Enhanced Multi-Task Learning and Knowledge Graph-Based Recommender System

Min Gao, *Student Member, IEEE*, Jian-Yu Li, *Student Member, IEEE*, Chun-Hua Chen, *Member, IEEE*, Yun Li, *Fellow, IEEE*, and Jun Zhang, *Fellow, IEEE,* Zhi-Hui Zhan, *Senior Member, IEEE*

Abstract—In recent years, the *m*ulti-task learning for *k*nowledge graph-based *r*ecommender system, termed MKR, has shown its promising performance and has attracted increasing interest, because a recommendation task and a knowledge graph embedding (KGE) task can help each other to improve the recommendation. However, MKR still has two difficult issues. The first is how fully to capture users' historical behavior pattern in the recommendation task and how fully to utilize deep multi-relation semantic information in the KGE task. The second is how to deal with datasets with different sparsity. Tackling these challenging issues, this paper proposes an enhanced MKR (EMKR) approach with two novelties. First, we propose to utilize the attention mechanism to aggregate users' historical behavior for more accurately mining preferences in the recommendation task, and utilize the relation-aware graph convolutional neural network to fully capture the deep multi-relation neighborhood features in the KGE task, so as to address the first issue. Second, a two-part modeling strategy is proposed for a better representation of users in the recommendation task to expand the expressive ability of the model for adapting to datasets with different sparsity, so as to address the second issue. Extensive experiments are conducted on widely-used datasets and 11 approaches are used for comparison. The results show that the proposed EMKR can achieve substantial gains over the compared state-of-the-art approaches, especially in the situation where user-item interactions are sparse.

Index Terms—Recommender System, Knowledge Graph Embedding, Attention Mechanism, Multi-task Learning.

1 Introduction

With the proliferation of the Internet, data have grown exponentially, making users difficult to pick what they are or may be interested in. To improve users' experience, recommender systems (RSs) have been proposed to recommend items that meet the need of users and have been applied in various scenarios [1], [2], [3]. One of the most popular recommendation techniques is collaborative filtering [4], which utilizes users' historical interactions and makes recommendation based on users' similar preferences. However, traditional RSs, such as collaborative filtering-based RSs, often suffer from many deficiencies [5]. For example, data sparsity makes it difficult to calculate the similarity between users or items. Moreover, cold-start problem often occurs in traditional RSs when the recommendation involves a new user or a new item, because the new user or the new item has no item interaction records for determining the user-user and item-item similarity. To address these issues, many studies have proposed hybrid recommendation techniques, in which a variety of side information is introduced [6], such as item attributes [7], [8], item reviews [9], [10], and social networks [11], [12].

In recent years, introducing knowledge graphs (KGs) as a kind of side information into RSs has become a trending research direction. A KG organizes entities and relations of the real world with a graph, which effectively expresses semantic relations between entities [13]. By constructing the item KG, user KG, or user-item KG, more accurate users' preferences can be captured by mining the relations between the entities [5]. Compared with traditional RSs, KG-based RSs (KGRSs) make the reasoning process available, which can improve the explainability of RS [14], [15].

One of the most feasible and common ways to design KGRSs is to use KG embedding (KGE) [13] approaches to train a low-dimension vector for each entity and relation, while preserving the inherent structure of the KG. Then, useful vectors obtained by KGE can be utilized to enhance the representations of users and/or items in RS. For example, collaborative knowledge-based embedding (CKE) [2] uses TransR [16] for KGE and integrates entity

Manuscript received . . . .; revised . . . .; accepted . . . .. This work was supported in part by the National Natural Science Foundations of China (NSFC under Grants 62176094 and 92270105, in part by the Guangdong Natural Science Foundation Research Team under Grants 2021B1515120078 and 2018B030312003, in part by the Ministry of Science and Technology of China under grant G2022032012L, and in part by the National Research Foundation of Korea (NRF) funded by the Korean Government under Grants NRF-2022H1D3A2A01093478 and NRF-2020R1C1C1013806. (Min Gao and Jian-Yu Li are co-first authors.) (Corresponding authors: Chun-Hua Chen; Zhi-Hui Zhan.)

Min Gao and Zhi-Hui Zhan are with the School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China (e-mail: zhanapollo@163.com).

Chun-Hua Chen is with the School of Software Engineering, South China University of Technology, Guangzhou 510006, China.

Yun Li is with Shenzhen Institute for Advanced Study, University of Electronic Science and Technology of China, Shenzhen 518110, China, and i4AI Ltd, London WCIN3AX, United Kingdom.

Jian-Yu Li and Jun Zhang are with Hanyang University, Ansan, 15588, South Korea.

embedding, text embedding, and image embedding of items to enrich the representation of items. A deep knowledge-aware network (DKN) [17] uses TransD [18] for KGE and combines entity embedding and word embedding together to enrich the representation of news. However, TransR and TransD are designed to perform KG-related tasks but not recommendation task [19]. Thus, a more effective KGE should be designed to improve recommendation. Collaborative knowledge-aware attentive network for RSs (CKAN) [20] constructs the entity set of users and items and propagates knowledge associations along links in KG. Vectors of users and items can be obtained by training entity embeddings. It can be found that these approaches all rely on effective KGE. However, due to the incompleteness and errors of KG, directly using KGE vectors may deteriorate the recommendation.

Considering the relevance between recommendation task and KGE task, some researchers also study multi-task learning (MTL) approaches to alleviate the influence caused by the difference between KGE and recommendation tasks and to improve the generalization performance of KGRSs. MTL shares common knowledge and learns multiple tasks jointly or alternately, which is better than training them independently [21]. In the current MTL-based KGRSs, how to establish associations between multiple tasks and how to train multiple tasks are two important issues [22], [23], [24], [25]. Recently, a new *M*TL for *K*G-enhanced *r*ecommendation approach named MKR proposed by Wang et al. [19] has achieved satisfactory performance and efficiency for collaborative enhancement between the KGE task and the recommendation task. Different from other MTL-based KGRSs that directly use vectors obtained by KGE to represent users or items in RS, or simply use sum operator to establish associations between the two tasks (i.e., the KGE task and the recommendation task), MKR achieves interaction between items in RS and entities in KGE by using a soft parameter sharing mechanism [21]. MKR alternately trains the MTL framework to achieve satisfactory performance, showing the effectiveness of the parameter sharing mechanism.

However, MKR still faces two difficult issues. The first issue is the insufficient data utilization due to the poor feature learning ability of both RS module and KGE module. Specifically, in the RS module, when modeling users, users' latent features are extracted by only using the simple multilayer perceptron. Therefore, users' historical behavior pattern in RS is not sufficiently considered, which reduces the recommendation performance. Moreover, in the KGE module, the deep multi-relation neighborhood data in KG is still not fully utilized because MKR also extracts entities' features in KG by only using the simple multilayer perceptron. The second issue is the difficulty in adapting to datasets with different sparsity. Specifically, when users' behavior is very complex, it is not enough to express and model users' other possible potential behavior preferences by aggregating only several historical interactions, which degrades the recommendation performance.

To address the issues of MKR, we propose a more effective enhanced MKR (EMKR) approach with two novelties. The first novelty is that EMKR addresses the issue of insufficient data utilization in original MKR from two aspects. First, in the RS module, EMKR uses the attention mechanism [26] to aggregate users' historical behavior. As the attention mechanism can automatically learn the importance of different historical items, it can help the RS focus more on important patterns and can more fully capture users' historical behavior preferences. Second, in the KGE module, EMKR uses the relation-aware graph convolutional neural network (GCN) for capturing neighborhood features. By updating the graph and distinguish the importance of different relations, the relation-aware GCN can fully capture entities' deep multi-relation neighborhood features in KG to help recommendation. The second novelty is that EMKR uses a two-part modeling strategy to further enhance the expressive ability, so as to adapt to datasets with different sparsity. Specifically, the strategy includes a context part and an auxiliary part. Users' context part is beneficial for mining historical preferences. Users' auxiliary part is beneficial for expressing diverse and other possible users' preferences. In addition, to alleviate the impact of differences between the two tasks to further improve the generalization ability, EMKR extends the MTL sharing unit in MKR [19] (i.e., the soft parameter sharing mechanism) to establish the association of the relevant parts between recommendation task and KGE task. Therefore, like MKR, each module of EMKR can be redesigned. The novelties and contributions are summarized as follows:

1) For sufficient data utilization, EMKR has two module enhancements when compared with the original MKR. In the enhanced RS module, we propose to utilize the attention mechanism to aggregate users' historical behavior for more accurately mining historical preferences. In the enhanced KGE module, we propose to utilize the relation-aware GCN to fully capture the entities' deep multi-relation neighborhood features in KGs.

2) For adapting to datasets with different sparsity, EMKR uses the effective two-part modeling strategy in user modeling. To the best of our knowledge, EMKR is the first that both learns users' historical behavior and users' possible behavior for better representation, which is novel compared to existing works including MKR.

3) EMKR achieves satisfactory results in both click-through rate (CTR) prediction and Top-K recommendation scenarios on 4 datasets with different sparsity, especially when user-item interactions are sparse.

The following of the paper is organized as follows. Section 2 illustrates the problem formulation. Section 3 reviews related works. Section 4 describes the EMKR. Section 5 conducts experiments and analysis. Finally, Section 6 concludes the paper.

2 Problems Formulation

2.1 Definition of RS

Given a set of users $U = \{u_1, u_2, \dots, u_M\}$ and a set of items $V = \{v_1, v_2, \dots, v_N\}$, where M and N denote the number of users and items, respectively. A binary feedback matrix

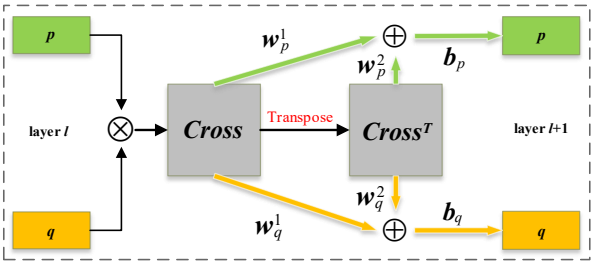

Fig. 1. MTL sharing unit used in MKR.

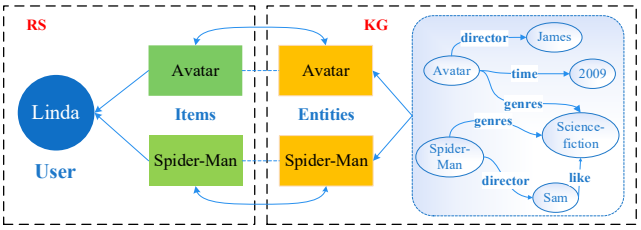

Fig. 2. Illustration of information delivery process between KG and RS. Given the interaction history and a KG, the arrow indicates the direction of information deliver.

$Y = \mathbb{R}^{M \times N}$ can be constructed based on users' historical behavior which contains watching, clicking, or rating. $y_{u,v}$ =1 indicates that there is an interaction between user u and item v, otherwise $y_{u,v}$ =0. A RS aims to predict whether a user u has a potential preference for an item v. We can formulate the predictive procedure as follows. Firstly, the RS learns a representation $\boldsymbol{u}$ and $\boldsymbol{v}$ (notes that we use **bold** symbols to represent the vector). Secondly, it measures the preferences for each user-item pair by calculating a scoring function $f_{RS}: \boldsymbol{u} \times \boldsymbol{v} \rightarrow \hat{y}_{u,v}$. Finally, the RS sorts items based on the predicted scores and then generates the recommendation list.

2.2 Definition of KG

In general, a KG with an entity set $\mathcal{E}$ and a relation set $\mathcal{R}$ is defined as $G=\{(h,r,t) \mid h,t \in \mathcal{E}, r \in \mathcal{R}\}$ where each triple (h, r, t) denotes a fact of the relation r from head entity h to tail entity t. For example, a triple (*Linda, brother, Mike*) means that *Mike is Linda's brother*. In this paper, each item in RS has a corresponding entity in KG.

A KG is an important side information to enhance the performance of RS. The rich links in KG complement the features of items in RS to a certain extent. Our work designs EMKR to alternately train the recommendation task and the KGE task. The problem is described as follows: given users U, items V, user-item interaction matrix Y, and KG G. The EMKR aims to learn KGE and recommend items to each user based on link information in KG and users' interaction history. The EMKR outputs embeddings of users, items, entities, and relations and the recommends item lists for users based on the scoring function.

3 Related Works

3.1 KGE Approaches

KGE are to embed the entities and relations of a KG into a low-dimension vector space while preserving the KG's structure [13]. They can be classified into two categories. One is translation-based approaches that use distance to predict the correctness of triples, such as TransE [27] and its variants [16], [18], [28], and the other is semantic matching approaches. Different from translation-based KGE, semantic matching approaches measure the plausibility of triples by exploiting similarity-based scoring functions, which mainly contain RESCAL [29], ANALOGY [30], and DistMult [31]. Recently, researchers also propose other KGE approaches based on graph neural networks, such as RGCN [32] and KGAT [33].

However, the above KGE approaches assumed all facts in KG are reliable and did not consider the incompleteness and errors, causing unreliable representation. Therefore, MTL was utilized to alleviate this issue by sharing useful information among several related tasks. Zhang et al. [34] designed a multi-task reinforcement learning framework for enhancing KGE. It jointly learned policy-based agents and KGE, where policy-based agents optimized the selection process of triples and used the score function to compute reward. Xu et al. [35] analyzed the several key relations of the product KGs and adopted different approaches to learn different relation prediction tasks and then trained the MTL framework.

Different from the above approaches, which are dedicated to improving the performance of KGE, EMKR aims to enhance the performance of RS. Essentially, the two studies [34], [35] design various tasks according to data characteristics of KGs and the needs of KGE, while EMKR views KGs as externally introduced side information. From another perspective, for the KGE task, the data of RSs can also be viewed as externally introduced side information. Therefore, existing popular KGE approaches can be incorporated into the proposed EMKR as the implementation of the KGE module. Designing and choosing suitable KGE approaches are also important research directions for future work.

3.2 MTL in Deep Learning

In the research process of deep learning, MTL approaches have been studied and applied widely in computer vision [36], [37], natural language processing [38], [39], [40] and other fields because they can balance the noise patterns of different tasks and reduce the risk of overfitting by sharing knowledge among related tasks. There are two common mechanisms of MTL, namely, hard parameter sharing and soft parameter sharing [21]. Hard parameter sharing mechanism allows different tasks to share hidden layers and separate task-specific layers. However, it is not suitable for improving the performance of multiple tasks with low correlation [41]. Soft parameter sharing mechanism allows different tasks to hold their own model and parameters, independently, only a part of useful information is shared among tasks, which alleviates the limitation of the hard mechanism. For example, Cross-stitch network [36] designs a cross-stitch unit to automatically learn the sharing of tasks by introducing trainable parameters.

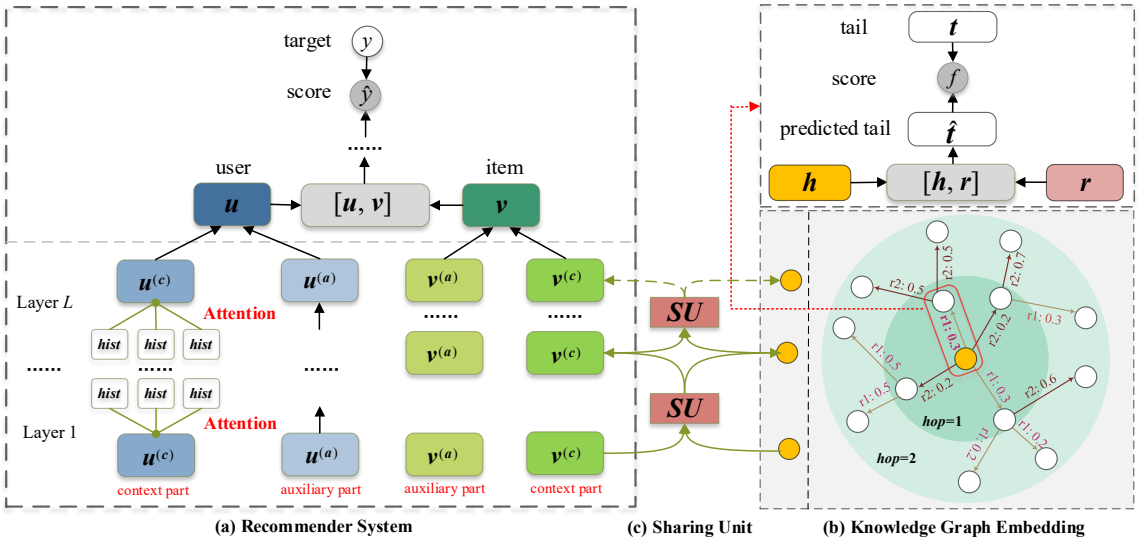

Fig. 3. The proposed EMKR framework.

3.3 MTL-Based KGRSs

Because the entities of KG overlap with items and/or users of RS, there is a correlation between KG-related task and recommendation task. Therefore, we can regard them as separate but related tasks. Using MTL can share useful knowledge and improve the generalization ability of RS model. For example, literature [22] alternately learns recommendation and KGE. The RS module mines users' preferences, and the KGE module mines the relationship between items. Literature [23] constructs rules by mining the paths in KG. It regards the difference of the relation path between two items as a rule. There are two modules in the MTL framework. One is used to learn rules, and the other is used to introduce rules for recommendation. The derived rules can improve the interpretability and performance of recommendation. In KG enhanced neural collaborative recommendation [24], the vectors obtained by KGE are used to enhance the representation of items, and it designs convolutional hidden layers to capture the complex correlations. In the effort of literature [25], KGE and recommendation tasks are associated with a joint learning framework, in which the item representation is obtained after fusing the entity and its neighborhood information.

However, most of the existing works simply use the vectors obtained by KGE task to enhance the representation of the RS data, without considering the negative impact of the missing links in KG nor considering the incompleteness or errors of the KG itself on KGE and recommendation. The MKR proposed by Wang et al. [19] is the first effort to use MTL soft parameter sharing to utilize side information of KG. MKR designs a new MTL sharing unit that is also used in [42] to share the relevant parts of the two tasks. We assume that $\boldsymbol{p}$ and $\boldsymbol{q}$ are the relevant vectors of two tasks in the l-th layer. The sharing unit is shown in Fig. 1. First, it will calculate a square matrix $\boldsymbol{Cross}$ by multiplying two vectors and obtain its transpose matrix $\boldsymbol{Cross}^T$. Then, it introduces trainable parameters $\boldsymbol{w}_p^1, \boldsymbol{w}_q^1, \boldsymbol{w}_p^2, \boldsymbol{w}_q^2, \boldsymbol{b}_p$, and $\boldsymbol{b}_q$ to map the two matrices to the latent vector space, and obtains the representation of the l+1-th layer, this process is formulated as

$$\begin{aligned} \boldsymbol{p} &= \boldsymbol{Cross} \cdot \boldsymbol{w}_p^1 + \boldsymbol{Cross}^T \cdot \boldsymbol{w}_p^2 + \boldsymbol{b}_p \\ \boldsymbol{q} &= \boldsymbol{Cross} \cdot \boldsymbol{w}_q^1 + \boldsymbol{Cross}^T \cdot \boldsymbol{w}_q^2 + \boldsymbol{b}_q \end{aligned} \tag{1}$$

4 Enhanced MKR

4.1 Framework

The relationship among the three parts (user, item, and KG) of KGRS is reconsidered herein. As shown in Fig. 2, users' historical click items imply users' behavioral characteristics. Due to alignment between items and entities, the rich link information of a KG will be propagated to items via entities, and then affects users. Meanwhile, items will also affect the entities because of this alignment.

The historical click items are not enough to fully reflect users' possible behavior features. For example, if a user purchased a science fiction novel, we can infer that he/she has a preference for that kind of items. Next time, this user may purchase a suspense novel, but this behavior feature is not included in historical interactions. Considering such an objective fact, we design a two-part modeling strategy, which models the historical behavior as the context part of users and introduces an auxiliary part to prevent the model from relying too much on historical data for expanding the expressive ability of the model. Similarly, items can be split into two parts. The context part captures the rich neighborhood features in KGs, and the auxiliary part prevents the model from relying too much on KG features for expending the expressive ability of items. The context part and auxiliary part of the user u are denoted as $u^{(c)}$ and $u^{(a)}$, respectively, while the context part and auxiliary part of the item v are denoted as $v^{(c)}$ and $v^{(a)}$, respectively.

The overall framework of our proposed EMKR is illustrated in Fig. 3. which includes MTL sharing unit, RS module, and KGE module. the proposed two-part strategy is used in the RS module. Considering the readability of the EMKR, we will first simply describe how the MTL sharing unit can be integrated into EMKR. Then, we in-

troduce two task modules in turn.

4.2 MTL Sharing Units

In EMKR, the MTL sharing unit designed by MKR is also retained to connect the relevant parts of KGE and recommendation. However, different from MKR, the relevant parts of both tasks are items' context part $\boldsymbol{v}^{(c)}$ and the aligned entity vector $\boldsymbol{e}$ in EMKR as shown in Fig. 3. For simplicity, we denote MTL sharing unit as

$$\begin{aligned} [\boldsymbol{v}^{(c)}, \boldsymbol{e}]_{l+1} &= SU[\boldsymbol{v}^{(c)}, \boldsymbol{e}]_l \\ [\boldsymbol{v}^{(c)}]_{l+1} &= SU[\boldsymbol{v}^{(c)}, \boldsymbol{e}]_l \to RS \\ [\boldsymbol{e}]_{l+1} &= SU[\boldsymbol{v}^{(c)}, \boldsymbol{e}]_l \to KG \end{aligned} \tag{2}$$

where $\to RS$ and $\to KG$ are used to distinguish RS output and KGE output, respectively. The entity can be a head entity $\boldsymbol{h}$ or tail entity $\boldsymbol{t}$. SU denotes a MTL sharing unit operation. In the l-th layer, two vectors are fed to SU as input and output their representation in the (l+1)-th layer. By doing so, EMKR achieves mutual learning of useful parameters in both tasks.

4.3 RS Module

As mentioned above, we design a new two-part modeling strategy in the RS module. Therefore, in the RS module, we represent both users and items in EMKR by two parts, which are the context part and the auxiliary part. Especially, for user modeling, the context part mainly captures the historical behavior of users, while the auxiliary part extends users' other possible behavior besides the historical behavior. For item modeling, the context part mainly captures multi-hop semantic features in the KG, while the auxiliary part represents other possible features besides the KG features.

4.3.1 User Modeling

The overall latent representation of a user $\boldsymbol{u}$ is the direct sum of the context part $\boldsymbol{u}^{(c)}$ and the auxiliary part $\boldsymbol{u}^{(a)}$ as

$$\boldsymbol{u} = \boldsymbol{u}^{(c)} + \boldsymbol{u}^{(a)} \tag{3}$$

We assume that the historical interaction items of the user u are constructed as a set $H = \{i1, i2, \cdots, iC\}$, where C denotes the number of historical items of user u.

We propose to use the attention mechanism [26] to model users' historical behavior for addressing the issue of insufficient data utilization in original MKR where users' historical behavior is not considered. Compared with assigning the same importance to each item [43], the attention mechanism is more flexible because it can automatically learn the different importance of different historical items, which can be defined as

$$\begin{aligned} \beta_k &= \boldsymbol{u}^{(c)T} \cdot \boldsymbol{W}_\beta \cdot \boldsymbol{v}_{ik} \\ \alpha_k &= \frac{\exp(\beta_k)}{\sum_{j=1}^{C} \exp(\beta_j)} \\ \boldsymbol{u}^{(c)} &= \sum_{k=1}^{C} \alpha_k \cdot \boldsymbol{v}_{ik} \end{aligned} \tag{4}$$

where β_k is attention value, $\boldsymbol{W}_\beta$ is a parameter that can be learned automatically. α_k is normalized attention value, which denotes the importance of $\boldsymbol{v}_{ik}$. Because a user may have many historical interactions, there will be several problems if aggregating all historical items. First, as we have to calculate a certain weight for each item, if there are too many historical interactions, some items may be assigned to a minimal weight that is almost negligible. Moreover, more historical interactions mean more parameters and computation, which will increase computational and time costs. Therefore, we set a hyperparameter *hist* to control the number of the historical items that are sampled for conducting the attention mechanism.

4.3.2 Item Modeling

Similar to the user representation, the latent vector representation of an item $\boldsymbol{v}$ is the direct sum of the context part $\boldsymbol{v}^{(c)}$ and the auxiliary part $\boldsymbol{v}^{(a)}$ as

$$\boldsymbol{v} = \boldsymbol{v}^{(c)} + \boldsymbol{v}^{(a)} \tag{5}$$

Note that in both the user modeling and the item modeling, the context part and the auxiliary part are directly summed because of several reasons. First, in the previous work [24], the author represents items as the sum of two parts (i.e., context and auxiliary). Therefore, such a direct, simple, and effective modeling strategy is inherited and extended to both users and items modeling. Second, the effect of direct summation is experimentally proved to be effective in Section 5.6. Third, summing the two parts directly is a simple yet efficient way. If the two parts are modeled in a weighted way, how to determine the weight of the two parts is also a problem to be solved.

4.3.3 Preference Prediction

We use L-1 sharing units to learn useful interactions between RS and KG. The final representation of items' context part $\boldsymbol{v}^{(c)}$ can be calculated by

$$\boldsymbol{v}^{(c)} = [\boldsymbol{v}^{(c)}]_L = SU^{L-1}[\boldsymbol{v}^{(c)}, \boldsymbol{e}]_{L-1} \to RS \tag{6}$$

Then, we use Equations (3) and (5) to combine two parts to obtain the final representation of users and items, respectively. Finally, we utilize the normalized inner product as the choice of predict function as

$$\hat{y}_{u,v} = f_{RS}(\boldsymbol{u}, \boldsymbol{v}) \tag{7}$$

4.4 KGE Module

4.4.1 Single Relation-aware GCN Layer

To capture multi-relation semantic features in KG, we use a relation-aware GCN layer to update the graph and distinguish the importance of different relations. Specifically, for a head entity h, a score function $g: \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is used to characterize the importance score between relation r to entity h as

$$\pi_r^h = g(\boldsymbol{h}, \boldsymbol{r}) \tag{8}$$

where $\boldsymbol{h} \in \mathbb{R}^d$ and $\boldsymbol{r} \in \mathbb{R}^d$ are the representations of entity h and relation r, respectively, d is the dimension of the representation vector. For a specific entity h, there are many neighborhood triples in the KG. These triples contain different relations, which reflect different semantic information. For example, existence triples *(Linda, brother, Mike)* and *(Linda, brother, John)*, "*Mike*" and "*John*" have the same importance to "*Linda*" under the relationship of "*brother*". We use $S_h^1 = \{(h, r_1, t_1), (h, r_2, t_2), \cdots, (h, r_s, t_s)\}$ to denote the set of triples directly connected to h and calculate the scores of all triples in S_h^1. Then, the normal-

ized score of the *i*-th triple is obtained by using softmax as

$$\tau_i = \frac{\exp\left(\pi_{r_i}^h\right)}{\sum_{j=1}^{s}\exp\left(\pi_{r_j}^h\right)} \tag{9}$$

A fusion representation of the neighborhood $\hat{\boldsymbol{h}}$ can be calculated by

$$\hat{\boldsymbol{h}} = \sum_{j=1}^{s}\tau_j \times \boldsymbol{t}_j \tag{10}$$

Then, a first-order representation of *h* finally can be obtained by aggregating the information of *h* and its neighborhood information. There are three aggregation operations (i.e., sum aggregator, concat aggregator, and neighbor aggregator) that can be used in GCN.

Sum aggregator first conducts add operator between two vectors and then uses a nonlinear transformation to calculate the latent representation as

$$aggregator_{sum} = \sigma\left(\boldsymbol{W}_k \cdot \left(\boldsymbol{h} + \hat{\boldsymbol{h}}\right) + \boldsymbol{b}_k\right) \tag{11}$$

where $\boldsymbol{W}_k$ and $\boldsymbol{b}_k$ are weight matrix and bias vector, respectively. σ is the activation function, which is the rectified linear unit (ReLU) activate function.

Concat aggregator [44] first concatenates the two vectors and then it uses a nonlinear transformation as

$$aggregator_{concat} = \sigma\left(\boldsymbol{W}_k \cdot concat\left(\boldsymbol{h}, \hat{\boldsymbol{h}}\right) + \boldsymbol{b}_k\right) \tag{12}$$

Neighbor aggregator [15] directly conduct a nonlinear transformation as

$$aggregator_{neighbor} = \sigma\left(\boldsymbol{W}_k \cdot \hat{\boldsymbol{h}} + \boldsymbol{b}_k\right) \tag{13}$$

In EMKR, we use the neighbor aggregator as equation (13). In Section 5.5, we will study the influence of different aggregators.

4.4.2 Relation-aware GCN with Multiple layers

After aggregating the immediate neighborhood information through a single GCN layer, we can obtain the representation of the entity, named 1-order entity representation. In order to explore entity's deeper semantic features, we stack multiple GCN layers to aggregate deep multi-relation neighborhood information in KG. The representation of each entity in the current layer will be used to calculate the representation of the next layer.

4.4.3 Complete KGE

A multi-layer GCN can capture dependencies across several relational steps and distinguish the importance of different relationships. We obtain the entity's *hop*-order latent representation $\boldsymbol{h}^{(hop)}$ via *hop* layers. Then, EMKR utilizes *L*-1 layer feature sharing unit to process $\boldsymbol{h}$, and we use *L* multilayer perceptron (MLP) to extract condensed features of relation $\boldsymbol{r}$. Moreover, $\boldsymbol{h}$ and $\boldsymbol{r}$ are concatenated, then *H*-layer MLP is used to predict the tail entity as

$$\begin{aligned} \boldsymbol{h} &= [\boldsymbol{h}]_L = SU^{L-1}[\boldsymbol{v}^{(c)}, \boldsymbol{h}]_{L-1} \rightarrow KG \\ \boldsymbol{r} &= MLP^L\left(\boldsymbol{r}\right) \\ \hat{\boldsymbol{t}} &= MLP^H\left(concat\left(\boldsymbol{h}, \boldsymbol{r}\right)\right) \end{aligned} \tag{14}$$

where $\hat{\boldsymbol{t}}$ denotes the predicted tail vector and *concat*() denotes concatenated operation. We score the triples by calculating the distance between the predicted tail entity $\hat{\boldsymbol{t}}$ and the real tail entity $\boldsymbol{t}$. The score of the triple (*h*, *r*, *t*) will be calculated by

$$score\left(h, r, t\right) = f_{KG}\left(\boldsymbol{t}, \hat{\boldsymbol{t}}\right) \tag{15}$$

where f_{KG} can use the normalized inner product.

Algorithm 1: Multi-task Alternately Training of EMKR

Input: Y - the interaction matrix of the task recommendation;
G - the knowledge graph;
Output: $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{h}, \boldsymbol{t}, \boldsymbol{r}$ - users, items, heads, tails, and relations;
$\mathcal{F}(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{h}, \boldsymbol{t}, \boldsymbol{r} \mid \Theta)$ - the prediction function;

1: **Begin**
2: Initialization;
3: **For** s = 1 to *epo // epo is the number of training times*
4: *// train recommendation task*
5: **For** i = 1 to n *// n is the number of iterations*
6: Sample *bs* interactions from Y;
7: Update parameters in Equations (2)-(6), (16);
8: **End For**
9: *// train KGE task*
10: Sample *bs* triples from G;
11: Update parameters in Equations (2), (8)-(16);
12: **End For**
13: **End**

4.5 Training Algorithm

The loss function of EMKR is defined as

$$\begin{aligned} \mathcal{L} &= \mathcal{L}_{RS} + \mathcal{L}_{KGE} + \mathcal{L}_{REG} \\ \mathcal{L}_{RS} &= \sum_{u\in U, v\in V} \mathcal{J}\left(\hat{y}_{u,v}, y_{u,v}\right) \\ \mathcal{L}_{KGE} &= -\left(\sum_{(h,r,t)\in\mathcal{G}} score\left(h,r,t\right) - \sum_{(h',r,t')\notin\mathcal{G}} score\left(h',r,t'\right)\right) \\ \mathcal{L}_{REG} &= \lambda \parallel \Theta \parallel_2^2 \end{aligned} \tag{16}$$

The first term uses a cross-entropy loss function $\mathcal{J}$ to calculate the loss of recommendation task. The second term calculates the loss of KGE and the negative samples (i.e., samples not in G) are constructed by replacing the head entity or the tail entity of the positive samples (i.e., samples in G). The last item is used to prevent overfitting, and λ is the weight of L2 regularization. When training EMKR, we employ a multi-task alternately training algorithm as presented in Algorithm 1. The training epoch is denoted as *epo* and each epoch consists of two stages. In the first stage, i.e., lines 4-8, EMKR trains recommendation task. In the second stage, i.e., lines 9-11, EMKR trains KGE task. We sample *bs* samples as a batch for training. Since we prefer to improve the performance of recommendation task, we introduce a hyperparameter *n* to control the frequency of alternately training KGE. Only when

TABLE 1
BASIC STATISTICS OF FOUR DATASETS.

DATASETS	USER	ITEM	REC	TRI	DEN
MovieLens-1M	6,036	2,347	753,772	20,195	5.32%
Book-Crossing	17,860	14,910	139,746	19,793	0.12%
Last.FM	1,872	3,864	42,346	15,518	0.59%
Dianping-Food	2,298,698	1,362	23,416,418	160,519	0.75%

recommendation task is trained n times, KGE will be trained once.

5 Experimental Verification and Comparison

5.1 Datasets

We conduct extensive experiments on MovieLens-1M, Book-Crossing, Last.FM, and Dianping-Food datasets.

The MovieLens-1M is collected from MovieLens website and records explicit ratings ranging from 1 to 5. Herein, the explicit records are converted into positive and negative samples based on a threshold proposed in [19]. Specifically, according to the literature [19], the threshold is set as 4 so that the records with rating less than 4 will be marked with 0, otherwise 1, where 0 indicates the negative sample and 1 indicates the positive sample. The Book-Crossing records explicit ratings range from 0-10. The Last.FM is collected from Last.fm online music system and it records how many times people listen. According to the literature [19], we view all records in these two datasets as positive samples. Moreover, for each user, we randomly sample from the user's unwatched items to obtain negative samples, so that the number of a user's negative samples is equal to that of his positive samples. The Dianping-Food contains more than 20 million records between around 2 million users and 1000 restaurants, which is collected from Dianping.com.

In the MovieLens-1M, Last.FM, and Book-Crossing datasets, the KG data are adopted from the one constructed in the work of MKR [19]. In the Dianping-Food, the KG data are adopted from the one used in the work of CKAN [20].

Table 1 reports statistics including the number of users, items, interaction records, triples in KG, and the density, where **DEN** is used to represent the density of the dataset. The smaller **DEN** is, the sparser the dataset is.

5.2 Compared Approaches

We compare the performance of EMKR with following most well-known and state-of-the-art approaches, including PER [45], CKE [2], DKN [17], LibFM [46], Wide & Deep [47], RippleNet [14], KGCN [15], MKR [19], CKAN [20], KGARA [48], and HKC [49]. Unless otherwise stated, the hyperparameters settings of these approaches are the same as reported in original papers or official codes.

PER [45] is a typical path-based approach that relies on predefined meta-paths features to capture users' preferences for items.

CKE [2] is a typical embedding-based CF recommendation approach, in which the latent representation of items is the sum of entity embeddings, text embeddings, and image embeddings.

DKN [17] is a recommendation framework especially proposed for news recommendation. The word embeddings of keywords in the news title and their corresponding entity embeddings and context embedding are concatenated as the input of Kim CNN [50] to obtain the vector of news.

LibFM [46] is a factorization machine approach with flexible feature engineering, which is a widely used CTR.

TABLE 2
Basic Setting of Hyperparameters of Four Datasets.

DATASETS	*d*	*epo*	*L*	*bs*	*lr01*	*lr02*	*hist*	*hop*
MovieLens-1M	16	20	2	4096	2e-2	1e-3	1	2
Book-Crossing	8	2	2	256	1e-3	5e-3	6	2
Last.FM	8	3	2	256	1e-3	1e-2	5	2
Dianping-Food	32	2	2	4096	5e-3	1e-5	4	2

Wide & Deep [47] is a deep recommendation approach that consists of a simple linear part and a deep nonlinear part, which enhance memorization and generalization.

RippleNet [14] understands the propagation process of users' preferences in KG as the spread of ripples of water. As the number of hop increases, users' potential preference for the entity in the ripples will weaken.

KGCN [15] models the importance of relations to a user. The importance coefficient is utilized to learn the neighborhood information of items based on GCN, which captures users' personalized and potential interests within KG.

MKR [19] comprehensively considers the correlation between recommendation and KGE tasks and uses soft sharing unit to automatically learn useful knowledge from two tasks. Both tasks only use multiple nonlinear transformation layers to extract latent features, but MKR obtained satisfactory results, illustrating the effectiveness of building MTL framework.

CKAN [20] constructs the entity set of users and items and encodes the collaborative signals by collaboration propagation. It proposes a natural way to combine collaborative signals with knowledge associations together.

KAGRA [48] uses a bi-Interaction model to learn the neighborhood information based on KGCN for accurately capturing users' preferences.

HKC [49] is an attempt to combine MKR and KGCN. Different from MKR, it uses KGCN to model the neighborhood information of nodes in KG as a receptive field.

5.3 Experiment Setup

In EMKR, we use the inner product as the scoring function of recommendation. We set the number of sampling neighborhood triples as 4 and λ=10e−6. We use grid search to find the best settings for EMKR, and the search scope for several important hyperparameters is as follows: L = {1, 2, 3}, *hop* = {1, 2, 3}, *lr01* = {0.0001, 0.0002, 0.001, 0.002, 0.005, 0.01, 0.02}, *lr02* = {0.0001, 0.0002, 0.001, 0.002, 0.005, 0.01, 0.02}, *hist* = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}. The setting of hyperparameters is given in Table 2. In Table 2, d denotes the vector dimension of all users, items, entities, and relations. L is the number of low layers to extract features and conduct MTL sharing unit. *bs* is batch size. *lr01* and *lr02* are the learning rate of recommendation and KGE tasks, respectively. *hist* is the number of historical items when modeling the users' context part. *hop* is the number of GCN layers when implementing KGE. For each dataset, according to the experimental configuration of MKR [19], we divide it into train dataset, valid dataset, and test dataset with a ratio of 6:2:2 for fair and scientific comparison. We evaluate EMKR in CTR prediction and

TABLE 3
THE RESULTS OF EMKR AND COMPARED APPROACHES ON AUC AND ACC IN CTR PREDICTION SCENARIO.

APPROACHES		MovieLens-1M		Book-Crossing		Last.FM		Dianping-Food	
		AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC
PER [45]	*Ave.*	0.710 (-21.3%)	0.664 (-18.4%)	0.623 (-12.3%)	0.588 (-11.7%)	0.633 (-17.5%)	0.596 (-15.7%)	/	/
CKE [2]	*Ave.*	0.801 (-12.2%)	0.742 (-10.6%)	0.671 (-7.5%)	0.633 (-7.2%)	0.744 (-6.4%)	0.673 (-8.0%)	/	/
DKN [17]	*Ave.*	0.655 (-26.8%)	0.589 (-25.9%)	0.622 (-12.4%)	0.589 (-11.6%)	0.601 (-20.7%)	0.581 (-17.2%)	/	/
LibFM [46]	*Ave.*	0.892 (-3.1%)	0.812 (-3.6%)	0.685 (-6.1%)	0.640 (-6.5%)	0.777 (-3.1%)	0.709 (-4.4%)	/	/
Wide & Deep [47]	*Ave.*	0.898 (-2.5%)	0.820 (-2.8%)	0.712 (-3.4%)	0.624 (-8.1%)	0.756 (-5.2%)	0.688 (-6.5%)	/	/
RippleNet [14]	*Ave.*	0.916 (-0.7%)	0.842 (-0.6%)	0.701 (-4.5%)	0.646 (-5.9%)	0.748 (-6.0%)	0.713 (-4.0%)	0.878 (-1.2%)	0.798 (-1.6%)
	Std	0.001	0.001	0.011	0.101	0.008	0.006	0.000	0.000
KGCN [15]	*Ave.*	0.901 (-2.2%)	0.824 (-2.4%)	0.677 (-6.9%)	0.622 (-8.3%)	0.801 (-0.7%)	0.729 (-2.4%)	0.834 (-5.6%)	0.764 (-5.0%)
	Std	0.002	0.002	0.004	0.003	0.005	0.004	0.001	0.001
MKR [19]	*Ave.*	0.914 (-0.9%)	0.840 (-0.8%)	0.727 (-1.9%)	0.678 (-2.7%)	0.791 (-1.6%)	0.742 (-1.1%)	0.500 (-39%)	0.499 (-31.5%)
	Std	0.001	0.002	0.008	0.013	0.008	0.011	0.000	0.005
CKAN [20]	*Ave.*	0.887 (3.6%)	0.811 (-3.7%)	0.739 (-0.7%)	0.655 (-5.0%)	**0.842 (+3.4%)**	0.745 (-0.8%)	0.878 (-1.2%)	0.801 (-1.3%)
	Std	0.000	0.000	0.001	0.001	0.001	0.003	0.000	0.000
KGARA [48]	*Ave.*	0.901 (-2.2%)	0.827 (-2.1%)	0.685 (6.1%)	0.630 (-7.5%)	0.791 (-1.7%)	0.715 (-3.8%)	0.842 (-4.8%)	0.766 (-4.8%)
	Std	0.001	0.001	0.004	0.004	0.005	0.006	0.000	0.001
HKC [49]	*Ave.*	0.922 (-0.1%)	**0.848 (-0%)**	0.741 (-0.5%)	0.697 (-0.8%)	0.805 (-0.3%)	0.748 (-0.5%)	/	/
EMKR	*Ave.*	**0.923**	**0.848**	**0.746**	**0.705**	0.808	**0.753**	**0.890**	**0.814**
	Std	0.001	0.001	0.002	0.001	0.002	0.002	0.000	0.000
Rank of EMKR		**1**	**1**	**1**	**1**	2	**1**	**1**	**1**

*"(+/-)" indicates the improvement or decline of the corresponding approach relative to EMKR. "**Rank of EMKR**" indicates the rank of EMKR among all the approaches. The **bold** indicates the best value and the underlined indicates the second-best value. "Ave." indicates the average results. "Std" indicates the standard deviation of 10 trails.*

Top-K recommendation scenarios. We use AUC and accuracy (ACC) to evaluate the performance of CTR prediction and use Precision@K, Recall@K, and F1@K to evaluate the performance of Top-K recommendation. The reasons for choosing these metrics are that they are appropriate metrics for evaluating the performance of approaches in CTR prediction and Top-K recommendation and are widely used in the literatures [14], [15], [19], [20]. Moreover, the AUC can be more robust and can better reflect the overall prediction performance of an approach than ACC metric, because ACC needs to be calculated based on a predefined threshold. The results of PER, CKE, DKN, LibFM, and Wide & Deep on MovieLens-1M, Book-Crossing, and Last.FM are derived from the results reported in MKR. The results of HKC are from the original paper. The results of other approaches are obtained by running the official source code provided by the authors. In order to reduce the randomness error, these approaches run 10 times on each dataset. The average results and the standard deviation are reported.

5.4 Comparison with State-of-the-Art Approaches

We compare the performance of EMKR and several compared approaches in CTR prediction and Top-K recommendation scenarios.

The results of all approaches in CTR prediction are reported in Table 3. Among all approaches, PER, CKE, and DKN perform quite poorly on three datasets. LibFM and Wide & Deep perform well on MovieLens-1M, but on Book-Crossing and Last.FM, the performance is bad, which illustrates LibFM and Wide & Deep have difficulty in capturing users' preferences in sparse scenarios. KGCN performs relatively well among all approaches because it can make full use of the multi-hop neighborhood features in KG to mine users' preferences. KGARA performs slightly better than KGCN. However, KGARA fails to make full use of the historical interactions in RS, so users' preference is not sufficiently captured. RippleNet, MKR, HKC, and CKAN are the best four of all the compared

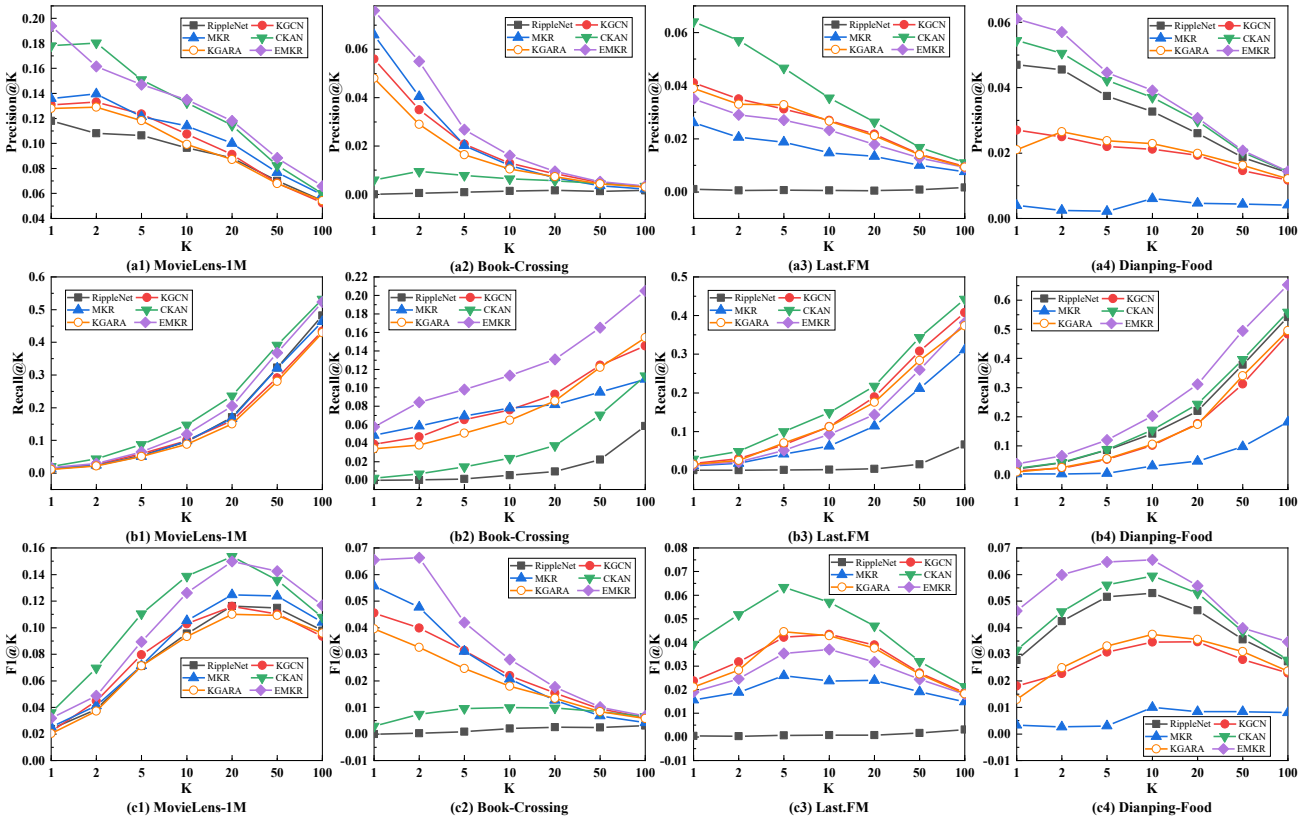

Fig. 4. The results of EMKR and compared approaches on Precision@K, Recall@K, and F1@K in Top-K recommendation scenario.

approaches. Specifically, the performance of HCK is the second best among all approaches on MovieLens-1M, and the performance of CKAN is the second best among all approaches on Dianping-Food.

EMKR performs best among all approaches on almost all metrics on all the four datasets. Specifically, EMKR obtains significant improvement on MovieLens-1M, Book-Crossing, and Dianping-Food. Although the AUC of Last.FM is not comparable to CKAN, EMKR still outperforms MKR and its overall performance ranks second. Compared with MKR, EMKR shows better performance on all datasets, indicating that EMKR is indeed an effective enhanced version of MKR, which can more effectively utilize the data in both KG and RS to improve the performance of recommendation. Moreover, EMKR performs well on sparse datasets such as Book-Crossing and Dianping-Food. This is because in sparse scenarios, users' preferences can be better understood by historical behavior. Even on a denser dataset such as MovieLens-1M where users' historical behavior is more complicated, EMKR still outperforms all compared approaches. We can also observe that the standard deviation of EMKR is smaller, indicating that the performance of EMKR is more stable. The above observations illustrate the general validity of EMKR.

Furthermore, we compare the Top-K recommendation performance of EMKR with RippleNet, KGCN, MKR, KGARA, and CKAN. The reason for only selecting these 5 approaches for comparison is that they perform significantly better than other compared approaches and the authors have provided source code. As shown in Fig. 4, EMKR also achieves outstanding or comparable performance on the MovieLens-1M, Book-Crossing, and Dianping-Food datasets. However, EMKR does not perform well on the Last.FM dataset. Nevertheless, EMKR is still superior to MKR, indicating that EMKR is indeed an enhanced version of MKR that can effectively utilize data.

Combining the results of both CTR prediction and Top-K recommendation scenarios, we can conclude that EMKR is very suitable for CTR prediction scenario with various scales of datasets and can accurately predict users' preference. The effect is especially pronounced on large-scale sparse dataset (i.e., Dianping-Food). In Top-K recommendation scenario, the advantage of EMKR is also very prominent. Only on a small-scale dataset (i.e., Last.FM) with a few historical records, the effect is unsatisfactory. This kind of datasets usually appears in some newly launched applications. In the real world, datasets with larger scale, more historical behavior, and sparser user-item interactions are more common. Therefore, EMKR has more practical application value to play an effect on this kind of datasets. Moreover, data sparsity has always been a challenge of RSs [5]. The significant effect on large-scale sparse datasets can better show the effectiveness of EMKR in coping with the challenge of data sparsity in RSs.

5.5 Comparison Among Different EMKR Variants

In this part, we investigate the influence of 3 aggregators, the influence of different multi-task sharing unit, and the influence of different KGE. Therefore, we design 4 variants of EMKR including EMKR (concat), EMKR (sum),

TABLE 4
THE RESULTS OF EMKR AND ITS FOUR VARIANTS ON AUC AND ACC IN CTR PREDICTION SCENARIO.

APPROACHES	MovieLens-1M		Book-Crossing		Last.FM		Dianping-Food	
	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC
EMKR	**0.923**	**0.848**	0.746	**0.705**	**0.808**	**0.753**	**0.890**	**0.814**
EMKR (concat)	**0.923**	**0.848**	**0.754 (+0.8%)**	0.695 (-1%)	0.800 (-0.8%)	0.743 (-1%)	0.889 (-0.1%)	**0.814**
EMKR (sum)	**0.923**	0.847 (-0.1%)	0.753 (+0.7%)	0.693 (-1.2%)	0.800 (-0.8%)	0.744 (-0.9%)	**0.890**	**0.814**
EMKR (stitch)	0.922 (-0.1%)	0.847 (-0.1%)	0.747 (+0.1%)	0.689 (-1.6%)	0.797 (-1.1%)	0.733 (-2%)	0.889 (-0.1%)	**0.814**
EMKR (simple)	0.922 (-0.1%)	0.847 (-0.1%)	0.751 (+0.5%)	0.690 (-1.5%)	0.802 (-0.6%)	0.750 (-0.3%)	0.889 (-0.1%)	**0.814**

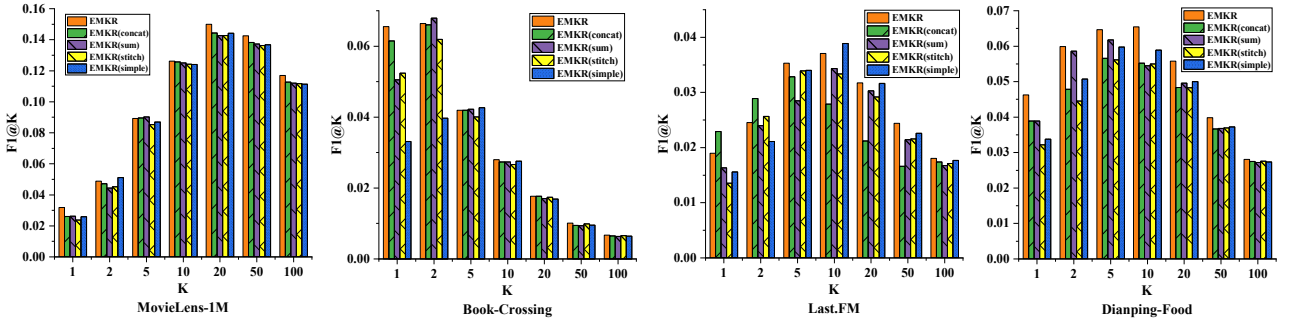

Fig. 5. The results of EMKR and its four variants on F1@K in Top-K recommendation scenario.

EMKR (stitch), and EMKR (simple). EMKR (concat) and EMKR (sum) use different aggregators introduced in Section 4.4.1. EMKR (stitch) uses the neighbor aggregator, but replaces SU by the cross-stitch unit in Cross-stitch networks [36]. EMKR (simple) replaces KGE module by simple KGE used in MKR. Table 4 and Fig. 5 report the performance of CTR prediction and Top-K recommendation, respectively. The results of Precision@K and Recall@K are reported in Fig S.1 in Supplementary Material.

After analyzing and comparing the results, we have the following observations and conclusions. (1) By comparing the results of EMKR (using neighbor aggregator), EMKR (concat), and EMKR (sum), we can find that EMKR performs best in both two recommendation scenarios. Specifically, in CTR prediction, the results of ACC and AUC metrics indicate that EMKR's performance on the four datasets is not worse than EMKR (sum) and EMKR (concat). In Top-K recommendation scenario, EMKR achieves the best results on almost all metrics of the four datasets. Above observations indicates the neighbor aggregator is the best choice for EMKR. Although neighbor aggregator only uses a non-linear transformation to process neighbor vector to represent the central node, the neglect of the central node does not necessarily damage performance. This is because, in our proposed MTL framework that uses the proposed two part-modeling strategy, the neighbor information in KG is modeled as the context part of items for enriching the presentation of items. Items with similar neighborhood are more similar, therefore, only using neighborhood to update the central node is enough for our approach. (2) By comparing EMKR and EMKR (stitch), we can find that EMKR performs better than EMKR (stitch) in almost all metrics of all datasets, which indicates that the sharing unit designed by MKR is a better choice because it learns a latent feature interaction from both horizontal and vertical directions. (3) By comparing EMKR and EMKR (simple), we can observe that the overall performance of EMKR is better than EMKR (simple) in almost all metrics of all datasets in both two scenarios, which further illustrates the effectiveness and sufficiency of our designed GCN-based KGE in utilizing the data in KG because their only difference is the KGE module.

5.6 Ablation Analysis

To prove the effectiveness of the two-part modeling strategy, we remove one part at a time for comparison. We report the decline in AUC and ACC metrics of several ablation experiments compared to the EMKR.

EMKR without users' context part $u^{(c)}$: we abandon users' context part to study the enhancement effect of historical interaction on the representation.

EMKR without users' auxiliary part $u^{(a)}$: we discard users' auxiliary part and represent users only using historical interactions to explore the extent to which the auxiliary part of users enhances users' representation.

EMKR without items' auxiliary part $v^{(a)}$: we remove items' auxiliary part of items and represent items by only using the related knowledge in KG.

The results are reported in Table S.1 of the Supplementary Material. We have the following observations: (1) On Book-Crossing and Last.FM datasets, no matter which part is removed, the overall performance of two metrics will not be better than the complete EMKR, which indicates the effectiveness of two-part modeling strategy. (2) When removing users' context part, the performance

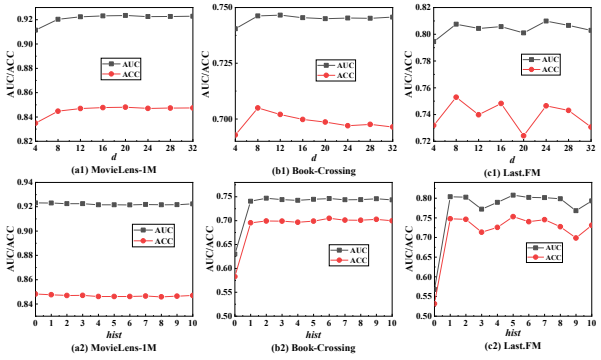

Fig. 6. The hyperparameter sensitivity of *d* and *hist*.

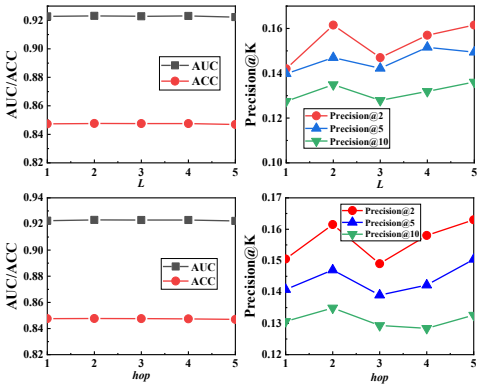

Fig. 7. The hyperparameter sensitivity of *L* and *hop*.

drops dramatically on 3 sparse datasets, that is, Book-Crossing, Last.FM, and Dianping-Food, which confirms the necessity and effectiveness of users' context part on enhancing the representation of users. This also indicates that whether to make full use of users' historical behavior will seriously affect the recommendation performance of the approaches on sparse datasets and our proposed EMKR can effectively and sufficiently utilize the historical data in RS. (3) On MovieLens-1M dataset, removing users' context part improves the performance slightly, while removing users' auxiliary part significantly damages the performance. This phenomenon indicates that the performance improvement of EMKR on this dataset relies more on users' possible behavior modeled by users' auxiliary part, and the historical behavior modeled by users' context part blurs users' preferences. This is because on a denser dataset, uses' behavior is more complex and changeable, the context part of users makes users prefer a few historical items, which cannot reflect users' real and complex preferences, thus limiting the expressive ability of the proposed EMKR. (4) On the Dianping-Food dataset, removing users' auxiliary part improves performance slightly, while removing users' context part damages performance, which indicates a sparse dataset relies more on historical behavior. Although removing a certain part may improve the performance slightly on some datasets, it severely hurts the performance on other datasets. The complete EMKR retaining two parts yields the overall best performance and is effective on datasets with different sparsity.

Note that, we do not study the role of items' context part when designing ablation experiments, because items' context part and the corresponding entity in KG are closely connected by MTL sharing unit. If we remove this part directly, not only EMKR has no connection with KG, but also MTL does not work. We still study the influence of items' context part in Section 5.8 to explore the mutual enhancement effect between recommendation and KGE tasks.

5.7 Parameter Sensitivity Analysis

5.7.1 Effects of the Dimension

We fix other hyperparameters, and vary *d* from 4 to 32 to study how the dimension affects EMKR. The results of AUC and ACC on three datasets are plotted in Fig. 6. It can be observed that on MovieLens-1M dataset, as the dimension increases, the performance tends to be stable; on Book-Crossing and Last.FM datasets, when *d* exceeds 8, although AUC may increase, ACC will decrease. Therefore, setting *d* to 8 can achieve a balance between AUC and ACC after considering the changing trend of the two metrics.

5.7.2 Effects of the Number of Historical Items

We fix other hyperparameters and set *hist* from 0 to 10 to investigate the influence of the size of sampling historical items. The results of AUC and ACC on three datasets are plotted in Fig. 6. On Book-Crossing and Last.FM datasets, when *hist* is set to 0, the performance of EMKR is the worst. However, when *hist* is set to 1, the performance is significantly improved, which indicates the dramatic effect of historical behavior on sparse datasets. As *hist* increases, the performance fluctuates slightly. Moreover, on MovieLens-1M dataset, the influence of *hist* is not obvious, indicating that users' preference prediction on this dataset does not depend on the attention modeling of historical behavior. This conclusion is consistent with the analysis in Section 5.6.

5.7.3 Effects of Layers and Hop

We use MovieLens-1M to investigate the influence of parameter *L* and *hop* by varying them from 1 to 5, and we report AUC and ACC to evaluate the CTR prediction performance and report Precision@K {K=2,5,10} to evaluate the performance of Top-K recommendation. The results are shown in Fig. 7. It can be observed that EMKR is insensitive to both the two hyperparameters. *L* and *hop* are suggested to be set to 2 according to the results of Precision@K. Note that, as the number of GCN layers increases, the performance of the approach does not decrease significantly, indicating the over-smoothing phenomenon [51] of GCN does not occur.

5.8 Effects of MTL on Both KGE and RS

We remove KGE module in one experiment and remove RS module in the other experiment to investigate whether MTL can benefit both the KGE task and recommendation task. Specifically, we study the performance of RS with or without KGE module and report AUC and ACC metrics in Table S.2 in Supplementary Material. Moreover, we use root mean square error (RMSE) to evaluate the performance of KGE with or without RS module and the results

are reported in Table S.3 in Supplementary Material. Combining the results of the above two experiments, we can conclude that MTL indeed benefits two tasks.

5.9 Time and Space Performance

In this part, we compare the training time (second) and analyze the space complexity. The average training time is reported in Table S.4 in Supplementary Material. It can be observed that the EMKR has the smallest time cost on 2 of 4 datasets. Especially in the large-scale Dianping-Food dataset, the training time of EMKR is significantly shorter than that of compared approaches, which can illustrate the time superiority of EMKR. Note that, although the time efficiency of EMKR on the other 2 datasets is not as good as that of compared approaches (including MKR), the EMKR can have significantly better performance (including AUC and ACC results) on all the datasets. Therefore, the additional time of EMKR over MKR is greatly deserved.

Moreover, we further analyze and compare the space complexity of different approaches. For example, the space of KGCN is $O((|U|+|\mathcal{E}|+|\mathcal{R}|)\times d+hop\times(d\times(d+1)))$, which contains two parts. The first part is to train d-dimensional vectors for users, entities, and relations, and the second part is to extend the GCN layers. Note that, *hop* in KGCN is usually set to a small and fixed constant, so the space consumption is negligible. Therefore, the space complexity can be simplified to $O((|U|+|\mathcal{E}|+|\mathcal{R}|)\times d)$. Similarly, the complexity of the other approaches is also analyzed and simplified. The analyzed results are reported in Table S.5 in Supplementary Material. Because EMKR adopts the proposed two-part modeling strategy, the space consumption is relatively large. However, EMKR has obvious advantages in recommendation accuracy, stability, and time consumption, the space consumption is greatly deserved.

6 Conclusion and Future Work

EMKR makes full use of the historical interaction data in RS and the deep multi-relation neighborhood data in KG to address the underutilization of data in MKR. Moreover, EMKR uses the proposed two-part modeling strategy to extend the expressive ability of the model. Extensive experiments verify the effectiveness and efficiency of EMKR in multiple recommender scenarios. Also, we find that by using the proposed two-part modeling strategy, EMKR can yield overall significant performance on datasets with different sparsity and different scales, which indicates the effectiveness of EMKR in capturing complex and changeable behavior of users and addressing data sparse challenges.

For future work, existing or new KGE approaches can be naturally introduced in the proposed framework. More efficient and suitable MTL mechanisms are to be explored. Moreover, current popular graph neural networks [52], [53], [54], [55] can be used to redesign the KGE module of our framework. In addition, for enhancing the proposed EMKR and related RSs, automatic and efficient hyperparameter optimization approaches (especially the evolutionary computation approaches [56], [57], [58], [59], [60]) deserve further exploration.

References

[1] 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. Knowledge Discovery and Data Mining*, 2018, pp. 1531–1540.

[2] 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. Knowledge Discovery and Data Mining*, 2016, pp. 353–362.

[3] H. Zhao, Q. Yao, J. Li, Y. Song, and D. L. Lee, "Meta-graph based recommendation fusion over heterogeneous information networks," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, 2017, pp. 635–644.

[4] K. Miyahara and M. J. Pazzani, "Collaborative filtering with the simple bayesian classifier," in *Proc. 6th Pacific Rim Int. Conf. Artificial Intelligence*, 2000, pp. 679–689.

[5] Q. Guo, F. Zhuang, C. Qin, H. Zhu, H. Xiong, and Q. He, "A survey on knowledge graph-based recommender systems," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 8, pp. 3549-3568, 2020.

[6] Z. Sun et al., "Research commentary on recommendations with side information: A survey and research directions," *Electronic Commerce Research and Application*, vol. 37, pp. 100879, 2019.

[7] S. Sen, J. Vig, and J. Riedl, "Tagommenders: Connecting users to items through tags," in *Proc. 18th Int. Conf. World Wide Web,* 2009, pp. 671-680.

[8] Y. Zhen, W. J. Li, and D. Y. Yeung, "Tagicofi: Tag informed collaborative filtering," in *Proc. 3rd ACM Conf. Recommender System*, 2009, pp. 69-76.

[9] L. Zheng, V. Noroozi, and P. S. Yu, "Joint deep modeling of users and items using reviews for recommendation," in *Proc. 10th ACM Int. Conf. Web Search and Data Mining*, 2017, pp. 425–434.

[10] Y. Xu, Y. Yang, J. Han, E. Wang, F. Zhuang, and H. Xiong, "Exploiting the sentimental bias between ratings and reviews for enhancing recommendation," in *Proc. IEEE Int. Conf. on Data Mining (ICDM)*, 2018, pp. 1356–1361.

[11] P. Massa and P. Avesani, "Trust-aware recommender systems," in *Proc. ACM Conf. Recommender Systems*, 2007, pp. 17-24.

[12] M. Jamali and M. Ester, "Trustwalker: A random walk model for combining trust-based and item-based recommendation," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery and Data Mining,* 2009, pp. 397-406.

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

[14] H. Wang et al., "RippleNet: Propagating user preferences on the knowledge graph for recommender systems," in *Proc. 27th ACM Int. Conf. Information and Knowledge Management*, 2018, pp. 417–426.

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

[16] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, "Learning entity and relation embeddings for knowledge graph completion," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp.2181-2187.

[17] H. Wang, F. Zhang, X. Xie, and M. Guo, "Dkn: Deep knowledge-aware network for news recommendation," in *Proc.*

World Wide Web, 2018, pp. 1835-1844.

[18] G. Ji, S. He, L. Xu, K. Liu, and J. Zhao, "Knowledge graph embedding via dynamic mapping matrix," in *Proc. 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing,* 2015, pp. 687–696.

[19] 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*, 2019, pp. 2000-2010.

[20] Z. Wang, G. Lin, H. Tan, Q. Chen, and X. Liu, "CKAN: Collaborative knowledge-aware attentive network for recommender systems," in *Proc. 43rd ACM SIGIR on Research and Development in Information Retrieval*, 2020, pp. 219–228.

[21] Y. Wang, W. Ding, R. Zhang, and H. Li, "Boundary-aware multitask learning for remote sensing imagery," *IEEE Journal of Selected Topics Applied Earth Observations and Remote Sensing*, vol. 14, pp. 951–963, 2021.

[22] Y. Cao, X. Wang, X. He, Z. Hu, and T. S. Chua, "Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences," in *Proc. World Wide Web*, 2019, pp. 151–161.

[23] W. Ma et al., "Jointly learning explainable rules for recommendation with knowledge graph," in *Proc. World Wide Web*, 2019, pp. 1210–1221.

[24] L. Sang, M. Xu, S. Qian, and X. Wu, "Knowledge graph enhanced neural collaborative recommendation," *Expert Syst. Appl.*, vol. 164, pp. 113992, 2021.

[25] C. Chen, M. Zhang, W. Ma, Y. Liu, and S. Ma, "Jointly non-sampling learning for knowledge graph enhanced recommendation," in *Proc. 43rd ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020, pp. 189–198.

[26] Y. Peng, Y. Fang, Z. Xie, and G. Zhou, "Topic-enhanced emotional conversation generation with attention mechanism," *Knowledge Based System*, vol. 163, pp. 429–437, 2019.

[27] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Proc. Neural Information Processing Systems*, 2013, pp. 2787-2795.

[28] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes," in *Proc. AAAI Artif. Intell.*, 2014, pp. 1112-1119.

[29] M. Nickel, V. Tresp, and H. P. Kriegel, "A three-way model for collective learning on multi-relational data," in *Proc. 28th Int. Conf. Machine Learning*, 2011, pp. 3104482-3104584.

[30] M. Nickel, L. Rosasco, and T. Poggio, "Holographic embeddings of knowledge graphs," in *Proc. AAAI Conf. Artif. Intell.*, 2016, pp. 1955-1961.

[31] B. Yang, W. Yih, X. He, J. Gao, and L. Deng, "Embedding entities and relations for learning and inference in knowledge bases," in *Proc. Int. Conf. Learning Representations (ICLR)*, 2015.

[32] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. van den Berg, I. Titov, and M. Welling, "Modeling relational data with graph convolutional networks," in *Proc. European Semantic Web Conf.*, 2018, pp. 593-607.

[33] D. Nathani, J. Chauhan, C. Sharma, and M. Kaul, "Learning attention-based embeddings for relation prediction in knowledge graphs," in *Proc. 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 4710-4723.

[34] Z. Zhang et al., "Towards robust knowledge graph embedding via multi-task reinforcement learning," *IEEE Trans. Knowl. Data Eng.*, 2021, doi: 10.1109/TKDE.2021.3127951.

[35] D. Xu, C. Ruan, E. Korpeoglu, S. Kumar, and K. Achan, "Product knowledge graph embedding for e-commerce," in *Proc. 13th Int. Conf. Web Search and Data Mining*, 2020, pp. 672–680.

[36] I. Misra, A. Shrivastava, A. Gupta, and M. Hebert, "Cross-stitch networks for multi-task learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 3994-4003.

[37] Y. Gao, J. Ma, M. Zhao, W. Liu, and A. L. Yuille, "NDDR-CNN: Layerwise feature fusing in multi-task CNNs by neural discriminative dimensionality reduction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 3205–3214.

[38] A. Søgaard and Y. Goldberg, "Deep multi-task learning with low level tasks supervised at lower layers," in *Proc. 54th Annual Meeting of the Association for Computational Linguistics*, 2016, pp. 231-235.

[39] K. Hashimoto, C. Xiong, Y. Tsuruoka, and R. Socher, "A joint many-task model: Growing a neural network for multiple nlp tasks," in *Proc. Conf. Empirical Methods Natural Language Process (EMNLP)*, 2017.

[40] V. Sanh, T. Wolf, and S. Ruder, "A hierarchical multi-task approach for learning embeddings from semantic tasks," in *Proc. AAAI Conf. Artificial Intelligence,* 2019, pp. 6949–6956.

[41] T. Sun et al., "Learning sparse sharing architectures for multiple tasks," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 8936–8943.

[42] Y. Wang, L. Dong, Y. Li, and H. Zhang, "Multitask feature learning approach for knowledge graph enhanced recommendations with RippleNet," *PLOS ONE*, vol. 16, no. 5, pp. e0251162, 2021.

[43] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Int. Conf. Learning Representations (ICLR)*, 2016, pp. 1-14.

[44] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in Neural Information Processing Systems*, 2017.

[45] X. Yu et al., "Personalized entity recommendation: A heterogeneous information network approach," in *Proc. 7th ACM Int. Conf. Web Search and Data Mining*, 2014, pp. 283–292.

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

[47] H. T. Cheng et al., "Wide & deep learning for recommender systems," in *Proc. 1st Workshop on Deep Learning for Recommender Systems*, 2016, pp. 7–10.

[48] Y. Zhang, M. Yuan, C. Zhao, M. Chen, and X. Liu, "Aggregating knowledge-aware graph neural network and adaptive relational attention for recommendation," *Applied Intelligence*, 2022, pp. 1-13.

[49] X. W. Guo, H. B. Xia, and Y. Liu, "Hybrid recommendation model of knowledge graph and graph convolutional network," *Journal of Frontiers of Computer Science and Technology,* vol. 16, no. 6, pp. 1343, 2022.

[50] Yoon Kim, "Convolutional neural networks for sentence classification," *EMNLP*, 2014.

[51] W. Cong, M. Ramezani, and M. Mahdavi, "On provable benefits of depth in training graph convolutional networks," *Advances in Neural Information Processing Systems*, vol. 34, pp. 9936-9949, 2021.

[52] D. Kreuzer, D. Beaini, W. Hamilton, V. Létourneau, and P. Tossou, "Rethinking graph transformers with spectral attention," in *Advances in Neural Information Processing Systems*, 2021, pp. 21618–21629.

[53] T. He, Y. S. Ong, and L. Bai, "Learning conjoint attentions for graph neural nets," in *Advances in Neural Information Processing Systems*, 2021, pp. 2641-2653.

[54] F. Monti, D. Boscaini, J. Masci, E. Rodola, J. Svoboda, and M. M. Bronstein, "Geometric deep learning on graphs and manifolds using mixture model CNNs," in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5115–5124.

[55] Z. H. Wu et al., "A comprehensive survey on graph neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 1, pp. 4-24, 2020.

[56] Z. H. Zhan, J. Y. Li, S. Kwong and J. Zhang, "Learning-aided evolution for optimization," *IEEE Transactions on Evolutionary Computation*, 2022, doi: 10.1109/TEVC.2022.3232776.

[57] Z. H. Zhan et al., "Matrix-based evolutionary computation," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 6, no. 2, pp. 315-328, 2022.

[58] J. Y. Li, Z. H. Zhan, J. Xu, S. Kwong, and J. Zhang, "Surrogate-assisted hybrid-model estimation of distribution algorithm for mixed-variable hyperparameters optimization in convolutional neural networks," *IEEE Trans. Neural Networks Learn. Syst.*, 2021, doi: 10.1109/TNNLS.2021.3106399.

[59] Y. Q. Wang, J. Y. Li, C. H. Chen, J. Zhang, and Z. H. Zhan, "Scale adaptive fitness evaluation-based particle swarm optimisation for hyperparameter and architecture optimisation in neural networks and deep learning," *CAAI Trans. Intell. Technol.*, 2022, doi: 10.1049/cit2.12106.

[60] Z. H. Zhan, J. Y. Li, and J. Zhang, "Evolutionary deep learning: A survey," *Neurocomputing*, vol. 483, pp. 42-58, 2022.

Min Gao (Student Member, IEEE) received the B.S. degree in mathematical science from the Nanjing Technology University, Nanjing, China, in 2020. She is currently pursuing the M.S. degree in computer science and technology from South China University and Technology, Guangzhou, China.

Her current research interests include evolutionary computation, recommender system, and knowledge graph.

Jian-Yu Li (Student Member, IEEE) received the Bachelor's degree in computer science and technology from the South China University of Technology, Guangzhou, China, in 2018, where he is currently pursuing the Ph.D. degree in computer science and technology with the School of Computer Science and Engineering.

He is currently a visitor with Hanyang University, Anshan, South Korea. His research interests mainly include computational intelligence, data-driven optimization, recommender system, machine learning including deep learning, and their applications in real-world problems, and in environments of distributed computing and big data. He serves as a regular reviewer of the *IEEE Transactions on Evolutionary Computation* and the *Neurocomputing*.

Chun-Hua Chen (Member, IEEE) received the bachelor's degree in Computer Science and Ph.D. degree in Information and Communication Engineering from the South China University of Technology, Guangzhou, China, in 2006 and 2012, respectively.

He is currently an assistant Professor with the School of Software Engineering, South China University of Technology, Guangzhou, China. His current research interests include evolutionary computation, knowledge graph, distributed system, component-based software engineering, and their applications in industrial sector.

Yun Li (Fellow, IEEE) received the B.S. degree from Sichuan University, Chengdu, China, the M.E. degree from University of Electronic Science and Technology of China (UESTC), Chengdu, China, and the Ph.D. degree from University of Strathclyde, Glasgow, U.K.

In 1989, he was a Control Engineer with the U.K. National Engineering Laboratory, Glasgow. In 1990, he was a Postdoctoral Research Engineer with Industrial Systems and Control Ltd, Glasgow. From 1991 to 2018, he was a Lecturer, Senior Lecturer, and Professor with University of Glasgow, Glasgow. Later, he served as the Founding Director of Dongguan Industry 4.0 Artificial Intelligence Laboratory, Dongguan, China, and the Technical Director of i4AI Ltd, London, U.K. He is currently Professor with Shenzhen Institute for Advanced Study, UESTC, Shenzhen, China. Since 1991, his research interest has been computational artificial intelligence and its applications. He has published 270 papers, one of which has been the most popular article every month since its publication in the *IEEE Transactions on Control System Technology*.

Prof. Li is a Chartered Engineer in the U.K. He is an Associate Editor of *IEEE Transactions on Neural Networks and Learning Systems*, and of *IEEE Transactions on Emerging Topics in Computational Intelligence*.

Jun Zhang (Fellow, IEEE) received the Ph.D. degree from the City University of Hong Kong, Kowloon, Hong Kong, in 2002.

He is currently a visiting professor with Hanyang University, Ansan, South Korea. His current research interests include computational intelligence, cloud computing, high performance computing, operations research, and power electronic circuits.

Dr. Zhang was a recipient of the Changjiang Chair Professor from the Ministry of Education, China, in 2013, the China National Funds for Distinguished Young Scientists from the National Natural Science Foundation of China in 2011 and the First-Grade Award in Natural Science Research from the Ministry of Education, China, in 2009. He is currently an Associate Editor of the *IEEE Transactions on Evolutionary Computation* and the *IEEE Transactions on Cybernetics*.

Zhi-Hui Zhan (Senior Member, IEEE) received the Bachelor's degree and the Ph. D. degree in Computer Science from the Sun Yat-Sen University, Guangzhou China, in 2007 and 2013, respectively.

He is currently the Changjiang Scholar Young Professor with the School of Computer Science and Engineering, South China University of Technology, Guangzhou, China. His current research interests include evolutionary computation, swarm intelligence, and their applications in real-world problems and in environments of recommender system, cloud computing, and big data.

Dr. Zhan was a recipient of the IEEE Computational Intelligence Society (CIS) Outstanding Early Career Award in 2021, the Outstanding Youth Science Foundation from National Natural Science Foundations of China (NSFC) in 2018, and the Wu Wen-Jun Artificial Intelligence Excellent Youth from the Chinese Association for Artificial Intelligence in 2017. His doctoral dissertation was awarded the IEEE CIS Outstanding Ph. D. Dissertation and the China Computer Federation Outstanding Ph. D. Dissertation. He is one of the World's Top 2% Scientists for both Career-Long Impact and Year Impact in Artificial Intelligence and one of the Highly Cited Chinese Researchers in Computer Science. He is currently the Chair of Membership Development Committee in IEEE Guangzhou Section and the Vice-Chair of IEEE CIS Guangzhou Chapter. He is currently an Associate Editor of the *IEEE Transactions on Evolutionary Computation*, the *Neurocomputing*, the *Memetic Computing*, and the *Machine Intelligence Research*.