Applied Intelligence (2022) 52:10032–10044
https://doi.org/10.1007/s10489-021-02872-8

Improving recommender system via knowledge graph based exploring user preference

Huilian Fan[1] · Yuanchang Zhong[2,3] · Guangpu Zeng[1] · Chenhao Ge[2,3]

Accepted: 16 September 2021 / Published online: 11 January 2022
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2021

Abstract
Knowledge graph(KG) has proven to improve recommendation performance. However, most efforts explore inter-entity relatedness by mining multi-hop relations on KG, thus failing to efficiently exploit these relations for enhanced user preference. To address this, we propose an end-to-end framework to improve the recommender system via a knowledge graph based on fusing entity relation(KGFER), which can sufficiently capture the users' preferences. The model samples from the 1-hop neighbors and relations in KG for the item with which the user interacts, and feeds them into TransR layer. Following this, CNN is employed to learn item features from entity-relations, and then aggregate item features with the interacting item by MLP. Finally, we apply user preference to project the refined item embeddings to the user latent space to predict the potential probability of the target item in which the user is interested in. Extensive experiments on four datasets about book, movie, music, and yelp2018 demonstrate that our approach outperforms state-of-the-art baselines. Also, further experiments show that the user preference matrix indeed makes a great contribution to our approach.

Keywords Recommender system · Knowledge graph · Item feature coefficients · User preference matrix

1 Introduction

Recommender system, which searches for a small set of entities to meet users' interests, plays an important role in many online services, such as music, movie, news, E-commerce, etc. A traditional recommendation technique is collaborative filtering(CF) based matrix factorization(MF), which projects the users and items into latent space, then models their interactions by inner product [1]. Despite the effectiveness of MF, the CF-based methods usually suffer from the sparsity of user-item interactions and simply combine the multiplication of latent features [2]. To address these limitations, more researchers have been turned to feature-rich scenarios and deep neural networks to improve the quality of recommendation, where attributes of the users and items are used to compensate for the sparsity, and capture users' potential interests via deep network layer [3]. For example, ECAE model [4] fuses separated training stages of generation and retrain into a unified framework to learn useful knowledge from discrete data of user feedbacks and balances the impact of knowledge and noise based on a novel distillation layer. DMGCF model [5] generate multiple graphs with a dynamic evolution mechanism to simulate side information for better performance, especially when side information is unavailable. A novel adaptive method [6], which integrate the role-based trust strength into recommendation model and considering both explicit and implicit information of trust and ratings, learn the trust influence between users with multiple roles of truster and trustee for recommendation.

Among various types of feature-rich scenarios, knowledge graph(KG), which is a directed heterogeneous graph in which nodes correspond to entities (or entity features)and edges correspond to relations, contains much more fruitful features and connections about entities [7]. The rich semantic relatedness among entities can help capture their latent

✉ Huilian Fan
fanhl@yznu.edu.cn

✉ Yuanchang Zhong
zyc@cqu.edu.cn

1 School of Big Data and Intelligent Engineering, Yangtze Normal University, 408100 Chongqing, China

2 School of Electrical Engineering, Chongqing University, Chongqing, China

3 China State Key Laboratory of Power Transmission Equipment & System Security and New Technology, 400044 Chongqing, China

connections, and help for exploring users' interests, thereby improve the precision of recommendation [8].

However, utilizing KG in recommendation system is rather challenging due to KG's heterogeneity, large volume, and high dimensionality [9].The first challenge is how to efficiently mine users' preferences base on entity relations.

For example, the film "Titanic" is linked with "James Cameron" (director), "Kate Winslet" (star), "U.S." (country), and "Disaster" (genre). These connections in KG provide us latent perspective to explore users' preferences.

To explore users' preferences, some methods try to capture both entity and relation features in multi-hop neighborhood of the interacted entity [10, 11]. However, if we don't catch the key factor the deeper we explore the farther away from the user's preference. As the above example, some users choose to watch the film because they like the star, some like the genre, like the director, etc. For example, GNUD [12] builds a bipartite graph based on user-news interactions and encodes high-order relationships into user and news representations by information propagation along the graph to explore users' preferences and achieve good results. However, we can explore the key factors why users interact with the entity in a shorter hop.

The second challenge is how to encode the extracted side information in an effective way for enhanced inference user preference. Several recent efforts have attempted to generalize and extend graph attention mechanisms to capture both entity and relation features [13–15], but most of them encode the obtained information with the entity embedding representation by inner product, sum, or conjunction operation. We believe that the user preferences are not fixed, but rather should change with different entities.

To this end, we propose an end-to-end model KGFER, which sufficiently captures the preference factors of a user by fusing entity and relation information via KG. The main contributions of our approach can be summarized as follows.

- We propose a novel method that can efficiently explore item features on the knowledge graph for recommender system. Be different from the previous works adopting the inner product, summation operation, and conjunction operation to integrate features of entities and relations, we propose a novel approach by performing matrix multiplication on the entity embedding vector and the relational embedding vector, then feed the result into convolutional neural network (CNN) to fuse the entity-relation features. After that, we integrate the embedding of the interacted (head) item by aggregating the fusing features through a multi-layer perceptron (MLP).
- Different from the previous works that make predictions directly based on the inner product results of item and user embeddings, we introduce a User Preference Matrix (UPM) into the predicting operation. The UPM is used to project refined item embeddings from their latent space into the user embedding space, and then evaluate the potential probability of the user preferences towards the target items through inner product results of user embedding with the projected item embedding representation. This UPM is a matrix of weighting coefficients that moderates the predicted outcome of the neural network and is learned implicitly from the prediction results based on the refined item embedding.
- We conduct experiments on four real-world recommendation scenarios, and the results demonstrate the efficacy of KGFER over state-of-the-art baselines.
- We release the code of KGFER to researchers for validating the reported results and conducting further research. These are available at https://github.com/fanhl/KGFER.

The rest of this paper is organized as follows. We provide a brief review of related work in Section 2. Section 3 introduce the details of proposed method. The datasets descriptions and experimental results are reported in Section 4. Lastly, the main work of this paper is summarized and the direction of future work is also pointed out in Section 5.

2 Related work

Recommender systems aim to find a small set of items for users to their interest from the item pool. Traditional recommendation methods, collaborative filtering(CF) models, which mine the similarities between users and items via historical interactions and make recommendation, has achieved great success [26]. Matrix factorization techniques [16] are widely used in recommender systems, Latent factor-based [17, 18] and SVD-based models [19, 20] also are common. However, CF-based methods usually suffer from the sparsity of user-item interactions. To ease the limitations, researchers have proposed incorporating side information into CF, such as social networks-based hybrid methods [27–29, 49]. For example, a named Sparse Stacked Denoising Autoencoder [30] aims to learn low- and high- level features from social information based on multi-layers neural and matrix factorization techniques. A novel correlative denoising autoencoder model [31] by utilizing three separated autoencoders to learn user features with roles of rater, truster, and trustee from a social network.

Due to KG usually contains much more fruitful features and connections about items, and it also contains the success of applying KG in wide variety of tasks, more and more researchers employ KG to improve the performance of recommender systems [21–24]. For example, AGSR [25] proposes annular-graph attention applying on the sub annular-graph to explore users' short-term preferences and introduces a latent factor model to explore long-term preferences and achieves good performance, etc. Existing KG-aware recommendations can be generally classified into three categories [13]:

Entity-relation embedding based methods Collaborative knowledge base embedding [32, 51] combines CF module with knowledge embedding via TransR[33] in a unified Bayesian framework. Deep Knowledge-aware Network [34] designs a CNN framework [48] to generates news embeddings by utilizing KG via TransD [35] for news recommendation. Unifying Knowledge Graph Learning and Recommendation [36] jointly learns the recommendation model and KG completion via TransH [37]. Relational Collaborative Filtering [38] combines user preference via attention mechanisms with item relations via DistMult. The Multi-Task Feature Learning model [39] utilizes KG embeddings to regularize recommendation tasks. Despite significant improvements have been achieved [40], the adopted KG algorithms in these methods are usually more suitable for in-graph applications such as link prediction than for capturing complex semantics of user-item connectivity for recommendation.

Path based methods These methods mine the various connections among items in KG based similarity to provide additional information for recommendations. HetNERec [41] constructs the co-occurrence networks by extracting multiple co-occurrence relationships from a recommendation-oriented heterogeneous network. Stacked Mixed-Order GCN [42] proposes a GCN-based framework [43] that can directly capture high-order connectivity among nodes KPRN [44] can generate path representations by composing the semantics of both entities and relations. In [23], a learning path generation algorithm is introduced which generates relational constraints based on individual learning needs to determine how to generate all possible paths. Path-based methods make use of KG in a more natural and intuitive way, however, decomposing the sophisticated user-item connectivity into separate linear paths would inevitably lead to information loss. For example, these methods try to capture both entity and relation features in multi-hop paths, but insufficiently explore the key factors why user interact with the entity, then the deeper we dig, the farther away from the user's preference.

Combining embedding-based and path-based methods Recent methods iteratively perform propagation over the whole KG to assist in the recommendation. KGAT [45] recursively performs propagation with attention-based aggregation over KGs via GCN to refine entity embeddings. Due to taking the whole KG as input, these methods learn embedding for each entity by aggregating information from all of its multi-hop neighbors, heavy computation (i.e., the large number of neighbors) would be inevitably introduced into the embedding learning process.

AKG E[13] exploits both semantics and topology of KGs by automatically extracts high-order subgraphs, and then encodes the subgraphs to learn accurate user preference by the proposed attentive graph neural network. It is implemented in a batch fashion, then can be operated on large datasets.

RippleNet [10, 46] automatically discovers users' hierarchical potential interests by iteratively propagating users' potential interests along with links in the KG, and incorporates the neighborhood information when computing the representation of a given entity.

PRSKG [52] employs self-attention mechanism to mine user preference from his/her historical interaction with KG. CGAT [53] introduce an attention mechanism to aggregate context information, which are extracted by a biased random walk sampling operation in KG.

The accuracy and personality of the recommendation are raised significantly. However, how to extract useful information, reduce the noise, and integrate side information from heterogeneous knowledge graph is quite a challenge. We also noticed that these models incorporate users' potential interests into entity embedding. Due to different users interact with the same entity for various preferences, we believe that user preferences should be constructed alone, rather than integrated into entities or user representations.

To this end, our approach KGFER explores the features of item in which user interact via fusing neighboring entities-relations based on convolutional neural network (CNN) and MLP, and learns the user preferences matrix for the interacted item by the refined item embedding to improve recommendation performance.

3 Methodology

In this section, we first introduce the notations and definitions used in the rest of the paper. Then, we present the exploring item feature and user preference framework for recommended tasks based KG and CNN. Finally, we show a method for calculating recommendation results based on vector projection.

3.1 Problem statement

In a typical recommendation scenario, we have user sets as $U = \{u_1, u_2 ..., u_m\}$ and item sets as $I = \{i_1, i_2, ..i_n\}$, and a user-item interaction martix $Y = \{y_{ui} | u \in U, i \in I\}$, where $y_{ui} = 1$ means that the feedback between user u and item i is observed, and 0 means otherwise [13, 46]. In addition to

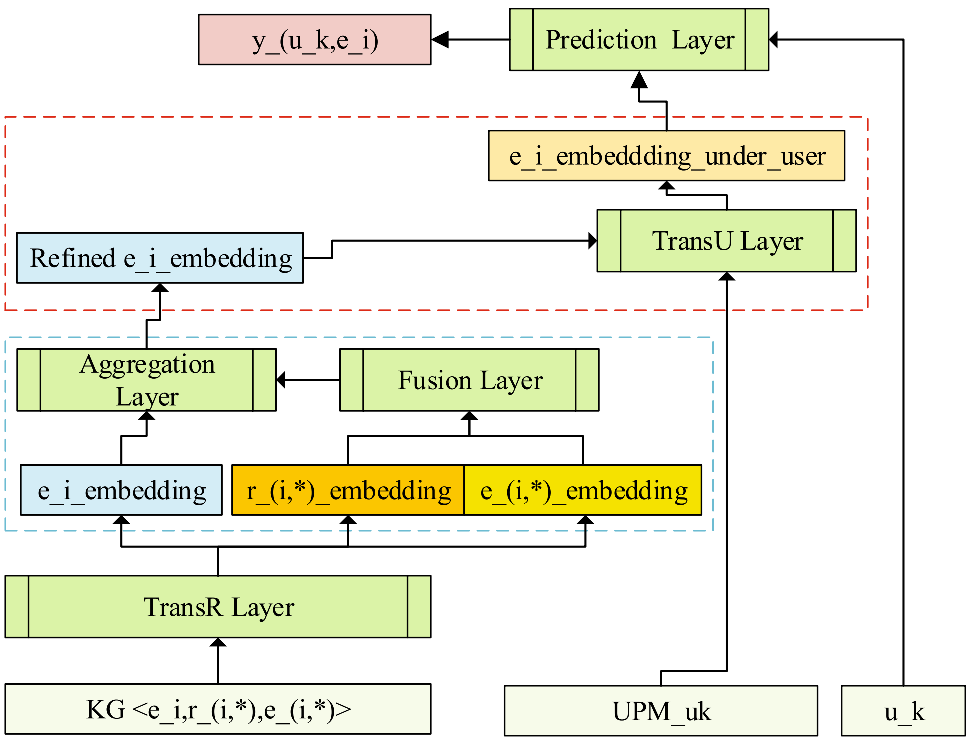

Fig. 1 The overall framework of KGFER

the user-item interaction matrix Y, we also have an available knowledge graph $\mathcal{G}$, which is denoted by $\mathcal{G} = (E, R)$. The KG $\mathcal{G}$ consists of massive entity relation-entity triples (h, r, t), where an entity can be an item or a property. The head h and tail entity t are all in E, and each relation $r \in R$. In general, the number of entity type $|E| > 1$ and the number of relation type $|R| > 1$.

Given the user-item interaction matrix Y and the knowledge graph $\mathcal{G}$, for each user $u \in U$, our task is to predict whether user u has a potential interest in item i. We aim to learn a prediction function $\hat{y}_{ui} = \mathcal{F}(u, i; \Theta, Y, \mathcal{G})$, where $\hat{y}_{ui}$ denotes the potential probability that user u is interested in item i, and Θ denotes the model parameters of function $\mathcal{F}$ [8].

3.2 Framework

The framework of KGFER aims to enhance the performance of recommended systems by capturing users' preferences from entities and relations in knowledge graph. The framework is presented in Fig. 1. There are four components: TransR-based entity and relation embedding layer, CNN-based entity relation fusing layer, MLP-based item features aggregating layer, and trans to userspace (TransU) based prediction layers.

The model takes user-item pairs as inputs. We extract all triplets, which head node is the user interacting item, from the knowledge graph and feed them into TransR layer. The item feature aggregating layer(in the blue dashed box) employs CNN to explore the side information of the head entity on the tail entity-relationship interaction matrix. Then the side information and the head entity(the interacted item) embedding are concatenated and fed into the MLP to get the refined embedding representation of the interacted item.

The TransU layer(in the red dashed box) is a fully connected neural network, which projects interacted items representation from their latent space to user embedding space by user preference matrix(UPM). The UPM is parameters to be learned from item features and optimized through the training process. After that, we employ the inner product to compute the potential probability that the user being interested in the item based on the projected item vector and user embedding. The UPM is implicitly learned from the prediction results based on the item embedding representation.

Specifically, for a candidate pair of user-item $\langle u_i, e_i \rangle$, we use $N(e_i) = \{e_{i1}, e_{i2}, ...\}$ to denote the set of entities which are directly connected to e_i, and use $R(e_i) = \{r_{i1}, r_{i2}, ...\}$ to represent the set of relations, where r_{ij} to denote the relation between entity e_i and e_{ij}. Item features fusing $N(e_i)$ and $R(e_i)$ are aggreageted with interacted item embedding for reconstructing its embedding. After that, we project the refined item embedding from their latent space to user embedding space by user preference matrix(UPM). Finally, we predict the potential probability of the user is interest in the item based on the inner product result.

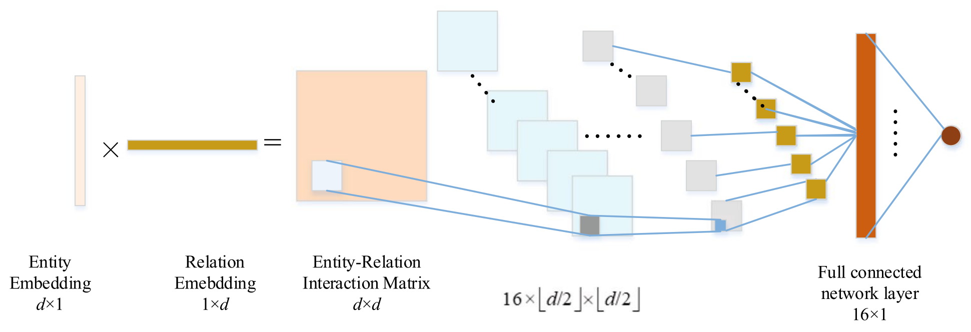
Entity Embedding $d\times 1$
Relation Emebdding $1\times d$
Entity-Relation Interaction Matrix $d\times d$
$16\times\lfloor d/2\rfloor\times\lfloor d/2\rfloor$
Full connected network layer 16×1

Fig. 2 The entity-relationship fusion operation

3.3 Raw input and embedding initialization

Our model KGFER requires user-entity interaction pairs and one-hop neighbors of that interacting entity and the corresponding relationships in the knowledge graph as input. We embed each user, interacting entity, and entity with which the interacting entity has a direct relationship into the low-dimensional space. Let u_i,e_j denote the embedding vector of user i, entity j, e_{ij} is directly neighbor of e_i, and r_{ij} denotes the relation between entity e_i and entity e_{ij}. To make computation more efficient, we make the embedding vectors of users, entities and relations have the same dimension d.

3.4 Embedding entity and relation

We adopt the widely used TransR in knowledge graph to embed entities and relations into entity space $\boldsymbol{R}^d$ and relation space $\boldsymbol{R}^d$. For a entity-relation-entity triple tuple $\langle h, r, t\rangle$, we use projection matrix $M_r \in \boldsymbol{R}^{d\times d}$ to map entities (h,t) from its own space to relation space by $h_r = hM_r$ and $t_r = tM_r$, then, the embedding can be accomplished by regarding as an vector operations, i.e., $h_r + r \approx t_r$ [50]. To achieve goal, the scoring function is defined as

$$f_r(h,t) = \left\| h_r + r - t_r \right\|_2^2 \tag{1}$$

The model is trained using soft-margin loss as

$$\mathcal{L}_{TransR} = \sum_{(h,r,t)\in\mathcal{S}} \sum_{(h,r,t')\in\mathcal{S}'} log(1 + (f_r(h,t) - f_r(h,t'))) \tag{2}$$

where $\mathcal{S}$ is the set of correct triples and $\mathcal{S}'$ is the set of incorrect. All $f_r(\cdot)$ are normalized.

3.5 Fusion layer and item embedding

Intuitively, each user preference could be learned from his historical interaction entities and the set of neighbors connected to these in Knowledge Graph.

Consider a candidate pair of user-entity interaction $\langle u, e\rangle$, the size of $N(e)$ varies significantly, so we uniformly sample a fixed-size set of neighbors from $N(e)$ instead of using its full neighbors. Later, we will discuss the effect of sampling number on the recommended results Specifically, for a user-entity interaction pair $\langle u_k, e_i\rangle$, we define the sampled neighborhood set of entity e_i as $N^s(e_i) = \{\langle e_i, r_{i,j}, e_{i,j}\rangle | e_{i,j} \in N(e_i), r_{i,j} \in R(e_i), j \in [1,S]\}$and $|N^s(e_i)| = S$ is a configurable constant. To capture features of the item e_i that interact with the user, we define the coefficient of item features as

$$\mu^k(r_{i,j}, e_{i,j}|e_i) = \sigma(W_r^T \times (r_{i,j} \oplus e_{i,j})) \tag{3}$$

where $\sigma(\cdot)$, $r_{i,j} \in \boldsymbol{R}^d$, $W_r \in \boldsymbol{R}^{d\times 1}$, denote a non-linear activation function, relation embedding, trainable weight vector. The $\mu^k(r_{i,j}, e_{i,j}|e_i)$ is fused with information of $r_{i,j}$ and $e_{i,j}$ in the triple $\langle e_i, r_{i,j}, e_{i,j}\rangle$ in knowledge graph for extending the feature information of the user interaction object e_i. Because relations and entities are not the same class of objects and have different potential space representations, the inner product operation and summation operation between different categories of objects are unreasonable, and the conjunction operation solves the above problem to some extent. Here, we apply matrix multiply to the entity and the relational latent vector and introduce a two-dimensional CNN to fuse the entity-relational features. Later, we compare and test the effects of different fusion operations on the recommendation results. The entity-relationship fusion operation is shown in Fig. 2. After obtaining the entity-relation interaction matrix, we apply 16 filter kernels of size 4×4 to the interaction matrix(size $d \times d$) and set the stride size to 2. The final output is a $1 \times 1 \times 16$ tensor, which is fed to a fully connected network layer to obtain the item features.

Following this, the item feature coefficients are normalized across all sample set of neighbors using softmax function as follows.

$$\alpha^k(r_{i,j}, e_{i,j}|e_i) = \frac{exp(\mu^k(r_{i,j}, e_{i,j}|e_i))}{\sum_{s\in[1,S]} exp(\mu^k(r_{i,s}, e_{i,s}|e_i))} \tag{4}$$

where S is size of the set $N^s(e_i)$.

Then, we tune the representation of interacted item embedding in the metric space via fusing relation-entity. It is formulated as follows.

$$fuse(r_{i,*}, e_{i,*}|e_i) = \sigma(W \times (e_i||\sigma(W_f^T(\alpha_{i,*} \odot e_{i,*}) + b_f))) + b) \tag{5}$$

where $* \in [1, S]$,$W_f \in R^{S\times1}$,$b_f \in R^d$, and the || represents conjunction operation. The $fuse(r_{i,*}, e_{i,*}|e_i)$ aggregates fusing item features and item embedding representation into a single vector.

3.6 Model prediction and optimization

Traditional recommendation algorithms assume user and item latent vector within the same space and directly use inner product operation on user and item embedding to predict the recommendation results. But users and items are completely different type objects, and user-item interaction is many-to-many relationship, so we employ matrix $E_u \in R^{k\times d}$ to project item from its latent space R^k to user latent space R^d. The matrix E_u is called the user preference matrix of user u. Notably, the dimension of user embedding can be unequal to the dimension of entity latent vector.

With the projection matrix, we define the projected vectors of item e for user-item interaction $\langle u, e\rangle$ as

$$e_u = e \times E_u \tag{6}$$

Then, we employ the inner product between user and the projected entity vector, as

$$\hat{y}_{u^k,e^i} = u_{k,e_i} \odot e_u^i \tag{7}$$

where the output $\hat{y}_{u_k,e_i}$ is used to predict probability that user k is interest in the target entity i. A higher $\hat{y}_{u_k,e_i}$ implies that the user k is more likely to have an interaction with the entity i and vice versa.

We also adopt minimize soft-margin loss to train the model.

$$\mathcal{L}_{cf} = \sum_{(u_k,e_i)\in\mathcal{I}^+} \sum_{(u_k,e_j)\in\mathcal{I}^-} log(1 + (\hat{y}_{u_k,e_j} - \hat{y}_{u_k,e_i}) \tag{8}$$

where $\mathcal{I}^+$ and $\mathcal{I}^-$ denote the sets of observed and non-observed user-entity interactions.

The complete loss function is as follows:

$$\mathcal{L} = \beta \times \mathcal{L}_{cf} + (1-\beta) \times \mathcal{L}_{TransR} + \lambda||\Theta||_2^2 \tag{9}$$

where β is parameter to trade-off among $\mathcal{L}_{cf}$ and $\mathcal{L}_{TransR}$, λ is regularization coefficient and Θ are parameters of the model. We conduct regularization on both model parameters and generated embeddings to prevent overfitting. The overall algorithm is presented in Algorithm 1.

Algorithm 1 KGFER algorithm

```
Input:
  Y: User Interaction tuple
  $\mathcal{G}$: knowledge graph
  $\Theta$: hyper-parameters
Output:
  Prediction function $\mathcal{F}(u, i; \Theta, Y, \mathcal{G})$
1: for epoch in range(n-epochs)
2:   for idx in range(n-batch)
3:     users, postive-items, neg-items=GetInteractionBatch(Y)
4:     heads, postive-realtions, postive-tails, neg-tails=GetKGBatch(pos-items)
5:     user-embed=initEmbedding(users)
6:     pos-item-embed=initEmbedding(postive-items)
7:     neg-item-embed=initEmbedding(neg-items)
8.     item-feature-coefficients=FusionER(realtions,tails)
9.     refined-item-embedding = Aggregate(item-embed,item-feature-coefficients)
10.    obtain recommend loss $\mathcal{L}_{cf}$ from Eq. (8)
11:    postive-item-embed,neg-item-embed = TransR(postive-item,postive-relations,postive-tails,neg-tails)
12:    obtain TransR loss $\mathcal{L}_{TransR}$ from Eq. (2)
13:    $\mathcal{F}$=multiply(user-embed, postive-item-embed)
14:    compute total loss $\mathcal{L}$ from Eq. (9)
15:    Update parameters by gradient descent
```

3.7 Model analysis

The computational complexity of KGFER contains the TransR layer, CNN based fusion layer, MLP based aggregate layer, the TransU layer, and the prediction layer.

The computational complexity of the TransR layer is $O(k \times d)$, where k denotes the embedding size of the entity and d is the embedding size of relation. The fusion layer contains a matrix multiply between entity embedding and relation embedding, its computational complexity is

Table 1 Statistics of the datasets

		MovieLens-20M	Amazon-book	Last-FM	Yelp2018
U-I Interaction	Users	138159	17860	1872	45919
	Items	13002	14910	3846	45538
	Interactions	6750811	69873	21173	1183528
KG	Entities	102569	77903	9366	136499
	Relations	32	25	60	42
	Triplets	499474	151500	15518	1853704

$O(S \times (k \times 1 \times d))$, and a CNN, the computational complexity of lth layer in CNN is $O(\sum_{l=1}^{D} M_l^2 \times K_l^2 \times C_{l-1} \times C_l)$. Where D is the number of convolution layers(the depth of the network), S is sampling size, M is feature map size, K is convolution kernel size and C is the number of channels. So, the total computational complexity of the fusion layer is $O(S \times (k \times 1 \times d + \sum_{l=1}^{D} M_l^2 \times K_l^2 \times C_{l-1} \times C_l))$. The aggregate layer is MLP, its computational complexity is $O(\sum_{h=1}^{H} m_{h-1} \times m_h)$, where m_h denotes the size of the hth layer in MLP. The TransU layer is a feedforward neural network with only one layer. Its computational complexity is $O(d \times d)$, there we also set the user embedding size to d.The computational complexity of the prediction layer is $O(d \times d)$.

Thus, the total computational complexity of KGFER is $O(N \times (k \times d + S \times (k \times 1 \times d + \sum_{l=1}^{D} M_l^2 \times K_l^2 \times C_{l-1} \times C_l)) + \sum_{h=1}^{H} m_{h-1} \times m_h + d \times d + d \times d))$, where N is batch size. The fusion and aggregate layer of the KGFER are the main factors affecting the computational complexity.

4 Experiments

In this section, we evaluate KGFER on four real-world scenarios:movie,book,music and business. Our experiments are aim to answer the following research questions:

- **RQ1** How does KGFER perfrom comapred with state-of-the-art knowledge-aware recommendation methods?
- **RQ2** How do different components affect KGFER?

4.1 Datasets

We utilize the following four benchmark datasets in our experiments [45]:

- MovieLens-20M[1] is a widely used benchmark dataset in movie recommendations, which consists of approximately 6 million explicit ratings on the MovieLens website.
- Amazon-book[2] Amazon-review is a widely used dataset for product recommendation. The benchmark dataset is composed of users and items with at least ten interactions, which consists of about 60,000 interactions.
- Last-FM[3] is music listening dataset collected from Last.fm online music systems. Wherein, the tracks are viewed as the items. In particular, we take the subset of the dataset where the timestamp is from Jan, 2015 to June, 2015. We use the same 10-core setting to ensure data quality.
- Yelp2018[4] is adopted from the 2018 edition of the Yelp challenge. Here we view the local businesses like restaurants and bars as the items. Similarly, we use the 10-core setting to ensure that each user and item have at least ten interactions.

Besides the user-item interactions, we also need to build a knowledge graph for each benchmark dataset. For Amazon-book and Last-FM, We follow the way in [47] to map items into Freebase entities via title matching if there is mapping available. For Yelp2018, we extract item knowledge from the local business information network (e.g., category, location, and attribute) as KG data.

We summarize the statistics of four datasets in Table 1.

4.2 Baselines

We compare the proposed KGFER with the following baselines, which are all KG-aware methods. Hyper-parameter settings for baselines are introduced in the next subsection.

- AKGE [13] extracts high-order user-item pairs subgraphs and encodes them by an attentive mechanism to learn user preference.
- KGAT [45] explicitly models the high-order connectivities in KG, recursively propagates the embeddings to refine entity representation, and uses an attention mechanism to explore the importance of the neighbors.
- KNCR [51] encodes the high-order connectivities of KG and combines KG structure information with collaborative recommendation in an end-to-end neural style.

[1] https://grouplens.org/datasets/movielens/20m/
[2] http://jmcauley.ucsd.edu/data/amazon/
[3] https://grouplens.org/datasets/hetrec-2011/
[4] https://www.yelp.com/dataset

Table 2 Values of the hyper parameters

	Embedding-dim	Batch-size	β	λ	Sampling number	Epochs
MovieLens-20M	128	2048	0.8	10^{-5}	4	100
Amazon-book	128	1024	0.8	10^{-5}	2	50
Last-FM	64	1024	0.8	10^{-5}	2	30
Yelp2018	128	1024	0.8	10^{-5}	8	100

Table 3 Overall performance comparison on the four datasets

Dataset	MovieLens-20M		Amazon-Book		Last-FM		Yelp2018	
Evaluation	AUC	F1	AUC	F1	AUC	F1	AUC	F1
AKGE	0.935	0.923	0.709	0.644	0.756	0.696	0.738	0.691
KGAT	0.948	0.929	0.712	0.658	0.772	0.695	0.756	0.701
KNCR	0.969	0.931	0.721	0.663	0.783	0.711	0.797	0.755
PRSKG	0.979	0.936	0.743	0.698	0.789	0.725	0.806	0.773
CGAT	0.981	0.943	0.752	0.706	0.791	0.733	0.831	0.812
KGFER	0.988	0.962	0.788	0.741	0.821	0.782	0.853	0.846

– PRSKG [52] employs a self-attention mechanism to mine user preference from his/her historical interaction with KG.
– CGAT [53] introduce an attention mechanism to aggregate context information, which is extracted by a biased random walk sampling operation in KG.

4.3 Performance comparison(RQ1)

In KGFER, we set activation function σ to ReLU for TransR,Fusion,Aggregate and Predict layer. Other hyper-parameter settings are presented in Table 2. For each dataset, the ratio of training, evaluation, and test set are 6:2:2. For all competitors, we used the implementation released by the original authors. AUC and F1 are employed to evaluate click-through rate(CTR) prediction. We repeat each experiment 5 times, and the average performance results are reported in Table 3. We have the following observations.

Attention mechanism based methods(i.e.,KGAT and AKGE) achieve better performance than the other compared methods, indicating that these models can enrich the representations of user and entity. However, due to taking the whole KG as input, heavy computation would be introduced noise into the embedding learning process.

KNCR takes a unified neural network model incorporating knowledge graph structures and user-item interactions for recommendation and all parameters are optimized in a federated manner but lack exploring user preferences.

PRSKG employs self-focus mechanism to mine user preferences from each user's historical behavior, but adopts a simple method, linear transformation over the concatenation of entities and relations, to fuse entity-relation.

In order to mine user preferences, CGAT employs a random walk sampling process to organize the item contextual environment and adopts recurrent neural network (RNN) to model the dependencies between entities and relations. Compared to our user preference matrix, the spatial transformation of the different vectors is missing.

The performance of PRSKG and CGAT verifies that incorporating high-hop neighboring items can enrich user representations. However, compared to our method, high-order connectivity could introduce more noise into the user preferences, thus leading to a negative effect.

The performance of KGFER is consistently better than the compared methods on all the datasets. It makes sense for the following reasons. First, sampling the associated entities and relationships of user interaction entities in the knowledge graph is conducive to discovering more characteristic information. The impact of the sampling size on performance is discussed in the following section. Then, we employ the matrix multiplication between the entity and the relationship vector to replace the dot product, summation, or concatenation operations usually used in feature

Table 4 Comparisons of time cost on four datasets

	Dimension	MovieLens-20M(s)	Amazon-Book(s)	Last-FM(s)	Yelp2018(s)
AKGE	128	6135	3126	532	12335
KGAT	128	6569	3751	555	12652
KNCR	64	6321	3633	521	12582
PRSKG	128	5108	2357	493	12167
CGAT	128	5082	2298	427	12012
KGFER	128	4879	2215	353	11823

Table 5 Results of ablation study

MethodID	Components		MovieLens-20M		Amazon-Book		Last-FM		Yelp2018	
	Fusion	TransU	AUC	F1	AUC	F1	AUC	F1	AUC	F1
1	×	×	0.882	0.856	0.665	0.617	0.693	0.652	0.692	0.655
2	×	√	0.911	0.876	0.672	0.619	0.715	0.661	0.701	0.670
3	√	×	0.947	0.912	0.751	0.713	0.789	0.755	0.815	0.796
4	√	√	0.988	0.962	0.788	0.741	0.821	0.782	0.853	0.846

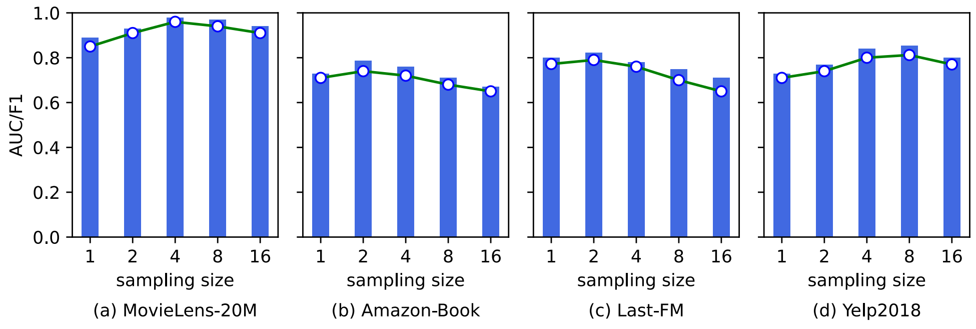

Fig. 3 Impact of neighbor sampling size

interaction to obtain sufficient feature interaction. After that, we adopt CNN to extract the item side features from the entity-relation interaction matrix efficiently. Due to user preferences are not fixed but will be different for different entities, using the user preference matrix to project the item embedding to the user's latent space lastly. The item's feature can be used to continuously learn the preference information of the user, and the preference matrix efficiently solved different preferences for different entities. So, the potential probability of user click can be efficiently predicted based on the projected item embedding under the user preference matrix.

To evaluate the computational efficiency, we conduct a series of experiments to compare the time cost of our model with other methods. These experiments are conducted on the system of ubuntu16.04 with a hardware configuration of Intel Xeon 4210 CPU and NVIDIA GeForce GTX2080Ti. All of the methods are conducted with 200 iterations. The results are shown in Table 4. From the result we can see, the time costs of KGFER for all datasets are lesser than the compared methods. It may be because of the following reason.

Compared to the full connection in compared methods, the CNN in our model reduces the computation complexity of the extract features operation. Moreover, due to the preprocess on the entity and relationship embedding based on TransR and the projection operation in the user latent space, we only employ one feedforward neural layer in the aggregate layer to fuse the entity features. In addition, the nearest neighbor sampling strategy in our algorithm further reduces the computational complexity of the KGFER. The CGAT method, which has a similar sampling strategy and adopts CNN to extract features, has relatively low time cost. And the compared methods all employ MLP to extract features from the result of inner product, summation, or conjunction operation, so their time cost is higher.

Therefore, the KGFER model is a very competitive method for recommendation applications.

4.4 Study of KGFER(RQ2)

We employ an ablation study to get deep insights KGFER. The ablation study is divided into four parts. The effects of the entity-relation fusion layer and user preference matrix we designed are discussed in the first parts. After that, we examine how neighbor sampling size, high order connectivity, and different aggregators affect the performance.

•*Effect of the fusion layer and UMP.*

To show just the fusion layer is added without UMP, we remove the TransU layer from our proposed model. The model id is named 3. In the same way, we remove the fusion layer from the original model and name the ablation model id as 2. The method, whose both the fusion and TransU layer are removed, is 1. Our proposed model id is 4. All experimental results are shown in Table 5. The hyperparameters of all the above variant methods, except for the removed components, have the same values as KGFER. We repeat each experiment 5 times, and the average performance results are reported in Table 5.

We observe that the overall trends of AUC and F1 become better with the addition of TransU and fusion layer. Besides, compared with the method which only adds the TransU layer, the improvement of the method which only adds a fusion layer is more obvious. The main reason is that the fusion layer can explore more side feature information of the user interaction

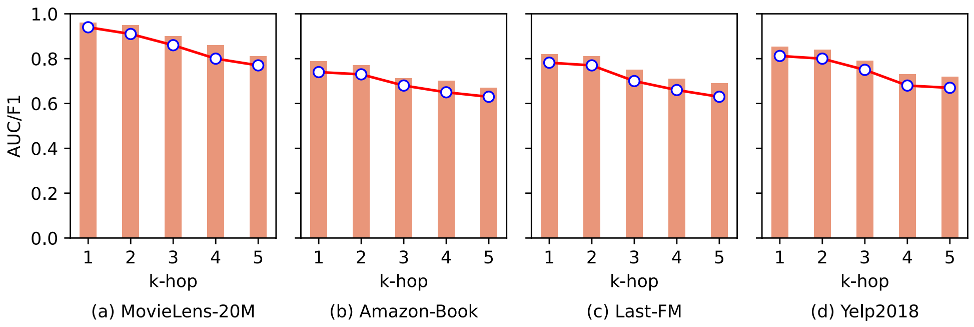

Fig. 4 Impact of high-order connectivity

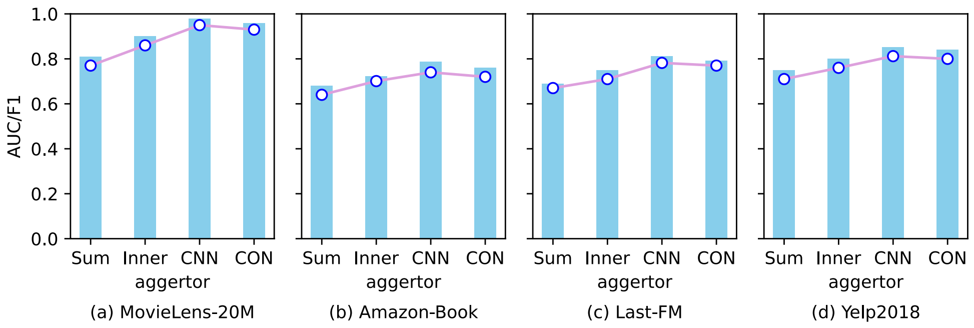

Fig. 5 Impact of different fusing aggregator

entities from the knowledge graph. While the separate user preference matrix, which does not incorporate into the knowledge graph, lacks side feature information of the interaction item, so user preferences could not be mine sufficiently. Once the fusion layer and TransU are combined, the performance is improved.

•*Effect of neighbor sampling size.* Consider a candidate pair of user-entity interaction $\langle u, e\rangle$, the size of e's neighbor varies, so we uniformly sample a fixed-size set of neighbors instead of using its full neighbors. The fixed-size S is a configurable constant. We vary the sampling size to study the efficiency of AUC. In particular, the sampling size is varied in the list of $\{1, 2, 4, 8, 16\}$. If the number of the neighbor is less than the sampling size, a repeated sampling strategy is employed. Figure 3 shows the results and we observe that KGFER achieves the best performance when S =2 or 4. It may be because of the following reason. There is at least one associated entity, and there is a maximum of 16 in the lined knowledge graph of the test dataset. 81% of the user interactive items in the MovieLens-20M have a degree 4. 76% of the interactive items in the Amazon-Book have a degree 2. 73% of the interactive items have a degree 2 in the Last-FM. In the Yelp2018, 72% of the interactive items have a degree 5. So a too small S does not have enough capacity to incorporate neighborhood information, while a too large S is prone to be misled by noises and increase computational complexity.

• *Effect of high-order connectivity.* We further study how the numbers of hops affect the performance of KGFER. Based head-relation1-tail1-realtion2-tail2 in knowledge graph, the k is varied in the range of $\{1, 2, 3, 4, 5\}$. Figure 4 shows the results for each dataset. We find that the KGFER is sensitive to k. We observe the occurrence of obvious performances drop when k=3. This may be because a larger k brings more noises to KGFER. This is also in accordance with our intuition, since a too long relation chain makes little sense. After accurately capturing user preferences, only 1-hop is enough for real cases according to the experiment results.

• *Effect of different fusing aggregator.* Lastly, we investigate the influence of different aggregators(Inner product, Sum, CNN, and Conjunction) in the fusing layer on the performance of KGFER. Figure 5 illustrates the AUC and F1 values of KGFER for different aggregators. From the results we find that: The CNN aggregator obviously boosts the AUC and F1. Sum aggregator performs worst, and Inner-aggregator performs best in general. This may be because the sum-aggregator losing useful information while add relation and entity embedding. Inner-product performs worse than Concatenate-aggregator due to the concatenate operator increase embedding dimension.

5 Conclusion

In this work, we explore user's preferences via knowledge graph based fusing entity-relation. We also implement the proposed model in a minibatch fashion, which is an end-to-end framework and able to operate on large datasets and knowledge graphs. It should be noted, the computational complexity is reduced due to the model sample from the direct neighbors for the entity which user interacts in KG as their receptive field. Our comparative experiments show that KGFER outperforms state-of-the-art baselines on four datasets about book, movie, music, and yelp2018. The result also indicates that KGFER can explore effective user preference and learn more well representations of user and entity.

There are three directions for our future work. (1)In this work, we focus on modeling item-end in user-item interaction based on a knowledge graph. Besides KG, other structural information such as social networks, indeed exists. By integrating social networks with KG, we can explore how user-end influence affects the recommendation. (2)We aim to further reduce the model complexity to design an algorithm to more well combine KG. (3)Based on recent interacted items and KG, predicting what items a user will interest in the next time is an exciting direction.

Acknowledgements This work is supported by the Natural Science Foundation of Chongqing (No.cstc2019jscx-msxmX0349), the Fundamental Research Funds for the Central Universities (No.2020CDCGTX055) and the Major Natural Science Funds of Chongqing Education Commission(No.KJZD-M201901401).

References

1. Cui Z, Xu X, Fei XUE, Cai X, Cao Y, Zhang W, Chen J (2020) Personalized recommendation system based on collaborative filtering for IoT scenarios. IEEE Trans Serv Comput 13(4):685–695
2. Nassar N, Jafar A, Rahhal Y (2020) A novel deep multi-criteria collaborative filtering model for recommendation system. Knowl-Based Syst 187:104811
3. Wang H, Wang Z, Hu S, Xu X, Chen S, Tu Z (2019) DUSKG: A fine-grained knowledge graph for effective personalized service recommendation. Futur Gener Comput Syst 100:600–617
4. Pan Y, He F, Yu H (2019) A novel enhanced collaborative autoencoder with knowledge distillation for top-N recommender systems. Neurocomputing 332:137–148
5. Tang H, Zhao G, Bu X, Qian X (2021) Dynamic evolution of multi-graph based collaborative filtering for recommendation systems. Knowl-Based Syst 228:107251
6. Pan Y, He F, Yu H, Li H (2020) Learning adaptive trust strength with user roles of truster and trustee for trust-aware recommender systems. Appl Intell 50(2):314–327
7. Chen X, Zhang Y, Xu H, Qin Z, Zha H (2019) Adversarial distillation for efficient recommendation with external knowledge. ACM Trans Inf Syst 37(1):12.1-12.28
8. Wang H, Zhao M, Xie X, Li W, Guo M (2019) Knowledge graph convolutional networks for recommender systems. The World Wide Web conference
9. Mezni H, Benslimane D, Bellatreche L (2021) Context-aware service recommendation based on knowledge graph embedding. IEEE Trans Knowl Data Eng 99:1–1
10. Wang H, Zhang F, Wang J, Zhao M, Guo M (2018) Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. Proceedings of the 27th ACM international conference on information and knowledge management
11. Nathani D, Chauhan J, Sharma C, Kaul M (2019) Learning attention-based embeddings for relation prediction in knowledge graphs. arXiv:1906.01195
12. Hu L, Xu S, Li C, Yang C, Shi C, Duan N, Xie X (2020) Graph neural news recommendation with unsupervised preference disentanglement. Proceedings of the 58th annual meeting of the association for computational linguistics
13. Sha X, Sun Z, Zhang J (2019) Attentive knowledge graph embedding for personalized recommendation. arXiv:1910.08288
14. Xie F, Zheng A, Chen L, Zheng Z (2020) Attentive meta-graph embedding for item recommendation in heterogeneous information networks - ScienceDirect. Knowl-Based Syst 211
15. Yang Z, Dong S (2020) HAGERec: Hierarchical attention graph convolutional network incorporating knowledge graph for explainable recommendation. Knowl-Based Syst 204:106194
16. Mehta R, Rana K (2017) A review on matrix factorization techniques in recommender systems. 2017 2nd international conference on communication systems, computing and IT applications (CSCITA), IEEE
17. Luo X, Zhou M, Li S, Shang M (2017) An inherently nonnegative latent factor model for high-dimensional and sparse matrices from industrial applications. IEEE Trans Ind Inf 14(5):2011–2022
18. Wang Q, Chen S, Luo X (2019) An adaptive latent factor model via particle swarm optimization. Neurocomputing 369:176–184
19. Pham MQ, Nguyen TTS, Do PMT, Kozierkiewicz A (2020) Incremental SVD-based collaborative filtering enhanced with diversity for personalized recommendation. International conference on computational collective intelligence, Springer, Cham
20. Sahoo AK, Pradhan C, Mishra BSP (2019) SVD based privacy preserving recommendation model using optimized hybrid item-based collaborative filtering. 2019 international conference on communication and signal processing (ICCSP), IEEE
21. Palumbo E, Monti D, Rizzo G, Troncy R, Baralis E (2020) Entity2rec: Property-specific knowledge graph embeddings for item recommendation. Expert Syst Appl 151:113235
22. Wang X, Xu Y, He X, Cao Y, Wang M, Chua TS (2020) Reinforced negative sampling over knowledge graph for recommendation. Proceedings of The Web Conference 2020
23. Shi D, Wang T, Xing H, Xu H (2020) A learning path recommendation model based on a multidimensional knowledge graph framework for e-learning. Knowl-Based Syst 195:105618
24. Wu B, He X, Sun Z, Chen L, Ye Y (2019) ATM: an attentive translation model for next-item recommendation. IEEE Trans Ind Inf 16(3):1448–1459
25. Hao J, Dun Y, Zhao G, Wu Y, Qian X (2021) Annular-graph attention model for personalized sequential recommendation. IEEE Trans Multimed
26. Najafabadi MK, Mohamed AHJ, Mahrin MN (2019) A survey on data mining techniques in recommender systems. Soft Comput 23(2):627–654
27. Xu K, Zheng X, Cai Y, Min H, Gao Z, Zhu B, Xie H, Wong T (2018) Improving user recommendation by extracting social topics and interest topics of users in uni-directional social networks. Knowl-Based Syst 140:120–133
28. Chen R, Hua Q, Wang B, Zheng M, Guan W, Ji X, Gao Q, Kong X (2019) A novel social recommendation method fusing user's social status and homophily based on matrix factorization techniques. IEEE Access 7:18783–18798
29. Wu L, Sun P, Hong R, Fu Y, Wang X, Wang M (2018) Social-GCN: an efficient graph convolutional network based model for social recommendation. arXiv:1811.02815
30. Yiteng P, He F, Yu H (2020) Learning social representations with deep autoencoder for recommender system. World Wide Web 23(4):2259–2279

31. Yiteng P, He F, Yu H (2020) A correlative denoising autoencoder to model social influence for top-N recommender system. Front Comput Sci 14(3):1–13
32. Zhang F, Yuan NJ, Lian D, Xie X, Ma W-Y (2016) Collaborative knowledge base embedding for recommender systems. Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining
33. Lin Y, Liu Z, Sun M, Liu Y, Zhu X (2015) Learning entity and relation embeddings for knowledge graph completion. Proceedings of the AAAI conference on artificial intelligence, vol 29, no 1
34. Wang H, Zhang F, Xie X, Guo M (2018) DKN: Deep knowledge-aware network for news recommendation. Proceedings of the 2018 World Wide Web conference
35. Ji G, He S, Xu L, Liu K, Zhao J (2015) Knowledge graph embedding via dynamic mapping matrix. Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: Long papers)
36. Cao Y, Wang X, He X, Hu Z, Chua T-S (2019) Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences. The World Wide Web conference
37. Zhu Q, Zhou X, Zhang P, Shi Y (2019) A neural translating general hyperplane for knowledge graph embedding. J Comput Sci 30:108–117
38. Xin X, He X, Zhang Y, Zhang Y, Jose J (2019) Relational collaborative filtering: Modeling multiple item relations for recommendation. Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval
39. Wang H, Zhang F, Zhao M, Li W, Xie X, Guo M (2019) Multi-task feature learning for knowledge graph enhanced recommendation. The World Wide Web Conference
40. 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
41. Zhao Z, Zhang X, Zhou H, Li C, Gong M (2020) HetNERec: Heterogeneous network embedding based recommendation. Knowl-Based Syst 204:106218
42. Hengrui Z, McAuley J (2020) Stacked mixed-order graph convolutional networks for collaborative filtering, Proceedings of the 2020 SIAM international conference on data mining. Society for industrial and applied mathematics
43. Zhiwei G, Wang H (2020) A deep graph neural network-based mechanism for social recommendations. IEEE Trans Indl Inf 17(4):2776–2783
44. Wang X, Wang D, Xu C, He X, Cao Y, Chua T-S (2019) Explainable reasoning over knowledge graphs for recommendation. Proceedings of the AAAI conference on artificial intelligence, vol 33, no 01
45. Wang X, He X, Cao Y, Liu M, Chua T-S (2019) KGAT: Knowledge graph attention network for recommendation. Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining
46. 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 Trans Inf Syst 37(3):1–26
47. Zhao WX, He G, Yang K, Dou H, Huang J, Ouyang S, Wen J-R (2019) Kb4rec: A data set for linking knowledge bases with recommender systems. Data Intell 1(2):121–136
48. Xing S, Liu F, Wang Q, Zhao X, Li T (2019) Content-aware point-of-interest recommendation based on convolutional neural network. Appl Intell 49(3):858–871
49. Nisha CC, Mohan A (2019) A social recommender system using deep architecture and network embedding. Appl Intell 49(5):1937–1953
50. Chen X, Jia S, Xiang Y (2020) A review: Knowledge reasoning over knowledge graph. Expert Syst Appl 141:112948
51. Sang L, Xu M, Qian S, Wu X (2021) Knowledge graph enhanced neural collaborative recommendation. Expert Syst Appl 164:113992
52. Hui B, Zhang L, Zhou X, Wen X, Nian Y (2021) Personalized recommendation system based on knowledge embedding and historical behavior. Appl Intell :1–13
53. Yang S, Liu Y, Xu Y, Miao C, Wu M, Zhang J (2021) Contextualized graph attention network for recommendation with item knowledge graph. IEEE Trans Knowl Data Eng

Huilian Fan received the B.S. and M.S degree in computing application from Chongqing University, China, in 2000 and 2006 respectively. Since 2012, he has been a professor with the college of computer Engineering Department, Yangtze Normal University, Chongqing, China. His main research interests include social computing, network science and intelligent computing.

Yuanchang Zhong received the B.S. degree in electrical engineering from the Changchun University of Science and Technology, Changchun, China, in 1988, and the M.S. degree in communication engineering and the Ph.D. degree in mechanical electronic engineering from Chongqing University, Chongqing, China, in 2002 and 2009, respectively, where he is currently a Professor with the College of Communication Engineering. His current research interests include wireless sensor networks, MEMS, and RFID technology.

Guangpu Zeng received the B.S. degree from Sichuan University, China, in 2000. He is currently an assistant professor in the school of computer engineering in Yangtze Normal University, Chongqing,China. His main research interests include network services, recommender systems and mobile computing.

Chenhao Ge received his B.S.degree in Communication Engineering from the School of Computer and Information of Hefei University of Technology,China,in 2013. He is currently pursuing his M.S.degree in Microelectronics and Communication Engineering at Chongqing University, China. His research interests include wreless energy transmission technology and laser technology.