CKEN: Collaborative Knowledge-Aware Enhanced Network for Recommender Systems

Wei Zeng[1,2], Jiwei Qin[1,2](✉), and Xiaole Wang[1,2]

[1] School of Information Science and Engineering, Xinjiang University, Urumqi 830046, China
jwqin_xju@163.com

[2] Key Laboratory of Signal Detection and Processing, Xinjiang Uygur Autonomous Region, Xinjiang University, Urumqi 830046, China

Abstract. Knowledge graph are widely used as a kind of side information to alleviate the data sparsity in collaborative filtering. However, a majority of existing recommendation methods incorporating knowledge graph emphasize how to encode knowledge associations in the knowledge graph, while ignoring the key collaboration information implied in the user-item history interactions and the shared information between the item and the knowledge graph entities, resulting in insufficient embedding representation of users and items. To that end, we propose a collaborative knowledge-aware enhanced network (CKEN). Specifically, CKEN uses a collaborative propagation approach to explicitly encode user-item collaborative information and then allows it to be propagated in the knowledge graph, and applies an attention mechanism to obtain contributions from neighbors with different knowledge during knowledge propagation. In addition, considering the association relationship between items and knowledge graph entities, this paper designs a feature interaction layer to facilitate feature sharing between items in the recommender systems and entities in the knowledge graph in order to further enrich the latent vector representation of items. Numerous experiments on several real-world scenario datasets have shown that CKEN's recommendations are significantly better than several other best-of-class baselines.

Keywords: Recommender systems · Collaborative propagation · Feature interaction · Knowledge graph

1 Introduction

Recommender systems (RS) aim to suggest certain items to users to meet their personalized preferences, thus alleviating the impact of information overload. One of the more widely used recommendation methods is collaborative filtering (CF) [5], which extracts users' preferences on items for recommendation by analyzing their historical interactions. However, collaborative filtering recommendation methods often face data sparsity and cold start problems. To tackle these problems, some researchers have introduced various types of side information in recommender systems, e.g., user social networks [4], user/item attributes features [11], and multimedia information (e.g., text [10], images [21]). Among a wide variety of side information, knowledge graph (KG) contain rich facts and connections about items have received increasing attention [1,14,17,18]. Figure 1 shows a movie recommendation scenario based on KG, the KG contains entities such as directors, actors and subject genres.

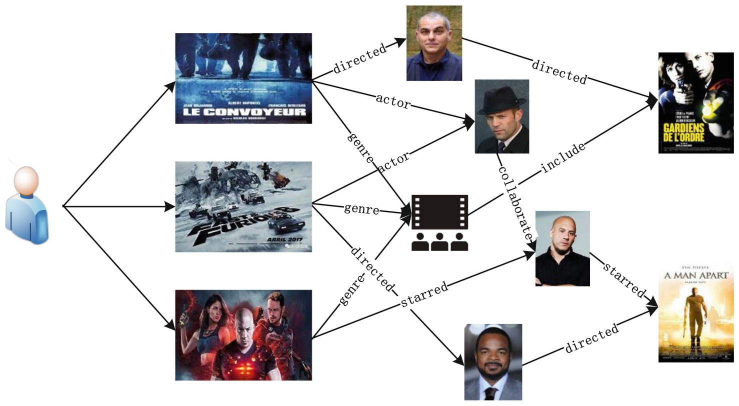

Fig. 1. Illustration of KG for movie recommendation.

Central to the combination of KG and RS is how to leverage the rich semantic information in KG to enhance the representation of users and items. For example, PER [20] and FMG [22] consider KG as heterogeneous information networks, representing the connectivity between users and items by withdrawing features of meta-paths. However, PER and FMG require manual design of meta-paths and the designed meta-paths are hardly optimal, which makes it difficult to be applied in general-purpose recommendation scenarios. DKN [13] designs a Convolutional Neural Network (CNN) framework that combines semantic and knowledge information from different channels to recommend news for users. However, since DKN are usually trained in a separate manner for entity embedding, Knowledge Graph Embedding (KGE) is not optimized for a specific task, which can lead to suboptimal results. RippleNet [12] is the first to combine entity embedding and path propagation of KG recommendation methods to propagate users' preferences in KG paths. However, RippleNet relies heavily on the representation of attributes, i.e., the construction of the KG, and the importance of the relations between entities is difficult to be described. CKE [21] extracts features from the structured knowledge, textual knowledge, and visual knowledge of the item, respectively, and then combines them with the CF module for joint learning, but the KGE module in the CKE model is more suitable for KG completion and link prediction tasks. MKR [15] is an end-to-end model that uses KGE tasks to assist recommendation tasks, which automatically share latent features between items and entities in the KG through the information sharing module designed in the model, but it ignores the user-side and item-side in the KG of propagating information, resulting in insufficient user and item embedding.

To tackle the limitations of extant knowledge graph recommendation methods, collaborative knowledge-aware enhanced network (CKEN) is proposed, which is an end-to-end model. Specifically, CKEN has two designs: 1) seamlessly combines collaborative information latent in user-item history interactions and knowledge associations in KG through the collaborative propagation layer (CPL) and the knowledge graph propagation layer (KPL), and then generates different attention weights of tail entities through a knowledge-aware attention mechanism to extract more valuable knowledge associations. 2) A feature interaction layer (FIL) is designed to explicitly model the high-level interactions between item features and entity features using a cross&compress approach and to control the knowledge transfer between item and KG entities. Through the FIL, the features of items and entities can supplement each other, which helps to obtain a richer latent vector representation of items, while avoiding fitting noise and improving generality.

In conclusion, the main contributions of CKEN are as follows :

1. We highlight the importance of the collaborative information latent in the user-item history interactions and the shared information between items and KG entities when obtaining embedding of users and items.
2. We propose CKEN, which is an end-to-end recommendation model. The design of CPL and KPL effectively combines the collaboration information latent in user-item history interactions with the side semantic associations in KG. The design of FIL enables feature sharing between items in the implicit interaction matrix and KG entities, further enhancing the latent vector representation of items.
3. We conducted extensive experiments on three realistic recommended scenarios demonstrate that CKEN surpasses compelling best-in-class baselines.

2 Problem Formulation

In the recommendation task of CKEN, we assume there exists a user set $U = \{u_1, u_{2,} \cdots, u_M\}$ consisting of M users and an item set $V = \{v_1, v_{2,} \cdots, v_N\}$ consisting of N items. Based on the implicit feedback from users in their history of actions (e.g. clicks, views and purchases), a user-item implicit feedback matrix $\mathrm{Y}^{M \times N}$ can be obtained, where $y_{uv} = 1$ indicates that the user u has clicked or viewed the item v, otherwise $y_{uv} = 0$. In addition, this paper constructs side information in the form of KG, which is usually represented using a triple, i.e.$G = \{(h, r, t) \,|h, t \in E, r \in R\}$, where $E = \{e_1, e_2, \cdots, e_n\}$ and $R = \{r_1, r_2, \cdots, r_n\}$ are the sets of entities and relations in the KG, and each KG triple (h, r, t) means that the head entity h and the tail entity t are connected by the relation r. And, we use the $A = \{(v, e) \,|v \in V, e \in E\}$ set to represent the alignment of items and entities.

Having obtained the implicit feedback matrix $Y^{M \times N}$ and konwledge graph G, our task is to obtain the probability that user u will click on an item v that he has not interacted with. Specifically, the goal of this paper is to learn the click-through rate(CTR) prediction function $\hat{y}_{uv} = P(u, v|\theta, G)$, where $\hat{y}_{uv}$ is the probability of obtaining and θ is the parameter of the function P.

3 Method

As shown in Fig. 2, CKEN has four main design components: collaborative propagation layer, knowledge graph propagation layer, feature interaction layer, and prediction layer. In the following subsections, we will elaborate on the different parts of CKEN.

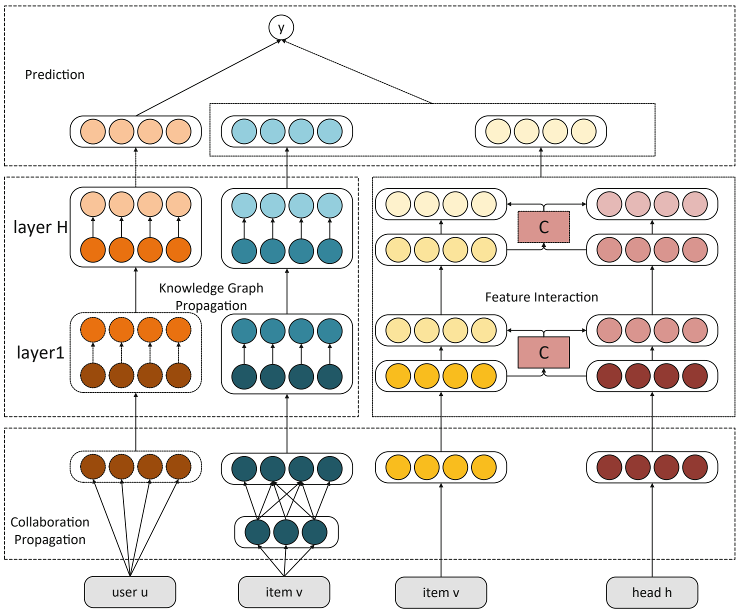

Fig. 2. Illustration of the proposed model CKEN that consists of four components: collaborative propagation layer, knowledge graph propagation layer, feature interaction layer, and prediction layer.

3.1 Collaborative Propagation Layer

According to the previous theory [19], the historical interactions of a known user u can characterize user's preferences to some extent. Therefore, we uses the set of related items V_u (first-order neighbors of the user) that user u historical interactions to represent user u, defined as follows:

$$V_u = \{v_u | v \in \{v | y_{uv} = 1\}\} \tag{1}$$

Then, we find the entities corresponding to the relevant set of items V_u based on the alignment set A of items and entities, and use them as the initial seed set for user u to propagate interest in KG. The definition is shown below:

$$\varepsilon_u^0 = \{e | (v, e) \in A \ and \ v \in V_u\} \tag{2}$$

The second-order neighbors of an item refer to other items that have been clicked by users who have generated interaction behavior with the same item v. Since user u will click on other items that are similar to the item because of their behavioral preferences, the second-order neighbors of an item can also be and are used to enhance the item representation. In this paper, we define collaborative neighbors for user history interaction items and set the set of collaborative items V_v of item v, which is defined as below:

$$V_v = \{v_u | u \in \{u | y_{uv} = 1\} \ and \ y_{uv_u} = 1\} \tag{3}$$

Based on the alignment set A between the items and KG entities, we can likewise find the set of entities corresponding to the set of collaborative items V_v of item v in KG and use it as the seed set for the propagation of item v, which is defined as shown below:

$$\varepsilon_v^0 = \{e | (v_u, e) \in A \ and \ v_u \in V_v\} \tag{4}$$

The CPL can effectively encode first-order neighbors expressing user preferences and second-order neighbors expressing item features explicitly into the seed set, thus enhancing the embedded representation of users/items.

3.2 Knowledge Graph Propagation Layer

The KPL mainly consists of two parts: knowledge graph propagation and knowledge-aware attention embedding. The knowledge graph propagation part mainly propagates knowledge associations along the links in KG in order to recursively supplement the supporting information for users and items. The knowledge-aware attention embedding part mainly learns the different weights of the same entity under different relations and generates weighted representations of the entities. They are described in detail as follows:

1) Knowledge graph propagation. The entities in KG are connected by some kind of relationship. The propagation along the connectivity of KG entities acquires extensions of the initial set of entities and triples of different distances, which can be considered as extensions of user preferences and item characteristics. The set of entities propagated by user u and item v in KG can be recursively defined as:

$$\varepsilon_*^k = \left\{t | (h, r, t) \in G \ and \ h \in \varepsilon_*^{k-1}\right\}, k = 1, 2, \cdots, H \tag{5}$$

where k denotes the distance from ε_*^0 and the symbol $*$ denotes user u or item v. Based on the above entity set definition, the set of *k-th* triples for user u and item v can be expressed as:

$$S_*^k = \left\{(h,r,t) \,|\, (h,r,t) \in G \textit{ and } h \in \varepsilon_*^{k-1}\right\}, k = 1, 2, \cdots, H \tag{6}$$

The initial set of entities obtained through collaborative propagation is similar to the center of a water ripple that keeps spreading outward. After the deep propagation of the KG, the CKEN model can obtain the complementary entity information of users and items at different distances in KG, thus greatly enriching the embedding representation of users and items.

2) Knowledge-aware attention embedding. In KG, when tail entity has different implications and potential vector representations when they are connected to different entities and relations. Based on this, this paper uses a knowledge-aware attention embedding method [9,19] to produce different attention scores for the tail entities. The embedding ∂_i of the tail entities is calculated as below:

$$\partial_i = \pi\left(e_i^h, r_i\right) e_i^t \tag{7}$$

where e_i^h is the embedding of the head entity of the *ith* triple, r_i is the embedding of the *ith* triple, and e_i^t is the embedding of the *ith* triple tail entity. $\pi\left(e_i^h, r_i\right)$ denotes the attention scores generated by the different triples. In this paper, the implementation of the function $\pi(\cdot)$ is similar to the literature [9] and is calculated as follows:

$$O_0 = ReLU\left(W_0\left(\left[e_i^h, r_i\right]\right) + b_0\right) \tag{8}$$

$$\pi\left(e_i^h, r_i\right) = \sigma\left(W_2 ReLU\left(W_1 O_0 + b_1\right) + b_2\right) \tag{9}$$

where σ denotes the Sigmoid [3] activation function, $[*]$ denotes the concatenation operation, W_* denotes the trainable weight matrix, and b_* denotes the bias vector. In this paper, the softmax [7] function is used to perform the normalization operation, which is calculated as follows:

$$\pi\left(e_i^h, r_i\right) = \texttt{softmax}(\pi(e_i^h, r_i)) = \frac{\exp\left(\pi\left(e_i^h, r_i\right)\right)}{\sum_{(h',r',t')\in s_*^k} \exp\left(\pi\left(e_i^{h'}, r_i^{'}\right)\right)} \tag{10}$$

where S_*^k denotes the *kth* level triple of the user or item. Through the knowledge-aware attention mechanism, the CKEN model can reasonably assign different weights to different neighboring tail entities, so as to capture the knowledge association more effectively. Finally, this paper obtains the latent representation of the *k-th* level triplet set of user/item by aggregating the attentional embedding representations of the tail entities, which is calculated as follows:

$$e_*^k = \sum_{i=1}^{\left|s_*^k\right|} \partial_i^k, k = 1, 2, \cdots, H \tag{11}$$

where $\left|S_*^k\right|$ denotes the number of triples in the set S_*^k at level k.

We know that the entities in the seed set are best used to characterize the original feature embedding of users and items. Therefore, in this paper, we add the initial entity sets for users and items, and the formula is expressed as follows:

$$e_*^0 = \frac{\sum_{e \in \varepsilon_*^0} e}{\left|\varepsilon_*^0\right|} \tag{12}$$

3.3 Feature Interaction Layer

In particular, item v itself has the key feature information of its original representation and it is related to one or more entities in KG and shares similar features in the latent feature space [6]. Therefore, in order to fully exploit the feature information of item v itself, we design FIL in CKEN model, which allows item v to interact with entity e in KG by cross&compress operation [15] to achieve feature sharing. The feature interaction matrix is shown as follows:

$$C_l = v_l e_l^T = \begin{bmatrix} v_l^{(1)} e_l^{(1)} & \cdots & v_l^{(1)} e_l^{(d)} \\ \vdots & \ddots & \vdots \\ v_l^{(d)} e_l^{(1)} & \cdots & v_l^{(d)} e_l^{(d)} \end{bmatrix} \tag{13}$$

where l is the number of layers of FIL and d is the hidden layer dimension. After calculation, the feature interaction matrix models all possible interactions $v_l^{(i)} e_l^{(i)}, \forall (i, j) \in \{1, \cdots, d\}^2$ of the item v with the entity e explicitly. The feature interaction matrix is then multiplied with the weight matrix to project them into the latent representation space, which in turn yields the input for the next level:

$$v_{l+1} = C_l W_l^{VV} + C_l^T W_l^{EV} + b_l^V = v_l e_l^T W_l^{VV} + e_l v_l^T W_l^{EV} + b_l^V \tag{14}$$

$$e_{l+1} = C_l W_l^{VE} + C_l^T W_l^{EE} + b_l^E = v_l e_l^T W_l^{VE} + e_l v_l^T W_l^{EE} + b_l^E \tag{15}$$

where $W_l^{**} \in \Re^d$ and $b_l^{**} \in \Re^d$ denote the weight matrix and bias vector. After computation, the feature interaction matrix from $\Re^{d \times d}$ space is in turn projected back into the $\Re^d$ feature space by the weight vectors. Finally, for readability, the output of FIL can be expressed as:

$$[v_{l+1}, e_{l+1}] = C(v_l, e_l) \tag{16}$$

In this paper, we denote the output items and entities in FIL by the symbols $[v]$ and $[e]$. After multiple layers of feature interactions, the final outputs of items and entities can be represented as:

$$v_e = C(v, e)[v] \tag{17}$$

$$h_e = C\left(v, e\right)\left[e\right] \tag{18}$$

With the FIL, CKEN can automatically coordinate the transfer of knowledge and fully explore the correlation between items and entities.

In CKEN, the depth of FIL should not be too large because in features tend to shift from general to specific in deep networks, and the migrability of features decreases significantly with increasing noise [15].

3.4 Prediction Layer

After knowledge-aware attention embedding and feature interaction, we set up representation sets for user u and item v that contain knowledge propagation-based attention-weighted representations and original relevant entity representations:

$$D_u = \left\{e_u^0, e_u^1, \cdots, e_u^H\right\} \tag{19}$$

$$D_v = \left\{v_e, e_v^0, e_v^1, \cdots, e_v^H\right\} \tag{20}$$

The representation in each layer of the propagation process can be interpreted as the potential impact of layering, which emphasizes different higher-order connectivity and preference similarities. In this paper, a summation aggregator is used to compute the final aggregation vector. Before applying the nonlinear transformation, the summation aggregator needs to sum the vectors for each layer:

$$agg_{sum}^{*} = \sum_{e_* \in D_*} e_* \tag{21}$$

Ultimately, e_u is used to represent the aggregation vector of users. Similarly, e_v denotes the aggregation vector of items. Then, we multiply e_v and e_u to obtain the prediction scores:

$$\hat{y}_{uv} = \sigma(e_u^T e_v) \tag{22}$$

where the activation function σ is Sigmoid [3].

3.5 Model Learning

The overall loss function for CKEN is shown below:

$$\begin{gathered} L = L_{RS} + L_{KG} + L_{REG} \\ = \sum_{u \in U, v \in V} J\left(\hat{y}_{uv}, y_{uv}\right) \\ -\lambda_1 \left(\sum_{(h,r,t) \in G} score\left(h, r, t\right) - \sum_{(h',r,t') \notin G} score\left(h', r, t'\right)\right) + \lambda_2 \left\|W\right\|_2^2 \end{gathered} \tag{23}$$

where $J(\cdot)$ denotes the cross-entropy loss and the task of the second term is to add the score distance between all true and false triples. The formula for calculating the triple score is shown below:

$$score\,(h, r, t) = \sigma\left(t^T \hat{t}\right) = \sigma\left(t^T M^K\left([h_e, r_e]\right)\right) \tag{24}$$

where t denotes the true tail entity vector, $\hat{t}$ denotes the prediction tail entity vector, r_e denotes the feature vector of the relation, $[*]$ denotes the connection operation, M^K denotes the MLP neural network with K layers, and σ is the Sigmoid [3].

The last term of the overall loss function is a regularization term for preventing overfitting, λ_1 and λ_2 are the balance parameters. And, we use the grid search method to pick parameters, with the following detailed parameter settings: $\lambda_1 = 0.1$, $\lambda_2 = 10^{-6}$.

4 Experiments

4.1 Dataset

To evaluate CKEN, we conduct experiments under several public available datasets: MovieLens-1M[1], Book-Crossing[2] and Last.FM[3]. We used MovieLens-1M dataset scores of 4 and 5 as a positive feedback signal, and all interactions in Book-Crossing and Last.FM are used as positive feedback signals. Table 1 shows the detailed statistical information.

Table 1. Detailed statistics of the three public datasets.

	MovieLens-1M	Book-Crossing	Last.FM
#users	6,036	17,860	1,872
#items	2,445	14,910	3,846
#interactions	753,772	139,746	42,346
#edge types	12	25	60
#entity	182,011	77,903	9,366
#relation	2483,990	303,000	31,036
#triplies	20,195	19,793	15,518
#sparsity level	0.948925	0.999447	0.994116

[1] https://grouplens.org/datasets/movielens/1m/.
[2] http://www.informatik.uni-freiburg.de/~cziegler/BX/.
[3] https://grouplens.org/datasets/hetrec-2011/.

4.2 Baselines

We have evaluated CKEN for performance comparison using the following baseline:

PER [20] uses meta-paths to characterize the relationship between users and items.

CKE [21] combines CF modules with knowledge from a variety of different sources to make recommendations.

DKN [13] combines entity and text embedding for news recommendations.

RippleNet [12] uses the user's a priori information to propagate user preferences in KG and enriches the user's representation in this way.

LibFM [8] is a feature-based decomposition model. In this paper, user IDs and item IDs, as well as related entity embeddings are concatenated as inputs to LibFM.

Wide & Deep [2] is a well-known model that trains a linear model jointly with a non-linear model to balance the overall model's ability to remember and generalise.

KGCN [16] is a recommendation model that fuses KG and GCN, which models only item-side representations.

MKR [15] is a multi-task learning framework that combines KG to assist in recommendation tasks by learning higher-order interactions between items and entities in the KG.

CKAN [19] is a recommendation model that fuses collaborative information and KG, which uses a heterogeneous propagation strategy to encode both knowledge attribute associations and user-item collaborative signals.

4.3 Experiments Setup

On the MovieLens-1M dataset, the number of FIL layers is 1, and the learning rates for the recommendation part and the knowledge graph part are 2×10^{-2} and 10^{-2}, respectively. On the Book-Crossing dataset, the number of FIL layers is 1, and the learning rates for the recommendation part and the knowledge graph part are 2×10^{-4} and 2×10^{-5}, respectively. On the Lasst.FM dataset, the number of FIL layers is 2, and the learning rates for the recommendation part and the knowledge graph part are 10^{-3} and 2×10^{-4}, respectively. In this paper, each dataset is divided into a training set, a validation set and a test set (6:2:2), and both AUC and ACC are used to evaluate the effectiveness of CTR prediction.

4.4 Performance Comparison

Table 2 shows the CTR prediction results for all models. The discussion is as follows:

- PER performs poorly because the manually designed meta-paths are hardly optimal, resulting in PER not being able to use heterogeneous information networks effectively.

Table 2. Detailed statistics of the three public datasets.

Model	MovieLens-1M		Book-Crossing		Last.FM	
	AUC	ACC	AUC	ACC	AUC	ACC
PER	0.710(−30.3%)	0.664(−28.0%)	0.623(−22.0%)	0.588(−19.4%)	0.633(−34.1%)	0.596(−30.5%)
CKE	0.801(−15.5%)	0.742(−14.6%)	0.671(−13.3%)	0.633(−10.9%)	0.744(−14.1%)	0.673(−15.6%)
DKN	0.655(−41.2%)	0.589(−44.3%)	0.622(−22.2%)	0.598(−17.4%)	0.602(−41.0%)	0.581(−33.9%)
RippleNet	0.913(−1.3%)	0.835(−1.8%)	0.729(−4.3%)	0.662(−6.0%)	0.768(−10.5%)	0.691(−12.6%)
LibFM	0.892(−3.7%)	0.812(−4.7%)	0.685(−10.9%)	0.640(−9.7%)	0.777(−9.3%)	0.709(−9.7%)
Wide&Deep	0.898(−3.0%)	0.820(−3.7%)	0.712(−6.7%)	0.624(−12.5%)	0.756(−12.3%)	0.688(−13.1%)
KGCN	0.919(−0.7%)	0.845(−0.6%)	0.694(−9.5%)	0.635(−10.6%)	0.794(−6.9%)	0.724(−7.5%)
MKR	0.917(−0.9%)	0.843(−0.8%)	0.734(−3.5%)	0.702(−0.0%)	0.797(−6.5%)	0.752(−3.5%)
CKAN	0.915(−1.1%)	0.840(−1.2%)	0.741(−2.6%)	0.655(−7.2%)	0.841(−1.0%)	0.744(−4.6%)
CKEN	**0.925**	**0.850**	**0.760**	**0.702**	**0.849**	**0.778**

- CKE outperforms PER because of the collaborative information incorporated in CKE.
- DKN performs the worst, which is because the names corresponding to the items in these datasets are too short and ambiguous, resulting in the inability to provide useful information.
- LibFM and Wide&Deep have a large improvement over PER and DKN, indicating that LibFM and Wide&Deep can better use the rich semantic relations in KG to improve the recommendation quality.
- MKR performs worse than CKEN because MKR does not take full advantage of the collaborative information as well as the propagation information in KG.
- CKAN performs less well than CKEN because the CKAN model ignores the shared information between items and knowledge graph entities. However, CKEN proposed in this paper seamlessly combines the collaborative information and knowledge associations in the KG, while further designing the FIL to mine the shared information and enrich the feature representation of items.
- Overall, our proposed CKEN performed the best on all three datasets. Specifically, the AUC increases by 0.7%, 2.6% and 1.0% in Movielens-1M, Book-Crossing and Last.FM datasets. This also proves the superiority of CKEN.

4.5 Study of CKEN

Effect of Dimension of Embedding. To facilitate the calculation, we set the embedding of entities and relationships in KG to the same dimensionality and study the effect of dimensionality on CKEN. From Fig. 3, we can see that increasing the embedding dimension in some range can enhance the recommendation effect of CKEN, but if the dimension is too large, the performance will decrease. The reason for this phenomenon is that when the dimensionality is too large, overfitting may occur.

Effect of Depths of Layer. In this paper, we choose different KPL depths to observe the variation of CKEN performance. Table 3 shows the experimental results for different depth layers, from which we can see that the model performs best when the propagation depth is 2 on MovieLens-1M and Book-Crossing

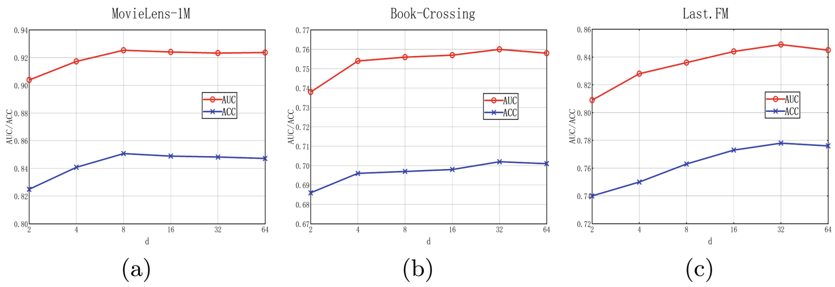

Fig. 3. The result of AU C w.r.t. different dimensions on all three datasets.

Table 3. The AUC results at different depths of layer.

Depth	MovieLens-1M	Book-Crossing	Last.FM
1	0.9248	0.7513	0.8426
2	**0.9254**	**0.7600**	0.8451
3	0.9251	0.7542	**0.8498**

Table 4. The AUC results for the effects of FIL, CPL, and knowledge perception attention mechanisms.

Model	MovieLens-1M		Book-Crossing		Last.FM	
	AUC	ACC	AUC	ACC	AUC	ACC
$CKEN_{w/o\ FIL}$	0.9239	0.8486	0.7224	0.6643	0.8160	0.7421
$CKEN_{w/o\ CPL}$	0.9242	0.8484	0.7286	0.6826	0.8115	0.7413
$CKEN_{w/o\ ATT}$	0.8775	0.7966	0.7427	0.6993	0.8069	0.7497
CKEN	**0.9254**	**0.8507**	**0.7600**	**0.7025**	**0.8498**	**0.7789**

Table 5. The AUC results for different initial entity set sizes on the Movielens-1M dataset.

User	Item				
	4	8	16	32	64
4	0.8961	0.9011	0.9002	0.9005	0.9009
8	0.9091	0.9099	0.9092	0.9116	0.9100
16	0.9151	0.9161	0.9170	0.9177	0.9171
32	0.9189	0.9204	0.9206	0.9212	0.9224
64	0.9228	0.9235	0.9229	**0.9254**	0.9251

Table 6. The AUC results for different initial entity set sizes on the Book-Crossing dataset.

User	Item				
	4	8	16	32	64
4	0.7453	0.7487	0.7476	0.7515	0.7517
8	0.7449	0.7528	0.7555	0.7564	0.7583
16	0.7468	0.7543	0.7577	0.7548	0.7559
32	0.7514	0.7548	0.7572	0.7584	**0.7600**
64	0.7516	0.7529	0.7561	0.7579	0.7589

Table 7. The AUC results for different initial entity set sizes on the Last-FM dataset.

User	Item				
	4	8	16	32	64
4	0.8314	0.8392	0.8386	0.8399	0.8434
8	0.8349	0.8415	0.8397	0.8448	0.8437
16	0.8407	0.8435	0.8461	0.8447	0.8457
32	0.8383	0.8390	0.8449	**0.8498**	0.8475
64	0.8419	0.8455	0.8477	0.8457	0.8492

datasets. On the Last.FM, the model performs best when the propagation depth is 3. This is due to the fact that deeper propagation brings in side information and also brings in a lot of noise to affect the performance of CKEN.

Effect of FIL, CPL and Knowledge-Aware Attention Mechanisms. Table 4 shows the ablation experimental results of the CKEN model, from the results in the table, we can see that the CKEN is better than the $CKEN_{w/o\ FIL}$ without FIL and the $CKEN_{w/o\ CPL}$ without CPL, the reason for this phenomenon is that the CKEN model fully captures and incorporates the collaborative information as well as the shared information between entities and items, which helps to enrich the latent vector representation of items. In addition, we replaces the knowledge-aware attention mechanism with a mean-value approach to study its effect on the experimental results of the CKEN model, and the results show that the CKEN is better than the $CKEN_{w/o\ ATT}$, which further demonstrates that the knowledge-aware attention mechanism can encode important knowledge associations into embedded representations of users and items, thus improve their representation capabilities.

Effect of Initial Entity Set Size. We change the initial entity set size of users and items to explore the upper limit of CKEN performance. Table 5, Table 6 and Table 7 show the results of AUC for MovieLens-1M dataset, Book-Crossing dataset and Last.FM dataset. We summarize the results and find that: increasing the initial entity set size can boost the performance, but if the initial entity set is too large, the results will be decreased. The reason for this phenomenon is the different number of the set of related items and the set of collaborative items, resulting in different number of entities in the KG associated with the user and the item, which is one of the key factors affecting the performance of CKEN.

5 Conclusion and Future Work

This paper proposes CKEN, a collaborative knowledge-aware enhanced network for recommender systems. Specifically, CKEN employs a collaborative propagation strategy to construct user-item collaborative relationships, which are then seamlessly combined with knowledge associations in the KG, while a feature interaction layer is designed to explicitly model the higher-order interactions between item features and KG entity features to facilitate feature sharing. Extensive experiments on multiple datasets in the real world have demonstrated that CKEN has significant performance improvement over other similar recommendation models.

In the future, we plan to untangle the user-item interaction bipartite graph to distinguish the relationship between users and items at fine-grained intent, while using dependency relationships in combination with the KG as a way to obtain fine-grained user preferences.

Acknowledgements. This work was supported by the Science Fund for Outstanding Youth of Xinjiang Uygur Autonomous Region under Grant No. 2021D01E14, the National Science Foundation of China under Grant No. 61867006, the Major science and technology project of Xinjiang Uygur Autonomous Region under Grant No. 2020A03001, the Key Laboratory Open Project of Science and Technology Department of Xinjiang Uygur Autonomous Region named Research on video information intelligent processing technology for Xinjiang regional security.

References

1. Cao, Y., Wang, X., He, X., Hu, Z., Chua, T.S.: Unifying knowledge graph learning and recommendation: towards a better understanding of user preferences. In: The World Wide Web Conference, pp. 151–161 (2019)
2. Cheng, H.T., et al.: Wide & deep learning for recommender systems. In: Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, pp. 7–10 (2016)
3. Han, J., Moraga, C.: The influence of the sigmoid function parameters on the speed of backpropagation learning. In: Mira, J., Sandoval, F. (eds.) IWANN 1995. LNCS, vol. 930, pp. 195–201. Springer, Heidelberg (1995). https://doi.org/10.1007/3-540-59497-3_175
4. Jamali, M., Ester, M.: A matrix factorization technique with trust propagation for recommendation in social networks. In: Proceedings of the Fourth ACM Conference on Recommender Systems, pp. 135–142 (2010)
5. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer **42**(8), 30–37 (2009)

6. Long, M., Cao, Z., Wang, J., Yu, P.S.: Learning multiple tasks with multilinear relationship networks. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
7. Memisevic, R., Zach, C., Pollefeys, M., Hinton, G.E.: Gated softmax classification. In: Advances in Neural Information Processing Systems, vol. 23 (2010)
8. Rendle, S.: Factorization machines with libfm. ACM Trans. Intell. Syst. Technol. (TIST) **3**(3), 1–22 (2012)
9. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
10. Wang, H., Wang, N., Yeung, D.Y.: Collaborative deep learning for recommender systems. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1235–1244 (2015)
11. Wang, H., Zhang, F., Hou, M., Xie, X., Guo, M., Liu, Q.: Shine: signed heterogeneous information network embedding for sentiment link prediction. In: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, pp. 592–600 (2018)
12. Wang, H., et al.: Ripplenet: propagating user preferences on the knowledge graph for recommender systems. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, pp. 417–426 (2018)
13. Wang, H., Zhang, F., Xie, X., Guo, M.: DKN: deep knowledge-aware network for news recommendation. In: Proceedings of the 2018 World Wide Web Conference, pp. 1835–1844 (2018)
14. Wang, H., et al.: Knowledge-aware graph neural networks with label smoothness regularization for recommender systems. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 968–977 (2019)
15. Wang, H., Zhang, F., Zhao, M., Li, W., Xie, X., Guo, M.: Multi-task feature learning for knowledge graph enhanced recommendation. In: The World Wide Web Conference, pp. 2000–2010 (2019)
16. Wang, H., Zhao, M., Xie, X., Li, W., Guo, M.: Knowledge graph convolutional networks for recommender systems. In: The World Wide Web Conference, pp. 3307–3313 (2019)
17. Wang, X., He, X., Cao, Y., Liu, M., Chua, T.S.: KGAT: knowledge graph attention network for recommendation. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 950–958 (2019)
18. Wang, X., Wang, D., Xu, C., He, X., Cao, Y., Chua, T.S.: Explainable reasoning over knowledge graphs for recommendation. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 5329–5336 (2019)
19. Wang, Z., Lin, G., Tan, H., Chen, Q., Liu, X.: 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, pp. 219–228 (2020)
20. Yu, X., et al.: Personalized entity recommendation: a heterogeneous information network approach. In: Proceedings of the 7th ACM International Conference on Web Search and Data Mining, pp. 283–292 (2014)

21. Zhang, F., Yuan, N.J., Lian, D., Xie, X., Ma, W.Y.: 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 (2016)
22. Zhao, H., Yao, Q., Li, J., Song, Y., Lee, D.L.: Meta-graph based recommendation fusion over heterogeneous information networks. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 635–644 (2017)