applied sciences

MDPI

Article

The Use of Attentive Knowledge Graph Perceptual Propagation for Improving Recommendations

Chenming Wang and Bo Huang *

School of Electronic and Electrical Engineering, Shanghai University of Engineering Science, Shanghai 201620, China
* Correspondence: huangbosues@sues.edu.cn

Abstract: Collaborative filtering (CF) usually suffers from data sparsity and cold starts. Knowledge graphs (KGs) are widely used to improve recommendation performance. To verify that knowledge graphs can further alleviate the above problems, this paper proposes an end-to-end framework that uses attentive knowledge graph perceptual propagation for recommendations (AKGP). This framework uses a knowledge graph as a source of auxiliary information to extract user–item interaction information and build a sub-knowledge base. The fusion of structural and contextual information is used to construct fine-grained knowledge graphs via knowledge graph embedding methods and to generate initial embedding representations. Through multi-layer propagation, the structured information and historical preference information are embedded into a unified vector space, and the potential user–item vector representation is expanded. This article used a knowledge perception attention module to achieve feature representation, and finally, the model was optimized using the stratified sampling joint learning method. Compared with the baseline model using MovieLens-1M, Last-FM, Book-Crossing and other data sets, the experimental results demonstrate that the model outperforms state-of-the-art KG-based recommendation methods, and the shortcomings of the existing model are improved. The model was applied to product design data and historical maintenance records provided by an automotive parts manufacturing company. The predictions of the recommended system are matched to the product requirements and possible failure records. This helped reduce costs and increase productivity, helping the company to quickly determine the cause of failures and reduce unplanned downtime.

Keywords: knowledge graph; recommendation system; multilayer propagation; attention mechanism; sparsity; cold start

check for updates

Citation: Wang, C.; Huang, B. The Use of Attentive Knowledge Graph Perceptual Propagation for Improving Recommendations. *Appl. Sci.* **2023**, *13*, 4667. https://doi.org/10.3390/app13084667

Academic Editor: José Machado

Received: 8 March 2023
Revised: 30 March 2023
Accepted: 5 April 2023
Published: 7 April 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

With the advancement of Internet technology, people can obtain lots of information by using computers or cell phones. It is difficult for people to quickly access the information they seek when faced with such a large amount of information, which leads to the problem of information overload [1,2], i.e., users cannot easily access the information they are searching for. On the other hand, providers of Internet content are facing a long-tail problem [3–5].

In response to the above problems, recommendation systems have become a solution. These systems have attracted much attention and have developed applications in academia and industry as they provide personalized items to meet users' interests and alleviate the problem of information explosion [6–8]. A recommendation system analyzes users' historical behaviors, preferences and interests to recommend content that meets their needs. It greatly reduces the time and cost of finding and selecting content and improves user experience and satisfaction. At the same time, it also provides Internet content providers with a more accurate target audience and higher revenue.

Early recommendation systems were mostly based on content-based recommendation algorithms [9,10], i.e., recommendations using the degree of similarity in content. The user's interests are modeled by analyzing the features and attributes of the items. The user's preferences are evaluated by calculating the similarity between their preferred items and other items. Although content-based recommendation algorithms have good explanatory power, they may suffer from "information filtering", i.e., if the user's history is too small or too homogeneous, the recommendation results may not be diverse enough.

Most subsequent work has focused on traditional collaborative filtering-based recommendation algorithms [11–18], which do not require prior feature extraction or modeling of items. Collaborative filtering-based recommendation algorithms are mainly divided into two types of algorithms: user-based [19,20], which recommends products of interest to similar users, and item-based [21,22], which recommends items to users of similar items. However, such algorithms generally suffer from user–item interaction data sparsity [23,24] (i.e., unsatisfactory recommendations when user history interaction data are sparse) and cold starts [25,26] (i.e., it is difficult to recommend products to new users while also recommending brand new items that have not been explored in depth).

To alleviate the shortcomings of the above algorithm problem, other information sources can be introduced, such as users' personal information [27], user comments [28,29], item attribute information [30] or knowledge graphs [31–38]. There is a large amount of auxiliary information that can be used by the recommendation algorithm, and the recommendation system can filter or sort the recommendation results based on this auxiliary information, so as to improve the diversity, real-time nature and accuracy of recommendations and meet the personalized needs of users.

A knowledge graph is essentially a semantic network graph information structure that consists of a set of entities and relationships between entities (which can also be called nodes and edges between nodes), and can be used to represent any type of knowledge [32]. As shown in Figure 1, the car manufacturing recommendation is used as an example. Different types of cars are considered as users and other entities as items. The knowledge graph stores various information on the production line, including cars, equipment, personnel, materials and production tasks, etc. Car1 and Car3 both prefer to use E1, E2 and E3 in their respective production lines, and they both use M2 and M3 in the selection of materials, which indicates that Car1 and Car3 have some correlation. The recommendation system can analyze this data and provide production planning and scheduling suggestions for the company. When the production demand of Car1 suddenly increases, the recommendation system can suggest shifting some production tasks to the production line of Car3 to reduce the pressure. It can also analyze equipment operation data, such as temperature, vibration and noise, to predict the health of the equipment. If a piece of equipment is about to break down, the recommendation system will send an early alert and schedule repairs. This helps reduce equipment downtime and production losses. In terms of human resources, the recommendation system can provide companies with reasonable personnel allocation suggestions based on production tasks and the equipment that workers can operate. W1 and W2 are similar in production tasks, and W1 and W3 can operate similar equipment. According to different relationships, different weights are assigned as a way to decide which relationship is more important.

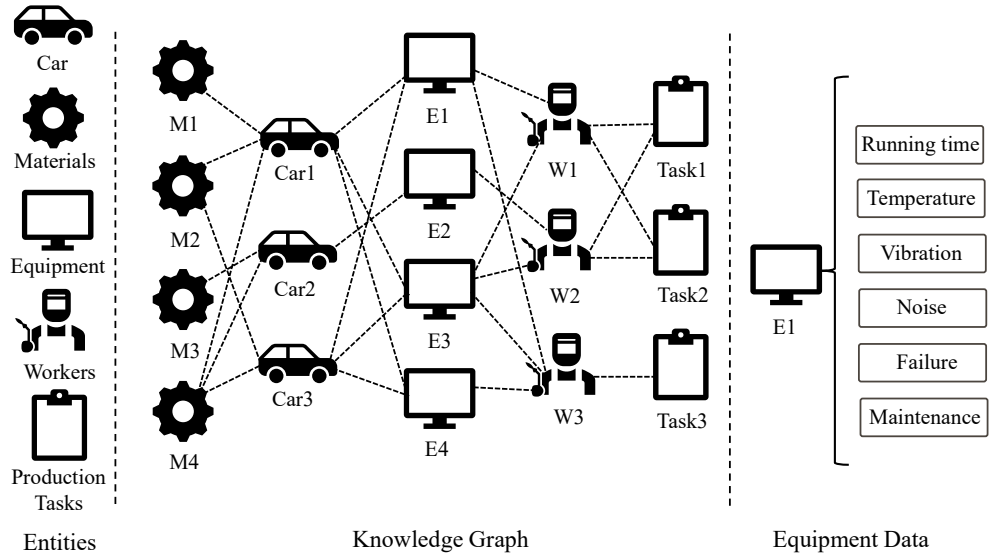


Figure 1. An example of car manufacturing scheduling recommendation with KG. Considering different models of cars as users, Car1 often uses E1, E2 and E3 on the production line, and commonly uses M2 and M3, similar to the production line situation of Car3. When the production demand of Car1 suddenly increases, the recommendation system will suggest transferring some production tasks to the production line of Car3 to reduce the pressure.

Users, items and their attributes can be mapped to a knowledge graph in a recommendation task, which can be used to further explore the potential relationship between users and items [33]. Feature learning through knowledge graphs can represent low-dimensional features of users and items [34], be useful for mining potential information at a deeper level of item granularity [36], and distinguish different levels of importance among items in a user's history [35], etc. Liang et al. proposed a Power Fault Retrieval and Recommendation Model (PF2RM) [39], where they use the graph-neighbor cluster polymerization method to update the cold-start prediction sequence for the power knowledge graph. Wang et al. proposed RippleNet [33], an end-to-end click-through prediction model using knowledge graphs as a source of auxiliary information, to continuously explore the potential interests of users by simulating the process of propagation of their interests. Users' interests are centered on their history records and, layer-by-layer, extend to other items or entities on the knowledge graph to achieve the propagation of users' interests. However, RippleNet is unable to distinguish the importance of items or entities within the propagation range. Wang et al. proposed the Knowledge Graph Convolutional Network (KGCN) [34], a recommendation method based on graph convolutional neural networks that can automatically capture higher-order structural and semantic information in the knowledge graph. Unlike RippleNet, which focuses more on the expansion of user history, KGCN focuses more on the expansion of item entities. Since users have different purchase intentions and items have different attributes, it is easy to ignore different aspects of user and item information, so the obtained embedding representation is suboptimal. Traditional recommendation algorithms assume that all user–item interactions are undifferentiated, which leads to learned user representations that contain only coarse-grained user–item interaction intent, i.e., the intent is coupled. Existing knowledge graph-based recommendation algorithms suffer from the problem that when modeling users using their histories, they do not mine information at the item granularity and cannot distinguish the importance of different items in the histories.

In response to the above limitations of existing recommendation algorithms, this paper proposes an end-to-end KG-based recommendation model named AKGP (attentive knowledge graph perceptual propagation for recommendation). The framework uses knowledge graphs as auxiliary information to obtain the embedded representations of

users and items, and through a multilayer propagation method, the extended entity sets and triples with different distances from the initial entity sets are obtained; these can effectively extend the potential vector representations of users and items. A knowledge-aware attention mechanism is used to generate different attention weights for tail entities to reveal their different meanings when different head entities and relationships are available. Knowledge graph embedding representations from propagation layer users and items are aggregated, and the predicted user preferences are output by the inner product. The model is further optimized using a stratified sampling strategy to retain entity features while reducing sample bias.

In summary, the contributions of this paper are as follows.

- This article proposes an end-to-end algorithmic framework, AKGP, that uses knowledge graphs as auxiliary information, fusing structured knowledge with contextual information to build fine-grained knowledge models.
- This article proposes an attention network that emphasizes the influence of different relationships between entities and explores users' higher-order interests through perceptual propagation methods. Moreover, a stratified sampling strategy is used to retain entity features and reduce sample bias.
- Extensive experiments are conducted on three datasets, and the experimental results demonstrate the effectiveness of the model in solving the sparsity and cold-start problems.

2. Related Work

Existing graph-based recommendation methods are divided into three main categories: embedding-based methods, path-based methods and propagation-based methods. This section briefly introduces these three methods.

2.1. Embedded-Based Methods

Direct relationships between connected entities in a knowledge graph are captured through a series of studies for entity embedding learning. Most of these approaches are based on traditional translation-based embedding techniques. For example, Zhang et al. proposed a model, Collaborative Knowledge Embedding (CKE) [37], based on collaborative filtering and knowledge graph embedding. It first obtains user and item vector representations via collaborative filtering. Then, the initial vector representation of items is enriched with text and image information related to the items. For knowledge information, CKE uses the knowledge graph embedding model TransR [40] to obtain knowledge representation. The embedding-based approach is more suitable for knowledge graph-related characters, such as connection prediction and node classification, and cannot fully exploit the allegorical richness and structure of knowledge graphs in recommendation tasks. Another example is Dynamic Knowledge-Aware Network (DKN) [41], which combines semantic layer representation and knowledge layer representation as different channels for news recommendation. Although significant improvements have been achieved, these approaches cannot capture the complex semantics of user–item connectivity because they only consider the direct relationships between entities in the knowledge graph. Zhang et al. proposed a KG recommendation model based on adversarial training (ATKGRM) [42], which can dynamically and adaptively adjust the aggregation weight of the knowledge graph, reasonably capture the content information of items, and better mine item features.

2.2. Path-Based Methods

Path-based approaches usually improve recommendations by exploring linear paths that connect pairs of entities in a knowledge graph. For example, Yu et al. proposed a meta-path-based recommendation method, Personalized Entity Recommendation (PER) [38], in known information networks, which forms a user preference feature matrix by pre-defining the user–item connectivity paths in the knowledge graph, and then, using a meta-path-based similarity algorithm to obtain the preference features between users and items. Although the user–item connectivity can be represented by different meta-path patterns,

which provide good recommendation effects and interpretability, defining effective meta-paths requires domain knowledge, which can be quite laborious for complex knowledge graphs with different entities and relationships. Moreover, path-based approaches are usually limited by the linear path representation capability when describing complex user–item connectivity.

2.3. Propagation-Based Methods

One can simulate user–item connections by propagating information across a knowledge graph. For example, KGCN [34] propagates item information in a knowledge graph through graph convolutional networks (GCN) to generate better item embeddings. Wang et al. proposed the Knowledge Graph Attention Network (KGAT) [35], an attention mechanism recommendation algorithm based on a knowledge graph, which fuses knowledge graph and user–item interaction information into the same icon in a space named the collaborative knowledge graph (CKG). Using a neural network framework, a multi-order neighbor information aggregation is implemented to model higher-order information, iteratively propagate information about entity neighbors to optimize their embedding and use attention mechanisms to distinguish the importance of neighbors. Zhang et al. proposed a model, Enhance Knowledge Propagation Perception Net (EKPNet) [43], where an attention mechanism with asymmetric semantics for the KG propagation is designed. It enhances the mining of user preferences by mapping the semantics of head and tail entities into different preference spaces. Although the above approaches have explored knowledge graph topology, the diffusion of information over the entire knowledge graph will inevitably introduce noise unrelated to the connection of specific user items, thus adversely affecting the approaches' embedded learning capabilities. In addition, these approaches do not explicitly encode user–item relatedness, but model it using only user and item embeddings, and thus, cannot fully exploit their relatedness to reveal user preferences.

3. Preliminaries

This section explains the notation used in the framework and introduces the flow of the recommendation task as a way to better represent our AKGP model.

This paper adopts the same setup as most knowledge graph-based recommendation system approaches, and uses $\mathcal{U} = \{u_1, u_2, \cdots, u_m\}$ to denote the set of users and $\mathcal{V} = \{v_1, v_2, \cdots, v_n\}$ to denote the set of items, where m and n denote the total number of users and items. In the recommendation scenario, the user–item interaction is recorded in the form of an interaction matrix, denoted as $Y = \{y_{uv} | u \in \mathcal{U}, v \in \mathcal{V}\}$, where y_{uv} is denoted as the element of the u row and the v column of the interaction matrix. When $y_{uv} = 1$, an interaction has been recorded between user u and item v, such as a click or a purchase. When $y_{uv} = 0$, no interaction has been recorded between the uth user and the vth item. The formula is as follows.

$$y_{uv} = \begin{cases} 1, & \text{User } u \text{ has interacted with item } v \\ 0, & \text{User } u \text{ has no interaction with item } v \end{cases} \quad (1)$$

In addition, this section introduces knowledge graphs as additional auxiliary information in the recommendation system. A knowledge graph is a method of organizing knowledge, such as entities, concepts, relationships and attributes into a graphical structure. It is a semantic network for representing and storing knowledge, similar to human thought patterns, in a collection of triples consisting of $\langle head, relation, tail \rangle$, representing a knowledge graph in which two entities can be related to each other. The knowledge graph is denoted by $\mathcal{G} = (\mathcal{E}, \mathcal{R})$, and generally, h, r and t are used to denote the head entities, relationships and tail entities in the knowledge graph triad, respectively, where $\mathcal{E} = \{e_1, e_2, \cdots\}$ denotes the set of entities and $\mathcal{R} = \{r_1, r_2, \cdots\}$ denotes the set of relationships. The alignment between items and entities is denoted by $\mathcal{A} = \{(v, e) | v \in \mathcal{V}, e \in \mathcal{E}\}$, i.e., every item v in the item set has an entity element e with which it is mapped in the entity set.

In this paper, the user–item interaction matrix Y and the knowledge graph $\mathcal{G}$ are used as inputs, and the aim of the recommendation task is to predict whether the user u is interested in an item v with which they have not interacted, i.e., the probability $\hat{y}_{uv}$ of user u clicking on item v. The prediction value $\hat{y}_{uv}$ will be a real number between 0 and 1, and a value closer to 1 means that the user is more interested in the item. In the recommendation scenario, after obtaining the predicted value of the user's preference for the item, the item is recommended to the user in descending order according to the predicted value, which completes the recommendation task.

4. Methodology

This section provides an overview of the AKGP framework and describe, in detail, each of the components, including user–item initial embedding representation, the multilayer perceptual propagation and attention network, aggregated prediction and hierarchical sampling optimization. The overall schematic of the framework is shown in Figure 2.

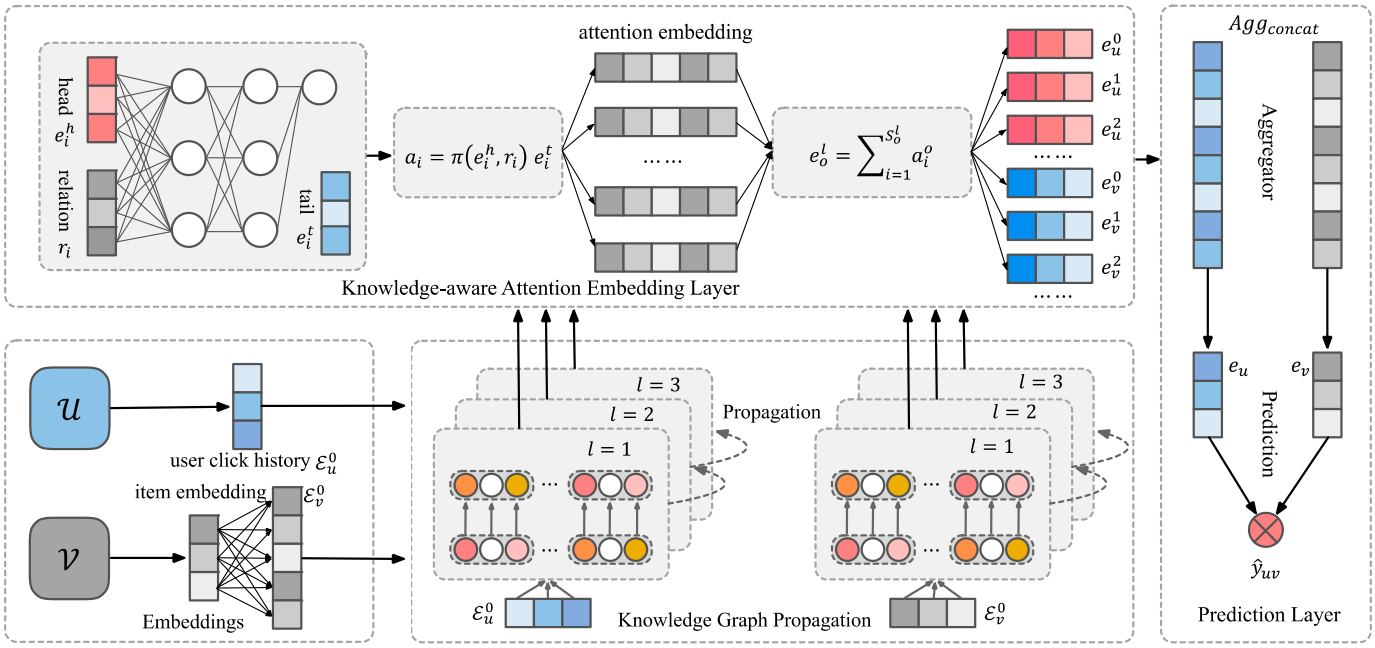


Figure 2. The general structure of the AKGP model. Upon inputting user–item interaction information and knowledge graph, the model calculates the weights of entities with different relationships through attention networks and propagates them in the knowledge graph as propagation factors. This enables us to aggregate and predict user preferences.

AKGP takes the user–item historical interaction information and the user–item related knowledge information in the knowledge graph as input, and models the structured knowledge at a fine-grained level using the TransD [44] and hierarchical modeling methods. The user–item information is projected onto multiple vector spaces to obtain multifaceted fine-grained embedding vectors. To explore the higher-order interests of users, item v is associated with entity e in the knowledge graph, and user preferences are propagated on the knowledge graph through correspondence between the user embedding representation and entity e in the knowledge graph. In the propagation process, an attention network is used to obtain the weights of different relationships in the knowledge graph that affect the user's interest, and the relationships are used as propagation factors over a certain number of neighbors to generate user-embedded representations and item-embedded representations. These embedded representations are aggregated for Click-Through Rate (CTR) prediction and Top-K recommendation. Based on the prediction results, the model is optimized using a stratified sampling strategy to reduce sample bias.

4.1. Initial Embedding Representation

The historical user–item interaction data, the specific corresponding set of domain knowledge graph triples, and the mapping relationships of entity items are collected to generate the corresponding embedded representations, which act as the input of this layer.

4.1.1. Knowledge Graph Embedding

Usually, a triple $\langle head, relation, tail \rangle$ is used to represent knowledge. Since there are too many entities and relationships with too many dimensions, one-hot vectors cannot capture the similarity when two entities or relationships are very close. Inspired by Wrod2Vec [45], the entities and relationships in this paper are represented by the distributed representation method. Additionally, the recommendation scenario may encounter a situation whereby user–item interactions are very sparse, in which case, the above method can be combined with user–item interaction information by introducing the knowledge graph as an auxiliary information method to improve the recommendation effect. Knowledge graphs generally contain three types of data: structured knowledge (triples), textual knowledge, and visual knowledge. This paper uses the TransD knowledge graph embedding method to model structured knowledge and learn the structured representation of each item in the knowledge graph.

The aim of the TransD approach is to project the head and tail onto the hyperplane using the same transition matrix, as shown in Figure 3. However, the head and tail are usually not entities of the same category (e.g., Robert Downey Jr., actor, Iron Man). In the given example, Robert Downey Jr. is an actor and Iron Man is a movie, which are two different categories of entities that should be transformed in different ways. Second, the projection should be related to the entities and their relationships, but the projection matrix is only determined by the relationships. TransD uses two vectors to represent each entity and relationship. The first vector represents the meaning of the entity or relationship, and the projection vector is used to construct the mapping matrix [44]. The formula is as follows:

$$M_{rh} = r_p h_p^T + I \tag{2}$$

$$M_{rt} = r_p t_p^T + I \tag{3}$$

where the mapping matrix is defined by entities and relationships, I is the unit matrix, and the generated matrix is used to modify the unit matrix. Like the classical Trans series knowledge graph embedding algorithm, the distance between entities is used to represent the score. The formula is as follows:

$$f_r(h,t) = \|h_\perp + r - t_\perp\|_2^2 \tag{4}$$

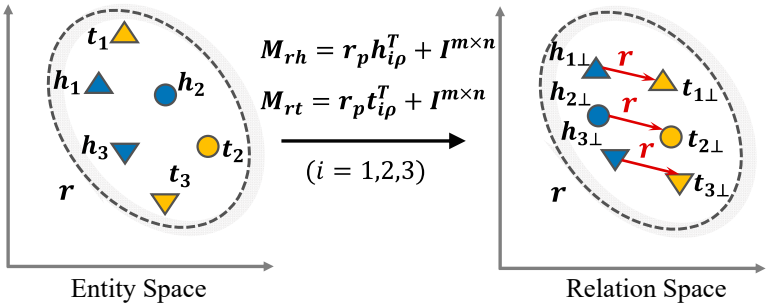


Figure 3. Schematic diagram of the TransD approach. The head and tail are projected to the hyperplane through the transition matrix and connected by relationships.

Since TransD is calculated on the hyperplane, it is obtained according to the mapping matrix.

$$h_{\perp} = M_{rh}h \tag{5}$$

$$t_{\perp} = M_{rt}t \tag{6}$$

The loss function uses a max-margin function with negative sampling. The formula is as follows:

$$L(h,r,t) = max\Big(0, f_r(h,t)_{pos} - f_r(h,t)_{neg} + margin\Big) \tag{7}$$

4.1.2. Fine-Grained Hierarchical Modeling

In coarse-grained modeling algorithms, the embedded representations of users and items are usually projected onto only one interest space and one feature space. This modeling approach assumes that there is only a single relationship between the interaction behaviors of users and the items, which cannot reflect the multiple interests and multiple intents of the interactions of users themselves, or the multiple attributes and multiple uses of items. The k vector spaces can be defined as the entities and relationships in the knowledge graph (corresponding to k mapping matrices $\{M_{\alpha,0}, M_{\alpha,1} \cdots, M_{\alpha,k-1}\}$, where $\alpha \in \{e, r\}$, $e \in \mathcal{E}$, $r \in \mathcal{R}$), thus producing a fine-grained model of the information in the knowledge graph and facilitating subsequent vector computation.

4.1.3. Initial User–Item Embedding Representation

The learned embedding information for individual entities is still limited, and additional contextual information needs to be extracted for each entity in order to help identify the location of the entity in the knowledge graph. As an example of a movie recommendation, a user has seen the movies "Deadpool" and "Green Lantern". Both movies are linked in the knowledge graph to the actor Ryan Reynolds, and the recommendation system may infer that the user is interested in movies starring Ryan Reynolds. Thus, it can recommend other movies and TV shows to this user that have starred Ryan Reynolds, such as "Free Guy" and "The Hitman's Bodyguard". Through the contextual information of entities in the knowledge graph, the recommendation system can identify potential connections between users and items, thus improving the performance of the recommendation system.

Predefined knowledge graphs can be used to associate user and project entities. Based on the identified entities, the subgraphs are constructed separately and all the relational links between them are extracted from the original knowledge graph. Since the relationships between the identified entities may be sparse and lack diversity, the current knowledge subgraph is extended to within one hop of the identified entities.

The user–item interaction information contains users' overall preferences, and the knowledge graph contains the auxiliary knowledge information related to users and items. The user–item interaction graph and the knowledge graph can be fused through the mapping relationship between items and entities. This paper uses a set $\mathcal{A} = \{(v,e)|v \in \mathcal{V}, e \in \mathcal{E}\}$ to represent the alignment of items with entities in the knowledge graph, so the initial set of entities for users is defined as follows:

$$\mathcal{E}_u^0 = \{e|(v,e) \in \mathcal{A}\ and\ v \in \{v|y_{uv} = 1\}\} \tag{8}$$

Similarly, the set of composite items $\mathcal{V}_u = \{v_u|u \in \{u|y_{uv} = 1\}\ and\ y_{uv_u} = 1\}$ and the set of pairs $\mathcal{A} = \{(v,e)|v \in \mathcal{V}, e \in \mathcal{E}\}$, which represent the initial set of entities of item v, are defined as follows:

$$\mathcal{E}_v^0 = \{e|(v_u,e) \in \mathcal{A}\ and\ v_u \in \mathcal{V}_u\} \tag{9}$$

The order interaction information can most effectively represent the underlying semantics of entities, and the initial embedding layer explicitly encodes it into the initial set of entities to enhance the representation of users and items and improve the recommendation

effect. This can also continuously emphasize the information of the original items and reduce bias due to multi-layer propagation.

4.2. Multi-Layer Perceptual Propagation and Attention Networks

This paper designs a multi-layer perceptual propagation method. The neighboring entities in the knowledge graph are always strongly related, so by propagating along the links in the knowledge graph, the extended entity sets and triples with different distances from the initial entity sets can be obtained, which can effectively extend the potential vector representations of users and items, as shown in Figure 4.

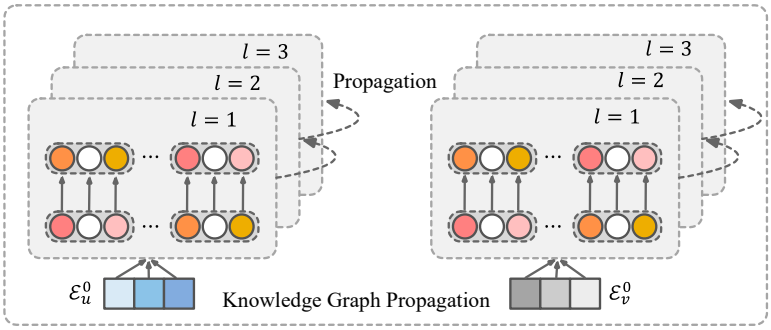


Figure 4. Structural diagram of knowledge graph propagation. The initial set of entities is similar to a sound wave source that propagates information layer-by-layer, from near to far, in the knowledge graph. Higher-order interaction information features of users and items are obtained via propagation.

The entity set definition of user u is recursively represented as follows:

$$\mathcal{E}_u^l = \left\{ t \middle| (h,r,t) \in \mathcal{G} \text{ and } h \in \mathcal{E}_u^{l-1} \right\} \tag{10}$$

Similarly, the entity set definition of item v is recursively represented as follows:

$$\mathcal{E}_v^l = \left\{ t \middle| (h,r,t) \in \mathcal{G} \text{ and } h \in \mathcal{E}_v^{l-1} \right\} \tag{11}$$

where $l = \{1,2,3,\cdots,L\}$ denotes the distance from the initial entity set. In addition, according to the definition of the entity set, the lth ternary set of user u is defined as follows:

$$S_u^l = \left\{ (h,r,t) \in \mathcal{G} \text{ and } h \in \mathcal{E}_u^{l-1} \right\} \tag{12}$$

Similarly, the lth ternary set of item v is defined as follows:

$$S_v^l = \left\{ (h,r,t) \in \mathcal{G} \text{ and } h \in \mathcal{E}_v^{l-1} \right\} \tag{13}$$

Using a knowledge graph as a source of edge information can enable us to construct more efficient models, since neighboring entities can be considered uncertain extensions of user preferences and item characteristics. The initial set of entities obtained through the initial embedding layer is similar to a sound wave source that propagates information layer-by-layer, from near to far, in the knowledge graph. Deep propagation based on the knowledge graph successfully acquires the higher-order interaction information features of users and items, which effectively improves the ability of the model to represent users and items under each behavior using potential vectors, thus alleviating the data sparsity and cold-start problems.

When each tail entity has different head entities and relationships in the knowledge graph, each tail entity has a different meaning and potential vector representation. Therefore, this paper proposes a neural network-based attention mechanism to generate different

attention weights of tail entities to express their different meanings when they have different head entities and relationships.

The attention embedding of the ith triplet of the ith layer is as follows:

$$a_i = \pi\left(e_i^h, r_i\right) e_i^t \tag{14}$$

where e_i^h is the embedding of the head entity, which is the embedding of the r_i relationship, and e_i^t is the embedding of the tail entity of the ith triplet. $\pi\left(e_i^h, r_i\right) e_i^t$ denotes the attention weights generated by the head entity and the relationship between the head and the tail. This section implements the function $\pi(\cdot)$ by means of a neural network-based attention mechanism. The formula is as follows:

$$\begin{aligned} \pi\left(e_i^h, r_i\right) &= \sigma\Big(W_2 ReLU\Big(W_1 ReLU\Big(W_0\Big(e_i^h \parallel r_i\Big) + b_0\Big) + b_1\Big) + b_2\Big) \\ &= \frac{exp\left(\pi\left(e_i^h, r_i\right)\right)}{\sum_{\left(h', r', t'\right) \in S_o^l} exp\left(\pi\left(e_i^{h'}, r_i'\right)\right)} \end{aligned} \tag{15}$$

where W and b are trainable weight matrices and deviation values, and their subscripts indicate that they are parameters of different layers; o denotes user u or item v; and S_o^l denotes the lth layer triad of user u or item v. This paper chooses $ReLU$ as the nonlinear activation function, uses the $Sigmoid$ activation function for series operation, and finally, uses the $softmax$ function to normalize the coefficients of the whole triad. Attention weights can indicate which neighboring tail entities should be given more attention in order to capture the knowledge graph more effectively. The final representation of the lth level triple for user u and item v is obtained as follows:

$$e_u^l = \sum_{i=1}^{S_u^l} a_i^u \tag{16}$$

$$e_v^l = \sum_{i=1}^{S_v^l} a_i^v \tag{17}$$

This yields a set of attention-weighted representations of user u and item v based on the knowledge graph, as follows:

$$\mathcal{W}_u = \left\{e_u^0, e_u^1, e_u^2, \cdots e_u^L\right\} \tag{18}$$

$$\mathcal{W}_v = \left\{e_v^0, e_v^1, e_v^2, \cdots e_v^L\right\} \tag{19}$$

Then, according to the importance of different relationships for user u and item v, this is used as a propagation factor to continue the propagation of the knowledge graph. Finally, the propagation stops after l layers, and the final entity e is obtained as the final representation of the user u and item v.

4.3. Aggregate and Prediction

The knowledge representation in each layer can be understood as the potential impact of different higher-order correlations and preference similarities. This paper uses the *Concat* aggregator, which aggregates knowledge graph embedding representations of the user u and item v from the propagation layer, to aggregate multiple entity representations from the equation into a single vector of users and items.

The *Concat* aggregator connects the representation vectors in the representation set and uses a nonlinear transformation as follows:

$$agg_{concat}^u = \sigma\left(W \cdot \left(e_u^0 \parallel e_u^1 \parallel \cdots \parallel e_u^l\right) + b\right) \tag{20}$$

$$agg_{concat}^{v} = \sigma\left(W \cdot \left(e_v^0 \parallel e_v^1 \parallel \cdots \parallel e_v^l\right) + b\right) \tag{21}$$

This paper uses e_u to represent the aggregation vector of users and e_v to represent the aggregation vector of items. Finally, users' preference scores for items are obtained using the inner products of the entity representations as follows:

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

4.4. Stratified Sampling and Iterations

Stratified sampling refers to stratifying a large collection of samples by feature or attribute, and then, randomly sampling each layer. In recommendation systems, the stratified sampling strategy can be used to solve the cold-start problem in migration learning.

Specifically, suppose there are two domain datasets (A and B). Dataset A contains the historical purchase records and rating information of users, while dataset B does not contain the historical records of users. To solve the cold-start problem in B, we can use the stratified sampling strategy to extract a certain number of samples from A for knowledge migration. The users in dataset A are stratified according to their purchase history or ratings, and then, a certain number of users and products are sampled from each layer, and these samples are used to train the recommendation model; then, the trained model is applied to B for recommendation. This method can retain the characteristics of the samples and avoid sample bias, and can also effectively use the historical data in A to carry out knowledge migration and thus improve the effectiveness of recommendations.

In this paper, a stratified sampling strategy is proposed. The data obtained by performing aggregated predictions are used as a sample set. The items are divided into multiple groups according to the user's degree of preference for the items. Items with similar preference values are grouped into the same group, and the sum of the preference levels of the samples in each group is the same. Then, the items in each group are identified as positive or negative according to the sample labels. In a similar way to supervised learning, the item characteristics are preserved, and sample bias is avoided when using stratified sampling. Additionally, the learning process of the model is continuously iterated, as shown in Figure 5.

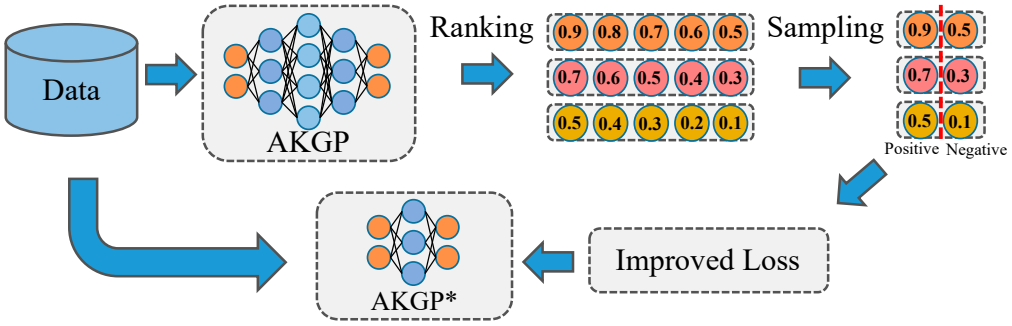


Figure 5. Structure diagram of stratified sampling method. The prediction samples obtained from AKGP are first grouped and arranged in descending order according to the preference of the samples. Then, samples with similar preference values are placed in the same group. The sum of the preference levels of samples in each group is the same. Then, the samples in each group are identified as positive or negative according to the sample labels. Finally, the scores are input into the adversarial training loss function, which is iterated into AKPG* model.

In order to solve the category imbalance problem, this paper ensures the effect of model training by balancing the number of positive and negative samples, and extracting the same number of negative samples as positive samples for each user. The algorithm

uses cross-entropy as the loss function of the whole AKGP model. The final loss function is as follows.

$$L = -\sum_{u \in U} \left(\sum_{v \in V} (y_{uv} \log \hat{y}_{uv} + (1 - y_{uv}) \log(1 - \hat{y}_{uv})) \right) + \lambda L_p \quad (23)$$

where L_p denotes a regularization term for the AKGP framework, which is added to the basic loss function to prevent overfitting, and λ is the regularization parameter.

5. Experiments

5.1. Datasets

To evaluate the performance of the AKGP framework proposed in this paper, three publicly available datasets (MovieLens-1M, Last-FM and Book-Crossing) were chosen as benchmark datasets for the experiments.

MovieLens-20M [46]: This dataset is a widely used benchmark dataset for movie recommendation, collected by the MovieLens website, that contains 20 million movie rating data and 465,000 marker data with ratings ranging from 1 to 5. The dataset contains six files, such as movies.csv, which contains movie information.

Last-FM: This dataset is a music listening dataset, collected from the Last.fm online music system, that contains users' play records and tagging information for songs, where tracks are considered items. The dataset contains two files.

Book-Crossing: This dataset contains ratings and tagging information for over 8,000,000 books from over 270,000 users in the Book-Crossing community, with ratings ranging from 0 to 10. The dataset contains three files, such as BX-Users.csv, which contains user information.

The user–item interactions for all three datasets are displayed in Table 1.

Table 1. Detailed statistics of the three datasets (# means 'the number of').

Data Type	Element	Last.FM	Book-Crossing	MovieLens-20M
User–Item Interaction	#Users	1872	19,676	138,159
	#Items	3846	20,003	16,954
	#Interactions	42,346	172,576	13,501,622
Knowledge Graph	#Entities	9366	25,787	102,569
	#Relationships	60	18	32
	#Triples	15,518	60,787	499,474

5.2. Baselines

To verify the effectiveness and generality of the AKGP framework proposed in this paper, models such as BPRMF (Bayesian Personalized Ranking Matrix Factorization) [47], NEUMF (Neural Matrix Factorization) [11], CKE [37], RippleNet [33], KGCN [34] and KGAT [35] were selected for use in the experiments.

BPRMF [47] is a recommendation model based on matrix decomposition. It uses Bayesian theory to maximize the posterior probability based on prior knowledge, and also uses two-by-two matrix decomposition to optimize implicit feedback.

NEUMF [11] is a recommendation model based on neural collaborative filtering. It uses the flexibility and nonlinearity of neural networks instead of the dot product of matrix decomposition to improve the expressiveness of the model with implicit feedback.

CKE [37] is an embedding-based model. It combines collaborative modules with structured, textual knowledge embeddings in a unified Bayesian framework. User and item representations are obtained via collaborative filtering. The initial item representations are then enriched with knowledge information related to the items.

RippleNet [33] is a propagation-based model. It is continuously extended to enrich user representation via preference propagation of the knowledge graph.

KGCN [34] is a propagation-based model. It captures the structural and semantic information of the knowledge graph by selectively and biasedly aggregating neighborhood

information. The neighbors of each entity are used as perceptual domains, which are extended to multi-hop neighbors to model higher-order neighborhood information, thus mining information about the personalization and potential interest of users.

KGAT [35] integrates the knowledge graph and a user–project interaction graph into the same graph space, known as collaborative knowledge graph. The modeling of higher-order information can be achieved using a collaborative knowledge graph with a three-layered structure of information dissemination, knowledge-based attention and information aggregation.

5.3. Experimental Settings

To ensure fairness, all baseline algorithms were trained using an Adam optimizer, and the learning rate was set to 0.01, the embedding dimension for the experiments was 64 and the batch size was 1024. The relevant parameters were set as shown in Table 2.

Table 2. Experimental settings.

Parameter Name	Parameter Value
Optimizer	Adam
Learning Rate	0.001
Embed Dimension	64
Batch Size	1024
L2 weighs	1×10^{-4}

The dataset was divided into training, validation and test sets in a 6:2:2 ratio. In this paper, the AKGP model is evaluated in two experimental scenarios as follows [48]. (1) CTR prediction: the trained model is applied to the test set to predict the click rate. The performance of CTR prediction is evaluated using AUC, F1. (2) Top-K recommendation: the trained model is used to select the K item with the highest predicted click rate for each user in the test set. This paper chooses Recall@K and Precision@K to evaluate the set of recommendations at the baseline. The formula was as follows:

$$AUC = \frac{TP + TN}{TP + TN + FP + FN} \tag{24}$$

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} \tag{25}$$

$$Precision = \frac{TP}{TP + FP} \tag{26}$$

$$Recall = \frac{TP}{TP + FN} \tag{27}$$

where TP denotes the prediction of positive samples as positive samples and FN denotes the prediction of negative samples as positive samples.

5.4. Experimental Results

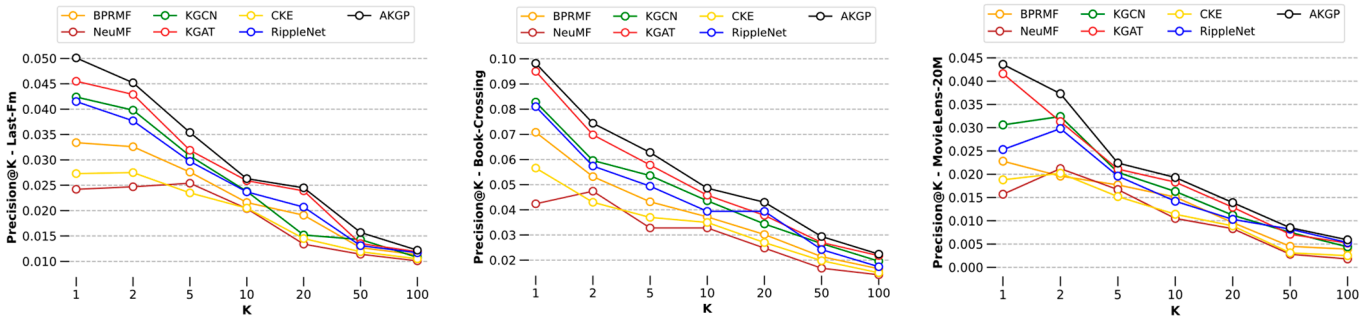
5.4.1. Performance Comparisons with Baselines

Table 3 shows the experimental results of the AUC and F1 metrics of the AKGP algorithm and the benchmark algorithm in the click-through rate prediction scenario.

Table 3. Performance comparisons with baselines.

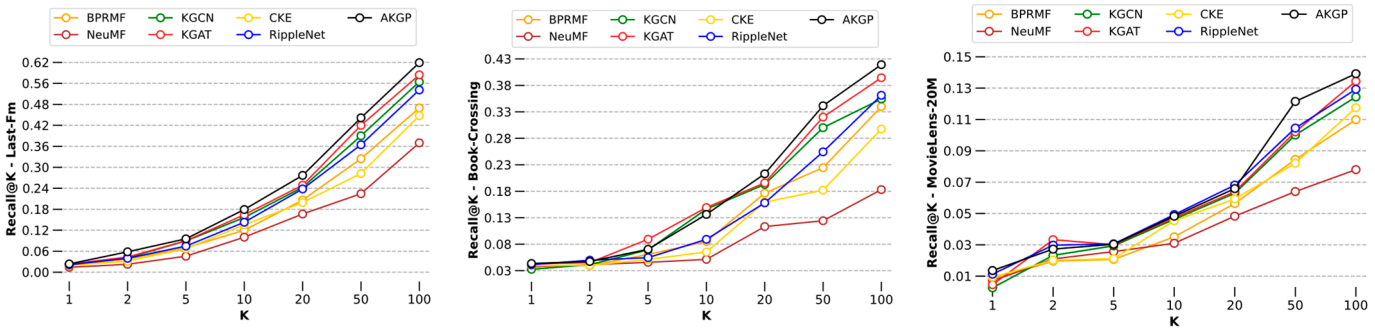
Model	Last-FM		Book-Crossing		Movielens-20M	
	AUC	F1	AUC	F1	AUC	F1
NEUMF	0.725 (−12.5%)	0.645 (−14.8%)	0.631 (−14.9%)	0.603 (−8.5%)	0.904 (−7.3%)	0.906 (−2.1%)
BPRMF	0.766 (−7.6%)	0.692 (−6.6%)	0.665 (−10.4%)	0.612 (−7.1%)	0.903 (−7.4%)	0.913 (−1.3%)
CKE	0.743 (−10.3%)	0.675 (−8.9%)	0.678 (−8.6%)	0.620 (−5.9%)	0.912 (−6.5%)	0.906 (−2.1%)
RippleNet	0.778 (−6.2%)	0.702 (−5.3%)	0.720 (−2.9%)	0.645 (−2.1%)	0.921 (−5.5%)	0.920 (−0.5%)
KGCN	0.802 (−3.2%)	0.722 (−2.6%)	0.686 (−7.5%)	0.631 (−4.2%)	0.974 (−0.2%)	0.924 (−0.1%)
KGAT	0.825 (−0.4%)	0.739 (−0.3%)	0.725 (−2.3%)	0.654 (−0.7%)	0.972 (−0.4%)	0.922 (−0.3%)
AKGP	0.829	0.741	0.742	0.659	0.976	0.925

As seen in the combined experimental results of each algorithm using the three datasets, most of the recommendation algorithms that incorporated knowledge graphs outperform the classical collaborative filtering algorithms, which indicates the positive impact of external auxiliary information on the recommendation effect. The neural collaborative filtering-based recommendation algorithm NEUMF performs the worst of all the benchmark algorithms among the different models for each metric recommendation. This may be due to its difficulty in making recommendations for data that do not appear in the user–item interaction matrix. The result for the embedding-based knowledge graph recommendation algorithm (CKE) is slightly lower than that of BPRMF for most metrics. This could be because it only considers the structural information of the knowledge graph and ignores the influence of its domain information, so it cannot effectively explore the potential interests of users. The recommended performance of AKGP is excellent, especially in the Last-FM and Book-Crossing datasets. Compared with baseline models such as BPRMF, NEUMF, CKE, RippleNet, KGCN, etc., the experimental results of AKGP validate the advantages of the fine-grained modeling of knowledge information and the effectiveness of using attention networks and perceptual propagation methods. A comparison of the precision and recall metrics for the Top-K recommendation scenario is shown in Figures 6 and 7. It can be seen that AKGP outperforms other benchmark algorithms in terms of recall for multiple K values.



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

Figure 6. The results of Precision in Top-K recommendation. (**a**) Results using the Last-FM dataset. (**b**) Results using the Book-Crossing dataset. (**c**) Results using the MovieLens-20M dataset.



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

Figure 7. The results of Recall in Top-K recommendation. (**a**) Results using the Last-FM dataset. (**b**) Results using the Book-Crossing dataset. (**c**) Results using the MovieLens-20M dataset.

5.4.2. Performance Comparisons with Variants

To further validate the effectiveness of the components of the AKGP framework, we compared three AKGP variants using ablation experiments. AKGP/E ablates parts of the knowledge graph embedding methods, fine-grained modeling and the extraction of contextual information. The entities and relationships in the user, item and knowledge graphs are represented by only one fixed-length embedding vector. AKGP/A represents the removal of the attention network from the AKGP framework. AKGP/S does not use hierarchical sampling methods, and instead uses direct summation pooling whereby it simply sums the pooling of domain information.

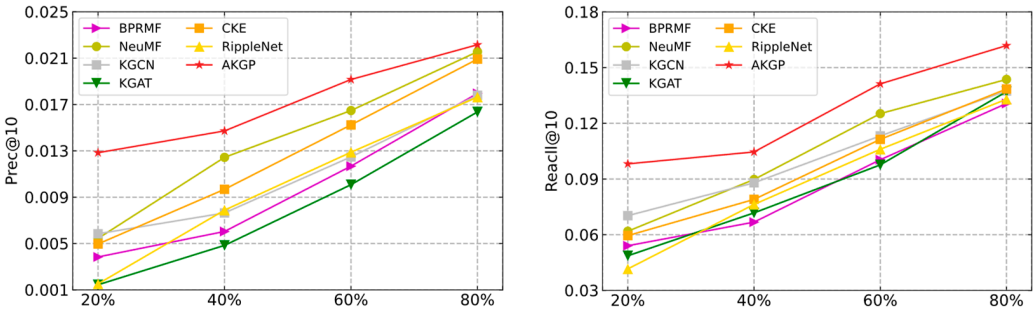
Table 4 shows the results of the AUC metrics for the three variants of the model. From the experimental results, it is clear that AKGP/E has a significant performance loss in all three datasets, which indicates that the user–item representations obtained from coarse-grained modeling are suboptimal and do not result in the best recommendation performance. The AUC metrics of AKGP/A and AKGP/S show a decrease in all three datasets, which reflects the importance of distinguishing domain information when aggregating domain information in attention networks, and the importance of hierarchical sampling optimization.

Table 4. Comparison of performance among variations of AKGP.

Model	Last-FM	Book-Crossing	MovieLens-20M
AKGP/E	0.788	0.702	0.898
AKGP/A	0.723	0.642	0.846
AKGP/S	0.791	0.727	0.912
AKGP	0.829	0.742	0.976

In summary, every part of the AKGP framework is necessary, and the results obtained after removing any of those parts are suboptimal.

Recommendation systems generally face the challenges of cold-start and data sparsity problems. The cold-start problem refers to the lack of sufficient interaction data to capture the interest of new users, which affects the accuracy of recommendations. The data sparsity problem refers to the fact that many users have only interacted with a few items, which leads to the inability of recommendation systems to provide accurate recommendations to users. To investigate whether the AKGP model can mitigate the effects of the cold-start problem and the data sparsity problem, the training data were divided into five equal sub-data of 20%, 40%, 60% and 80%, and the comparison results are shown in Figure 8. The results prove that the AKGP model performs well, especially in the case of sparse interaction records, and the improvement decreases as the training data increases.



(**a**) Prec@10 (**b**) Recall@10

Figure 8. Performance comparison over the sparsity distribution of data on Last-FM. (**a**) Prec@10 with increase in Data. (**b**) Recall@10 with increase in Data.

In AKGP, knowledge graph embedding methods, such as TransE [49], TransH [50], TransR [40] and TransD [44], are used for knowledge graph complementation tasks to extend the potential relationships between knowledge graph entities. The effects of using different representation learning methods on the AUC are shown in Table 5. The experiments show that using TransD for the representation learning of knowledge graph entities in AKGP is more effective than using other methods. TransD is the most complex model among the four embedding methods. It maps entity embeddings to the corresponding relationship space and learns low-dimensional representation vectors for each entity and relationship to preserve the structural information of the knowledge graph.

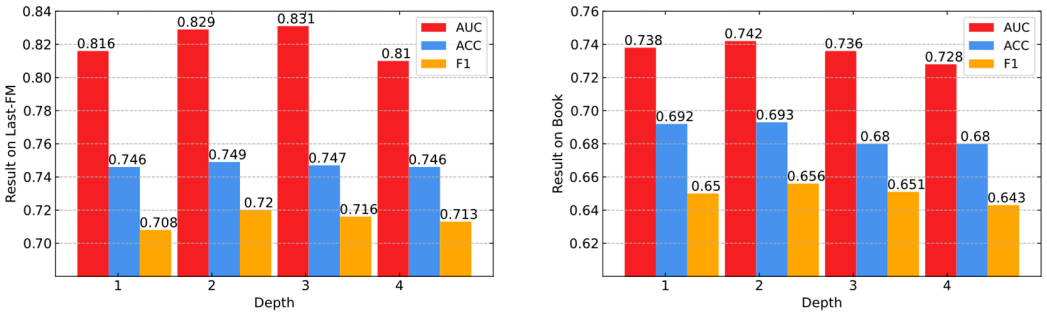
Table 5. Influence of different representation learning methods on AUC.

Model	Last-FM	Book-Crossing	MovieLens-20M
AKGP-TransE	0.815	0.723	0.973
AKGP-TransH	0.824	0.732	0.964
AKGP-TransR	0.830	0.735	0.957
AKGP-TransD	0.831	0.742	0.976

The effect on the AUC of changes in propagation depth in the multilayer propagation layer of AKGP is shown in Table 6 and Figure 9. From the experimental results, the performance of the AUC metric is optimal when the propagation depths of the three data sets are 1, 3 and 2. The experiments show that in different datasets, when the propagation depth is appropriately increased, extended entity sets and triples with different distances from the initial entity set are obtained. This can effectively extend the potential vector representation of users and items using external information, and enhance the capability of embedding representation. From the experimental results, it can be concluded that if the propagation depth is beyond a certain range, the distance of users and items from external information decreases, as does their relevance, which will interfere with and reduce this method's embedding representation ability.

Table 6. Influence of different propagation depths on the AUC.

Depth	Last-FM	Book-Crossing	MovieLens-20M
1	0.813	0.724	0.976
2	0.827	0.742	0.965
3	0.831	0.733	0.947
4	0.825	0.721	0.938



(**a**) Last-FM (**b**) Book-Crossing

Figure 9. The results of different propagation depths on two datasets. (**a**) Results on the Last-FM dataset. (**b**) Results on the Book-Crossing dataset.

6. Conclusions and Future Work

To address the data sparsity and cold-start problems of recommendation models, this paper proposes a recommendation algorithm, AKGP, that incorporates attention and knowledge graph-aware propagation. User–item interaction information and a knowledge graph are used as inputs. Using the knowledge graph embedding method, fine-grained modeling and the extraction of contextual information, the interests of users are effectively mined. The weights of entities with different relationships are computed through attention networks and used as propagation factors to propagate information from the knowledge graph, in order to extend domain information on users and items. Domain information is aggregated and preference values predicted using a stratified sampling strategy to retain item information features while avoiding sample bias. Finally, an end-to-end recommendation framework is formed. In this study, we conducted comparative experiments on publicly available datasets for three different scenarios. The results show that the AKGP model achieves significant performance improvement in both CTR prediction and Top-K recommendation compared to the baseline algorithm. The ablation experiments further validated the effectiveness of this method.

In addition, we applied the model to the product design data and historical maintenance records provided by the project partner of this paper (an automotive parts manufacturing company). The knowledge graph is used to store and organize data related to product design, and the entities include parts, materials, manufacturing processes, etc. Based on the product information, the recommendation system predicts the top 10 suitable materials, most of which meet the product requirements. This helps reduce costs and increase productivity. The knowledge graph can also store the historical maintenance records of the equipment, causes of failure and repair methods. The recommendation system analyzes the operating status of the equipment and predicts the top 10 possible failures, most of which match the situation in which the failure occurred. This helps companies quickly identify the cause of failures and reduce unplanned downtime.

Although the recommendation effect of this paper is improved in mitigating the cold-start and data sparsity problems, the performance of the model is yet to be tested in the face of large-scale datasets. Secondly, if the knowledge graph embedding representation is used directly without any constraint, it will easily lead to noise penetration into the underlying information. In future work, we will set constraints for representation learning based on the correlation between nodes. We also plan to introduce other auxiliary information into the recommendation algorithm, such as user reviews and social networks.

Author Contributions: C.W. performed the experiments, contributed to the construction of the knowledge graph and the design of the model, executed a detailed analysis, and wrote certain sections of the manuscript. B.H. set the direction of the research, wrote certain sections of the manuscript, and performed the final corrections. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Science and Technology Innovation 2030—Major Project of "New Generation Artificial Intelligence" granted by Ministry of Science and Technology, grant number 2020AAA0109300.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to express their sincere gratitude to all the teachers and students who helped to successfully complete this study.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Sun, Z.; Guo, Q.; Yang, J.; Fang, H.; Guo, G.; Zhang, J.; Burke, R. Research commentary on recommendations with side information: A survey and research directions. *Electron. Commer. Res. Appl.* **2019**, *37*, 100879. [CrossRef]
2. Sharma, R.; Ray, S. Explanations in recommender systems: An overview. *Int. J. Bus. Inf. Syst.* **2016**, *23*, 248–262. [CrossRef]
3. Song, Y.; Elkahky, A.M.; He, X. Multi-rate deep learning for temporal recommendation. In Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, Pisa, Italy, 17–21 July 2016; pp. 909–912.
4. Bai, B.; Fan, Y.; Tan, W.; Zhang, J. DLTSR: A deep learning framework for recommendations of long-tail web services. *IEEE Trans. Serv. Comput.* **2017**, *13*, 73–85. [CrossRef]
5. Feldmann, A.; Whitt, W. Fitting mixtures of exponentials to long-tail distributions to analyze network performance models. *Perform. Eval.* **1998**, *31*, 245–279. [CrossRef]
6. Bobadilla, J.; Ortega, F.; Hernando, A.; Gutiérrez, A. Recommender systems survey. *Knowl.-Based Syst.* **2013**, *46*, 109–132. [CrossRef]
7. Klašnja-Milićević, A.; Ivanović, M.; Nanopoulos, A. Recommender systems in e-learning environments: A survey of the state-of-the-art and possible extensions. *Artif. Intell. Rev.* **2015**, *44*, 571–604. [CrossRef]
8. Adomavicius, G.; Tuzhilin, A. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. Knowl. Data Eng.* **2005**, *17*, 734–749. [CrossRef]
9. Lops, P.; De Gemmis, M.; Semeraro, G. *Content-Based Recommender Systems: State of the Art and Trends*; Springer: Boston, MA, USA, 2011; pp. 73–105.
10. Wang, D.; Liang, Y.; Xu, D.; Feng, X.; Guan, R. A content-based recommender system for computer science publications. *Knowl.-Based Syst.* **2018**, *157*, 1–9. [CrossRef]
11. 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.
12. Zhang, H.; Shen, F.; Liu, W.; He, X.; Luan, H.; Chua, T.-S. Discrete collaborative filtering. In Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, Pisa, Italy, 17–21 July 2016; pp. 325–334.
13. Mongia, A.; Jhamb, N.; Chouzenoux, E.; Majumdar, A. Deep latent factor model for collaborative filtering. *Signal Process.* **2020**, *169*, 107366. [CrossRef]
14. Pang, S.; Yu, S.; Li, G.; Qiao, S.; Wang, M. Time-Sensitive Collaborative Filtering Algorithm with Feature Stability. *Comput. Inform.* **2020**, *39*, 141–155. [CrossRef]
15. Nikolakopoulos, A.N.; Karypis, G. Boosting item-based collaborative filtering via nearly uncoupled random walks. *ACM Trans. Knowl. Discov. Data (TKDD)* **2020**, *14*, 1–26. [CrossRef]
16. Herlocker, J.L.; Konstan, J.A.; Borchers, A.; Riedl, J. An algorithmic framework for performing collaborative filtering. In Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Berkeley, CA, USA, 15–19 August 1999; pp. 230–237.
17. Kim, B.M.; Li, Q.; Park, C.S.; Kim, S.G.; Kim, J.Y. A new approach for combining content-based and collaborative filters. *J. Intell. Inf. Syst.* **2006**, *27*, 79–91. [CrossRef]
18. Kim, B.-D.; Kim, S.-O. A new recommender system to combine content-based and collaborative filtering systems. *J. Database Mark. Cust. Strat. Manag.* **2001**, *8*, 244–252. [CrossRef]
19. Bellogín, A.; Castells, P.; Cantador, I. Neighbor selection and weighting in user-based collaborative filtering: A performance prediction approach. *ACM Trans. Web (TWEB)* **2014**, *8*, 1–30. [CrossRef]
20. Koohi, H.; Kiani, K. User based collaborative filtering using fuzzy C-means. *Measurement* **2016**, *91*, 134–139. [CrossRef]
21. Yue, W.; Wang, Z.; Liu, W.; Tian, B.; Lauria, S.; Liu, X. An optimally weighted user-and item-based collaborative filtering approach to predicting baseline data for Friedreich's Ataxia patients. *Neurocomputing* **2021**, *419*, 287–294. [CrossRef]
22. Xue, F.; He, X.; Wang, X.; Xu, J.; Liu, K.; Hong, R. Deep item-based collaborative filtering for top-n recommendation. *ACM Trans. Inf. Syst.* **2019**, *37*, 1–25. [CrossRef]
23. Çano, E.; Morisio, M. Hybrid recommender systems: A systematic literature review. *Intell. Data Anal.* **2017**, *21*, 1487–1524. [CrossRef]

24. Ahmadian, S.; Joorabloo, N.; Jalili, M.; Ahmadian, M. Alleviating data sparsity problem in time-aware recommender systems using a reliable rating profile enrichment approach. *Expert Syst. Appl.* **2022**, *187*, 115849. [CrossRef]
25. Kouki, P.; Fakhraei, S.; Foulds, J.; Eirinaki, M.; Getoor, L. Hyper: A flexible and extensible probabilistic framework for hybrid recommender systems. In Proceedings of the 9th ACM Conference on Recommender Systems, Vienna, Austria, 16–20 September 2015; pp. 99–106.
26. Zhang, Y.; Shi, Z.; Zuo, W.; Yue, L.; Liang, S.; Li, X. Joint Personalized Markov Chains with social network embedding for cold-start recommendation. *Neurocomputing* **2020**, *386*, 208–220. [CrossRef]
27. Catherine, R.; Cohen, W. Transnets: Learning to transform for recommendation. In Proceedings of the Eleventh ACM Conference on Recommender Systems, Como, Italy, 27–31 August 2017; pp. 288–296.
28. Tay, Y.; Luu, A.T.; Hui, S.C. Multi-pointer co-attention networks for recommendation. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 2309–2318.
29. Xu, Y.; Yang, Y.; Han, J.; Wang, E.; Zhuang, F.; Xiong, H. Exploiting the sentimental bias between ratings and reviews for enhancing recommendation. In Proceedings of the 2018 IEEE International Conference on Data Mining (ICDM), Singapore, 17–20 November 2018; pp. 1356–1361.
30. Rendle, S. Factorization machines. In Proceedings of the 2010 IEEE International Conference on Data Mining, Sydney, NSW, Australia, 13–17 December 2010; pp. 995–1000.
31. 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]
32. 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 Proceedings of the The World Wide Web Conference, San Francisco, CA, USA, 13–17 May 2019; pp. 151–161.
33. 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, Torino, Italy, 22–26 October 2018; pp. 417–426.
34. Wang, H.; Zhang, F.; Zhang, M.; Leskovec, J.; Zhao, M.; Li, W.; Wang, Z. 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, Anchorage, AK, USA, 4–8 August 2019; pp. 968–977.
35. 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.
36. 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, Virtual Event China, 25–30 July 2020; pp. 219–228.
37. 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.
38. 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.
39. Liang, K.; Zhou, B.; Zhang, Y.; Li, Y.; Zhang, B.; Zhang, X. PF2RM: A Power Fault Retrieval and Recommendation Model Based on Knowledge Graph. *Energies* **2022**, *15*, 1810. [CrossRef]
40. Lin, Y.; Liu, Z.; Sun, M.; Liu, Y.; Zhu, X. Learning entity and relation embeddings for knowledge graph completion. In Proceedings of the AAAI Conference on Artificial Intelligence, Austin, TX, USA, 25–30 January 2015.
41. 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.
42. Zhang, S.; Zhang, N.; Fan, S.; Gu, J.; Li, J. Knowledge Graph Recommendation Model Based on Adversarial Training. *Appl. Sci.* **2022**, *12*, 7434. [CrossRef]
43. Zhang, H.; Wang, Y.; Chen, C.; Liu, R.; Zhou, S.; Gao, T. Enhancing Knowledge of Propagation-Perception-Based Attention Recommender Systems. *Electronics* **2022**, *11*, 547. [CrossRef]
44. Ji, G.; He, S.; Xu, L.; Liu, K.; Zhao, J. Knowledge graph embedding via dynamic mapping matrix. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Beijing, China, 26–31 July 2015; pp. 687–696.
45. Goldberg, Y.; Levy, O. word2vec Explained: Deriving Mikolov et al.'s negative-sampling word-embedding method. *arXiv* **2014**, arXiv:1402.3722.
46. Harper, F.M.; Konstan, J. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.* **2015**, *5*, 1–19. [CrossRef]
47. Rendle, S.; Freudenthaler, C.; Gantner, Z.; Schmidt-Thieme, L. BPR: Bayesian personalized ranking from implicit feedback. *arXiv* **2012**, arXiv:1205.2618.
48. Zhang, L.; Kang, Z.; Sun, X.; Sun, H.; Zhang, B.; Pu, D. KCRec: Knowledge-aware representation graph convolutional network for recommendation. *Knowl.-Based Syst.* **2021**, *230*, 107399. [CrossRef]

49. Bordes, A.; Usunier, N.; Garcia-Duran, A.; Weston, J.; Yakhnenko, O. Translating embeddings for modeling multi-relational data. In Proceedings of the 26th International Conference on Neural Information Processing Systems, Lake Tahoe, NV, USA, 5–10 December 2013; Volume 2, pp. 2787–2795.
50. Wang, Z.; Zhang, J.; Feng, J.; Chen, Z. Knowledge graph embedding by translating on hyperplanes. In Proceedings of the AAAI Conference on Artificial Intelligence, Québec City, QC, Canada, 27–31 July 2014.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.