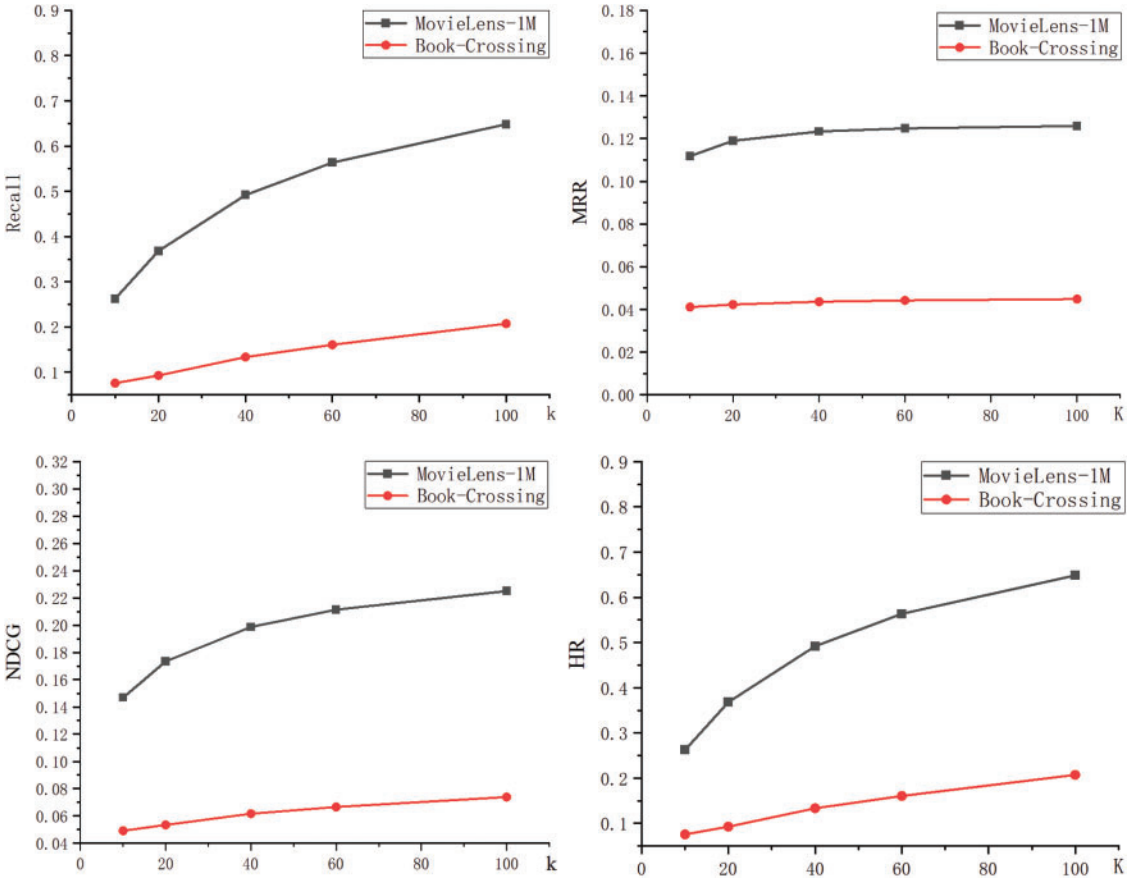

Figure 4: (Continued)

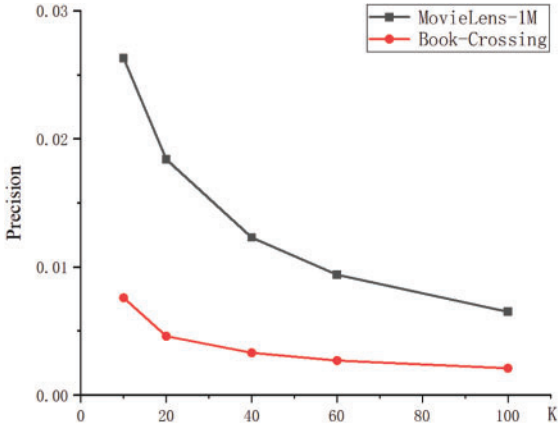

Figure 4: Performance of KGSR-GG under different lengths of the recommendation list

The superior performance of the KGSR-GG algorithm could be credited to the following three reasons. First, it aggregates the information of adjacent nodes to the central node by the graph convolutional network, establishes high-order connectivity, introduces the residual connection mechanism to the input and output layers of the graph convolutional network, and fulfills feature fusion of the items to the maximum degree. Second, it uses the TransR embedding technique to model the item attributes into knowledge graphs as auxiliary information, and the rich semantic information of the KG improves the interpretability and diversity of the recommendations. Third, the bi-directional GRU network fuses the item sequence and the KG entity sequence to model the user's short-term interest, and improves the accuracy of the recommendation.

5 Conclusions

In the present work, a new sequential recommender algorithm—KGSR-GG is proposed, which integrates the graph neural network-based recommender system, the knowledge graph and the gated recurrent unit into a unified recommender framework. Compared with conventional sequential recommender methods like the Markov chain and the recurrent neural networks, the proposed algorithm could capture more complicated dynamic information. Compared with other baseline models, the KGSR-GG algorithm, by dint of the high-order connectivity of graphs and the advantages of knowledge graphs, improves the accuracy and diversity of the recommendations, and achieves better performance than the baseline models on two datasets. Future work will focus on how to create high-quality knowledge graphs and dynamic sequences to improve the recommendation.

Acknowledgement: Conceptualized and designed the research work, J.P.Y and X.J.L; performed the implementation and evaluation of the research work, K.Y.C; wrote the manuscript, K.Y.C; performed investigation, made revisions to the manuscript and supervised the research work, Y.J.W. All authors have read and agreed to the published version of the manuscript.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] C. W. Cai, R. N. He and J. McAuley, "SPMC: Socially-aware personalized markov chains for sparse sequential recommendation," in *Artificial Intelligence. 26th Int. Joint Conf., IJCAI 2017*, Melbourne City, Australia, pp. 1476–1482, 2017.

[2] C. F. Xu, P. P. Zhao, Y. C. Liu, J. J. Xu, V. Sheng *et al.,* "Recurrent convolutional neural network for sequential recommendation," in *World Wide Web Conf.. 19th International Conf.*, San Francisco City, USA, pp. 3398–3404, 2019.

[3] T. Li, H. Li, S. Zhong, Y. Kang, Y. Zhang *et al.,* "Knowledge graph representation reasoning for recommendation system," *Journal of New Media*, vol. 2, no. 1, pp. 21–30, 2020.

[4] J. Huang, W. X. Zhao, H. J. Dou, J. R. Wen and E. Y. Chang, "Improving sequential recommendation with knowledge-enhanced memory networks," in *Int. ACM SIGIR Conf.. 41st Int. Conf.*, New York City, US, pp. 505–514, 2018.

[5] B. C. Wang and W. T. Cai, "Knowledge-enhanced graph neural networks for sequential recommendation," *Information-an International Interdisciplinary Journal*, vol. 11, no. 8, pp. 1–14, 2020.

[6] R. Sun, X. Z. Cao, Y. Zhao, J. C. Wan, K. Zhou *et al.,* "Multi-modal knowledge graphs for recommender systems," in *Information & Knowledge Management. 29th ACM Int. Conf.*, New York City, USA, pp. 15–25, 2020.

[7] L. Palaniappan and K. Selvaraj, "Profile and rating similarity analysis for recommendation systems using deep learning," *Computer Systems Science and Engineering*, vol. 41, no. 3, pp. 903–917, 2022.

[8] A. H. Hussein, Q. M. Kharma, F. M. Taweel, M. M. Abualhaj and Q. Y. Shambour, "A hybrid multi-criteria collaborative filtering model for effective personalized recommendations," *Intelligent Automation & Soft Computing*, vol. 31, no. 1, pp. 661–675, 2022.

[9] X. Zhao and P. Keikhosrokiani, "Sales prediction and product recommendation model through user behavior analytics," *Computers, Materials & Continua*, vol. 70, no. 2, pp. 3855–3874, 2022.

[10] R. Sabitha, S. Vaishnavi, S. Karthik and R. M. Bhavadharini, "User interaction based recommender system using machine learning," *Intelligent Automation & Soft Computing*, vol. 31, no. 2, pp. 1037–1049, 2022.

[11] X. W. Zhu, P. P. Zhao, J. J. Xu, J. H. Fang, L. Zhao *et al.,* "Knowledge graph attention network enhanced sequential recommendation," in *Proc. of the Asia-Pacific Web (APWeb) and Web-Age Information Management (WAIM) Joint Int. Conf. on Web and Big Data*, Tianjin City, China, pp. 181–195, 2020.

[12] L. Sang, M. Xu, S. S. Qian and X. D. Wu, "Knowledge graph enhanced neural collaborative recommendation," *Expert Systems with Applications*, vol. 164, no. 12, pp. 1–13, 2021.

[13] L. J. Xie, Z. M. Hu, X. J. Cai, W. S. Zhang and J. J. Chen, "Explainable recommendation based on knowledge graph and multi-objective optimization," *Complex & Intelligent Systems*, vol. 7, no. 9, pp. 1–12, 2021.

[14] S. Rendle, C. Freudenthaler and L. Schmidt-thieme, "Factorizing personalized markov chains for next-basket recommendation," in *World Wide Web. 19th Int. Conf.*, Raleigh City, USA, pp. 811–820, 2010.

[15] P. Wang, J. Guo, Y. Lan, J. Xu, S. Wan *et al.,* "Learning hierarchical representation model for nextbasket recommendation," in *38th Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, Santiago City, Chile, pp. 403–412, 2015.

[16] B. Hidasi, A. Karatzoglou, L. Baltrunas and D. Tikk, "Session-based recommendations with recurrent neural networks," *Computer Science*, vol. 1, no. 1, pp. 1–10, 2015.

[17] R. He, W. C. Kang and J. Mcauley, "Translation-based recommendation," in *Eleventh ACM Conf. on Recommender Systems*, Como City, Italy, pp. 161–169, 2017.

[18] J. Tang and K. Wang, "Personalized top-n sequential recommendation via convolutional sequence embedding," in *Eleventh ACM Int. Conf. on Web Search and Data Mining*, California City, USA, pp. 565–573, 2018.

[19] W. C. Kang and J. Mcauley, "Self-attentive sequential recommendation," in *Industrial Conf. on Data Mining. ICDM 2018*, Sentosa City, Singapore, pp. 197–206, 2018.