Applied Intelligence (2023) 53:1068–1083
https://doi.org/10.1007/s10489-022-03521-4

RFAN: Relation-fused multi-head attention network for knowledge graph enhanced recommendation

Huajuan Duan[1] · **Peiyu Liu**[1] · **Qi Ding**[1]

Accepted: 17 March 2022 / Published online: 25 April 2022

Abstract
Knowledge graphs (KGs) provide rich external structures and semantic information for recommendation systems and have attracted extensive attention recently. Some existing KG-based recommendation methods process triplets independently through knowledge graph embedding, and they fail to fully capture the complex and implicit relation information in the neighbourhood around a given entity. To solve this problem, we borrow Graph Attention Network (GAT) and propose a relation-fused multi-head attention network (RFAN) that integrates relations into an attention network in a KG to learn the representation of entities through different contributions of neighbours and then combine them with user-item interactions to capture user preferences. In addition, we use a non-sampling strategy to learn parameters from the whole training data to improve training efficiency. We apply the proposed model on three public benchmarks, and the experimental results show that the RFAN outperforms several state-of-the-art methods.

Keywords Knowledge graph · Recommendation systems · Multi-head attention network · Entity embedding

1 Introduction

According to the needs and interests of users, the recommendation system [1–3] mines the items that users are interested in from massive data. From various search engines, social platforms to news websites, recommendation systems have become an essential module in all applications that provide services for users. Most popular recommendation algorithms model user preferences and item characteristics respectively, and predict interaction scores through supervised learning models such as Neural FM(NFM) [4] and xDeepFM [5]. As one of the most classical algorithms in recommendation system, collaborative filtering (CF) [6–8] finds user preferences based on mining user historical behaviors. However, due to the lack of side information [9] such as user profiles, item attributes and context information, it performs poorly in dealing with data sparsity and cold start problems.

One solution is to utilize the graph of item side information, aka knowledge graph(KG) to supplement the item characteristics and construct recommendation models. Knowledge graphs [10–13] can provide relevant facts for items because of their rich semantic information and relational structure so that they have attracted extensive attention in the recommendation systems field. KG consists of countless triplets, in which nodes represent entities and edges correspond to relations. Each triplet is an objective fact that exists in reality. In the knowledge graph used in this paper, the item and some item attributes are considered entities, and there are various relation types between entities. By combining the user-item historical interaction and item knowledge graph to expand user interest, we provide users with a reasonable recommendation list and improve the accuracy and explainability of the recommendation system. As shown in Fig. 1, the user watched the movie *The Day After Tomorrow* which is a disaster film and the director is *Roland Emmerich*. The system recommends *Independence Day* to the user through a knowledge connection in the KG. However, it is not clear whether the system recommends it to the user according to genre or director. Therefore, it can be seen that user interests

✉ Peiyu Liu
liupy@sdnu.com.cn

Huajuan Duan
duanhuaj@163.com

Qi Ding
2632548757@qq.com

[1] School of Information Science and Engineering, Shandong Normal University, Jinan, 250358, Shandong, China

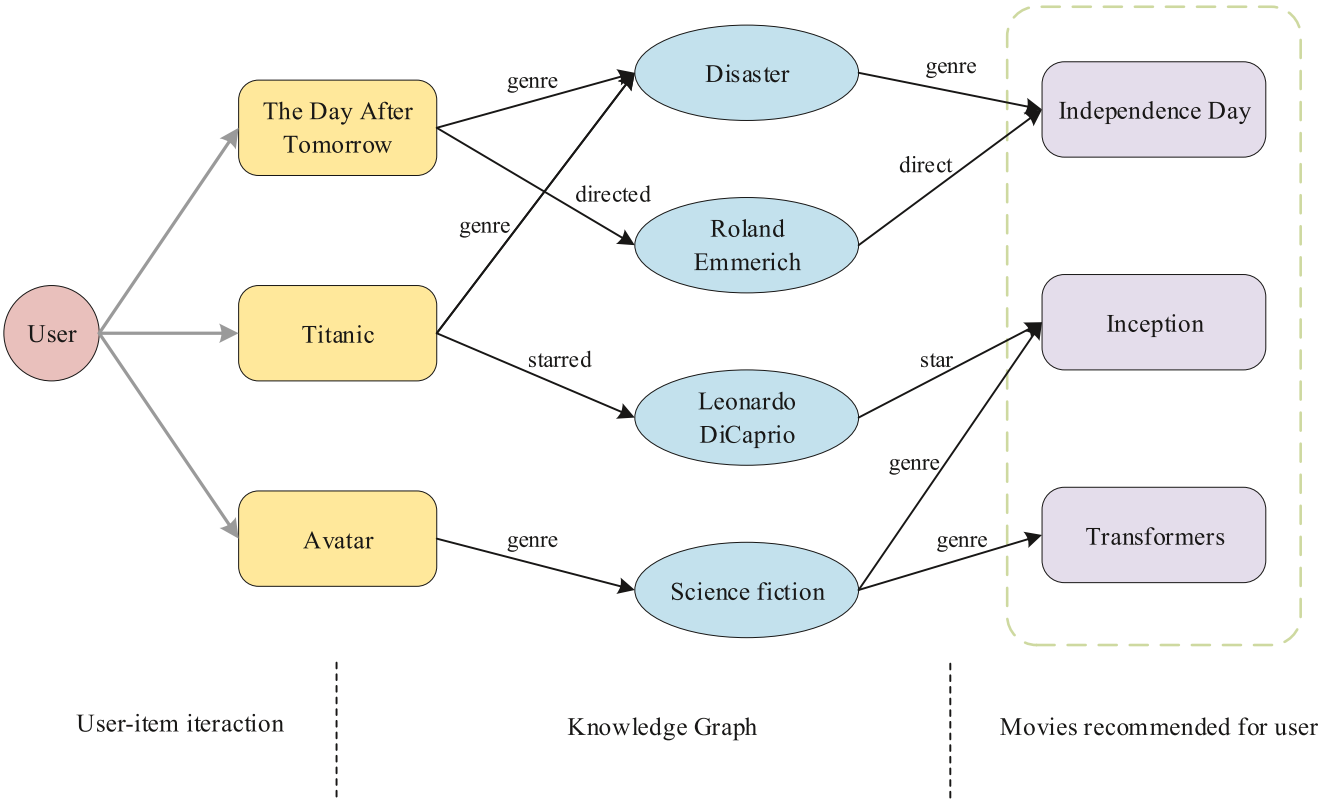


Fig. 1 Illustration of knowledge graph enhanced movie recommendation systems

are often diverse, so it is significant to model the relations in the KG. Some popular models such as RippleNet [14], KGCN [15], KGAT [16], CKAN [17] etc., consider the relations and have achieved good results.

Graph attention network(GAT) [18] has been proven to be able to focus on the most important part of the neighborhood. For a node, it can adaptively match the weights of different neighbors and aggregate the characteristic information of neighbors. Because GAT is a nodewise operation, it also has powerful functions in dealing with directed graphs. However, GAT is not suitable for the neighborhood aggregation of entities in KGs, because the research goal of GAT focuses on nodes and GAT ignores the relation characteristics, which is a very important part of the KG [19]. Nathani et al. improved GAT for relation prediction in knowledge graphs. We also modify GAT for the knowledge graph recommendation task to model the relations and entities in KGs.

In this paper, based on the idea of GAT, we present a Relation-Fused multi-head Attention Network (RFAN) for entity embedding applicable to knowledge graphs aimed at recommendation systems.RFAN combines the relation information to assign attention weight to the nodes in the neighborhood, which encapsulates the multi-hop relations while preserving the graph structure, as well as updates the node information through multi-layer propagation. Since the contribution of distant entities is very small and noise will be introduced, we will specify the appropriate number of layers by experiments. In the prediction stage, we specify the appropriate number of layers by experiments. In the prediction stage, we use an efficient non-sampling strategy to train the whole dataset; that is, all unobserved data are trained as negative samples, which improves the model performance. The experimental results show that RFAN achieves AUC gains of 0.2% to 16.9%, 2.3% to 16.8%,4.4% to 28.2% on MovieLens-20M, Book-Crossing and Last.FM, respectively.

The contributions of our work are summarized as follows:

1. We propose a relation-fused multi-head attention network RFAN for KG enhanced recommendation, which is an end-to-end recommendation framework.
2. We highlight the importance of relation modeling between entities in the knowledge graph, and expand user preferences through a variety of relations around an entity. RFAN extends multi-head attention mechanism to enhance user and item representations to recommend for users, which alleviates the cold start problem to a certain extent.

3. In the experimental part, we use the efficient non-sampling strategy to train the whole dataset. Extensive experiments on the datasets of three public benchmarks prove the effectiveness of the RFAN in knowledge-enhanced recommendation.

2 Related works

Recommendation systems based on knowledge graphs introduce KGs as auxiliary information into recommendation systems. At present, the mainstream research methods can be divided into two categories: path-based methods [20–26] and embedding-based methods [27–33].

The basic idea of path-based methods is to use the connectivity of entities in KG to find a variety of connection paths between users and items. Among them, some models calculate the semantic similarity between entities under different meta-paths to optimize the representation of users and items, such as HIN2Vec [20] and HERec [21]. Others learn the explicit path embedding between user and item, such as MCRec [22], RKGE [23] and KPRN [24]. The path-based methods provide explainability for the recommendation system, they need to manually extract and construct a large number of meta-paths from the data, and if the recommendation scene or knowledge graph changes, it needs to reconstruct the meta path. In addition, the long path will contain some noise because of the introduction of too much entity information, so it is very difficult to define many effective meta paths. Therefore, the RFAN avoids path construction and focuses on entity embedding.

Embedding-based methods usually project entities and relations into low-dimensional space by KGE algorithms. KSR [27] uses TransE [28] to learn sequential recommendations, KTUP [29] jointly learns both tasks of recommendation and KG completion via TransH [30], CKE [12] encodes the structured knowledge of items via TransR [31], and DKN [32] combines KG to obtain news embedding representation via TransD [33]. Although these embedding-based methods can obtain the specific representations of entities and relations in a KG, they ignore the importance of integrating relations into user-item interactions and find it difficult to fully capture the complex semantics of user-item connections. RFAN considers the relation between entities in KG, extracts the relation characteristics of multi-hop neighborhoods, and provides a rich knowledge base for user-item interaction.

Some recent methods consider the relations in KG and carry out multi-layer information propagation on KG to enhance recommendation. RippleNet [14] automatically discovers users' potential hierarchical interests via preference propagation in KG. KGCN [15] calculates the importance between users and relations and captures the local proximity structure through aggregating neighbor information. Similarly, KGAT [16] uses the relation-aware attention mechanism to learn the weight of each neighbor in the propagation process and models the high-order relation of user-item by a collaborative knowledge graph. To alleviate the cold start problem, CKAN [17] propagates the cooperation signal through user-item interaction and knowledge edge association in KGs. However, the high-order interaction of user-items may be completely different from the latent semantic representation of the original item, which will introduce noise to the embedding. These methods use a single-head attention mechanism for neighborhood aggregation. In contrast, RFAN separately uses user-item interaction graph and knowledge graph to design a relation-fused attention mechanism to encapsulate multi-hop relation semantics and enrich user preferences for items. Furthermore, the RFAN extends the multi-head attention mechanism to prevent overfitting and stabilize the learning results.

Existing works often use negative sampling strategy for training. Although negative sampling is easy to implement, many recent studies show that the robustness of a negative sampling strategy is poor and important training samples may be ignored, resulting in the model failing to converge to the optimal value. Chen et al. [34] proposed a nonsampling efficient neural matrix factorization ENMF. In the basic recommendation scenario, ENMF only uses the ID information of users and items, whose performance and training speed exceed many advanced algorithms. In addition, Chen et al. applied an efficient non-sampling strategy in multiple scenarios such as social recommendation [35], multi-task recommendation [36] and knowledge graph recommendation [37], and achieved valuable results. Therefore, we utilize the non-sampling strategy in the RFAN to train the whole data to improve the model robostness.

3 Relation-fused multi-head attention network based on knowledge graph

We now describe our RFAN model for knowledge graph recommendation, and the overall framework of the model is shown in Fig. 2. In this section, we first give the notations definition and overall framework, and then introduce RFAN in three parts: 1) knowledge graph embedding, which maps entities and relations in KG into vectors; 2) relation-fused multi-head attention network, which iteratively propagates multi-layer information to update the entity representation; 3) model prediction and optimization, which obtains user and item representations from the propagation layer and outputs prediction scores.

3.1 Notations definition and framework

In a recommendation scenario, we have a user set $\mathcal{U}$ and an item set $\mathcal{V}$. According to users implicit feedback such as watching, collecting and purchasing, we can define a user-item interaction matrix $Y = [y_{uv}] \in \{0, 1\}$, where $y_{uv} = 1$ indicates that user u interacts with item v, otherwise $y_{uv} = 0$. However, $y_{uv} = 0$ does not mean that user u does not like item v; perhaps user u ignores it by accident. In addition, we also have knowledge graph that can provide auxiliary information for items, which is defined as $\mathcal{G} = \{(h, r, t) \mid h, t \in \mathcal{E}, r \in \mathcal{R}\}$, in which each triplet (h, r, t) describes that there is a relation r between the head entity h and the tail entity t, $\mathcal{E}$ and R represent the set of entities and relations respectively, and entities contain items and their certain properties. Since we set that all items can find corresponding entities in the knowledge graph, we use an alignment set $\mathcal{A} = \{(v, e) \mid v \in \mathcal{V}, e \in \mathcal{E}\}$ to represent the alignment between items and entities. Given user-item interaction matrix Y and knowledge graph G, our goal is to learn a prediction function $\hat{y}_{uv} = \mathcal{F}(u, v \mid \Theta)$ to predict whether user u has a potential preference for an item v, where $\hat{y}_{uv}$ indicates the probability score that user u will interact with item v, and Θ represents the parameters of RFAN.

3.2 Knowledge graph embedding

Knowledge graph embedding can project entities and relations to a low-dimensional vector space while retaining the original structure of KG. We apply TransR proposed by [31], which is a method that has been widely used in KG embedding. TransR assumes that there are multiple relation spaces, and the head entity and tail entity have different representations in different relation spaces. For a triplet (h, r, t) that exists in the KG, TransR learns the embeddings of each entity and relation by optimizing the translation condition $W_r e_h + e_r \approx W_r e_t$, where $e_h, e_t \in \mathbb{R}^d$ and $e_r \in \mathbb{R}^{d_1}$ represent the embeddings of h, r and t, respectively; $W_r \in \mathbb{R}^{d \times d_1}$ is the transformation matrix of relation r,which maps entities from the d-dimension entity space into the d_1-dimension relation space. Our goal is to obtain more accurate triplets, so for a given triplet (h, r, t), its score function is expressed as follows:

$$f(h, r, t) = \| W_r e_h + e_r - W_r e_t \|_2^2 \tag{1}$$

The lower the score of $f(h, r, t)$, the more likely it is that the triplet is fact and vice versa. TransR utilizes pairwise ranking loss to train the triplets, which is given by (2):

$$\mathcal{L}_{KG} = \sum_{(h,r,t)\in\mathcal{T}} \sum_{(h',r,t')\in\mathcal{T}'} -ln\sigma\left(f\left(h', r, t'\right) - f\left(h, r, t\right)\right) \tag{2}$$

where $\mathcal{T}$ is the set of valid triplets and $\mathcal{T}'$ denotes the set of broken triplets which is defined as:

$$\mathcal{T}' = \left\{\left(h', r, t\right) \cup \left(h, r, t'\right) \mid \left(h', r, t\right) \notin \mathcal{G},\ \left(h, r, t'\right) \notin \mathcal{G}\right\} \tag{3}$$

where h' and t' are entities other than h and t, respectively. Note that h and t cannot be replaced at the same time. $\sigma(\cdot)$ is the sigmoid function. Training the embedding layer considers the relative ranking of valid triples and broken triples, which improves the computational efficiency of the model. Different from other methods that only use the entity embedding, we also combine the entity type information to enhance the entity representation. For an entity h, we first use an embedding look-up layer to project h (e.g., *Roland Emmerich* or *Titanic*) and its corresponding type (e.g., person or movie) into low-dimensional vectors $e_h \in \mathbb{R}^{d_1}, e'_h \in \mathbb{R}^{d_t}$, where d_1 and d_t are the respective embedding sizes. Therefore, the type-enhanced entity embedding of entity h consists of two parts, which is calculated as:

$$x'_h = \sigma\left(W_a\left(e_h \oplus e'_h\right) + b_1\right) \tag{4}$$

W_a and b_1 are the transformation matrix and bias, respectively; $\oplus$ is the concatenation operation. As described above, we initialize e_h with the pretrained embeddings via TransR.

3.3 Relation-fused attention embedding propagation

Although GAT has achieved success in many fields, it ignores the relation between nodes and is not suitable for knowledge graph modeling. A head entity h in the knowledge graph may connect multiple tail entities through various relations, that is, h may own multiple neighbor nodes. For example, as revealed in Fig. 1, (*Titanic*,*genre*, *Disaster*) and (*Titanic*, *starred*, *Leonardo DiCaprio*) are two triplets that have the same head entity *Titanic* and connect different tail entities through different relations. It is difficult for the original GAT to model triplets using relations. Hence, we borrow the idea of GAT to propose a relation-fused attention entity embedding method RFAN, which not only integrates neighbor entity characteristics into the attention mechanism, but also considers the effect of different relations on user preferences. As shown in Fig. 2, we draw the embedding propagation process of entities and relations in the relation-fused attention propagation layer, which is described in detail below.

We take the entity matrix $G'_e = \{x'_1, x'_2, \ldots, x'_h, \ldots, x'_M\}, x'_h \in \mathbb{R}^{F'_e}$and relation matrix $G'_r \in \mathbb{R}^{K \times F'_r}$ as the input layer, where M, K denote the number of entities and relations separately; F'_e and F'_r are feature dimensions

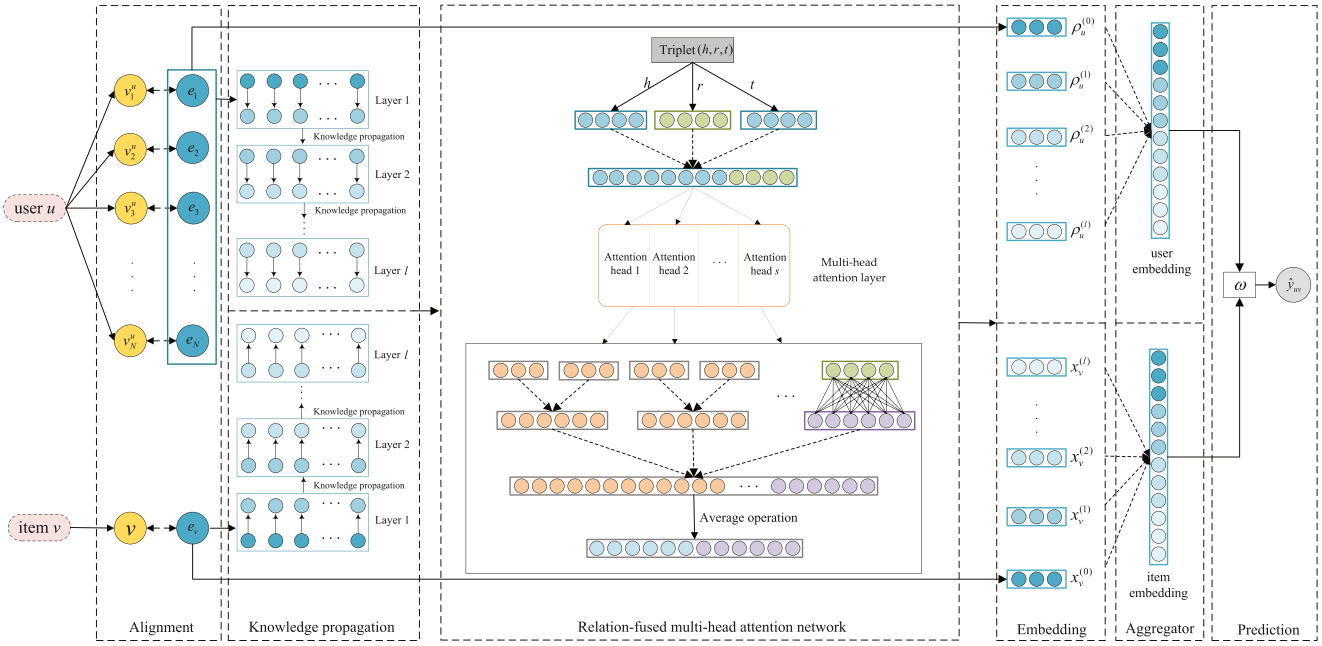


Fig. 2 Overall framework of the proposed RFAN model. Dashed arrows indicate the concatenation operation.The figure shows a relation-fused multi-head attention network, which acts on each knowledge propagation process

of each entity and relation embedding. We learn the hidden state of entity h through all triplets connected with h. The triplets are calculated by linear transformation over the concatenation of entity and relation features, given by:

$$\varphi_{hrt} = W_b \left[x'_h \parallel e_r \parallel x'_t \right] \tag{5}$$

where φ_{hrt} is the representation of a triplet (h, r, t), x'_h, x'_t and e_r are embeddings of the head entity, tail entity and relation, respectively.The calculation of x'_t is consistent with x'_h, which has been described in Section 3.2. W_b is a linear transformation matrix that acts on each node. Then, the RFAN learns the importance of each related triplet to entity h expressed as β_{hrt}:

$$\beta_{hrt} = LeakyReLU(W_c \varphi_{hrt}) \tag{6}$$

W_c is a weight matrix to carry out linear transformation to φ_{hrt}, and then we perform nonlinearity by the $LeakyReLU$ activation function. Similar to GAT, We apply $softmax$ to normalize the neighbor nodes of entity h to obtain the relative attention value of each triplet, as shown in (7). Figure 3 shows the calculation process of the relative attention value π_{hrt} of a triplet (h, r, t).

$$\pi_{hrt} = \frac{\exp(\beta_{hrt})}{\sum_{n \in N_h} \sum_{k \in \mathcal{R}_{hn}} \exp(\beta_{hkn})} \tag{7}$$

where N_h is the set of neighbor entities directly connected to h, and $\mathcal{R}_{hn}$ denotes the set of relations between h and N_h. The more important the neighbor tail entity is, the greater the attention weight of the triplet to highlight the impact of knowledge association on user preference modeling. The updated embedding of entity h is the aggregation of the weighted attention values corresponding to each adjacent triplet, given by the following formula:

$$\hat{x}_h = \sigma \left(\sum_{t \in N_h} \sum_{r \in \mathcal{R}_{ht}} \pi_{hrt} \varphi_{hrt} \right) \tag{8}$$

To stabilize the learning process and improve the generalization ability of the attention mechanism, we

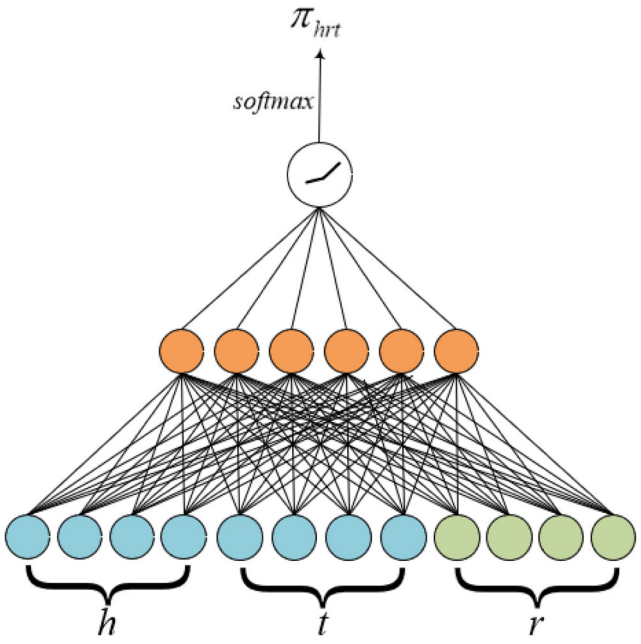


Fig. 3 Relation-fused attention network

also use multi-head attention like GAT. Specifically, S independent attention mechanisms work together to update the embedding of entity h. To maintain the dimension of feature vector, we adopt an average operation for multiple heads to calculate the embedding representation of entity h output by the attention layer, which is shown as (9):

$$x_h = \sigma\left(\frac{1}{S}\sum_{s=1}^{S}\sum_{t\in N_h}\sum_{r\in\mathcal{R}_{ht}}\pi_{hrt}(s)\varphi_{hrt}(s)\right) \tag{9}$$

The relation-fused attention layer finally outputs a new relation embedding matrix $G_r \in \mathbb{R}^{K\times F_r}$. Due to the change in dimension, we perform a linear transformation matrix $W_R \in \mathbb{R}^{F_r'\times F_r}$ on the initial relation matrix G_r', where F_r denotes the dimension of G_r:

$$G_r = W_R * G_r' \tag{10}$$

The relation-fused attention layer outputs an embedding representation of entities, denoted as:

$$G_e = \{x_1, x_2, \ldots, x_h, \ldots, x_M\}, x_h \in \mathbb{R}^{F_e} \tag{11}$$

where F_e is the dimension of entity representations produced by the relation-fused multi-head attention layer.

The representation of an entity depends on its ego-network. Using only the first-order neighbors of the entity may lose important information. Therefore, we expand multiple knowledge propagation layers to further explore the impact of high-order neighbors and the potential interests of users. The technology is specifically described as follows. In the first knowledge propagation layer, all entities obtain information from the first-order neighbors directly connected to them, and then we repeat this process. The second layer also propagates and aggregates the first-order neighbor entities. For instance, as illustrated in Fig. 1, *Titanic* gathers information from *Disaster* and *Leonardo DiCaprio* in the first layer, which are the 1-order direct neighbors connecting *Titanic*. In the second layer, *Titanic* obtains information from *Independence Day* and *Inception* which are the second-order neighbors of *Titanic*. Due to the first-order neighbor entity already stores the representations of second-order ones from the previous layer, the entity can capture the information of the second-order neighbors, and so on, in the l-th knowledge propagation layer, the entity can obtain the information from l-order neighbors.

3.4 Model prediction

Owing to the alignment between items and entities, each item must find its corresponding entity in the knowledge graph. For the target user u, the related item set namely $\mathcal{V}^u = \left\{v_1^u, v_2^u, \ldots, v_N^u\right\}$ directly interacting with the user can be obtained according to the historical interaction behavior, which can be transformed into the initial entity seed set of user u in the knowledge graph via the alignment set of items and entities, denoted as:

$$\mathcal{E}_u^{(0)} = \left\{e_j \mid (v_j^u, e_j) \in \mathcal{A},\ j = 1, ..., N\right\} \tag{12}$$

According to the relation-fused attention embedding method proposed by this paper, we can obtain multiple embedding representations for user u after executing l knowledge propagation layers:

$$\mathcal{P}_u = \left\{\rho_u^{(0)}, \rho_u^{(1)}, \ldots, \rho_u^{(l)}\right\} \tag{13}$$

where $\rho_u^{(0)} = \frac{\sum_{e_j\in\mathcal{E}_u^{(0)}} e_j}{|\mathcal{E}_u^{(0)}|}$ is the representation of the initial entity set that directly contacts user u.For each knowledge propagation layer, a representation of the user node will be obtained. $\left\{\rho_u^{(1)}, \ldots, \rho_u^{(l)}\right\}$ are the outputs from the first knowledge propagation layer to the last layer. The user representation is completely calculated by user-item interaction and knowledge graph propagation, which alleviates the cold start problem in the recommendation system.

For a target item v, the item has its initial embedding representation shown as e_v, which is converted to $x_v^{(0)}$. Then we can construct a l-layer knowledge graph auxiliary representation set of items, which explicitly encodes the multi-hop neighborhood of item v in the form of attention weight. Similar to the user representation set, multiple item representations can be acquired:

$$\mathcal{Q}_v = \left\{x_v^{(0)}, x_v^{(1)}, \ldots, x_v^{(l)}\right\} \tag{14}$$

The output representations of different layers can be interpreted as the high-order impact of different depths on users and items. In order to retain the hidden embedding information to a greater extent, we aggregate the representations of all layers into a single vector by a concatenate operation.

$$\begin{aligned} p_u &= \sigma\left(W_d\left(\rho_u^{(0)} \parallel \rho_u^{(1)} \parallel \ldots \parallel \rho_u^{(l)}\right) + b_2\right) \\ q_v &= \sigma\left(W_d\left(x_v^{(0)} \parallel x_v^{(1)} \parallel \ldots \parallel x_v^{(l)}\right) + b_2\right) \end{aligned} \tag{15}$$

where $p_u \in \mathbb{R}^{d_2}$ and $q_v \in \mathbb{R}^{d_2}$ are final embedding representations of the user and item, respectively. W_d and b_2 are the trainable weight and bias. The nonlinear activation function is set as Sigmoid. We discuss the effects of different aggregators in Section 4.4.3.

We utilize the neural form of MF as the prediction function to predict the interaction score of target users and items:

$$\hat{y}_{uv} = \omega^\top (p_u \odot q_v) \tag{16}$$

where ω is the predicted vector;$\odot$ denotes the element-wise product operation.

3.5 Optimization and training

In order to improve the model robustness, we adopt an efficient non-sampling strategy to optimize the model, that is, all unmarked data are taken as negative samples during training. Specifically, for implicit data, the non-sampling loss minimizes the disparity between user real feedback y_{uv} and predicted value $\hat{y}_{uv}$:

$$\mathcal{L}_{RS} = \sum_{u \in U} \sum_{v \in V} c_{uv} \left(y_{uv} - \hat{y}_{uv} \right)^2 \tag{17}$$

where c_{uv} is the weight of y_{uv}. For a batch of items $\mathcal{B}$, we expand and transform (17) to obtain the non-sampling loss function, which is expressed as follows:

$$\begin{aligned} \mathcal{L}_{RS} = & \sum_{u \in \mathcal{U}^+} \sum_{v \in \mathcal{B}} \left(\left(c_v^+ - c_v^- \right) \hat{y}_{uv}^2 - 2 c_v^+ \hat{y}_{uv} \right) \\ & + \sum_{i=1}^{d} \sum_{j=1}^{d} \left(\left(\omega_i \omega_j \right) \left(\sum_{u \in \mathcal{U}} p_{u,i} p_{u,j} \right) \left(\sum_{v \in \mathcal{B}} q_{u,i} q_{u,j} \right) \right) \end{aligned} \tag{18}$$

Finally, we combine the knowledge graph embedding loss function $\mathcal{L}_{KG}$ with the recommended loss function $\mathcal{L}_{RS}$ as the objective function:

$$\mathcal{L}_{RFAN} = \mathcal{L}_{KG} + \mathcal{L}_{RS} + \lambda \left\| \Theta \right\|_2^2 \tag{19}$$

where $\Theta = \left\{ E, R, W_i, b_j, \forall i \in \{r, a, b, c, d\}, j \in \{1, 2\} \right\}$ is the set of trainable model parameters. E and R are the embedding tables for all entities and relations. L_2 regularization parameterized by λ on Θ is conducted to prevent overfitting.

3.6 Time complexity analysis

We further discuss the time complexity of our model in this section. The time cost of RFAN mainly comes from three parts. For the knowledge graph embedding, the embedding principle has time complexity $O(|\mathcal{G}|d^2)$. For the attention embedding propagation scheme, updating entity representations has computational complexity $O(l|\mathcal{G}|d)$. For the efficient non-sampling strategy, one epoch takes $O((|\mathcal{V}| + \frac{|\mathcal{U}||\mathcal{V}|}{|\mathcal{B}|})d^2 + |\mathcal{Y}|d)$ time, where $\mathcal{Y}$ denotes positive user-item interactions. Consequently, the overall time complexity of the RFAN is $O(|\mathcal{G}|d^2 + l|\mathcal{G}|d + (|\mathcal{V}| + \frac{|\mathcal{U}||\mathcal{V}|}{|\mathcal{B}|})d^2 + |\mathcal{Y}|d)$.

4 Experiments

In this section, we evaluate the performance of RFAN based on three real datasets and analyse its effectiveness and accuracy by comparing it with the existing baselines.

Table 1 Statistics of the datasets

Dataset		MovieLens-20M	Book-Crossing	Last.FM
User-item Interaction	#Users	138,159	17,860	1,872
	#Items	16,954	14,967	3,846
	#Interactions	13,501,622	139,746	42,346
Knowledge Graph	#Entities	102,569	77,903	9366
	#Relations	32	25	60
	#Triplets	499,474	151,500	15,518
Hyperparameter Settings	d	64	128	256
	l	2	2	3
	η	2×10^{-3}	5×10^{-3}	10^{-2}
	λ	10^{-7}	10^{-5}	10^{-4}

4.1 Datasets

We expand experiments on three real datasets for movie, book and music recommendations:(1) MovieLens-20M is a stable benchmark dataset which contains approximately 20 million explicit ratings ranging from 1 to 5. (2) Book-Crossing consists of 1 million ratings of different books by diverse readers, ranging from 0 to 10, which is one of the sparsest datasets. (3) Last.FM provides a dataset of music recommendations, including social networks, tags and music artist listening information of 1,872 users.

These datasets were processed and used in [15, 17, 38, 39], and a corresponding knowledge graph was constructed for each dataset using Microsoft Satori. To facilitate performance comparison, we directly employ the processed datasets for experiments. We give the specific information of the three datasets and their corresponding knowledge graphs, including the number of users, items and interactions, as well as the number of entities, relations and triplets in the knowledge graph, as shown in Table 1.

4.2 Experimental settings

4.2.1 Baselines

For the sake of proving the effectiveness of the RFAN model, we compared it with the following knowledge graph recommendation baselines, and described the hyperparametric settings for baselines.

(1) CKE [12] combines collaborative filtering with item structural, textual and visual information into a unified framework, which is a representative recommendation

system for knowledge graph feature learning. In this paper, the dimension and learning rate of the three datasets are set as follows: $d = 64$, $\eta = 0.5$ for MovieLens-20M; $d = 128$, $\eta = 0.5$ for Book-Crossing; $d = 64$, $\eta = 0.1$ for Last.FM.

(2) DKN [32] regards word embedding, entity embedding and entity context embedding as multi-channels input to the CNN for news recommendation. Here we use movie/book/music names as the textual input for DKN to implement it. The dimension of word embedding and entity embedding is 64. The number of filters is 128 for each window size of 1,2 and 3.

(3) LibFM [40] is a generic feature-based recommendation method. We take user ID and item ID as the input of libFM, in which the knowledge graph can be weakened into item features for TransE embedding. The dimension of TransE is 32.

(4) RippleNet [14] is an advanced knowledge propagation-based algorithm that naturally integrates the KGE method into the recommendation system through preference propagation, and automatically finds the potential hierarchical interests of users. The important hyperparameters of RippleNet are as follows: $d = 8, l = 2, \eta = 10^{-2}, \lambda_1 = 10^{-6}, \lambda_2 = 10^{-2}$ for MovieLens-20M; $d = 4$, $l = 3$, $\eta = 10^{-3}$, $\lambda_1 = 10^{-5}$, $\lambda_2 = 10^{-2}$ for Book-Crossing; $d = 8$, $l = 2$, $\eta = 2 \times 10^{-2}$, $\lambda_1 = 10^{-6}$, $\lambda_2 = 10^{-2}$ for Last.FM.

(5) KGCN [15] is a state-of-the-art propagation-based model that expands the receptive field of each entity in the knowledge graph, aggregates the neighborhood information of a given entity to capture users' high-order personalized interests and models the potential relations between items. We set $d = 64$, $l = 1$, $\eta = 10^{-4}$ and $\lambda = 2 \times 10^{-5}$ for Book-Crossing.

4.2.2 Parameters setup

We apply a grid search for hyperparameter settings. The learning rate η is adjusted among $\{10^{-3}, 2 \times 10^{-3}, 5 \times 10^{-3}, 10^{-2}, 2 \times 10^{-2}, 5 \times 10^{-2}\}$, the coefficient of L2 normalization λ is searched in $\{10^{-7}, 10^{-6}, \ldots, 10^{-1}\}$, and the entity embedding size d is tested in {8, 16, 32, 64, 128, 256, 512}. Moreover, the embedding sizes of relation and entity type are set as 32. For multi-head attention network, the hidden layer size of the attention mechanism is set as 128 and we set 8 heads for each hidden layer. The hyperparameters of the RFAN are given in Table 1, including the embedding size d, the number of layers l, the learning rate η and the coefficient of L2 normalization λ. The remaining hyperparameters of baselines are the same as their original paper.

4.2.3 Evaluation metrics

In each dataset, we randomly extract 80% of the user interaction data as the training set and see the remaining 20% as the test set. We select 20% of the training set data to construct the validation set for hyperparameter adjustment. To verify the feasibility and accuracy of the RFAN, we conduct experiments on CTR prediction and top-K recommendation scenarios respectively. CTR prediction is to predict whether the user interacts with the target item and output the predicted click probability. AUC and ACC are utilized to evaluate the prediction results in CTR scenarios. In top-K recommendation, we apply the trained model to match the K items with the highest click probability for each user, and use $Precision@K$, $Recall@K$, $F1@K$ as evaluation metrics for top-K recommendation(K=1,2,5,10,20,50,100,200). We take the average of the five experimental results as the final result to ensure the fairness of the experiment.

4.3 Performance comparison with baselines

The experimental results of all models in CTR prediction and top-K recommendation are shown in Table 2 and Figs. 4, 5 and 6, respectively. The following conclusions can be obtained by observing the experimental results.

RFAN shows better performance than baselines on three datasets. In the CTR prediction scenario, RFAN has a significant improvement in Book-Crossing and Last.FM, in which AUC surpasses the second place by 2.3% and 4.4% respectively, and the ACC raises by 3.9% and 1.8%. However, the AUC of the RFAN only increases lightly on MovieLens-20M, and even ACC decreases by 0.2% compared with KGCN. It can be seen that the number of users interacting with items in MovieLens-20M is so large that the effect of knowledge propagation is not obvious. In top-K recommendation, RFAN shows the best.

Performance with respect to $Precision$, $Recall$ and $F1$ value, and RFAN still performs well with the increase in K. We calculate the average $Precision$, $Recall$ and $F1$ over all K values as shown in Figs. 4(d), 5(d) and 6(d). The RFAN outperforms baselines by a significant margin. In general, the average $Precision$, $Recall$ and $F1$ of RFAN over all K values increased by -0.5%, 6.4%, and 4.6% respectively compared to the second place on MovieLens-20M, 5.3%, 7.8%, 10.8% on Book-Crossing, 4.2%, 13.6%, 3.6% on Last.FM.

RFAN performs well on the two relatively sparse datasets, Book-Crossing and Last.FM, which proves that the RFAN can handle sparse data well. The results contrasted with RippleNet prove importance and effectiveness of modeling relations into a knowledge attention mechanism. The reason why the performance of RFAN is better

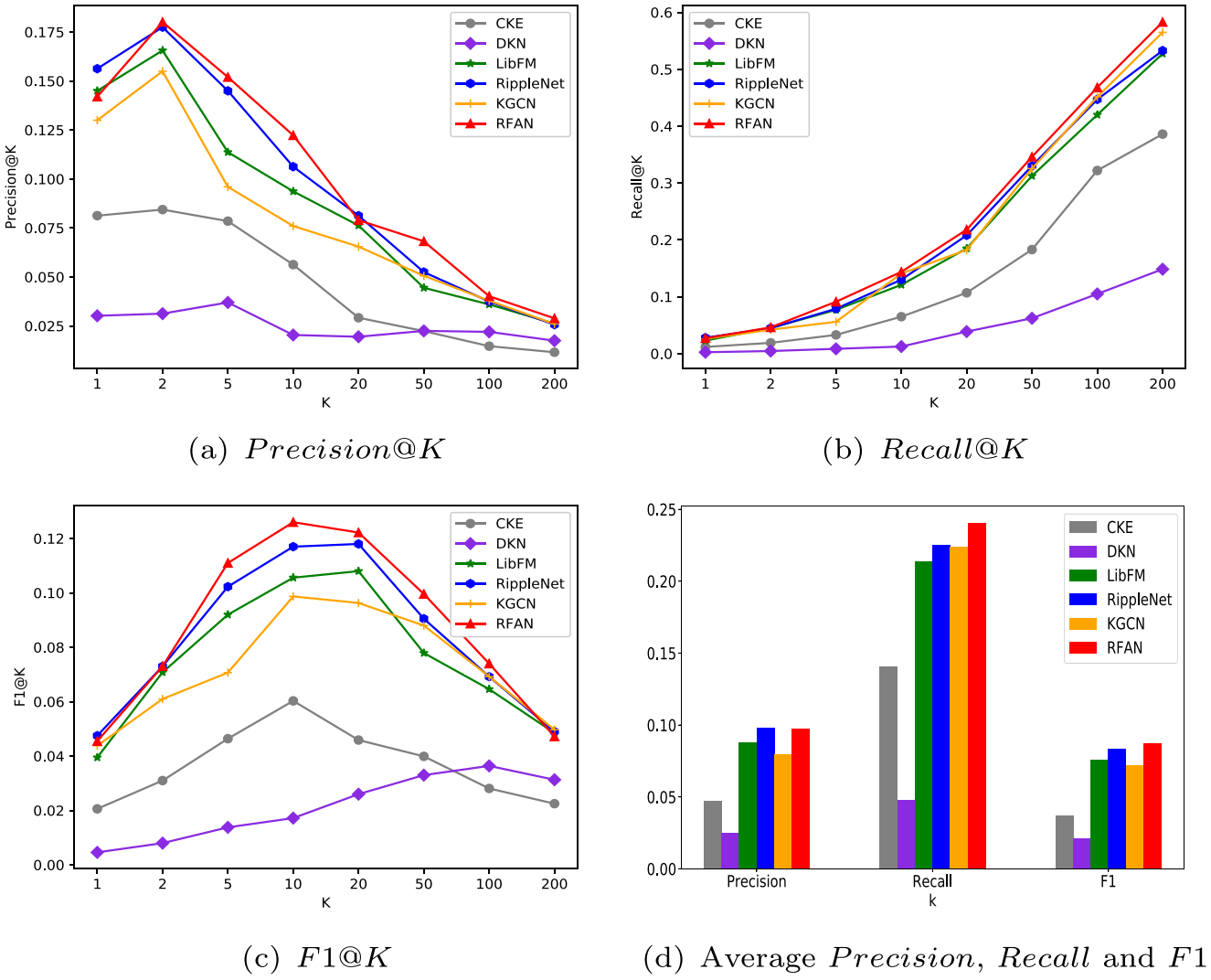
Table 2 The results of AUC and ACC in CTR prediction

Model	MovieLens-20M		Book-Crossing		Last.FM	
	AUC	ACC	AUC	ACC	AUC	ACC
CKE	0.924(-5.7%)	0.880(-5.3%)	0.674(-13.2%)	0.635(-7.8%)	0.744(-11.3%)	0.673(-8.1%)
DKN	0.814(-16.9%)	0.733(-21.2%)	0.621(-16.8%)	0.598(-13.2%)	0.602(-28.2%)	0.581(-20.6%)
LibFM	0.966(-1.4%)	0.914(-1.7%)	0.685(-8.2%)	0.639(-7.2%)	0.777(-7.4%)	0.709(-3.1%)
RippleNet	0.968(-1.2%)	0.925(-0.5%)	0.729(-2.3%)	0.662(-3.9%)	0.768(-8.5%)	0.691(-5.6%)
KGCN	0.978(-0.2%)	**0.932(+0.2%)**	0.684(-8.3%)	0.655(-4.9%)	0.802(-4.4%)	0.719(-1.8%)
RFAN	**0.980***	0.930	**0.746***	**0.689***	**0.839***	**0.732***

The contents in brackets are the average gains of RFAN for the baselines. The champion is bold and the runner up is underlined."*" denotes the improvement of RFAN is significant based on paired t-test for $p = 0.01$

than KGCN is that RFAN uses the multi-head attention mechanism to learn richer representations for each entity, so as to strengthen the representation of users and items. In addition, RFAN uses non-sampling strategy to train and optimize parameters on the whole data, while the baselines only extract a small number of negative samples, which is one of the reasons for the good performance of RFAN.

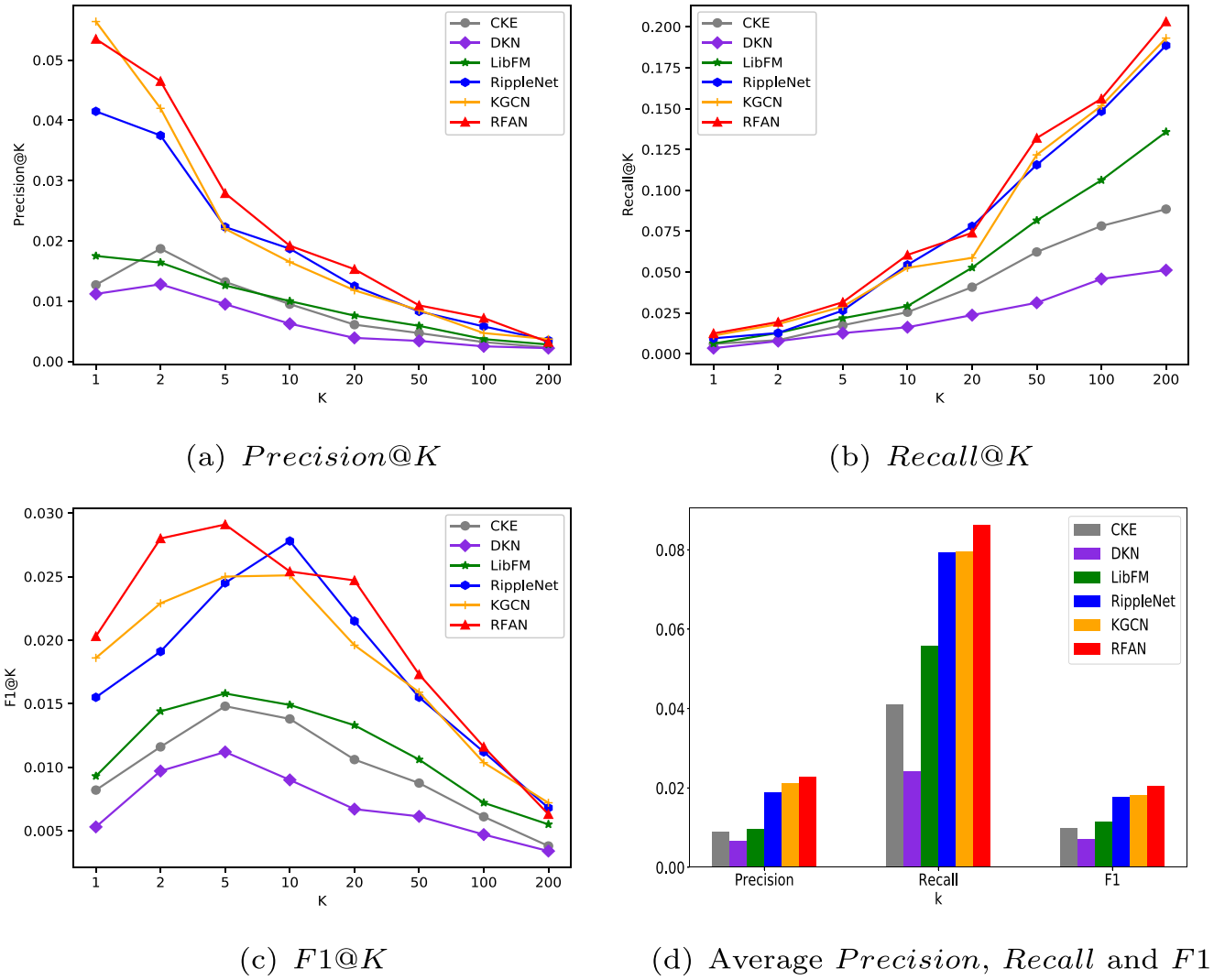
DKN performed worst compared with other baselines because the name of the input movie/book/music was too short to provide rich information for DKN. The CKE performance exceeds that of DKN on the three datasets because CKE better integrates a knowledge graph into CF to improve the model effect. However, CKE only uses structural knowledge without other additional



(a) $Precision@K$ (b) $Recall@K$

(c) $F1@K$ (d) Average $Precision$, $Recall$ and $F1$

Fig. 4 The results of $Precision@K$, $Recall@K$, $F1@K$ and average over all K for MovieLens-20M



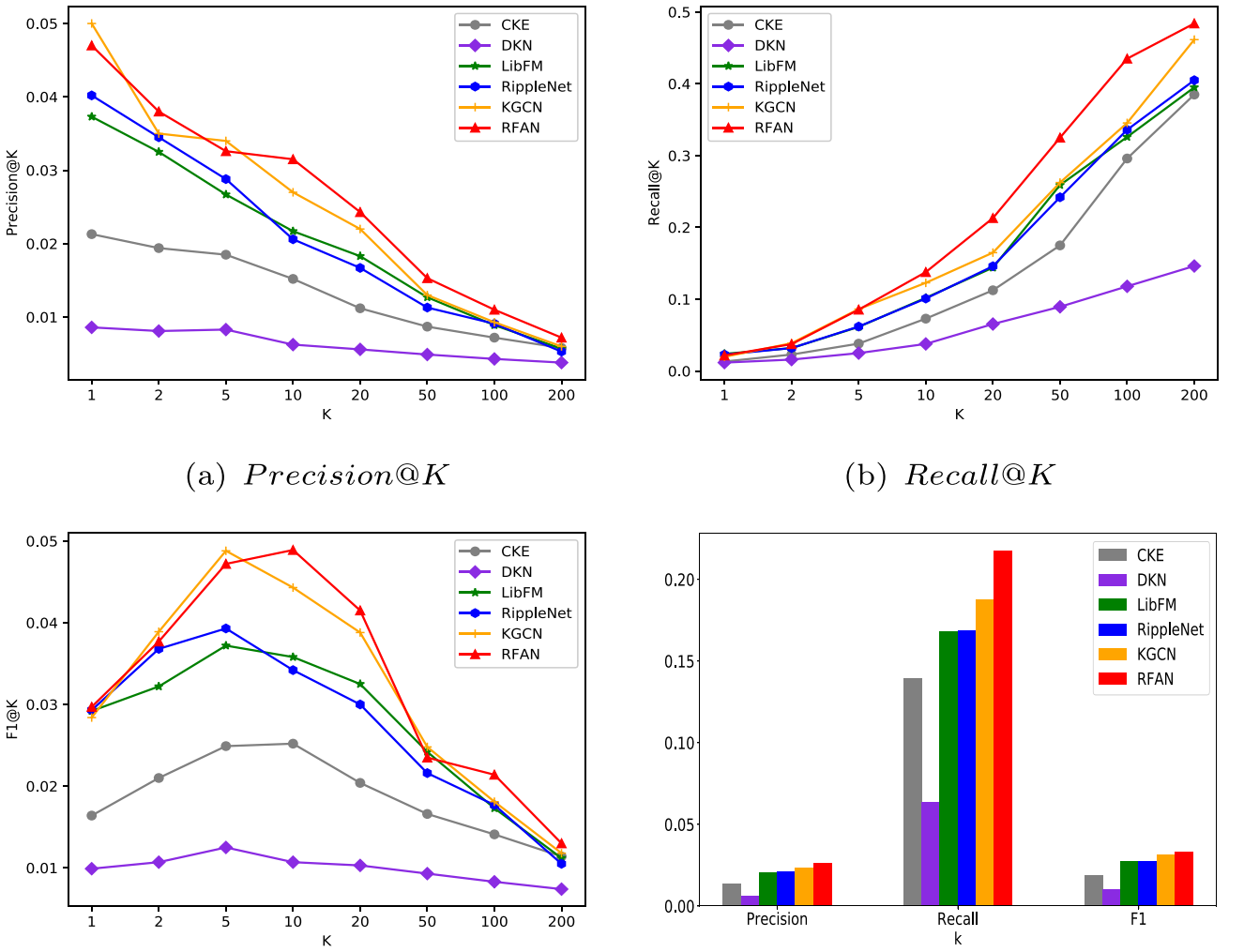
(a) $Precision@K$ (b) $Recall@K$

(c) $F1@K$ (d) Average $Precision$, $Recall$ and $F1$

Fig. 5 The results of $Precision@K$, $Recall@K$, $F1@K$ and average over all K for Book-Crossing

information, so its performance is not as good as RippleNet and KGCN. Note that CKE performs better in movie recommendation than book and music. This may be because Book-Crossing and Last.FM are sparser and are not suitable for collaborative filtering of CKE. LibFM achieves satisfactory results on three datasets, showing that it can make good use of KG knowledge, but it is difficult to use multi-hop knowledge propagation and introduce relation information so that its performance is worse than that of propagation-based methods such as RippleNet, KGCN and RFAN. RippleNet reveals strong performance especially in CTR prediction on Book-Crossing and Top-K recommendation on MovieLens-20M second only to RFAN, which demonstrates that RippleNet can accurately capture user preferences by multi-hop neighborhood structure, and makes full use of KG's high-order connectivity information. However, it reduces the importance of the relations and makes it difficult to capture the information contained in KG relations. As the best overall baseline model, KGCN not only uses the high-order connectivity of KGs, but also employs the attention mechanism to aggregate neighborhood information, in which the neighbor weights depends on the relations between entities to measure the user preferences for different relations.

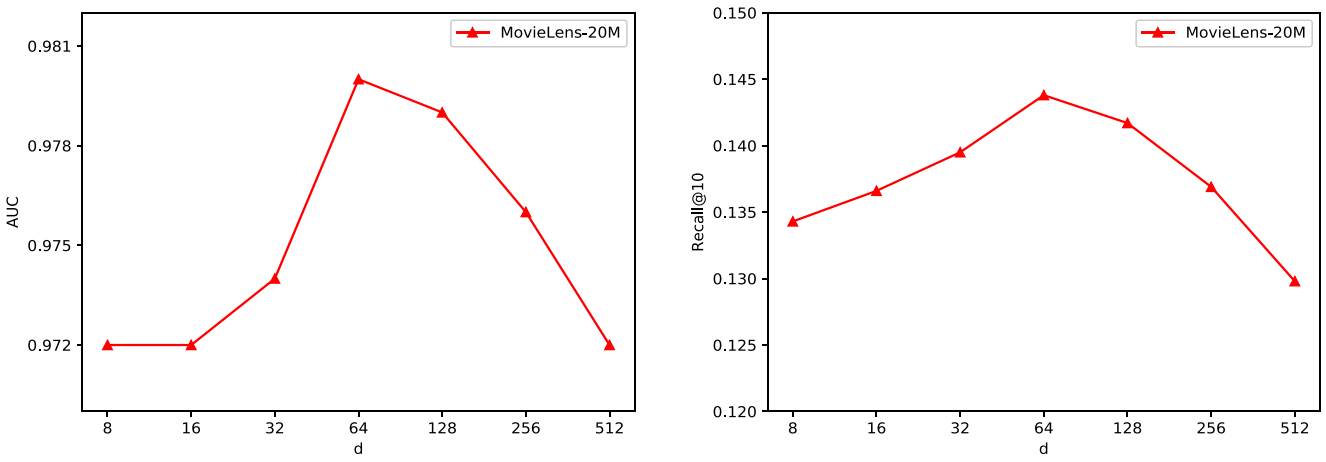
Note that the ranking of the outcomes of all model performances on the three datasets from high to low is MovieLens-20M, Last.FM, Book-Crossing. We reasonably believe that the interaction between users and items in MovieLens-20M may be richer, which can provide more information for the training of the model. In the absence of sufficient user-item interactions and knowledge links in Book-Crossing and Last.FM, it is difficult for recommendation models to learn latent embeddings.



(a) $Precision@K$ (b) $Recall@K$

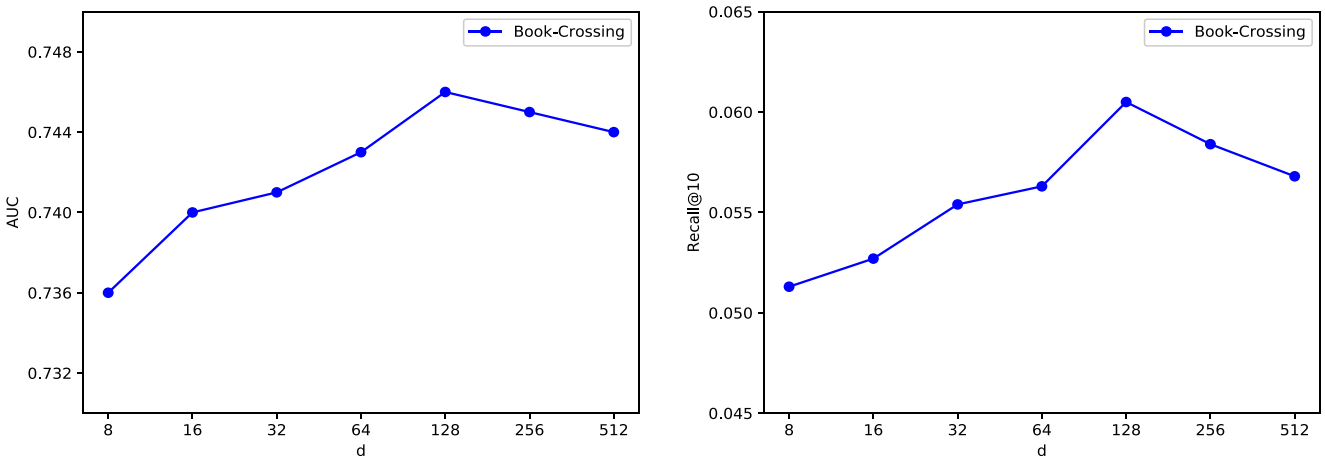
(c) $F1@K$ (d) Average $Precision$, $Recall$ and $F1$

Fig. 6 The results of $Precision@K$, $Recall@K$, $F1@K$ and average over all K for Last.FM



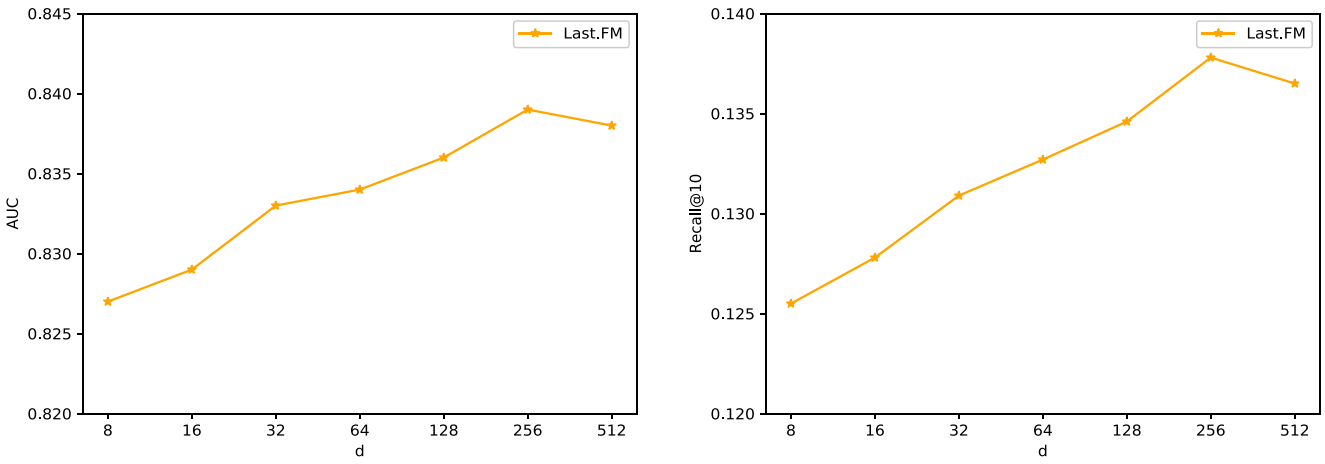
(a) AUC (b) $Recall@10$

Fig. 7 AUC and $Recall@10$ of RFAN w.r.t dimension of embedding on MovieLens-20M



(a) AUC (b) $Recall@10$

Fig. 8 AUC and $Recall@10$ of RFAN w.r.t dimension of embedding on Book-Crossing



(a) AUC (b) $Recall@10$

Fig. 9 AUC and $Recall@10$ of RFAN w.r.t dimension of embedding on Last.FM

Table 3 AUC of RFAN w.r.t different depth of knowledge propagation

Depth of Layer	1	2	3	4
MovieLens-20M	0.973	**0.980**	0.956	0.644
Book-Crossing	0.744	**0.746**	0.728	0.572
Last.FM	0.834	0.838	**0.839**	0.829

Table 4 AUC of RFAN w.r.t different depth of knowledge propagation

Aggregator	RFANsum	RFANpool	RFANconcat
MovieLens-20M	0.967	0.962	**0.979**
Book-Crossing	0.724	0.737	**0.746**
Last.FM	0.828	0.825	**0.839**

Table 5 AUC of RFAN w.r.t different knowledge graph embedding methods

Embedding methods	TransE	TransH	TransR
MovieLens-20M	0.968	0.974	**0.980**
Book-Crossing	0.736	0.742	**0.745**
Last.FM	0.826	0.833	**0.839**

4.4 Effect of parameter setting

In this section, we explore the effect of different parameters on the performance of RFAN.

4.4.1 Effect of dimension of embedding

We discuss the performance of RFAN by changing the entity embedding dimensions, as demonstrated in Figs. 7, 8 and 9. It can be seen that the AUC value and $recall@10$ reach the peak when the embedding dimensions on MovieLens-20M, Book-Crossing and Last.FM are 64, 128, and 256 respectively. Then, as the entity embedding dimension increases, the AUC and $recall@10$ gradually decrease. The results in Figs. 7, 8 and 9 indicate that increasing the dimension d of embeddings within a certain range can effectively encode more KG information, but when d exceeds the threshold, it causes overfitting, so AUC and $Recall@10$ present a trend of first rising and then falling.

4.4.2 Effect of knowledge propagation layers depths

We discuss the impact of the depth of knowledge propagation layer by changing the number of layers from 1 to 4 as demonstrated in Table 3. The results indicate that the AUC value reaches the maximum when the number of layers on MovieLens-20M, Book-Crossing and Last.FM are 2, 2, and 3, respectively. A reasonable explanation is that it is not enough to use only the first-order neighborhood entity information. With the increase in the number of knowledge propagation layers to 3 or 4, too many entities are incorporated into the model to provide more knowledge information while introducing more noise, making the representation of all users and items tend to be similar. Therefore, excessively long-distance propagation leads to over-smoothing.

4.4.3 Effect of aggregators

In this section, we explore the impact of different aggregators on aggregating user and project representations. Three aggregators are selected:

(1) The concat aggregator is introduced in Section 3.4, in which the same symbol is used for the other two aggregators.

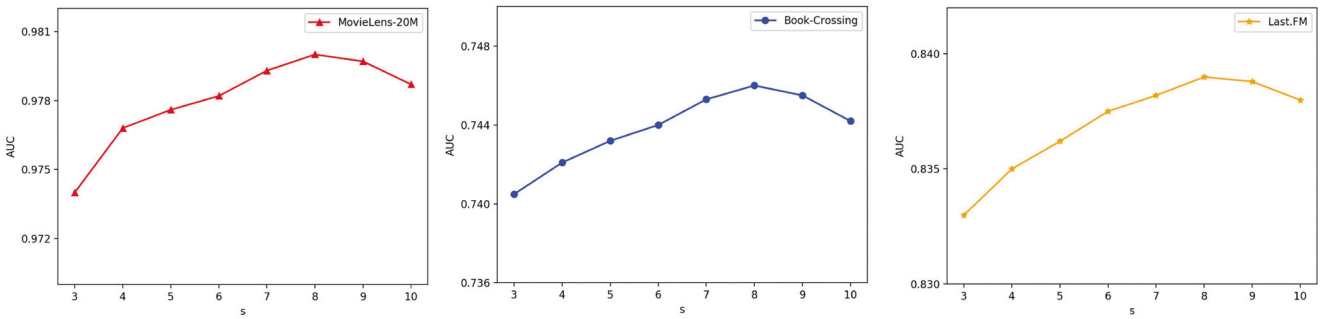
(2) The sum aggregator performs the summation operation on multiple representation vectors followed by a nonlinear transformation:

$$\begin{aligned} p_u &= \sigma\left(W_d * \sum_{i=1}^{l} \rho_u^{(i)} + b_2\right) \\ q_v &= \sigma\left(W_d * \sum_{i=1}^{l} x_v^{(i)} + b_2\right) \end{aligned} \tag{20}$$

(3) The maximum pooling aggregator takes the maximum value in multiple representation vectors before nonlinear transformation:

$$\begin{aligned} p_u &= \sigma\left(W_d * pool_{max} \rho_u^{(i)} + b_2\right) \\ q_v &= \sigma\left(W_d * pool_{max} x_v^{(i)} + b_2\right) \end{aligned} \tag{21}$$

Table 4 shows the AUC of the three aggregators on the RFAN. Obviously, the concat aggregator has won a complete victory because the concatenation operation can retain more original information hidden in the embeddings. The summation and pooling operations may destroy or discard much useful information, resulting in poor performance.



(a) MovieLens-20M (b) Book-Crossing (c) Last.FM

Fig. 10 AUC of RFAN w.r.t attention heads

4.4.4 Effect of knowledge graph embedding methods

We compare the effects of three knowledge graph embedding methods on RFAN performance, as shown in Table 5. It can be observed that the embedding method based on TransR works best. This is because TransE and TransH project entities and relations in the same semantic space, but entities contain multiple attributes. In the entity space, some entities are similar and close to each other, but some specific entity attributes are quite different, and therefore are far apart from each other in the related relation space. In contrast, TransR maps entities and relations into different semantic spaces so it can better handle complex knowledge relations.

4.4.5 Effect of the number of attention heads

In this section, we discuss the influence of the number of attention heads on the model. As shown in Fig. 10, it can be seen that on the three datasets, AUC shows a trend of rising first and then falling. When the number of attention heads is 8, AUC reaches the maximum. The results show that the number of attention heads does have an impact on the performance of the RFAN. This is because multiple heads are equivalent to multiple single-heads working together, which can stabilize the learning process and prevent overfitting. A small number of attention heads is insufficient to capture various semantic information of the entity, while using too many heads introduces redundant information.

4.4.6 Effect of important components

We verify the impact of important components on the performance of the RFAN through ablation studies and design four comparative variants of the RFAN.

(1) RFAN-t: RFAN without type-enhanced entity embedding and only with entity original embedding.
(2) RFAN-a: RFAN without multi-head attention mechanism to update the entity representation and set π_{hrt} as $1/|N_h|$.
(3) RFAN-m: RFAN without multi-head attention mechanism to update the entity representation and replaced by single-head attention.
(4) RFAN-n: RFAN without non-sampling strategy when training negative samples and with negative sampling.

We compared RFAN with its four variants, as revealed in Fig. 11. It is clear that RFAN-a has the worst performance on the three datasets. It sets the weight of all triplets to the same value so that it cannot distinguish the different contributions of triplets, which reflects the importance and necessity of attention mechanism. RFAN-m proves the excellence of the multi-head attention mechanism, which enriches the ability of the model and stabilizes the training process compared to the single-head attention mechanism.

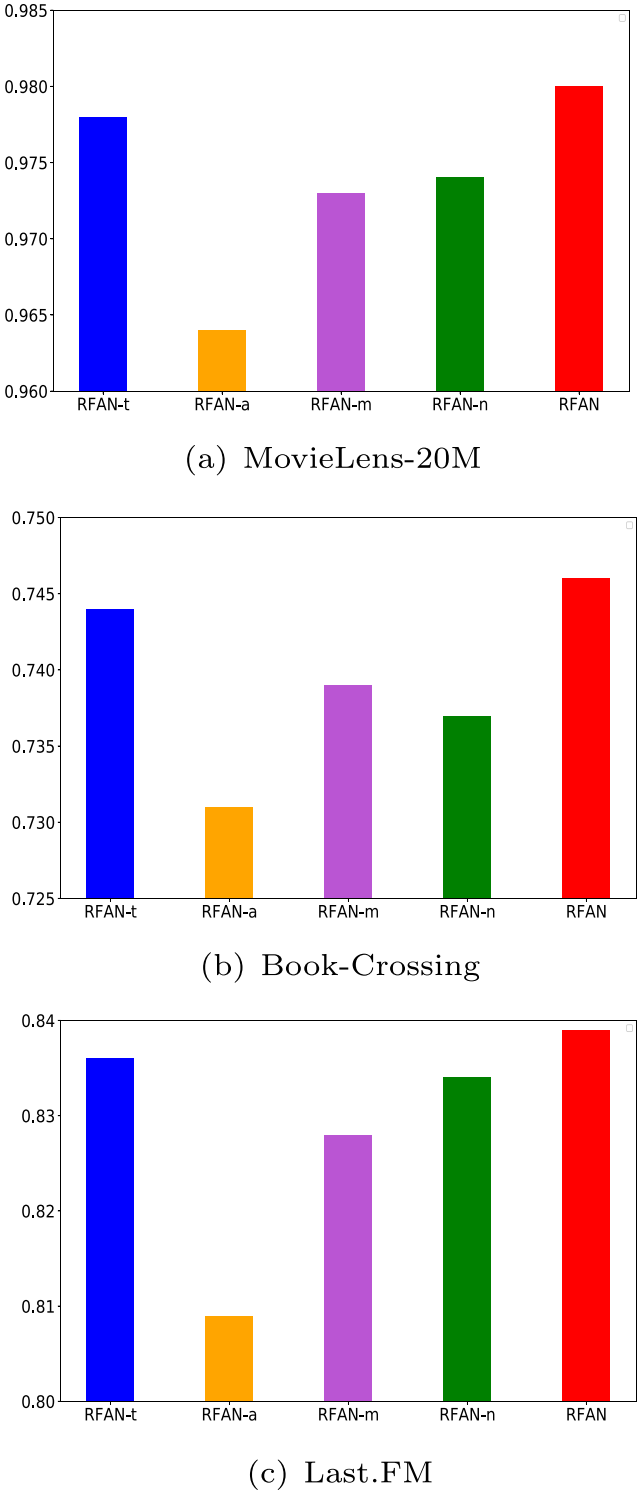


Fig. 11 AUC of RFAN w.r.t important components

We also verify the success of non-sampling strategy. Due to negative sampling can only extract part of the negative examples which cannot represent all the samples. The robustness of negative sampling is poor, and the model has difficulty achieving optimization. Moreover, we encode entity types into entity embedding, which is also helpful to improve the performance of RFAN.

5 Conclusion and future work

This paper proposed a relation-fused multi-head attention network for knowledge graph enhancement recommendation called RFAN. We improved the limitations of traditional graph attention networks on knowledge graphs, emphasized the importance of relations in KGs, and used multi-head attention mechanism to update entity embedding representation. RFAN combines user-item interaction graphs and knowledge graphs, uses high-order connectivity for knowledge propagation and constructs aggregate representations of users and items. Finally, RFAN applies a non-sampling strategy to train the whole data. Experiments on three public datasets achieved satisfactory performance in CTR prediction and Top-K recommendation.

In a realistic scenario, there are many interactive relations between users and items and there may be more than one relation between entities in the knowledge graph, and these relations may affect each other. Our future work will focus on how to fine-grained model the interaction relations between different types of user-items and the entity-relations of the knowledge graph.

Acknowledgements This work was supported in part Key R & D project of Shandong Province 2019JZZY010129, and in part by the Shandong Provincial Social Science Planning Project under Award 19BJCJ51, Award 18CXWJ01, and Award 18BJYJ04.

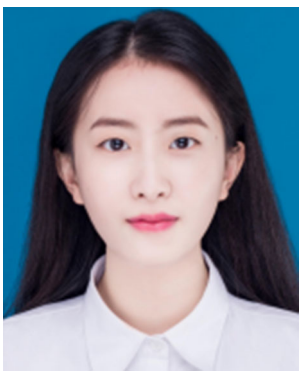
References

1. Wu L, He X, Wang X, Zhang K, Wang M (2021) A survey on accuracy-oriented neural recommendation: From collaborative filtering to information-rich recommendation
2. Bi JW, Liu Y, Fan ZP (2020) A deep neural networks based recommendation algorithm using user and item basic data. Int J Mach Learn cybern 11(4):763–777
3. Da'U A, Salim N (2020) Recommendation system based on deep learning methods: a systematic review and new directions. Artificial Intelligence Review 53(6)
4. He X, Chua TS (2017) Neural factorization machines for sparse predictive analytics. In: ACM, pp 355–364
5. Lian J, Zhou X, Zhang F, Chen Z, Xie X, Sun G (2018) Xdeepfm: Combining explicit and implicit feature interactions for recommender systems. In: The 24th ACM SIGKDD international conference
6. Zhang HR, Min F, Zhang ZH, Wang S (2018) Efficient collaborative filtering recommendations with multi-channel feature vectors. International Journal of Machine Learning and Cybernetics
7. Chen J, Wang B, Ouyang Z, Wang Z (2020) Dynamic clustering collaborative filtering recommendation algorithm based on double-layer network. International Journal of Machine Learning and Cybernetics (1):1–17
8. Tewari AS (2020) Generating items recommendations by fusing content and user-item based collaborative filtering. Procedia Computer Science 167:1934–1940
9. Han J, Zheng L, Xu Y, Zhang B, Zuo W (2019) Adaptive deep modeling of users and items using side information for recommendation. IEEE Trans Neural Netw Learn Syst PP(99): 1–12
10. Wang C, Zhang M, Ma W, Liu Y, Shaoping Ma YL (2020) Make it a chorus: Knowledge- and time-aware item modeling for sequential recommendation, 109–118
11. Wang Q, Mao Z, Wang B, Guo L (2017) Knowledge graph embedding: a survey of approaches and applications. IEEE Trans Knowl Data Eng 29(12):2724–2743
12. Zhang F, Defu Lian NJY, Xie X, Ma W (2016) Collaborative knowledge base embedding for recommender systems. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp 353–362
13. Yang Y, Zhu Y, Li Y (2021) Personalized recommendation with knowledge graph via dual-autoencoder. Appl Intell, 1–12
14. Wang H, Zhang F, Wang J, Zhao M, Guo M (2018) Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. ACM
15. Wang H, Zhao M, Xie X, Li W, Guo M (2019) Knowledge graph convolutional networks for recommender systems. In: The world wide web conference, WWW 2019, pp 3307–3313
16. Wang X, He X, Cao Y, Liu M, Chua T (2019) KGAT: Knowledge graph attention network for recommendation. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining, KDD 2019, pp 950–958
17. Wang Z, Lin G, Tan H, Chen Q, Liu X (2020) CKAN: Collaborative knowledge-aware attentive network for recommender systems. In: Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval, SIGIR 2020, pp 219–228
18. Velikovi P, Cucurull G, Casanova A, Romero A, Liò P, Bengio Y (2017) Graph attention networks
19. Nathani D, Chauhan J, Sharma C, Kaul M (2019) Learning attention-based embeddings for relation prediction in knowledge graphs
20. Fu TY, Lee WC, Lei Z (2017) Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning. In: The 2017 ACM
21. Shi C, Hu B, Zhao X, Yu P (2017) Heterogeneous information network embedding for recommendation. IEEE Trans Knowl Data Eng, 1–1
22. Hu B, Shi C, Zhao W (2018) Leveraging meta-path based context for top- n recommendation with a neural co-attention model. In: The 24th ACM SIGKDD international conference
23. Sun Z, Yang J, Zhang J, Bozzon A, Huang L, Xu C (2018) Recurrent knowledge graph embedding for effective recommendation. In: Proceedings of the 12th ACM conference on recommender systems, RecSys 2018, pp 297–305
24. Wang X, Wang D, Xu C, He X, Cao Y, Chua TS (2018) Explainable reasoning over knowledge graphs for recommendation
25. Sha X, Sun Z, Zhang J (2019) Attentive knowledge graph embedding for personalized recommendation
26. Hui B, Zhang L, Zhou X, Wen X, Nian Y (2021) Personalized recommendation system based on knowledge embedding and historical behavior. Applied Intelligence (7)

27. Huang J, Zhao WX, Dou H, Wen J, Chang EY (2018) Improving sequential recommendation with knowledge-enhanced memory networks. In: The 41st international ACM SIGIR conference on research & development in information retrieval, SIGIR 2018, ann arbor, MI, USA, July 08-12, 2018, pp 505–514
28. Bordes A, Usunier N, García-durán A, Weston J, Yakhnenko O (2013) Translating embeddings for modeling multi-relational data. In: Advances in neural information processing systems 26: 27th annual conference on neural information processing systems 2013. Proceedings of a Meeting Held December 5-8, 2013, Lake Tahoe, Nevada, United States, pp 2787–2795
29. Cao Y, Wang X, He X, Hu Z, Chua TS (2019) Unifying knowledge graph learning and recommendation. Towards a better understanding of user preferences
30. Zhang J (2014) Knowledge graph embedding by translating on hyperplanes AAAI - Association for the Advancement of Artificial Intelligence
31. Lin Y, Liu Z, Sun M, Liu Y, Zhu X (2015) Learning entity and relation embeddings for knowledge graph completion. In: Proceedings of the Twenty-Ninth AAAI conference on artificial intelligence, Jan 25-30, 2015, Austin, Texas, USA, pp 2181–2187
32. Wang H, Zhang F, Xie X, Guo M (2018) DKN: Deep knowledge-aware network for news recommendation. In: Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018, pp 1835–1844
33. Ji G, He S, Xu L, Liu K, Zhao J (2015) Knowledge graph embedding via dynamic mapping matrix. In: Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing of the asian federation of natural language processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers, pp 687–696
34. Chen C, Zhang M, Zhang Y, Liu Y, Ma S (2020) Efficient neural matrix factorization without sampling for recommendation. ACM Trans Inf Syst 38(2):14–11428
35. Chen C, Zhang M, Wang C, Ma W, Li M, Liu Y, Ma S (2019) An efficient adaptive transfer neural network for social-aware recommendation. In: Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval, SIGIR 2019, Paris, France, July 21-25, 2019, pp 225–234
36. Chen C, Zhang M, Zhang Y, Ma W, Liu Y, Ma S (2020) Efficient heterogeneous collaborative filtering without negative sampling for recommendation. In: The thirty-fourth AAAI conference on artificial intelligence, AAAI 2020, the thirty-second innovative applications of artificial intelligence conference, IAAI 2020, the tenth AAAI symposium on educational advances in artificial intelligence, EAAI 2020, new york, NY, USA, February 7-12, 2020, pp 19–26
37. Chen C, Zhang M, Ma W, Liu Y, Ma S (2020) Jointly non-sampling learning for knowledge graph enhanced recommendation. In: Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020, pp 189–198
38. Wang H, Zhang F, Wang J, Zhao M, Li W, Xie X, Guo M (2019) Exploring high-order user preference on the knowledge graph for recommender systems. ACM Transactions on Information Systems (TOIS)
39. Wang H, Zhang F, Zhang M, Leskovec J, Zhao M, Li W, Wang Z (2019) Knowledge-aware graph neural networks with label smoothness regularization for recommender systems. In: The 25th ACM SIGKDD international conference
40. Rendle S (2012) Factorization machines with libfm ACM Transactions on Intelligent Systems and Technology (TIST)

Huajuan Duan is currently pursuing a doctorate in the school of information science and engineering, Shandong Normal University, China. Her research interests include deep learning, knowledge graph embedding and personalized recommendation.

Peiyu Liu received the master's degree from East China normal university in 1986. He is currently a Second-level professor, doctoral supervisor, with the School of information science and engineering, Shandong Normal University, China. He is the national outstanding science and technology worker, middle-aged and young expert with outstanding contribution in Shandong province, famous teacher of Shandong province. His research interests include network information security, information retrieval, natural language processing, and artificial intelligence.



Qi Ding was born in 1997. She is currently pursuing the master's degree in the school of information science and engineering, Shandong Normal University, China. Her research interests include recommendation system, data mining and robustness research.