applied sciences

MDPI

Article

Knowledge-Aware Enhanced Network Combining Neighborhood Information for Recommendations

Xiaole Wang [1,2], Jiwei Qin [1,2,*], Shangju Deng [1,2] and Wei Zeng [1,2]

[1] School of Information Science and Engineering, Xinjiang University, Urumqi 830046, China
[2] Key Laboratory of Signal Detection and Processing, Xinjiang Uygur Autonomous Region, Xinjiang University, Urumqi 830046, China
[*] Correspondence: jwqin@xju.edu.cn

Abstract: In recent years, the application of knowledge graphs to alleviate cold start and data sparsity problems of users and items in recommendation systems, has aroused great interest. In this paper, in order to address the insufficient representation of user and item embeddings in existing knowledge graph-based recommendation methods, a knowledge-aware enhanced network, combining neighborhood information recommendation (KCNR), is proposed. Specifically, KCNR first encodes prior information about the user–item interaction, and obtains the user's different knowledge neighbors by propagating them in the knowledge graph, and uses a knowledge-aware attention network to distinguish and aggregate the contributions of the different neighbors in the knowledge graph, as a way to enrich the user's description. Similarly, KCNR samples multiple-hop neighbors of item entities in the knowledge graph, and has a bias to aggregate the neighborhood information, to enhance the item embedding representation. With the above processing, KCNR can automatically discover structural and associative semantic information in the knowledge graph, and capture users' latent distant personalized preferences, by propagating them across the knowledge graph. In addition, considering the relevance of items to entities in the knowledge graph, KCNR has designed an information complementarity module, which automatically shares potential interaction characteristics of items and entities, and enables items and entities to complement the available information. We have verified that KCNR has excellent recommendation performance through extensive experiments in three real-life scenes: movies, books, and music.

Keywords: attention network; graph neural network; knowledge graph; recommender systems

check for updates

Citation: Wang, X.; Qin, J.; Deng, S.; Zeng, W. Knowledge-Aware Enhanced Network Combining Neighborhood Information for Recommendations. *Appl. Sci.* **2023**, *13*, 4577. https://doi.org/10.3390/app13074577

Academic Editor: Krzysztof Koszela

Received: 2 March 2023
Revised: 23 March 2023
Accepted: 31 March 2023
Published: 4 April 2023

1. Introduction

In the era of the internet of everything, the amount of data in real life has grown exponentially. Recommender systems have become one of the most effective methods to solve data overload. Among them, the classic collaborative filtering [1–3] recommendation method is widely used. It obtains the preferences of the target user, by finding the set of users who have the same preferences as the target user, to perform the recommendation. However, due to the dramatic increase in the number of users and items in a recommendation system, collaborative filtering-based approaches often suffer from insufficient prior information about users or items, resulting in lower user satisfaction with recommendations. Therefore, some researchers have proposed to introduce auxiliary information (e.g., social network [4], user/item attributes [5], multimodal features [6], etc.), to complement the representation of users or items and enhance the recommendation effect.

In recent studies, researchers have used the correlation between items and item attributes [7–9] to construct a knowledge graph (KG) for the recommendation, with good results. A KG is a heterogeneous information network, containing multiple nodes and multiple connected edges between nodes, where the nodes are different entities in the KG, and the edges are relationships between entities. In a recommendation scenario, the entity can be an item (movie) or an attribute of the item (movie's director). Compared with other recommendation methods, introducing a KG into recommender systems has three advantages [10]. (1) We can use the extensive item and item attribute information in the KG to supplement the item representation and improve recommendation accuracy. (2) We can use various types of relationships in the KG to propagate user preferences and increase the diversity of recommendation results. (3) We can use the user's historical behavior in the KG for the recommendation, which brings certain interpretability to the recommender system. Figure 1 shows the KG in a movie recommendation scenario. The movie KG in Figure 1, contains related entities, such as the movie's stars, genre, and production company. We use various relationships between entities to connect them, to form the movie KG. For example, the "genre" relation in the figure, can associate a movie with its genre entity. Then, we recommend unwatched movies that the user is expected to like, for the user in the KG, based on the user's history of watching movies. For example, assuming that user u_1 has watched movies A, B, and C, we can recommend unwatched movie F to the user, because movies F and A have the same movie genre and production company.

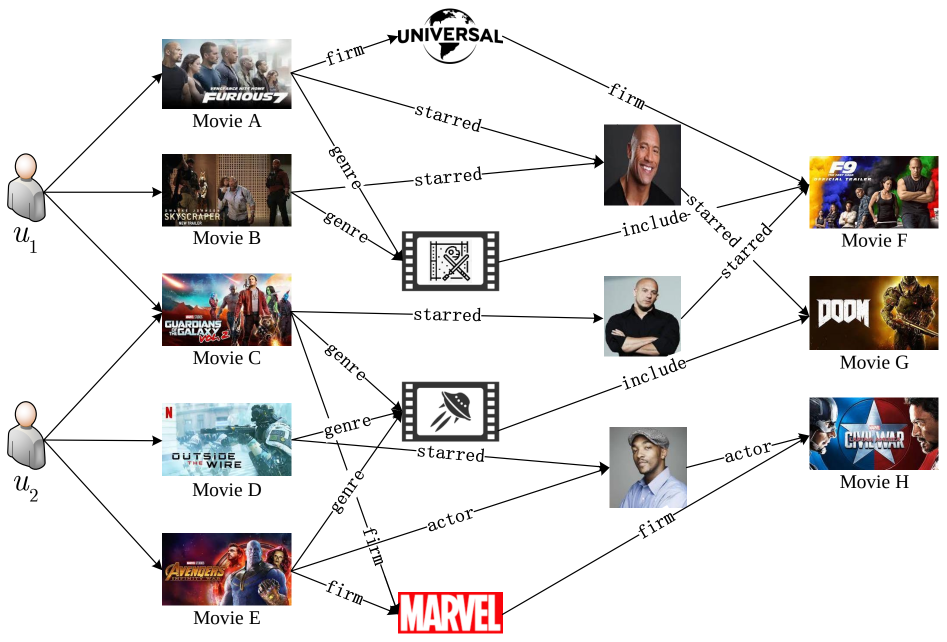


Figure 1. A movie recommendation knowledge graph.

A KG, introduces more associated semantic information into the recommender system, and we can dig deeper into the user's personalized preferences in the connection between items and item attributes, thereby improving the performance of the recommender system. For example, Wang et al. [11] designed a DKN network, which combines different semantic and knowledge information for news recommendation tasks, through a designed convolutional neural network framework. However, DKN requires embedding entities before they can be used. This results in a lack of an end-to-end training approach. Another drawback of DKN, is that it can only make use of text-based auxiliary information, making it highly limited in its applicability to scenarios. Yu et al. [12] proposed the PER model, which uses the heterogeneous network structure of a KG to characterize the connectivity of users and items, in terms of the potential features of the different types of relationships extracted. It is important to note that PER relies heavily on the design of meta-paths, a design that requires significant domain expertise, which limits the application of PER in general-purpose scenarios. Zhang et al. [13], proposed the collaborative knowledge embedding model, that combines collaborative filtering with entity embedding in the KG, but the knowledge graph embedding (KGE) module in its model, is better for entity completion or link prediction in the KG, and it is difficult to achieve feature interaction between the explicit semantics of the KG and the implicit semantics of the user. Wang et al. [10] designed a RippleNet model, which combines entity embedding with entity path connectivity, introducing the concept of user preference propagation in a KG, for the first time. However, RippleNet relies heavily on the representation of attributes, which largely weakens the importance of edges in the KG. In addition, the set of user preferences will change in unknown ways as the size of the KG increases, and most likely cause a large memory overhead. Wang et al. [14] also designed and proposed a multi-task learning framework based on KG and MKR, which allows the user preference prediction recommendation task and the KGE task to be performed separately, and uses a specially designed interaction unit to associate the two tasks. However, MKR ignores the richer information of users and items in KG, resulting in the insufficient embedding of users and items in the recommendation system.

To address the limitations of existing KG-based recommendation algorithms, inspired by MKR, a knowledge-aware enhanced network combining neighborhood information for recommendation (KCNR), is proposed, which is an end-to-end model. KCNR mainly consists of a recommendation module, an information complementary module (ICM), and a KGE module. In the recommendation module, KCNR firstly combines the prior information of user interaction items with the rich association semantic information in the KG, and then obtains and aggregates the weights of different related entities in the KG through a knowledge-aware attention network, so as to enrich the user's embedding representation. Similarly, KCNR uses the idea of a graph convolutional network [15] to sample the neighborhoods of item entities in the KG, and bias aggregates the item neighborhood information to enrich the embedding representation of the original items. This bias mainly refers to the use of connectivity relations and user-specific generated scores in the KG, to weight the aggregated item neighborhood information. Through the above operations, we can not only characterize the associated semantic information in the KG, but also capture the personalized interest of users in the relationship. In addition, KCNR also designs an ICM, to combine the recommendation module and the KGE module. The main idea of the ICM is to exploit the one-to-many relationships between item entities in the KG, and to explicitly model enhanced item representations and higher-order interaction features between entities, using cross-compression awareness to automatically control the knowledge transfer between the two. This also brings regularization to the model, to a certain extent.

The contribution of KCNR can be summarized in three aspects as shown below:

1. We propose KCNR, a knowledge-aware enhanced network combining neighborhood information, which is an end-to-end model. KCNR effectively utilizes users' prior information and the rich associative semantic information in the KG, and greatly enhances user and item representations, by designing a user representation layer (URL) and item representation layer (IRL), to alleviate the data sparsity problem of the RS.
2. We design an ICM, to enable information sharing and complementarity between items in the RS and related entities in the KG. The model generalization capability is improved, based on further enriching the item embedding representation.
3. We conduct experiments in three realistic recommendation scenarios, and the results show that KCNR outperforms the baseline approach.

2. Related Work

KG uses heterogeneous information to build a network with powerful semantic representation. In recent years, many studies have introduced KGs as auxiliary information to recommender systems and achieved good results. Combining KGs with recommender systems is also an emerging research area. There are three types of knowledge graph-based recommendation methods [16]: embedding-based methods, path-based methods, and unified methods.

The main idea of the embedding-based method, is to use the embedding representation of entities in the KG, obtained with the KGE algorithm [17,18], directly in the RS, to enrich user or item representations. Zhang et al. [13] proposed a CKE model, which focuses on how to exploit heterogeneous information of items to improve recommendation effectiveness. It unites item content information (e.g., text and visual), structural knowledge information, and user preference information, to learn together, significantly improving the performance of collaborative filtering recommendation methods. Wang et al. [11] proposed the DKN model, which combines KGs with a convolutional neural network for news recommendation, to more accurately recommend news that may be of interest to users, by combining the connections between news at the knowledge level, and to capture the dynamic interest of users in the news, using the designed news following module. Huang et al. [19] designed the KSR model, which uses the key-value memory network to incorporate KG information into recommender systems, and combines the KG information with item information to capture users' preferences, to enhance the model capabilities. Wang et al. [14] designed a multi-task feature learning framework called MKR, which utilizes the correspondence between items and entities in a KG, to link the recommendation task and the entity embedding task. In the framework, the recommendation task learns representations of users and items and eventually makes predictions, while the entity embedding task learns representations of item-related entities and relationships, and both share latent features using cross-compression units. Embedding-based methods have high flexibility, but the directly learned entity embedding vectors sometimes cannot directly describe the recommendation relationship between items, and it ignores a large number of inter-entity connections in KG and lacks reasoning ability.

The main idea of the path-based method, is to provide auxiliary information to the recommender system by using various connection patterns of the unique structure of the KG. For example, Yu et al. [20] proposed the HeteRec model, which supplements the user–item interaction matrix, by calculating the similarity of user (item) meta-paths in the KG, so that richer representations of items can be extracted to improve the recommendation quality. Zhao et al. [21] designed the FMG model, which uses a meta-graph containing more rich connectivity information instead of meta-paths, incorporating complex semantic information into the model, so that the model can more accurately obtain the similarity of entities. Hu et al. [22] proposed the MCRec model, which uses a context-based meta-path and a neural network model of a co-attention mechanism, in a heterogeneous information network, to build recommender systems. The path-based method can describe the information in the KG more intuitively, and using the correlation of entities in the KG brings a certain degree of interpretability to the recommendation. However, this method excessively depends on the meta-paths designed according to the domain of expertise, and needs to reconstruct the meta-paths when the recommendation scenarios or KGs change, thus limiting its usage scenarios.

The above-mentioned two methods only utilize the unilateral information of the KG. In order to fully exploit and utilize the information in the KG, the semantic knowledge of entities and relationships in the KG has been used jointly with association information for the recommendation, which is called the unified method. For example, Wang et al. [10] designed the RippleNet model, which inputs user–item interaction pairs and outputs whether the user clicked on an item. It gradually spreads out, like water ripples, and vividly describes the user preference spread along the associated path in the KG, from historical interests. Wang et al. [15] also proposed the KGCN model, which utilizes a graph convolutional network to sample multiple neighbor entities of an item entity in a KG and aggregate them onto the original item, to enhance the item representation. Additionally, Qu et al. [23] proposed the KNI model, which also considers the interaction between users and item neighbors, further enriching the connectivity of nodes. Wang et al. [24] proposed the CKAN network, which encodes prior user–item interaction information and then combines it with the designed attention module, for the recommendation. The unified methods organically unify the ideas of embedding and propagation, and enrich the representation of entities using the linking structure of entities in the KG, which greatly enhances the embedding representation of users and items.

3. Our Approach

In this section, we present the KCNR model in detail. We first present the recommendation problem of this paper. Then, we describe the KCNR model framework and the design of each module in detail. Finally, we discuss the learning algorithms for different parts of the KCNR.

3.1. Problem Definition

In our recommendation module, it is assumed that there is a user set $\mathbf{U} = \{u_1, u_2, \cdots, u_M\}$, and an item set $\mathbf{V} = \{v_1, v_2, \cdots, v_N\}$, where M and N are the number of users and the number of items in the RS, respectively. The user–item interaction matrix $\mathbf{Y}^{M\times N} = \{y_{uv} | u \in \mathbf{U}, v \in \mathbf{V}\}$ is defined as follows:

$$y_{uv} = \begin{cases} 1, & \text{user } u \text{ interacts with item } v, \\ 0, & \text{otherwise,} \end{cases} \tag{1}$$

where $y_{uv} = 1$ indicates that user u has interacted with item v, i.e., has purchased, browsed, clicked, etc. $y_{uv} = 0$ here, simply indicates that user u has not clicked on item v. It does not indicate that the user does not like the item. It is also possible that the user has not seen the item or has unintentionally skipped it. In addition, we construct a KG $\mathbf{G} = \{(h, r, t) | h, t \in E, r \in R\}$ containing a large number of triplets, where $E = \{e_1, e_2, \cdots, e_i\}$ denotes a large number of entities in the KG and $R = \{r_1, r_2, \cdots, r_j\}$ denotes a large number of connection relationships between entities in the KG. Any of the triplets (h, r, t) in G consists of an entity–relationship–entity, representing the head entity h, the tail entity t, and the relationship r between them, respectively. For example, the triplet *(Fast & Furious 9, film.film.star, Vin Diesel)*, indicates that *Vin Diesel* starred in the film *Fast & Furious 9*. Then, we use $D = \{(v, e) | v \in \mathbf{V}, e \in E\}$ to represent the one-to-one correspondence between items and entities in the KG, where (v, e) denotes the correspondence between v and e. In most recommended scenarios, an item often corresponds to only one entity.

In short, after obtaining the implicit interaction matrix $\mathbf{Y}^{M\times N}$, and the KG $\mathbf{G}$, our main task is to build a prediction function to predict the possible interactions between users and their uninteracted items. The prediction function $\mathcal{F}(\cdot)$, is defined as follows:

$$\hat{y}_{uv} = \mathcal{F}(u, v | \Theta, \mathbf{G}), \tag{2}$$

where $\hat{y}_{uv}$ denotes the possibility that user u interacts with item v, and Θ denotes the argument of function $\mathcal{F}(\cdot)$.

3.2. Model Framework

As shown in Figure 2, KCNR consists of a recommendation module, an ICM, and a KGE module. Among them, the recommendation module contains a URL, an IRL, and a prediction layer. The URL mainly encodes the prior information of user–item interactions explicitly as user representations, and captures important entity information associated with them in the KG, using a knowledge-aware attention network to make the model describe user representations more richly. Similarly, the IRL takes advantage of the correlation between entities in the KG to bias aggregate multiple-hop neighborhood information of item entities, to enhance the item embedding representation. Then, the prediction layer combines the obtained enhanced user/item representations, to output the final predicted item click probabilities. In addition, we designed an ICM, for feature interaction between items in the RS and head entities in the KG, to complement the available information between items and entities. The KGE module mainly connects the head entities output from the ICM with the relational entities, and then goes through a multi-layer neural network to predict the tail entities, thus assisting the recommendation module and bringing regularization to the model.

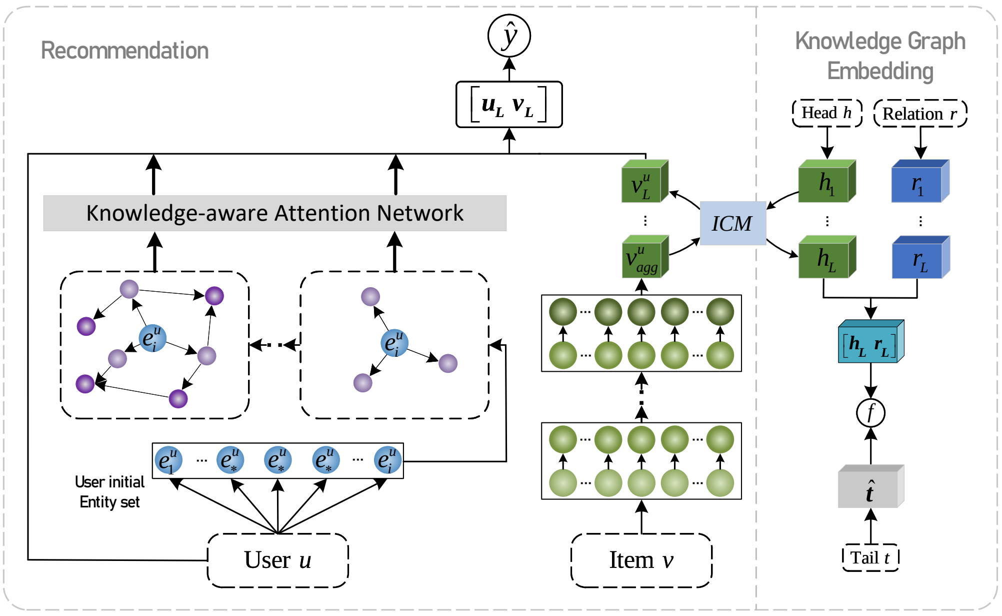


Figure 2. Illustration of the proposed KCNR model, which consists of a recommendation module, knowledge graph embedding module, and information complementary module.

3.3. Recommendation Module

In this subsection, we detail the URL, IRL, and prediction layer of the KCNR model.

3.3.1. URL

The URL includes a collaborative propagation part and a knowledge graph-aware propagation part. We will discuss both parts in detail below.

Collaborative propagation: We know that the user's prior information (i.e., historical interaction items) largely represents the information about the user's personalized preferences. Therefore, to enrich the description of the user, KCNR does not use a simple embedding representation of the user alone, but combines it with the user's prior information and the knowledge information in the KG, to describe the user. Specifically, KCNR first finds multiple entities in the KG, corresponding to multiple interaction items of user u, based on the correspondence between items and entities in the KG, which constitutes the initial entity propagation set of user u, and lets it propagate in the KG, to obtain user personalized preferences. The initial entity propagation set of user u is defined as follows:

$$\varphi_u^0 = \{e|(v,e) \in D \; and \; v \in |y_{uv} = 1\}. \tag{3}$$

With the above processing, we can add the interaction information that best represents the potential semantics of the user to the initial set of entity propagation of the user u, which is used to enhance the user embedding representation.

Knowledge graph-aware propagation: In the KG, entities connected by a variety of relationships often have strong knowledge associations. Thus, by propagating entities along the associated paths in the KG, we can obtain the extended entity set of the user's initial entity propagation set, as well as the set of triplets of different depths, which can also further enhance the user's embedding representation, due to the strong correlation between them. The extended set of entities of user u in the KG, can be recursively represented as:

$$\varphi_u^l = \left\{t|(h,r,t) \in \mathbf{G} \; and \; h \in \varphi_u^{l-1}\right\}, \quad l = 1,2,\cdots,L, \tag{4}$$

where l represents different depths. Then, we define the user's l-th triplet set as:

$$\phi_u^l = \left\{(h,r,t)\middle|(h,r,t) \in \mathbf{G} \; and \; h \in \varphi_u^{l-1}\right\}, \quad l = 1,2,\cdots,L. \tag{5}$$

Through collaborative propagation and knowledge graph-aware propagation, users' preferences are continuously diffused in the KG, and we are then able to capture users' knowledge-based higher-order interaction information, effectively enhancing the ability of the model to augment user representations with latent vectors. However, each tail entity in the KG tends to have different head entities and their corresponding relationships. For example, *Farewell Atlantis* and *Interstellar* are relatively similar in the genre of films, but the actors and directors are less similar. Therefore, we adopt a knowledge-aware attention network [24] to generate different scores for tail entities of triplets (i.e., different weights for different head entities and relations), which are used to indicate that, when different head entities are obtained, different meanings when entities and relationships are used. The knowledge-aware network is shown in Figure 3. Assuming that (h, r, t) is the *i-th* triplet in the set of user extended entities in the set of triplets at the *l-th* layer, the attention embedding of the tail entity is as follows:

$$att_i = \delta\left(e_i^h, r_i\right)e_i^t, \tag{6}$$

where e_i^h is an embedding representation of the head entity, r_i is an embedding representation of relation, and e_i^t is an embedding representation of the tail entity. We use $\delta\left(e_i^h, r_i\right)$ to control the attention scores (weights) generated by e_i^h and r_i. In this paper, a network similar to the attention network [25] is used to implement the function $\delta(\cdot)$, formulated as follows:

$$\delta\left(e_i^h, r_i\right) = Sigmoid(w_2 ReLU(w_1 z_0 + b_1) + b_2), \tag{7}$$

where

$$z_0 = ReLU(w_0(e_i^h, r_i)_{concat} + b_0), \tag{8}$$

where w_* and b_* are the parameters of the nonlinear layer. Then, we use softmax [26] to normalize the weights as follows:

$$\delta\left(e_i^h, r_i\right) = \frac{\exp\left(\delta\left(e_i^h, r_i\right)\right)}{\sum_{(h',r',t')\in\phi_u^l}\exp\left(\delta\left(e_i^{h'}, r_i'\right)\right)}, \tag{9}$$

where ϕ_u^l denotes the triplet set of the user's *l-th* layer. Through the knowledge-aware attention network, KCNR can assign different weights to different tail entities, so that we can obtain knowledge with higher relevance to the user, to enrich the user description. Then, we use the obtained different weights to aggregate the tail entities, to obtain a potential expression of the *l-th* layer triplet set for the user u, as shown below:

$$e_u^l = \sum_{i=1}^{|\phi_u^l|} att_i^u, \tag{10}$$

where $\left|\phi_u^l\right|$ denotes the number of triplets. Note in particular, that the entities in the user's initial entity propagation set tend to be closest to the user's original representation. Therefore, we include these representations in the user representation. This is shown as follows:

$$e_u^0 = \frac{\sum_{e\in\varphi_u^0} e}{|\varphi_u^0|}. \tag{11}$$

We finally obtain the set of user representations containing the user simple embedding representation, the user first-order entity set, and the weighted extended entity set. The representations are as follows:

$$\psi_u = \left\{u, e_u^0, e_u^1, \cdots, e_u^l\right\}. \tag{12}$$

Since the representations in each layer of the propagation process have their different potential impacts, it emphasizes different higher-order connectivity and preference similarity. Therefore, in this paper, the summation approach is used to obtain the embedded representations of users as follows:

$$u_L = \sum_{e_u \in \psi_u} e_u. \tag{13}$$

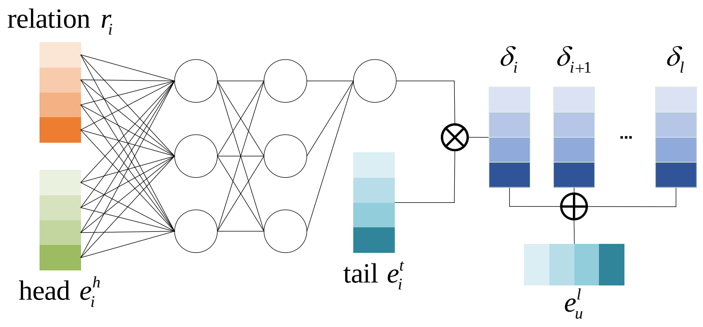


Figure 3. Knowledge-aware attention network.

3.3.2. IRL

The IRL mainly samples the neighboring entities of the item entity in the KG, and then combines the neighboring entity information with a bias, to complement the embedded representation of the item entity. We can also extend the perceptual depth of items in the KG to multi-hop, to obtain potential personalized preferences of users over long distances. We assume that there are users u and items v, and we use $\mathcal{Q}(v)$ to denote the set of first-order entities in the KG, corresponding to item v. To enrich the embedding representation vector of items, we have the bias to aggregate these entities as follows:

$$v^u_{\mathcal{Q}(v)} = \sum\nolimits_{e \in \mathcal{Q}(v)} \tilde{\pi}^u_{r_{v,e}} e, \tag{14}$$

where $r_{v,e}$ is the relational representation of entities v and e in the KG, and $\tilde{\pi}^u_{r_{v,e}}$ is the score between u and r, defined as follows:

$$\tilde{\pi}^u_{r_{v,e}} = \frac{\exp(\pi^u_{r_{v,e}})}{\sum_{e \in \mathcal{Q}(v)} \exp(\pi^u_{r_{v,e}})}, \tag{15}$$

where

$$\pi^u_r = \Delta(u, r), \tag{16}$$

where u and r are user embedded representations and relation embedded representations, respectively, π^u_r denotes the score between a user and a relation, and $\tilde{\pi}^u_{r_{v,e}}$ is the normalization operation on π^u_r. The function $\Delta(\cdot)$ (inner product), is used to calculate the score π^u_r. This operates because we consider that different relations are important to users in different ways. For example, a particular user may have more personalized preferences for music they have historically enjoyed by the same artists, while other users may be more concerned with the genre of the music. Therefore, we use the user and relationship scores as a filter for the user's personalized preferences, allowing us to bias specific scores when aggregating neighborhood information. Note that, in the real world, the $\mathcal{Q}(v)$ of some entities may be large. Therefore, to ensure higher computational efficiency, we draw the set of neighbors of the same size uniformly for each entity. Briefly, we denote the neighborhood representation of entity v as $v^u_{\mathcal{O}(v)}$, where $\mathcal{O}(v) = \{e|e \in \mathcal{Q}(v)\}$ and $|\mathcal{O}(v)| = K$. K denotes the size of the nearest neighbor, which is a tunable parameter, and we will discuss its effect on the model in the experimental section. As shown in Figure 4, we give examples of the two-layer depth of perception of the item entity.

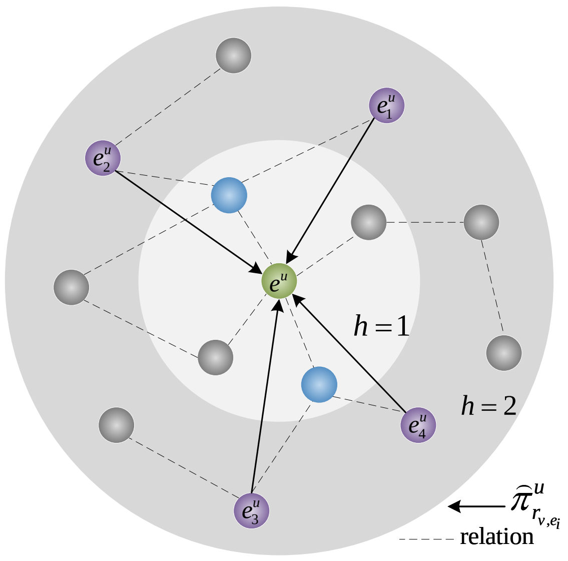


Figure 4. A two-layer perception depth of item entity in a KG.

Finally, we have to combine the initial embedding representation of item v, with its aggregated neighborhood representation $v^u_{\mathcal{O}(v)}$, to obtain the neighborhood-enhanced representation v^u_{agg}, of item v. This paper implements the following four types of aggregators:

- *Summation aggregator* [27] directly sums v and $v^u_{\mathcal{O}(v)}$, and then goes through a nonlinear layer, which is defined as follows:
$$v^u_{agg_{sum}} = ReLU(w \cdot (v + v^u_{\mathcal{O}(v)}) + b). \tag{17}$$
- *Concat aggregator* [28] concatenates v and $v^u_{\mathcal{O}(v)}$ before going through the nonlinear layer, which is defined as follows:
$$v^u_{agg_{concat}} = ReLU(w \cdot concat(v, v^u_{\mathcal{O}(v)}) + b). \tag{18}$$
- *Neighbor aggregator* [29] only considers $v^u_{\mathcal{O}(v)}$ as the output representation, which is defined as follows:
$$v^u_{agg_{neighbor}} = ReLU(w \cdot v^u_{\mathcal{O}(v)} + b). \tag{19}$$
- *Bi-interaction aggregator* [30] considers two information interactions of v and $v^u_{\mathcal{O}(v)}$, defined as follows:
$$v^u_{agg_{Bi}} = ReLU(w_1 \cdot (v + v^u_{\mathcal{O}(v)})) + ReLU(w_2 \cdot (v \odot v^u_{\mathcal{O}(v)})). \tag{20}$$

Among the above four aggregators, w_* and b denote the weights and bias, respectively. In particular, the $\odot$ in the bi-interaction aggregator represents the Hadamard product. Aggregation is an important step in our model, and we will evaluate its performance in the experimental section.

3.3.3. Prediction Layer

After obtaining the final embedded representations of users and items, we combine the two through the prediction function f_{RS} (inner product), to make the final prediction. The probability of user u clicking on item v is as follows:

$$\hat{y}_{uv} = \sigma(f_{RS}(u_L, v_L^u)), \tag{21}$$

where

$$v_L^u = \mathbb{E}_{h \sim s(v)}\left[ICM\left(v_{agg}^u, h\right)[v_{agg}^u]\right], \tag{22}$$

where $s(v)$ represents the set of related entities of item v, and $ICM(\cdot)$ is the information complementary module we designed, which we will introduce in detail in the next subsection.

3.4. ICM

To model higher-order feature interactions between enhanced item representations in RS and entity representations in KGs, and let them complement the available information, we design an ICM in KCNR, as shown in Figure 5. The ICM utilizes a cross-compression unit [14], to automatically control the enhanced item representation and knowledge transfer between entities. Specifically, for the neighbor aggregate representation v_{agg}^u of item v, and a related entity e of the item in the KG, we construct the interaction of their potential features v_l and e_l from the l-th layer (for convenience, in the following description we replace v_{agg}^u with q):

$$J_l = q_l e_l^{\mathrm{T}} = \begin{bmatrix} q_l^1 e_l^1 & \cdots & q_l^1 e_l^d \\ \vdots & \ddots & \vdots \\ q_l^d e_l^1 & \cdots & q_l^d e_l^d \end{bmatrix}, \tag{23}$$

$$J_l^T = e_l q_l^{\mathrm{T}} = \begin{bmatrix} e_l^1 q_l^1 & \cdots & e_l^1 q_l^d \\ \vdots & \ddots & \vdots \\ e_l^d q_l^1 & \cdots & e_l^d q_l^d \end{bmatrix}, \tag{24}$$

where both J_l and J_l^T are called the cross feature matrix, and d is the entity embedding dimension. Through the above operations, all possible feature interactions $q_l^i e_l^j, \forall(i,j) \in \{1,\cdots,d\}$ represented by the neighbor aggregate representation q of item v and related entities e, are explicitly modeled. Next, we project J_l and J_l^T into their latent representation space, and use the output as input for the next layer of interaction, expressed as follows:

$$v_{l+1} = J_l w_l^{QQ} + J_l^T w_l^{EQ} + b_l^Q = q_l e_l^T w_l^{QQ} + e_l q_l^T w_l^{EQ} + b_l^Q, \tag{25}$$

$$e_{l+1} = J_l w_l^{QE} + J_l^T w_l^{EE} + b_l^E = q_l e_l^T w_l^{QE} + e_l q_l^T w_l^{EE} + b_l^E, \tag{26}$$

where $w_l^{\cdot\cdot}$ and $b_l^{\cdot}$ are weight and bias. For the convenience of description, we denote the ICM as:

$$[q_{l+1}, e_{l+1}] = ICM(q_l, e_l). \tag{27}$$

In particular, the ICM generally has a better feature interaction effect at the lower layer, because as the number of network layers increases, the degree of mixing of various features grows, and the mixed features lack clear associations and are not suitable for sharing [14].

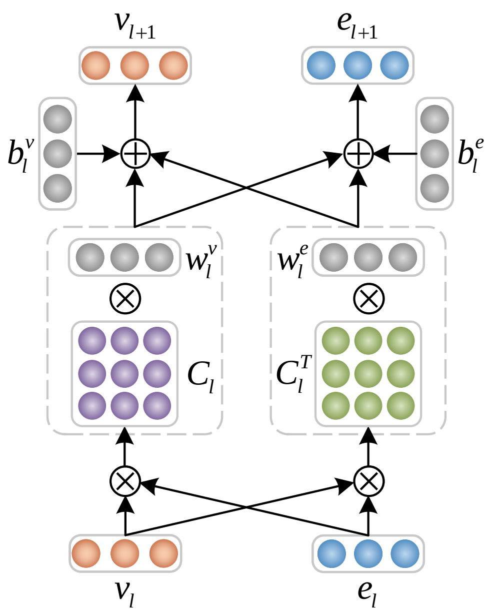


Figure 5. Information complementary module.

3.5. KGE Module

KGE is used to represent entities and relationships in a continuous vector space. In KCNR, we first use the ICM to obtain the interactive features of the head entity h, and item neighbor aggregated representation v_{agg}^u in the triplet (h, r, t), and then we combine the head entity representation h_L output by the ICM, with the nonlinear layer processing. The relational entities r_L are combined, and the tail entity is predicted after K-layer multi-layer perceptron (MLP). The specific operation steps are as follows:

$$h_L = \mathbb{E}_{v \sim s(h)} \left[ICM^L \left(v_{agg}^u, h \right) [e] \right], \tag{28}$$

$$r_L = \mathcal{M}^L(r), \tag{29}$$

$$\hat{t} = \mathcal{M}^K \left(\begin{bmatrix} h_L \\ r_L \end{bmatrix} \right), \tag{30}$$

where $s(h)$ is the set of associated items of entity h, and $\hat{t}$ is the predicted tail entity representation, we then calculate the score of the triad using the score function f_{KGE}:

$$score(h, r, t) = f_{KGE}(t, \hat{t}) = \sigma \left(t^T \hat{t} \right), \tag{31}$$

where $\sigma(\cdot)$ is a nonlinear activation function.

3.6. Learning Algorithm

The loss function of KCNR is shown below:

$$\begin{aligned} L &= L_{RS} + L_{KGE} + L_{REG} \\ &= \sum_{u \in \mathbf{U}, v \in \mathbf{V}} J(\hat{y}_{uv}, y_{uv}) \\ &\quad -\lambda_1 \left(\sum_{(h,r,t) \in \mathbf{G}} score(h, r, t) - \sum_{(h',r,t') \notin \mathbf{G}} score(h', r, t') \right) \\ &\quad +\lambda_2 \|\mathbf{W}\|_2^2, \end{aligned} \tag{32}$$

where $J(\cdot)$ represents the cross-entropy loss, and L_{KGE} is the loss of the KGE module, which calculates the score of the triplet. L_{REG} is the regularization term of KCNR.

4. Experiments

In this section, we conduct a large number of experiments on KCNR, and analyze and discuss the experimental results and related influencing factors.

4.1. Datasets

KCNR conducted experiments on the following three datasets.

- **MovieLens-1M**, https://grouplens.org/datasets/movielens/1m/, (accessed on 20 September 2022) is the more widely utilized movie recommendation dataset in recommendation systems. In the dataset, 6040 favorite movies of users are reflected as ratings from 1 to 5.
- **Book-Crossing**, http://www2.informatik.uni-freiburg.de/cziegler/BX/, (accessed on 12 October 2022) is used for book recommendations, and it consists of 90,000 user ratings from 0 to 10, for 1,149,780 books.
- **Last.FM**, https://grouplens.org/datasets/hetrec-2011/, (accessed on 6 September 2022) is a dataset for music recommendation, which contains music interaction information from 2000 users.

Since the interaction data in the above three datasets are all explicit feedback data, we first transform them into implicit feedback data by data preprocessing. For MovieLens-1M, we labeled the interactions of ratings 4 and 5 as positive feedback signals, and for Book-Crossing and Last.FM, we recorded all interactions as positive feedback behaviors. Microsoft's Satori was used to extract the knowledge triad used to construct the KG. For the specific knowledge graph construction process, we followed the literature [10]. Table 1 shows the details of the datasets.

Table 1. Statistical information for the three datasets.

	MovieLens-1M	Book-Crossing	Last.FM
#users	6036	17,860	1872
#items	2445	14,910	3846
#inter.	753,772	139,746	42,346
#entity	182,011	77,903	9366
#relation	2,483,990	303,000	31,036
#triplets	20,195	19,793	15,518
#sparsity	0.9489	0.9994	0.9941

4.2. Baseline Model

In this paper, to more comprehensively evaluate the performance of KCNR, we compare it with mainstream baseline methods, including graph network-based recommendation methods (FairGo, PER), knowledge graph-based recommendation methods (CKE, RippleNet, KGCN, MKR, CAKR), and other recommendation methods (LibFM, Wide&Deep). The descriptions of these methods are shown below:

- **LibFM** [31] is used in the CTR recommendation scenario, and it does not use KG.
- **PER** [12] is a path-based recommendation method that regards the KG as a heterogeneous information network.
- **CKE** [13] introduces various types of heterogeneous data for the RS, combined with the initial item representation, to improve the recommendation quality.
- **Wide&Deep** [32] combines a deep model and linear model, to obtain more information, to improve the recommendation effect.
- **RippleNet** [10] proposed to use the user's prior information to propagate in the KG, to capture the user's personalized preferences.
- **KGCN** [15] propagates item entities in the KG and has a bias to aggregate the item's neighbor information to complement the item embedding.
- **MKR** [14] is a multi-task recommendation model trained by combining the recommendation task and the KGE task.

- **FairGo** [33] uses adversarial learning techniques to consider the fairness of graph recommendation.
- **CAKR** [34] improves the interaction unit of MKR, to better capture characteristic interactions between entities.

4.3. Experimental Setup

As shown in Table 2, we set the corresponding hyperparameters for these three datasets, where the d parameter is the embedding dimension of the model and L is the layer number of the ICM. λ_{RS} is the learning rate for the recommendation module, and λ_2 represents the $L2$ regularization weight. The hyperparameters of the other baseline models are the same as in the original papers. In this paper, AUC and F1 were used to evaluate the KCNR model, and the higher the values, the better. In terms of dataset partitioning, we divide the dataset into 6:2:2 (training set:validation set:test set).

Table 2. Hyperparameter settings for the KCNR model.

Dataset	**Parameters**
MovieLens-1M	$d = 32, L = 2, \lambda_{RS} = 2 \times 10^{-2}, \lambda_2 = 10^{-6}$
Book-Crossing	$d = 16, L = 2, \lambda_{RS} = 2 \times 10^{-4}, \lambda_2 = 10^{-6}$
Last.FM	$d = 4, L = 2, \lambda_{RS} = 10^{-3}, \lambda_2 = 10^{-6}$

4.4. Experimental Results

In this section, we conduct experiments on click-through rate prediction for different models. We show the final results in Table 3. The analysis is as follows:

- Compared with other baseline models, the performance of PER is relatively poor, because the design of artificial paths often requires more professional domain knowledge, resulting in its inability to use the information in the KG efficiently.
- From the experimental results, LibFM, Wide&Deep, and CKE, achieved better experimental results than PER, indicating that they can utilize the rich auxiliary information in the KG to improve the recommendation performance.
- RippleNet shows excellent performance, suggesting that obtaining auxiliary information by propagating user preferences in the KG is effective in improving the recommendation effect. However, by comparing the experimental results of the three datasets, RippleNet performs poorly on the dataset with greater data sparsity, indicating its strong dependence on data sparsity.
- For both MKR, CAKR, and KGCN, MKR and CAKR outperform all baseline methods on the Book-Crossing dataset, suggesting that the cross-compression unit in MKR and CAKR can learn more additional information, to alleviate the data sparsity problem in recommendation scenarios with high sparsity. KGCN, on the other hand, is the least effective on the Book-Crossing dataset, which is also due to the fact that data sparsity causes KGCN to easily introduce noise, degrading the model performance when aggregating neighbor information.
- Overall, the KCNR model proposed in this paper outperformed all baseline models on all three datasets. On the MovieLens-1M dataset, the AUC increased by 0.7%. On the Book-Crossing dataset, the AUC increased by 0.6%. On the Last.FM dataset, the AUC increased by 1.1%. It can be seen that KCNR utilizes the rich semantic information in the KG, and the special network structure of the KG, to enhance the embedded representation of users and items. This also alleviates the data sparsity problem of the recommendation system, to a certain extent.
- The last four rows in Table 3 show the results of KCNR using four different aggregators, when aggregating item neighbors. It can be observed that KCNR$_{neighbor}$ has the worst performance, because it uses only the information of the item's neighbors to represent the item, losing the original information of the item. This information is important for the item. KCNR$_{sum}$ and KCNR$_{concat}$ achieve better results than KCNR$_{neighbor}$, because

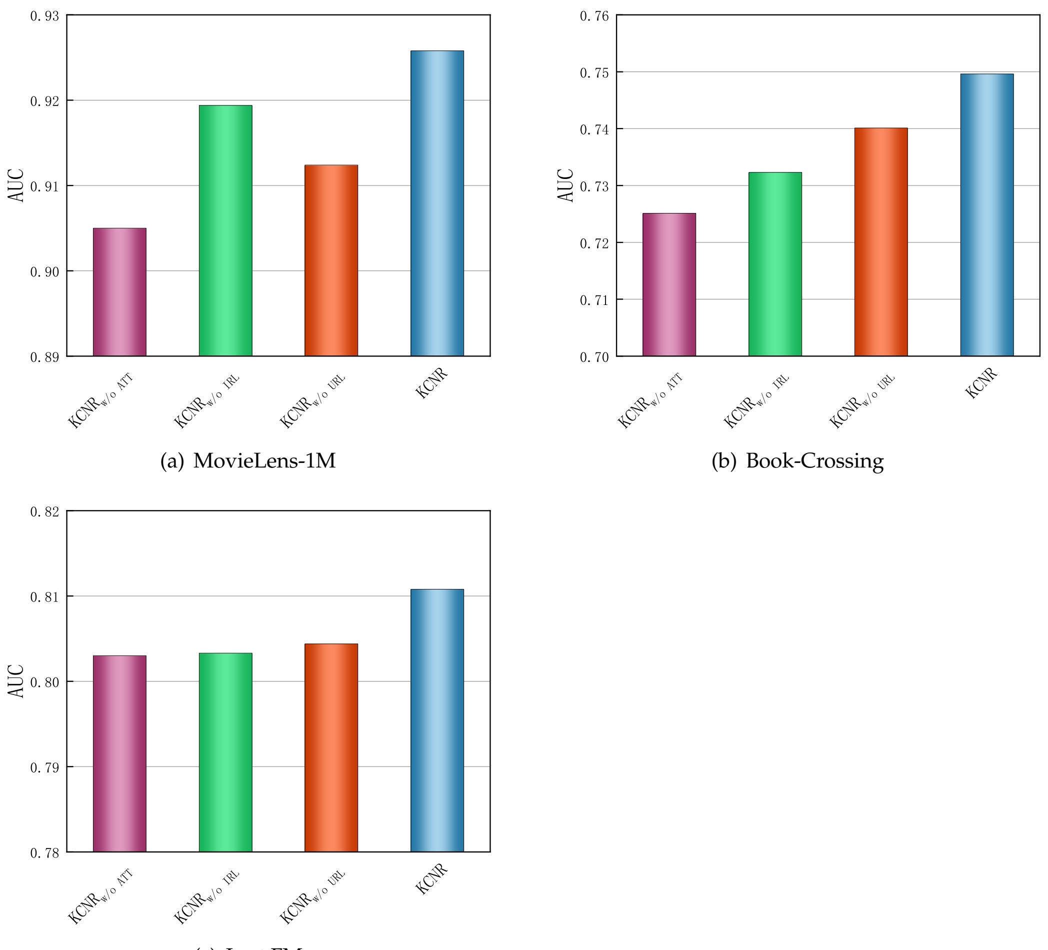
they consider the importance of the item's original information and neighboring entities' information. Based on KCNR$_{sum}$, KCNR$_{Bi}$ adds additional feature interaction between the item's original information and the neighboring entities' information, to supplement the information further. So it achieved better results. This also shows that the information disseminated in the KG is sensitive to the association between $v^u_{\mathcal{O}(v)}$ and v. It is also demonstrates that the rich semantic information in the KG can enhance user and item embedding representations and effectively improve the recommendation quality.

Table 3. The results of AUC and F1.

Model	**MovieLens-1M**		**Book-Crossing**		**Last.FM**	
	AUC	**F1**	**AUC**	**F1**	**AUC**	**F1**
LibFM	0.892	0.763	0.685	0.618	0.777	0.710
PER	0.706	0.639	0.624	0.562	0.632	0.596
CKE	0.801	0.703	0.671	0.611	0.744	0.673
Wide&Deep	0.898	0.791	0.712	0.645	0.756	0.654
RippleNet	0.912	0.812	0.725	0.650	0.766	0.702
MKR	0.911	0.838	0.727	0.665	0.795	0.729
KGCN	0.908	0.834	0.690	0.634	0.798	0.718
FairGo	0.907	0.838	0.716	0.661	0.796	0.700
CAKR	0.919	0.844	0.744	0.648	0.800	0.725
KCNR$_{\mathbf{Bi}}$	**0.926**	**0.852**	**0.750**	**0.666**	**0.811**	**0.732**
KCNR$_{sum}$	0.926	0.851	0.743	0.661	0.808	0.732
KCNR$_{concat}$	0.925	0.851	0.741	0.662	0.805	0.734
KCNR$_{neighbor}$	0.924	0.851	0.738	0.654	0.805	0.734

4.5. Influence of Different Modules

In this part, we discuss the impact of the URL module, the IRL module, and the knowledge-aware attention network on KCNR, as shown in Figure 6. As can be seen from the figure, the results of KCNR outperform $KCNR_{w/o\ URL}$ without URL and $KCNR_{w/o\ IRL}$ without IRL. This indicates that the KCNR model adds more supplementary information to users and items through URL and IRL, which makes the model describe users and items more accurately. By processing in this way, we can better understand users' personalized preferences and thus provide them with more accurate recommendation results. In addition, the results of KCNR outperform $KCNR_{w/o\ ATT}$ without knowledge-aware attention networks, and it has the most significant performance improvement. This shows that, when the model selects neighbor entities for users, it takes into account entities that are more relevant to the user, by using the knowledge-aware attention network, and combines the information of these entities with the original embedding of the user, which significantly improves user representational ability. The main purpose of a recommendation system, is to model the user's preferences to provide higher-quality recommendation results to the user. Therefore, the improvement of the model by knowledge-aware attention networks is relatively significant.



(a) MovieLens-1M (b) Book-Crossing

(c) Last.FM

Figure 6. AUC results for different modules of KCNR.

4.6. Experimental Parameter Analysis

In this subsection, we discuss the effect of parameters such as dimensionality on the KCNR model, provided that other parameters are fixed.

4.6.1. Impact of Embedding Dimension

In this part, we use the same embedding dimensions for the KG and recommendation system, and experimentally explore the effect of different dimensions on KCNR models. As shown in Table 4, we find that for the MovieLens-1M dataset, the model performance increases when the embedding dimension increases from 4 to 32, and when the dimension is larger than 32, the performance does not increase and starts to decrease. For the Book-Crossing dataset, the best model performance is achieved when the embedding dimension is 16, and the performance starts to decrease when the embedding dimension continues to increase. For the Last.FM dataset, the model performance is optimal when the embedding dimension is 4 and starts to decrease when the embedding dimension starts to increase. We can infer that KCNR's performance will keep changing with the embedding dimension, and when the peak is reached, continuing to increase the dimension will lead to a decrease in the performance of KCNR. This suggests that a suitable dimensional embedding can improve the performance of the model, but when the dimensionality is too large, overfitting problems may occur. In addition, an excessively large embedding dimension may introduce some noise while encoding more information, which is also a possible reason for the decline in model performance.

Table 4. Results of AUC on three datasets with different embedding dimensions.

d	**4**	**8**	**16**	**32**	**64**	**128**
MovieLens-1M	0.918	0.924	0.925	**0.926**	0.925	0.922
Book-Crossing	0.743	0.744	**0.750**	0.742	0.743	0.738
Last.FM	**0.811**	0.808	0.808	0.807	0.806	0.803

4.6.2. Impact of Item Perception Depth

We explore the impact of item entities on the performance of KCNR, by setting their different depth of feeling in the KG. As shown in Table 5, the optimal performance is obtained when h is two or three, for the three datasets. We can also see from the table, that when the perceptual depth is too large, it will lead to bad results. This is because too long a KG propagation chain, sometimes does not add more useful information to the initial item entities and may even add more noise, affecting the model performance.

Table 5. Results of AUC on three datasets with different item perception depths.

h	**1**	**2**	**3**	**4**
MovieLens-1M	0.924	**0.926**	0.923	0.923
Book-Crossing	0.742	**0.750**	0.740	0.735
Last.FM	0.807	0.808	**0.811**	0.806

4.6.3. Impact of Item Neighbor Sampling Size

We chose different item sampling neighbor sizes, to investigate their effect on KCNR's performance. From Table 6, we observe that KCNR's performance is optimal in the MovieLens-1M and Last.FM datasets when the neighbors are six and eight, respectively, and in the Book-Crossing dataset when the neighbors are two. This is because the Book-Crossing dataset is more sparse, and more neighbors may introduce more useless information and reduce the predictive power of the model.

Table 6. Results of AUC on three datasets with different item neighbor sampling sizes.

K	**2**	**4**	**6**	**8**	**16**	**32**
MovieLens-1M	0.923	0.923	**0.926**	0.924	0.923	0.923
Book-Crossing	**0.750**	0.744	0.743	0.743	0.742	0.734
Last.FM	0.805	0.807	0.808	**0.811**	0.808	0.807

4.6.4. Impact of the User's Initial Entity Size

We explored the upper bounds of KCNR's performance, by varying the size of the user's initial entity. Table 7 shows the results of the AUC evaluation metrics for the MovieLens-1M dataset, the Book-Crossing dataset, and the Last.FM dataset. We find that a suitable user initial set size is beneficial to the model. The reason is, that the number of users' initial entities determines the number of triplets associated with KG, which ultimately affects the model recommendation results.

Table 7. Results of AUC on three datasets with different user's initial entity sizes.

s	**4**	**8**	**16**	**32**	**64**
MovieLens-1M	0.906	0.910	0.916	0.920	**0.926**
Book-Crossing	0.743	**0.750**	0.744	0.743	0.739
Last.FM	0.808	0.809	**0.811**	0.807	0.807

4.6.5. Impact of User Propagation Depth

In this part, we observe the changes in KCNR's performance, by varying the depth of user perception propagation. Table 8 shows the AUC evaluation metrics for the three datasets. From the table, we can see that the model performance is optimal at a propagation depth of two for the MovieLens-1M dataset, three for the Book-Crossing dataset, and one for the Last.FM dataset. This is because propagating too far in the KG, in addition to bringing in more supplementary information to enhance the user representation, may bring in more noise. Therefore, a reasonable propagation depth can be set, to obtain the optimal model prediction capability in different recommendation scenarios.

Table 8. Results of AUC on three datasets with different user propagation depths.

l	1	2	3
MovieLens-1M	0.924	**0.926**	0.922
Book-Crossing	0.743	0.744	**0.750**
Last.FM	**0.811**	0.808	0.805

5. Conclusions and Future Work

In this paper, we propose an end-to-end knowledge-aware enhanced network combining neighborhood information for recommendation (KCNR). KCNR consists of a recommendation module, KGE module, and ICM. In the recommendation module, we use user prior information and item entities propagated in the KG, to enhance user and item embeddings, as a way to alleviate the data sparsity problem of the RS. The KGE module uses the properties of triplets in the KG to extract latent features. We then associate the recommendation module with the KGE module, by explicitly modeling the interaction between the enhanced item representations and the features of the entities in the KG, using the designed ICM. Our numerous experiments in real recommendation scenarios, show that KCNR outperforms the baseline models. In future work, we will try to construct user-related KGs for recommendation, and consider adding temporal information to capture the dynamic preferences of users at different time periods.

Author Contributions: Methodology, X.W., J.Q., S.D. and W.Z.; Software, X.W. and W.Z.; Validation, X.W. and S.D.; Formal analysis, X.W. and J.Q.; Resources, J.Q.; Data curation, S.D.; Writing—original draft, X.W.; Visualization, X.W. and W.Z.; Supervision, J.Q. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Science Fund for Outstanding Youth of Xinjiang Uygur Autonomous Region, under grant no. 2021D01E14.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest: The authors declare that they have no conflict of interest.

References

1. He, X.; Liao, L.; Zhang, H.; Nie, L.; Hu, X.; Chua, T.S. Neural collaborative filtering. In Proceedings of the 26th International Conference on World Wide Web, Perth, Australia, 3–7 April 2017; pp. 173–182.
2. Cui, Z.; Xu, X.; Fei, X.; Cai, X.; Cao, Y.; Zhang, W.; Chen, J. Personalized recommendation system based on collaborative filtering for IoT scenarios. *IEEE Trans. Serv. Comput.* **2020**, *13*, 685–695. [CrossRef]
3. Yu, X.; Jiang, F.; Du, J.; Gong, D. A cross-domain collaborative filtering algorithm with expanding user and item features via the latent factor space of auxiliary domains. *Pattern Recognit.* **2019**, *94*, 96–109. [CrossRef]
4. Guo, Z.; Wang, H. A deep graph neural network-based mechanism for social recommendations. *IEEE Trans. Ind. Inform.* **2020**, *17*, 2776–2783. [CrossRef]
5. Yang, X.; Zhou, S.; Cao, M. An approach to alleviate the sparsity problem of hybrid collaborative filtering based recommendations: The product-attribute perspective from user reviews. *Mob. Netw. Appl.* **2020**, *25*, 376–390. [CrossRef]
6. Wu, C.; Wu, F.; Qi, T.; Huang, Y. MM-Rec: Multimodal News Recommendation. *arXiv* **2021**, arXiv:2104.07407.
7. Zhu, Q.; Zhou, X.; Wu, J.; Tan, J.; Guo, L. A knowledge-aware attentional reasoning network for recommendation. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 6999–7006.

8. 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, Honolulu, HI, USA, 27 January 2019; Volume 33, pp. 5329–5336.
9. Shi, C.; Hu, B.; Zhao, W.X.; Philip, S.Y. Heterogeneous information network embedding for recommendation. *IEEE Trans. Knowl. Data Eng.* **2018**, *31*, 357–370. [CrossRef]
10. Wang, H.; Zhang, F.; Wang, J.; Zhao, M.; Li, W.; Xie, X.; Guo, M. Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In Proceedings of the 27th ACM International Conference on Information and Knowledge Management, Turin, Italy, 22–26 October 2018; pp. 417–426.
11. 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, Lyon, France, 23–27 April 2018; pp. 1835–1844.
12. Yu, X.; Ren, X.; Sun, Y.; Gu, Q.; Sturt, B.; Khandelwal, U.; Norick, B.; Han, J. Personalized entity recommendation: A heterogeneous information network approach. In Proceedings of the 7th ACM international Conference on Web Search and Data Mining, New York, NY, USA, 24–28 February 2014; pp. 283–292.
13. 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, San Francisco, CA, USA, 13–17 August 2016; pp. 353–362.
14. Wang, H.; Zhang, F.; Zhao, M.; Li, W.; Xie, X.; Guo, M. Multi-task feature learning for knowledge graph enhanced recommendation. In Proceedings of the World Wide Web Conference, San Francisco, CA, USA, 13–17 May 2019; pp. 2000–2010.
15. Wang, H.; Zhao, M.; Xie, X.; Li, W.; Guo, M. Knowledge graph convolutional networks for recommender systems. In Proceedings of the World Wide Web Conference, San Francisco, CA, USA, 13–17 May 2019; pp. 3307–3313.
16. Guo, Q.; Zhuang, F.; Qin, C.; Zhu, H.; Xie, X.; Xiong, H.; He, Q. A survey on knowledge graph-based recommender systems. *IEEE Trans. Knowl. Data Eng.* **2020**, *34*, 3549–3568. [CrossRef]
17. Ji, S.; Pan, S.; Cambria, E.; Marttinen, P.; Philip, S.Y. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *33*, 494–514. [CrossRef] [PubMed]
18. Liu, H.; Wu, Y.; Yang, Y. Analogical inference for multi-relational embeddings. In Proceedings of the International Conference on Machine Learning, PMLR, Sydney, Australia, 6–11 August 2017; pp. 2168–2178.
19. Huang, J.; Zhao, W.X.; Dou, H.; Wen, J.R.; Chang, E.Y. Improving sequential recommendation with knowledge-enhanced memory networks. In Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, Ann Arbor, MI, USA, 8–12 July 2018; pp. 505–514.
20. Yu, X.; Ren, X.; Sun, Y.; Sturt, B.; Khandelwal, U.; Gu, Q.; Norick, B.; Han, J. Recommendation in heterogeneous information networks with implicit user feedback. In Proceedings of the 7th ACM Conference on Recommender Systems, Hong Kong, China, 12–16 October 2013; pp. 347–350.
21. 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, Halifax, NS, Canada, 13–17 August 2017; pp. 635–644.
22. Hu, B.; Shi, C.; Zhao, W.X.; Yu, P.S. Leveraging meta-path based context for top-n recommendation with a neural co-attention model. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 1531–1540.
23. Qu, Y.; Bai, T.; Zhang, W.; Nie, J.; Tang, J. An end-to-end neighborhood-based interaction model for knowledge-enhanced recommendation. In Proceedings of the 1st International Workshop on Deep Learning Practice for High-Dimensional Sparse Data, Anchorage, AK, USA, 5 August 2019; pp. 1–9.
24. 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, Xi'an, China, 25–30 August 2020; pp. 219–228.
25. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 5998–6008.
26. Memisevic, R.; Zach, C.; Pollefeys, M.; Hinton, G.E. Gated softmax classification. *Adv. Neural Inf. Process. Syst.* **2010**, *23*, 1603–1611.
27. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv* **2016**, arXiv:1609.02907.
28. Hamilton, W.; Ying, Z.; Leskovec, J. Inductive representation learning on large graphs. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 1024–1034.
29. Velickovic, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; Bengio, Y. Graph attention networks. *arXiv* **2017**, arXiv:1710.10903.
30. 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, Anchorage, AK, USA, 4–8 August 2019; pp. 950–958.
31. Rendle, S. Factorization machines with libfm. *ACM Trans. Intell. Syst. Technol.* **2012**, *3*, 1–22. [CrossRef]
32. Cheng, H.T.; Koc, L.; Harmsen, J.; Shaked, T.; Chandra, T.; Aradhye, H.; Anderson, G.; Corrado, G.; Chai, W.; Ispir, M.; et al. Wide & deep learning for recommender systems. In Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, Boston, MA, USA, 15 September 2016; pp. 7–10.

33. Wu, L.; Chen, L.; Shao, P.; Hong, R.; Wang, X.; Wang, M. Learning fair representations for recommendation: A graph-based perspective. In Proceedings of the Web Conference 2021, Ljubljana, Slovenia, 19–23 April 2021; pp. 2198–2208.
34. Huang, W.; Wu, J.; Song, W.; Wang, Z. Cross attention fusion for knowledge graph optimized recommendation. *Appl. Intell.* **2022**, *52*, 10297–10306. [CrossRef]