IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 27 March 2023, accepted 4 April 2023, date of publication 13 April 2023, date of current version 28 April 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3266835

RESEARCH ARTICLE

MPL-TransKR: Multi-Perspective Learning Based on Transformer Knowledge Graph Enhanced Recommendation

JIANKANG SHI AND KAI YANG
School of Computer Science and Software Engineering, University of Science and Technology Liaoning, Anshan 114051, China
Corresponding author: Kai Yang (asyangkai@126.com)

This work was supported in part by the Basic Research Project for Institutes of Higher Learning funded by the Liaoning Provincial Department of Education under Grant LJKZ0285.

ABSTRACT The emergence of recommender system is aimed at solving the problems brought by information explosion to human life and even the development of human society. As a traditional recommendation technique, collaborative filtering often encounters sparsity and cold start problems in many recommendation scenarios. Therefore, researchers have found that the introduction of side information can solve these problems to a certain extent and improve the performance of recommender systems. The knowledge graph is a heterogeneous graph that contains rich semantic relationships among items. The Multi-Perspective Learning based on Transformer Knowledge Graph Enhanced Recommendation (MPL-TransKR) proposed in this paper uses the knowledge graph as the side information for input and introduces the multi-head self-attention mechanism by reasonably combining the transformer idea. While learning the high-order neighborhood information of the items, the long-distance information between the items is captured, and the weight value is assigned to the user through the attention mechanism to strengthen the user representation to realize user-item multi-perspective learning to enhance the performance of the recommendation model. Through extensive experiments using public datasets, we demonstrated that MPL-TransKR performs well in book and music recommendations, surpassing state-of-the-art baselines on several metrics.

INDEX TERMS Recommender systems, knowledge graph, transformer, attention mechanism.

I. INTRODUCTION

With the continuous development of society, living in the current information age, we are not only served by ever-expanding data, but also impacted by the high overload of data. It is difficult for people to quickly and accurately lock their goals in massive amounts of data, which leads to a waste of time and resources. Therefore, it is important to design an efficient and accurate recommender system (RS).

Collaborative filtering (CF) [1] is a classic recommendation technique. It finds the preferences of users by mining the historical behavior data of users, so as to predict the items that users like and make recommendations. However, the cold start problem and sparse matrix problem faced by the algorithm have not been well alleviated, which also leads to excessive reliance of the algorithm on historical data and the user-item interaction matrix, which is not very friendly to new users and new items.

As one of the most challenging problems in recommender systems, the core of the cold start problem is the recommendation of new users or items. "New" means that there is not enough interaction matrix between users or items to be used as a reference. Therefore, it is difficult to determine the interest preferences of new users and characteristics of cold start items. For this reason, we tried to propose our own solution. First, we need to know that knowledge graph is a semantic network that expresses the relationship between items in the real world through attributes [2]. It is composed of numerous "entity-relation-entity" triples, and entities are connected through relationships to form a network knowledge structure. The method adopted in this study is to input the knowledge graph as side information. When the cold-started item has

The associate editor coordinating the review of this manuscript and approving it for publication was Hui Ma.

only limited interactive data, the rich structural knowledge in the knowledge graph can be mined to improve the item itself, such as the author or publisher of a book, etc., and the item representation can be further enriched through this associated information, and then recommended in a way that determines which users are likely to be interested in the book, which to some extent alleviates the cold start problem. Simultaneously, the knowledge graph-based recommender system integrates the semantic relations among items into the collaborative signal, effectively mining the high-level correlation information between items, and then understanding the user's intention at the semantic level to enhance the personalized recommendation performance of the recommendation model and the interpretability of the recommendation results.

Previous knowledge graph-based recommender systems are roughly divided into two categories: meta-path based methods and embedding based methods. The meta-path based [3], [4], [5] methods need to input manually defined meta-paths with high-order information into the prediction model in advance. However, because of their excessive dependence on artificially designed meta-paths, they have certain requirements for knowledge in related fields, and this is a very difficult task for complex knowledge graphs [6]; for example, in news recommendation, the relationship and various entities are in different areas, so it is impossible to design meta-paths in an artificial way. The embedding-based methods [7], [8], [9], [10], [11] are used to preprocess the knowledge graph through the knowledge graph embedding (KGE) method, and embed the entities and relations in the knowledge graph into a continuous vector space. Thus, entity and relation embeddings can be easily used in other tasks based on preserving the inherent structure of the knowledge graph [12]. KGE has high flexibility in using knowledge graphs to assist recommender systems, but it focuses on modeling strict semantic relevance, which makes it more suitable for in-graph applications such as link prediction, classification, and relation fact extraction than processing recommendation tasks in practical applications [13]. At the same time, KGE often requires pre-training, therefore, end-to-end training methods are lacking.

In general, to overcome the limitations of existing methods, we propose MPL-TransKR. The design goal of the model is to fully collect the high-order association information between items and entities in the knowledge graph, at the same time, introduce the attention mechanism to strengthen user and item representation, and conduct training and learning from multiple perspectives of users and items, so as to make the model carry out more targeted recommendation and enhance the performance of the recommendation algorithm. Compared with previous traditional recommender systems, MPL-TransKR has three main advantages: (1) Training and learning from multiple perspectives of users and items compensate for the problem that traditional recommendation algorithms only focus on unilateral learning of user representation or item representation, resulting in insufficient interpretation of user representation semantic information or insufficient mining of structural information between items and item representation semantic information. (2) We input the knowledge graph as side information, and design the Knowledge Graph Enhancement Block (KGE-Block) to capture the structured neighborhood information between items in the knowledge graph, and then fuse the item itself with the neighborhood information to enhance the item representation, which enhances the accuracy of the recommendation model and the interpretability of the results to a certain extent. (3) We designed a Sim-Transformer block (STrans-Block) based on the idea of Transformer [14], [15] On the one hand, it can help the item capture long-distance information, and on the other hand, it can allocate weights reasonably according to the user's attention, which greatly enhances the pertinence of the recommendation model when dealing with Click-through Rate (CTR) prediction tasks [16]

We applied MPL-TransKR to Book-Crossing (book) and Last.FM (music) datasets for the CTR prediction. The experimental results show that the proposed model achieves an average AUC gain of 15.1% and 8.4% for book and music recommendations, respectively, compared with other baselines The contributions of this study are summarized as follows:

- The proposed MPL-TransKR simultaneously collects information from multiple perspectives of users and items, i.e., assigns weights to different entities according to user attention, and simultaneously captures the neighborhood information and long-distance information of the item, which increases the accuracy and interpretability of the recommendation model.
- We performed CTR predictions for two real recommendation scenarios, movie and music, and the experimental results showed that MPL-TransKR performs better than several baselines.

The remaining section of the paper includes an introduction to related work in Section II and a summary of the tasks to be processed in Section III. Section IV includes the description of the proposed model structure in detail. The experimental comparison results of MPL-TransKR and the baselines are discussed in Section V. Finally, Section VI summarizes the study and briefly discusses a novel direction for future work.

II. RELATED WORK

In this section, we will introduce the related work in two parts: KG-aware recommendation and attention mechanisms applied in recommender systems (RS)

A. KG-AWARE RECOMMENDATION METHODS

With the deepening of the research on KG-aware recommendation, some end-to-end hybrid recommendation algorithms have emerged, such as RippleNet [17] and KGCN [18]. Compared with the traditional KG-aware recommendation algorithm, RippleNet does not need to manually design a metapath. Instead, RippleNet uses the propagation of user interaction entity records in the knowledge graph to explore the potential preference degree of users for the recommendation item. The KGCN takes each item as the central point, samples its neighboring entities, and captures the higher-order structure and semantic information between item and entities by mining the relevant attributes in the knowledge graph. The core idea of its design is to expand the neighborhood scope and enrich the item representation by means of multi-hops. However, excessive multi-hops make it easy for the training model to overfit and affect the recommendation results. DUPN [19] uses LSTM to model the user behavior sequence, then uses it for user modeling, and finally learns common user representations on multiple tasks. But DUPN is not suitable for most recommendation scenarios (such as news recommendations) because it relies on a complete sequence of user behavior. KGAT [6] implements the explicitly modeling of high-order connectivity on knowledge graph. Based on the feature that the entities in knowledge graph are naturally connect together through different relationships to form paths, high-level user-item connectivity can be fully mined

B. ATTENTION MECHANISMS APPLIED IN RS

In recent years attention mechanisms have been widely applied in various scenarios of recommender systems, such as Alibaba's Deep Interest Network (DIN) [20] and Deep Interest Evolution Network (DIEN) [21] DIN considers different items interacting with user's historical behavior by weighting the Attention of user's historical behavior sequence. The success of DIN is mainly to dynamically depict user's interests based on the attention mechanism, which solves the problem that only k independent interests can be expressed by the embedding of k-dimensional users. DIEN, as an evolutionary version of DIN, innovatively proposes the interest evolutionary network. Although DIEN can better model the migration of user interest, using GRU to model the user sequence makes the time consumption problem become its biggest bottleneck.

Inspired by the successful application of the attention mechanism originated from the field of Neural Machine Translation in recommender systems, we apply the graph convolutional network to explore the user-item representation based on the alignment of user-item interaction matrix and knowledge graph, and incorporate the idea of attention mechanism. Experimental results show that MPL-TransKR has good performance in several indicators

III. TASK DESCRIPTION

The recommended problems to be solved by MPL-TransKR are described as follows: First, the user set and the item set are represented by $\mathrm{U} = \{u_1, u_2, \cdots, u_\mathrm{M}\}$ and $\mathrm{V} = \{v_1, v_2, \cdots, v_\mathrm{N}\}$ respectively. The matrix $Y \in \mathrm{R}^{\mathrm{M}\times\mathrm{N}}$ represents the user-item interaction matrix formed according to the implicit feedback of the user. When $y_{uv} = 1$, it indicates that user u interacts with item v, such as playing, liking or commenting, and when $y_{uv} = 0$, there is no interaction between the user and item. Then, the knowledge graph is represented by G, $G = \{(h, r, t) \mid h, t \in E, r \in R\}$ which is composed of many relational triples (h, r, t), where $h, t \in E$ represent the head and tail entities of the triplet, respectively, and $r \in R$ represents the relation. For example, a simple relational triple (The Old Man and the Sea, book.written_work.author, Ernest Miller Hemingway) indicates that Ernest Hemingway wrote a work called Old Man and the Sea. Our task is to predict whether user u has a potential interest in the item v that has not been interacted with by the given user-item interaction matrix Y and knowledge graph G, i.e., whether user u is likely to interact with item v. By training the model, a prediction function $\hat{y}_{uv} = \mathcal{F}(u, v|\vartheta, Y, G)$ is learned, where $\hat{y}_{uv}$ represents the proba-

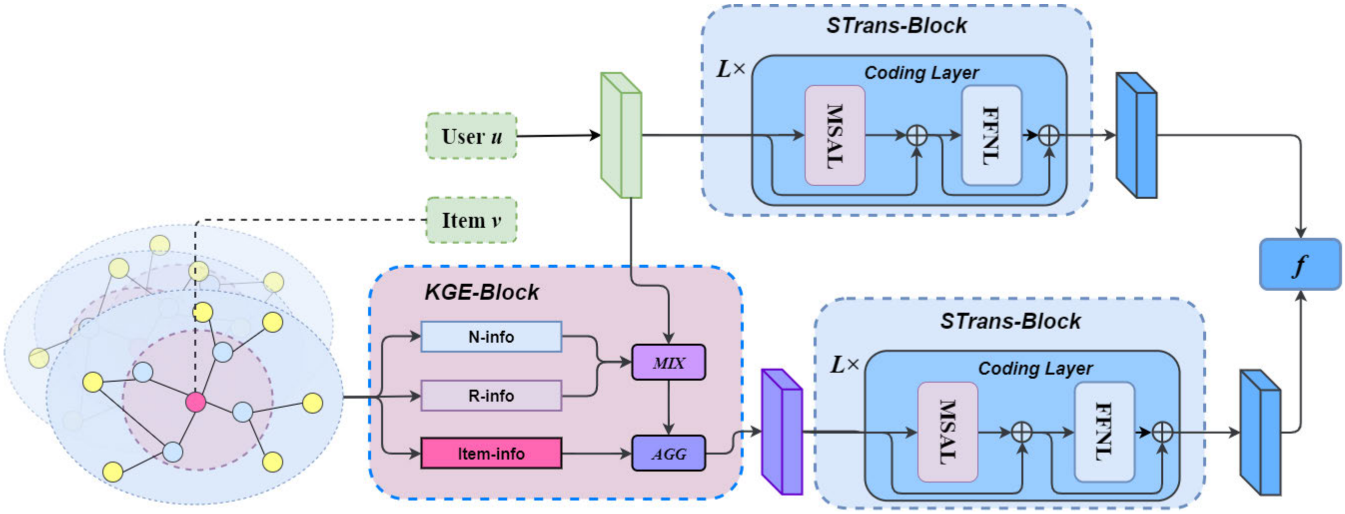

FIGURE 1. The overall architecture of MPL-TransKR. In KGE-Block, N-info represents the neighborhood entity information of v, R-info represents the relationship information between v and its connected entities, Item-info represents the v itself. In addition, the role of *MIX* is to integrate the neighborhood information of v, *AGG* is the aggregation function, f is the prediction function, and L is the number of Coding Layers for each STrans-Block.

bility that user u may interact with item v, and ϑ represents the parameters of the training model.

IV. MPL-TRANSKR

A. OVERALL ARCHITECTURE

An overview of the proposed MPL-TransKR is presented in Fig1. The MPL-TransKR is composed of a KGE-Block and two STrans-Blocks. From the user's perspective, we first obtain the user feature vector through the embedding method and then input it into the STrans-Block. User embedding is divided into different subspaces to realize the parallel attention calculation of multiple subspaces. Finally, after information enhancement, user embedding is obtained according to the user's attention of different types. From the perspective of items, we first mined the neighborhood information in the knowledge graph associated with the item through KGE-Block, fused the aggregated neighborhood information with the item to obtain item embedding, and then captured the long-distance information of the item through STrans-Block to complete further information enhancement. This means that after learning, user embedding and item embedding with enhanced information are input into the prediction function to obtain the prediction result.

The above content is a summary of the model architecture of MPL-TransKR. Then, we will introduce the specific design ideas of KGE-Block and STrans-Block. Finally, we introduce the implementation process and complete algorithm of MPL-TransKR.

B. KGE-BLOCK

Knowledge graphs contain rich semantic information between items, and mining structured information in knowledge graph can effectively improve a user's potential interaction items representation. For example, the user reads the literary work "The Old Man and the Sea", and its author is Hemingway. After enriching the item representation through the knowledge graph, it was found that Hemingway also published the work "A Farewell to Arms", so the user has potential interaction needs for the literary work "A Farewell to Arms". Overall, compared with simply defining the user's potential interest based on the user's click history, it is more practical to define a user's potential interest using the expansion information extracted from the knowledge graph. Therefore, we designed the KGE-Block, which is composed of two layers: a Neighborhood Information Extraction Layer and Information Fusion Layer. However, please note that unlike KGCN, MPL-TransKR improves the item presentation by only capturing the neighborhood information of the item through multi-hop operation, but enhances the item representation by combining KGE-Block to capture the information of the entities directly related to the item and using STrans-Block to capture the long-distance information of the item. This not only successfully integrates the structured semantic information in the knowledge graph, but also avoids unnecessary overfitting caused by redundant multi-hop operations that affects the performance of the recommendation model.

1) NEIGHBORHOOD INFORMATION EXTRACTION LAYER

First, we select a set of user u and item v, where $u \in \mathrm{U}$, $v \in \mathrm{V}$. In a realistic recommendation scenario, the number of entities directly associated with item v is often uncertain. In order to ensure that the computer remains normal, stable and efficient during the training of the model, we use $D(v)$ to represent the set of entities directly connected to the item, and then define the number of samples of $D(v)$ as a fixed value N. After training the model using two different datasets, we found that the experimental results were optimal when $\mathrm{N} = 4$. Using r_{v,v^i} to represent the relation vector between item v and the directly related entity v^i $(i = 1, \cdots, \mathrm{N})$, the preference of user u for the relation r_{v,v^i} is calculated using the inner product function $\mathcal{T}$:

$$\mu^u_{r_{v,v^i}} = \mathcal{T}\left(u, r\right) \tag{1}$$

$$\tilde{\mu}^u_{r_{v,v^i}} = \frac{exp\left(\mu^u_{r_{v,v^i}}\right)}{\sum_{e\in D(v)} exp\left(\mu^u_{r_{v,v^i}}\right)} \tag{2}$$

where preference coefficient $\tilde{\mu}^u_{r_{v,v^i}}$ is obtained after normalization of $\mu^u_{r_{v,v^i}}$. In realistic recommendation scenarios, some users may prefer to read a certain type of literary works, such as classical literature, historical classics and science fiction, while others may prefer to read works published by a certain writer or publications published by a certain publishing house. Therefore, this degree coefficient can improve recommendation accuracy to a certain extent.

After the preference coefficient $\tilde{\mu}^u_{r_{v,v^i}}$ weighted integration of entities in $D(v)$, we obtain the neighborhood information of item v:

$$v^u_{D(v)} = \sum\nolimits_{v^i\in D(v)} \tilde{\mu}^u_{r_{v,v^i}} v^i \tag{3}$$

2) INFORMATION FUSION LAYER

We used the aggregation function AGG to fuse the neighborhood information $v^u_{D(v)}$ of item v extracted from the knowledge graph with item v itself to enhance the representation of the original item and obtain $\tilde{v}$

$$\tilde{v} = AGG\left(v, v^u_{D(v)}\right) = Tanh\left(W \cdot \left(v + v^u_{D(v)}\right) + b\right) \tag{4}$$

where W is the weight of the linear transformation and b is the deviation. We sample the nonlinear activation function *Tanh*, which solves the problem that the output is not centered on zero, resulting in slower convergence compared with the traditional activation function *Sigmoid*

C. STRANS-BLOCK

Because MPL-TransKR has designed STrans-Block from both user and item perspectives with similar structure, we introduce the Strans-Block from these two perspectives. Note that we only briefly explain repetitive or similar parts.

From the item perspective, we input item embedding $\tilde{v}$ after information fusion into the STrans-Block, which is composed of multiple Coding Layers, and the structure of the Coding Layer is similar to that of the encoder in Transformer [14]. Then, the STrans-Block improves item embedding by capturing longdistance information. From the user perspective, we directly input user embedding into another STrans-Block, and strengthen the user representation by assigning weights to different entities based on the user's attention. Each Coding Layer in STrans-Block consists of two sub-layers, the Multi-head Self-Attention Layer (MSAL) and the Feed-Forward Network Layer (FFNL), whose specific structure is shown in Fig.2. In MSAL, user embedding and item embedding after the fusion of neighborhood information are divided into several sub-spaces. Each attention head only focuses on the feature information in the local space, then the captured attention head information is aggregated and processed into the FFNL. Finally, the user embedding $\hat{u}$ after the integrated preference weight and item embedding $\hat{v}$ after the integrated long-distance information were obtained. A residual connection was conducted around the two sub-layers, and normalization was performed.

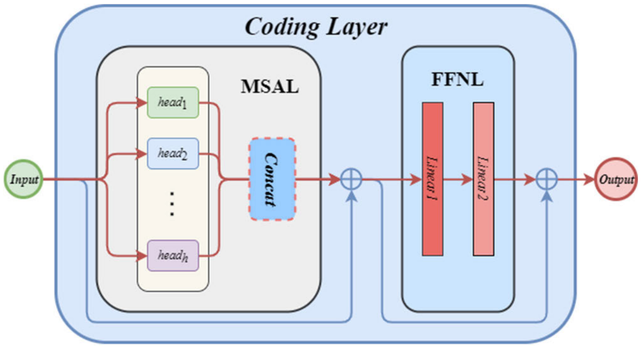

FIGURE 2. The architecture of coding layer.

Considering the user perspective as an example, we first extract the features of the user representation to obtain user embedding E_u which is input into the MSAL of the Coding Layer. In each self-attention head, E_u was linearly transformed several times to obtain Q, K and V matrices. The equations used are as follows:

$$Q = Linear\left(E_u\right) = E_u W^Q \tag{5}$$

$$K = Linear\left(E_u\right) = E_u W^K \tag{6}$$

$$V = Linear\left(E_u\right) = E_u W^V \tag{7}$$

where $W^Q \in R^{d\times d}$, $W^K \in R^{d\times d}$ and $W^V \in R^{d\times d}$ represent the weight matrices of the three groups of linear transformation layer learning. Then, the dot product method was used to calculate the correlation between matrix Q and matrix K to obtain the attention matrix. In order to ensure that the gradient can be stable after the dot product, a normalization process is carried out. The attention matrix is divided by the dimension d_k of matrix Q and matrix K to obtain the matrix $Score = \left(\frac{QK^T}{\sqrt{d_k}}\right)$. The softmax function was used to convert the matrix $Score$ into a probability matrix distributed in the interval of 0 to 1. Multiply matrix V to output the final attention matrix

$$Attention\left(Q, K, V\right) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{8}$$

Because each attention head in MSAL separately trains different Q, K, and V matrices, we finally concatenate the calculation results of h attention heads and multiply by the weight matrix $W^O \in R^{d\times d}$ to obtain the matrix S_u

$$\begin{aligned} S_u &= MultiHead\left(Q, K, V\right) \\ &= Concat\left(head_1, \ldots, head_h\right) W^O \end{aligned} \tag{9}$$

$$head_i = Attention\left(E_u W^Q, E_u W^K, E_u W^V\right) \tag{10}$$

$$\tilde{S}_u = LayerNorm\left(E_u + S_u\right) \tag{11}$$

where h represents the number of attention heads of the MSAL, and matrix $\tilde{S}_u$ is obtained after the residual linking and normalization of matrix S_u. Matrix $\tilde{S}_u$ contains all the attention head information, which is input into the FFNL to obtain the information enhanced user embedding $E_{\hat{u}}$

$$\mathrm{FFNL}\left(\tilde{S}_u\right) = \max\left(0, \tilde{S}_u W_1 + b_1\right) W_2 + b_2 \tag{12}$$

$$E_{\hat{u}} = LayerNorm\left(\tilde{S}_u + \mathrm{FFNL}\left(\tilde{S}_u\right)\right) \tag{13}$$

where, W_1 and W_2 are the weight matrices, and b_1 and b_2 are the bias vectors. Accordingly, from the item perspective, we used $E_{\tilde{v}}$ to represent item embedding enhanced by neighborhood information. After training and learning through MSAL and FFNL, we obtained the item embedding $E_{\hat{v}}$ integrated with long-distance information. The equations used are as follows:

$$\begin{aligned} S_v &= MultiHead\left(Q, K, V\right) \\ &= Concat\left(head_1, \ldots, head_h\right) W^O \end{aligned} \tag{14}$$

$$\tilde{S}_v = LayerNorm\left(E_{\tilde{v}} + S_v\right) \tag{15}$$

$$\mathrm{FFNL}\left(\tilde{S}_v\right) = \max\left(0, \tilde{S}_v W_1 + b_1\right) W_2 + b_2 \tag{16}$$

$$E_{\hat{v}} = LayerNorm\left(\tilde{S}_v + \mathrm{FFNL}\left(\tilde{S}_v\right)\right) \tag{17}$$

D. ALGORITHM

This section introduces the algorithm of MPL-TransKR, and the specific process description is presented in Algorithm 1. Note that to facilitate the reader's understanding, we replace E_u, $E_{\hat{u}}$, $E_{\tilde{v}}$ and $E_{\hat{v}}$ with u, $\hat{u}$, $\tilde{v}$ and $\hat{v}$ in the algorithm description. MPL-TransKR takes the user-item interaction matrix Y and knowledge graph G as the input to extract the feature information of the users and items respectively For a given user-item pair (u, v) (line 2), we first randomly sample the set $D(v)$ of entities directly associated with item v to obtain an item set $Neighbor.\_Filed$[N] of neighborhood size N, and then obtain the neighborhood information $v^u_{D(v)}$ of item v (line 3, 20-24) Finally, the item representation is improved by the aggregation function AGG to obtain $\tilde{v}$ (line 4). We input the $\tilde{v}$ into STrans-Block, which consists of a number of L Coding Layers. The final item representation $\hat{v}$ is obtained by passing MSAL and FFNL in the Coding Layer (line 5-10). After feature extraction, user u is directly input into STrans-Block, and then through MSAL and FFNL, the user's final representation $\hat{u}$ is obtained (line 11-16). Finally, both $\hat{u}$ and $\hat{v}$ are input into the prediction function f for probability prediction

Algorithm 1 *MPL-TransKR*

```
Input: Interaction matrix Y; knowledge graph $G$;
        Trainable parameters: $\{u\} \in$ U, $\{v\} \in$ V, $\{r\} \in$ R
        Hyper-parameters: N,$h$, $dim$, $L$, $\mathcal{T}(\cdot)$, $f(\cdot)$
              $AGG(\cdot)$, $LayerNorm(\cdot)$
Output: Prediction function: $\mathcal{F}(u, v|\vartheta, Y, G)$
1: while MPL-TransKR not converge do
2:   for $(u, v)$ in Y do
3:     $v^u_{D(v)} \leftarrow Extract\_Neighbor.\_Information(v)$;
4:     $\tilde{v} \xleftarrow{AGG} Tanh\left(W \cdot \left(v + v^u_{D(v)}\right) + b\right)$;
5:     for $l = 1, \ldots, L$ do
6:      for $i = 1, \ldots, h$ do
7:         $head_i \leftarrow Attention(Q_{\tilde{v}}, K_{\tilde{v}}, V_{\tilde{v}})$;
8:      $S_v \leftarrow Concat(head_1, \ldots, head_h)\, W^O$;
9:      $\tilde{S}_v \leftarrow LayerNorm(\tilde{v} + S_v)$;
10:     $\hat{v} \leftarrow LayerNorm\left(\tilde{S}_v + \mathrm{FFNL}\left(\tilde{S}_v\right)\right)$;
11:  for $l = 1, \ldots, L$ do
12:     for $i = 1, \ldots, h$ do
13:        $head_i \leftarrow Attention(Q_u, K_u, V_u)$;
14:     $S_u \leftarrow Concat(head_1, \ldots, head_h)\, W^O$;
15:     $\tilde{S}_u \leftarrow LayerNorm(u + S_u)$;
16:     $\hat{u} \leftarrow LayerNorm\left(\tilde{S}_u + \mathrm{FFNL}\left(\tilde{S}_u\right)\right)$;
17:   Calculate predicted probability $\hat{y} = f(\hat{u}, \hat{v})$;
18:   Update parameters by gradient descent;
19: return $\mathcal{F}$;
20: Function: $Extract\_Neighbor.\_Information(v)$
21:   $Neighbor.\_Filed[\mathrm{N}] \leftarrow D(v)$;
22:   for $i = 1, \ldots, \mathrm{N}$ do
23:      $v^u_{D(v)} \leftarrow v^u_{D(v)} \cup Neighbor.\_Filed[i]$;
24:   return $v^u_{D(v)}$;
```

The prediction function f for probability prediction and complete loss function $Loss$ are as follows:

$$\hat{y}_{uv} = f(\hat{u}, \hat{v}) \tag{18}$$

$$\begin{aligned} Loss(\hat{y}_{uv}, y_{uv}) = & -w_{uv}[(y_{uv} \times \ln \hat{y}_{uv}) \\ & + (1 - y_{uv}) \times \ln(1 - \hat{y}_{uv})] \end{aligned} \tag{19}$$

where w_{uv} represents the weight, y_{uv} represents the actual label given in the dataset, and $\hat{y}_{uv}$ represents the click-through rate of user u on candidate item v calculated by the prediction function. We take the actual label y_{uv} and the prediction label $\hat{y}_{uv}$ as the input of the loss function $Loss$. Note that the prediction label $\hat{y}_{uv}$ requires the $Sigmoid$ function to limit its prediction value to between zero and one before the input.

V. EXPERIMENTS AND RESULTS

A. DATASETS

We selected two datasets, one large and one small in two fields of book and music, to train MPL-TransKR, and the experimental process tried to simulate the real recommendation scenario as much as possible.

TABLE 1. Basic statistics for two datasets of Book-Crossing and Last.FM, including the number of users and items, the number of interactions and the number of KG triples.

Dataset	Book-Crossing	Last.FM
users	17,860	1,872
items	14,910	3,846
interactions	139,746	42,346
KG triples	19,793	15,518

- **Book-Crossing**[1] is one of the most widely used datasets in the field of recommender systems, and contains more than 1 million explicit ratings (ranging from 0 to 10) of books in the Book-Crossing community.
- **Last.FM**[2] contains more than 40,000 music playback records from a group of nearly 2 thousand users of Last.FM, including each user's music playlist and favorite singer list.

B. BASELINES

We selected six different baselines for comparison with MPL-TransKR. Among them, except that LibFM [22] is a feature-based factorization model, the other five baselines belong to the KG-aware methods.

- **CKE** [7] is a typical regularization-based method that enhances the matrix factorization of semantic information derived from TransR and utilizes heterogeneous information in the knowledge base to improve the quality of the recommender system.
- **LibFM** is a feature-based factorization model that is widely used in the task of processing CTR prediction. We use user ID and item ID as input of LibFM.
- **RippleNet** [17] takes a memory-network-like approach by extending the users' potential interest along the network of relationships in the knowledge graph, using a set of entities in the knowledge graph to gradually enrich user interest preferences for recommendation.
- **KGCN** [18] is an end-to-end network structure that uses item-centered knowledge graph neighborhood information for recommendation. This method can fully exploit the correlation between items and alleviate the impact of data sparsity on model recommendation performance to a certain extent.
- **KCRec** [23] is an end-to-end framework. It can effectively capture the inter-user and inter-item relatedness by propagating the relationship between item

[1] http://www2.informatik.uni-freiburg.de/~cziegler/BX/

[2] https://grouplens.org/datasets/hetrec-2011/

neighborhoods in KG, aggregating item features, and further combining with graph convolutional networks.

- **CKAN** [24] is a KG-aware methods which explicitly encodes the collaborative signals by collaboration propagation and proposes a natural way of combining collaborative signals with knowledge associations together.

C. EXPERIMENTS SETUP

The hyper-parameter Settings of MPL-TransKR for different datasets are listed in Table 2. When measuring the hyper-parameters, we used AUC as the evaluation standard, repeated the experiment ten times with each value of the hyper-parameter, and then took the average value as the final result. In the book recommendation, the ratio of training, evaluation and test set was set as 6:2:2, whereas in the music recommendation, the ratio of training, evaluation and test set was set as 8:1:1. In the CTR prediction experiment scenario, we put the test set into the trained model for testing, and used AUC, F1 and ACC to evaluate the ability of the model to handle the CTR prediction task. We implemented the MPL-TransKR code in Python 3.9, PyTorch 1.10.2, and Numpy 1.20.3.

TABLE 2. Hyper-parameter settings for the two datasets (*lr*: learning rate, N: neighbor sampling size, *L*: the number of Coding Layers, *h*: the Number of heads of multi-head self-attention layer).

	Book-Crossing	Last.FM
batch size	256	128
dim	64	16
lr	2×10^{-4}	5×10^{-4}
N	4	4
L	3	2
h	2	3

The hyper-parameter settings for the baselines were as follows: For the details of the hyper-parameter settings of part of the baselines, we refer to the design ideas of papers MKR [11] and KGCN. For CKE, the dimensions of the user and item embeddings for the two datasets were set to 128 and 32, respectively. Among them, the training weight for KG part is 0.1 for two datasets, and the learning rate $lr = 0.5$ and 0.1 for two datasets, respectively. The hyper-parameter settings for LibFM were the same as those in the baseline comparison experiment of KGCN. For RippleNet, $dim = 64$, $H = 2$, $\lambda_1 = 10^5$, $\lambda_2 = 0.1$, $lr = 0.005$ for Book-Crossing; $dim = 8$, $H = 2$, $\lambda_1 = 10^{-6}$, $\lambda_2 = 0.01$, $lr = 0.002$ for Last.FM. For KGCN $dim = 64$, $H = 1$, $\lambda = 2 \times 10^5$, $lr = 2 \times 1^{-4}$ for Book-Crossing; $dim = 16$, $H = 1$, $\lambda = 10^{-4}$, $lr = 5 \times 1^{-4}$ for Last.FM. The hyper-parameter settings of KCRec and CKAN are consistent with the original paper.

D. RESULTS

The results of MPL-TransKR and the baselines for CTR prediction are shown in Table 3. The analysis of the experimental results is summarized as follows:

- The results show that in the two recommendation scenarios of music and books, the KG-aware method CKE performs worse than the traditional feature-based factorization method LibFM in the three evaluation indices of AUC, F1 and ACC. The main reason is that CKE fail to make full use of knowledge graph due to the underlying design, resulting in a lack of visual and textual data.
- It is not difficult to find from Table 3 that KGCN has significantly improved in several indexes compared with previous baselines. The main reason for the analysis is that the KGCN captures the neighborhood information of items in the knowledge graph as much as possible by using the multi-hop neighborhood structure, which indicates that the knowledge graph is very important for recommendation. However, it only focuses on single-perspective learning of items, which greatly limits the improvement of its recommendation performance. Consequently, the final performance did not to exceed that of the proposed MPL-TransKR.
- The AUC results of KCRec and CKAN are better than other baselines. The analysis is mainly because multi-perspective learning can simultaneously learn from the perspectives of users and items, and more fully explore the inter-user and inter-item relatedness, which is helpful to improve the performance of the recommendation algorithm.
- Among all the baselines, CKAN has the best performance, mainly because compared with the previous KG-aware methods, CKAN tries a new way of combining collaborative information with knowledge information. This also proves the effectiveness of the knowledge-aware attention mechanism.
- In this experiment, we selected two datasets, music and books, and achieved an average AUC gain of 15.1% and 8.4% compared with each baseline, which indicates that MPL-TransKR can adapt well to recommendation tasks in different recommendation scenarios, and also proves that multi-perspective learning is necessary.

We proposed two variant models of MPL-TransKR, and the results are shown in the last three rows of Table 3. Among them, SPL-KR refers to only considering the perspective of the item and learning the item representation by KGE-Block mining the neighborhood information of the item in the knowledge graph, canceling the two STrans-Blocks in MPL-TransKR. MPL-KR is improved to multi- perspective learning based on SPL-KR, and the STrans-Block is added to the user's perspective. The experimental results show that multi-perspective learning is helpful for improving the performance of recommender system. However, MPL-KR failed to capture longdistance information between items compared with MPL-TransKR, which resulted in a less than optimal final performance

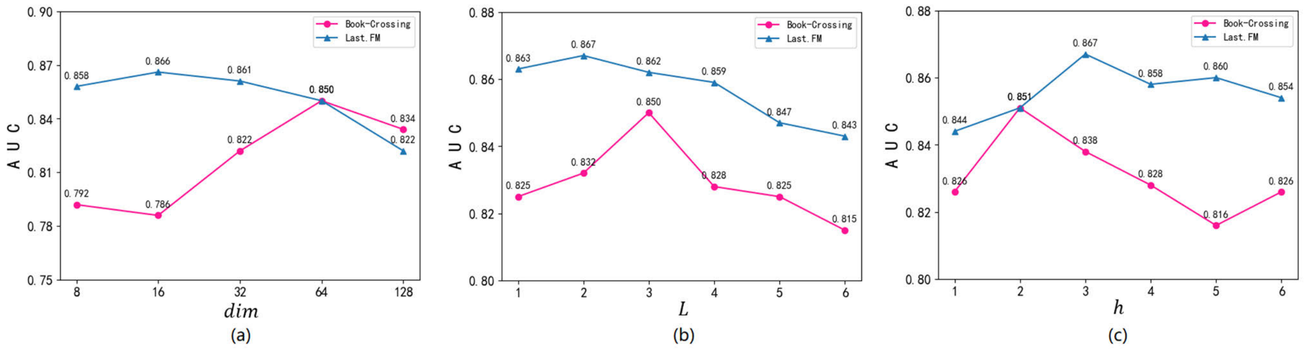

FIGURE 3. AUC results of MPL-TransKR with different parameter adjustment for Book-Crossing and Last.FM. a) AUC result of MPL-TransKR with different dimension of embedding. b) AUC result of MPL-TransKR with different number of Coding Layers. c) AUC result of MPL-TransKR with different number of heads in MSAL.

TABLE 3. The results of AUC, F1 and ACC in CTR prediction, which includes six baselines, two variants of MPL-TransKR and MPL-TransKR.

Model	Book-Crossing			Last.FM		
	AUC	F1	ACC	AUC	F1	ACC
CKE	0.675(-20.7%)	0.609(-22.2%)	0.636(-20.2%)	0.744(-14.2%)	0.673(-14.2%)	0.673(-15.0%)
LibFM	0.689(-19.0%)	0.619(-20.9%)	0.645(-19.1%)	0.776(-10.4%)	0.709(-9.6%)	0.702(-11.4%)
RippleNet	0.735(-13.6%)	0.655(-16.3%)	0.664(-16.7%)	0.778(-10.3%)	0.702(-10.5%)	0.698(-11.9%)
KGCN	0.739(-13.1%)	0.691(-11.7%)	0.689(-13.5%)	0.802(-7.5%)	0.725(-7.5%)	0.730(-7.8%)
KCRec	0.745(-12.5%)	0.655(-16.3%)	0.694(-12.9%)	0.819(-5.5%)	0.718(-8.4%)	0.748(-5.6%)
CKAN	0.754(-11.4%)	0.675((-13.8%)	0.695(-12.8%)	0.844(-2.7%)	0.771(-1.7)	0.750(-5.3%)
MPL-TransKR	**0.851**	**0.783**	**0.797**	**0.867**	**0.784**	**0.792**
MPL-KR	0.826(-2.9%)	0.755(-3.6%)	0.769(-3.5%)	0.850(-2.0%)	0.767(-2.2%)	0.771(-2.7%)
SPL-KR	0.711(-16.5%)	0.606(-22.6%)	0.654(-17.9%)	0.839(-3.2%)	0.750(-4.3%)	0.762(-3.8%)

1) IMPACT OF DIMENSION OF EMBEDDING

We first tested the influence of different embedding dimensions on the performance of the MPL-TransKR. As visualized in Fig. 3(a), the AUC results of MPL-TransKR fluctuate with increasing dim. The book dataset reached its maximum when $dim = 64$, whereas the music dataset reached its maximum when $dim = 16$. It can be seen that an increase in the dimensions can improve the performance of MPL-TransKR to a certain extent. However, if the dimension is too large, it affects the performance of the recommendation algorithm because it is prone to overfitting.

2) IMPACT OF LAYERS OF CODING LAYER

We tested the influence of different number of Coding Layers on the performance of MPL-TransKR by taking the value of L from one to six. From Fig. 3(b), we can see that when the value of L reaches four to six, in the recommendation of books and music, the AUC decrease significantly. Analyzing the reasons, we believe that too many Coding Layers can easily add more noise information, thus affecting recommendation performance.

3) IMPACT OF NUMBER OF HEADS IN MSAL

Finally, we studied the influence of the multi-head self-attention mechanism on the performance of MPL-TransKR by changing the number of heads in the MSAL. The result is shown in Fig. 3(c), we found that compared with single-headed self-attention mechanism, multi-headed self-attention mechanism is helpful to improve the performance of MPL-TransKR, and the number of heads in MSAL should not be too large. In terms of book and music recommendations, the AUC reached a peak when $h = 2$ and 3, respectively.

VI. CONCLUSION

The MPL-TransKR proposed in this paper is an end-to-end framework based on user-item multi-perspective learning, which is composed of two types of modules: KGE-Block and STrans-Block introduce multi-head self-attention mechanism while making KG-aware recommendation, which not only fully mining neighborhood information of items in knowledge graph through KGE-Block, STrans-Block is also used to capture the longdistance information of the item to further complete the item embedding and use the attention mechanism to assign weights to different users to enhance the user embedding. We conducted extensive experiments in two different recommended scenarios, book and music, and the results showed that the MPL-TransKR performed well on several indicators, significantly better than the six baselines. In addition, we conducted a large number of ablation experiments, which demonstrated the rationality of the final model structure settings and various hyperparameter Settings of MPL-TransKR.

REFERENCES

[1] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proc. 26th Int. Conf. World Wide*, 2017, pp. 173–182.

[2] S. Wu, F. Sun, W. Zhang, X. Xie, and B. Cui, "Graph neural networks in recommender systems: A survey," *ACM Comput. Surv.*, vol. 55, no. 5, pp. 1–37, Dec. 2022.

[3] X. Yu, X. Ren, Y. Sun, Q. Gu, B. Sturt, U. Khandelwal, B. Norick, and J. Han, "Personalized entity recommendation: A heterogeneous information network approach," in *Proc. 7th ACM Int. Conf. Web Search Data Mining*, Feb. 2014, pp. 283–292.

[4] Z. Huang, Y. Zheng, R. Cheng, Y. Sun, N. Mamoulis, and X. Li, "Meta structure: Computing relevance in large heterogeneous information networks," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 1595–1604.

[5] B. Hu, C. Shi, W. X. Zhao, and P. S. Yu, "Leveraging meta-path based context for top-*N* recommendation with a neural co-attention model," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 1531–1540.

[6] X. Wang, X. He, Y. Cao, M. Liu, and T. S. Chua, "KGAT: Knowledge graph attention network for recommendation," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 950–958.

[7] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W.-Y. Ma, "Collaborative knowledge base embedding for recommender systems," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 353–362.

[8] J. Huang, W. X. Zhao, H. Dou, J. R. Wen, and E. Y. Chang, "Improving sequential recommendation with knowledge enhanced memory networks," in *Proc. 41th Int. ACM SIGIR Conf. Res. Dev. Inf. Retr.*, Jun. 2018, pp. 505–514.

[9] H. Wang, F. Zhang, X. Xie, and M. Guo, "DKN: Deep knowledge-aware network for news recommendation," in *Proc. World Wide Web Conf.*, Apr. 2018, pp. 1835–1844.

[10] A. Dadoun, R. Troncy, O. Ratier, and R. Petitti, "Location embeddings for next trip recommendation," in *Proc. World Wide Web Conf.*, May 2019, pp. 896–903.

[11] H. Wang, F. Zhang, M. Zhao, W. Li, X. Xie, and M. Guo, "Multi-task feature learning for knowledge graph enhanced recommendation," in *Proc. World Wide Web Conf.*, May 2019, pp. 2000–2010.

[12] Q. Wang, Z. Mao, and B. Wang, "Knowledge graph embedding: A survey of approaches and applications," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 12, pp. 2724–2743, Dec. 2017.

[13] H. Wang, F. Zhang, M. Zhang, J. Leskovec, M. Zhao, W. Li, and Z. Wang, "Knowledge-aware graph neural networks with label smoothness regularization for recommender systems," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 968–977.

[14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, Dec. 2017, pp. 5998–6008.

[15] L. Xia, C. Huang, Y. Xu, P. Dai, X. Zhang, H. Yang, J. Pei, and L. Bo, "Knowledge-enhanced hierarchical graph transformer network for multi-behavior recommendation," in *Proc. AAAI Conf. Artif. Intell.*, May 2021, pp. 4486–4493.

[16] W. Guo, R. Su, R. Tan, H. Guo, Y. Zhang, Z. Liu, R. Tang, and X. He, "Dual graph enhanced embedding neural network for CTR prediction," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Aug. 2021, pp. 496–504.

[17] H. Wang, F. Zhang, J. Wang, M. Zhao, W. Li, X. Xie, and M. Guo, "RippleNet: Propagating user preferences on the knowledge graph for recommender systems," in *Proc. 27th ACM Int. Conf. Inf. Knowl. Manag.*, Oct. 2018, pp. 417–426.

[18] H. Wang, M. Zhao, X. Xie, W. Li, and M. Guo, "Knowledge graph convolutional networks for recommender systems," in *Proc. World Wide Web Conf.*, May 2019, pp. 3307–3313.

[19] Y. Ni, D. Ou, S. Liu, X. Li, W. Ou, A. Zeng, and L. Si, "Perceive your users in depth: Learning universal user representations from multiple e-commerce tasks," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 596–605.

[20] G. Zhou, X. Zhu, C. Song, Y. Fan, H. Zhu, X. Ma, Y. Yan, J. Jin, H. Li, and K. Gai, "Deep interest network for click-through rate prediction," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2018, pp. 1059–1068.

[21] G. Zhou, N. Mou, Y. Fan, Q. Pi, W. Bian, C. Zhou, X. Zhu, and K. Gai, "Deep interest evolution network for click-through rate prediction," in *Proc. AAAI Conf. Artif. Intell.*, Jul. 2019, pp. 5941–5948.

[22] S. Rendle, "Factorization machines with libFM," *ACM Trans. Intell. Syst. Technol.*, vol. 3, no. 3, pp. 1–22, May 2012.

[23] L. Zhang, Z. Kang, X. Sun, H. Sun, B. Zhang, and D. Pu, "KCRec: Knowledge-aware representation graph convolutional network for recommendation," *Knowl.-Based Syst.*, vol. 230, Oct. 2021, Art. no. 107399.

[24] Z. Wang, G. Lin, H. Tan, Q. Chen, and X. Liu, "CKAN: Collaborative knowledge-aware attentive network for recommender systems," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2020, pp. 219–228.

JIANKANG SHI was born in Binzhou, China, in 1998. He is currently pursuing the M.S. degree in electronic information engineering with the University of Science and Technology Liaoning, China. His research interests include artificial intelligence and recommender systems.

KAI YANG received the B.S. and M.S. degrees in computer application technology and the Ph.D. degree in metallurgical engineering from the University of Science and Technology Liaoning, China, in 2017. He is currently an Associate Professor and a Master's Supervisor with the School of Computer Science and Software Engineering, University of Science and Technology Liaoning. His current research interests include data mining, machine learning, and the modeling of complex industry process.