applied sciences

MDPI

Article

MDAR: A Knowledge-Graph-Enhanced Multi-Task Recommendation System Based on a DeepAFM and a Relation-Fused Multi-Gead Graph Attention Network

Songjiang Li [1], Qingxia Xue [1] and Peng Wang [1,2,*]

[1] College of Computer Science and Technology, Changchun University of Science and Technology, Changchun 130022, China; lsj@cust.edu.cn (S.L.); xqx@mails.cust.edu.cn (Q.X.)
[2] Changchun University of Science and Technology Chongqing Research Institute, Chongqing 401120, China
* Correspondence: wangpeng@cust.edu.cn

Abstract: In recent years, MKR has attracted increasing attention due to its ability to enhance the accuracy of recommendation systems through cooperation between the RS tasks and the KGE tasks, allowing for complementarity of the information. However, there are still three challenging issues: historical behavior preferences, missing data, and knowledge graph completion. To tackle these challenging problems, we propose MDAR, a multi-task learning approach that combines DeepFM with an attention mechanism (DeepAFM) and a relation-fused multi-head graph attention network (RMGAT). Firstly, we propose to leverage the attention mechanism in the DeepAFM to distinguish the importance of different features for target prediction by assigning different weights to different interaction features of the user and the item, which solves the first problem. Secondly, we introduce deep neural networks (DNNs) to extract the deep semantic information in the cross-compressed units by obtaining the high-dimensional features of the interactions between the RS task and the KG task to solve the second problem. Lastly, we design a multi-head graph attention network for relationship fusion (RMGAT) in the KGE task, which learns entity representations through the different contributions of the neighbors by aggregating the relationships into the attention network of the knowledge graph and by obtaining information about the neighbors with different importance for different relationships, effectively solving the third problem. Through experimenting on real-world public datasets, we demonstrate that MDAR obtained substantial results over state-of-the-art baselines for recommendations for movie, book, and music datasets. Our results underscore the effectiveness of MDAR and its potential to advance recommendation systems in various domains.

Keywords: multi-task recommendation system; knowledge graph embedding; multi-head graph attention network with relation fusion; attention mechanism

check for updates

Citation: Li, S.; Xue, Q.; Wang, P. MDAR: A Knowledge-Graph-Enhanced Multi-Task Recommendation System Based on a DeepAFM and a Relation-Fused Multi-Gead Graph Attention Network. *Appl. Sci.* **2023**, *13*, 8697. https://doi.org/10.3390/app13158697

Academic Editors: Shoujin Wang and Qi Zhang

Received: 5 July 2023
Revised: 19 July 2023
Accepted: 25 July 2023
Published: 27 July 2023

1. Introduction

In recent years, knowledge-graph-based recommendation systems (KGRSs) have attracted increasing attention, which utilize knowledge graphs (KGs) as side information to address issues of sparse data and cold start problems commonly encountered in traditional recommendation systems (RSs) [1–5]. A knowledge graph (KG) is essentially a directed heterogeneous graph [6], where nodes denote the entities and edges denote the relations [7]. Some notable academic and commercial knowledge graphs that have been proposed include NELL, DBpedia, Google Knowledge Graph, and Microsoft Satori [8].

One of the most popular and widely used methods in KGRSs is the utilization of knowledge graph embedding (KGE) methods [9]. These methods require representing entities and connections within a dense vector space, maintaining the original structural details of the knowledge graph. For instance, Zhang et al., proposed CKE [10] to utilize TransR [11] to determine the likelihood that a user will select a specific item and to obtain the user's preference ranking. DKN [12] combines semantic and knowledge information to represent news articles by using TransD [13], resulting in the improved performance of news recommendations. Wang et al., introduced MKR [14], which enhances the performance of recommendations by mutual cooperation between the RS module and the KGE module. To capture the linear characteristics of users and items, some researchers have applied the memory of FM [15], AFM [16], and DeepFM [17] to create linear combinations of features and utilize the feature vectors as a kind of side information to improve recommendation accuracy. Meanwhile, in order to capture the extensive interactions and supplementary information, we can extract the relationships between entities in the knowledge graph. Graph neural networks (GNNs) that aggregate node attributes from neighborhoods using neural networks are usually exploited to learn the representations in KGs in an efficient, explicit, and end-to-end manner. In this way, auxiliary information can be exploited to enrich the representations of users, entities, and relations, such as constructing user KGs [18–21], item KGs [22–25], or user–item KGs [26–30], and we can extract more accurate preferences of users by excavating the relationships between entities. In recent years, several researchers have applied factorization machines and neural networks to MKR, drawing on the strengths of both by alternate training, thus improving the performance of the recommendation systems. For example, DFM_GCN [31], DeepFM_GCN [32], EMKR [33], etc.

However, MKR still confronts three challenging problems. The first problem is that when modeling users and items, the potential features of users and items are excavated by only using the multilayer perceptron (MLP) in the RS module. This approach fails to capture crucial user–item interaction features and obtain users' behavioral preferences, which reduces recommendation accuracy. The second problem is missing data in the cross-compression unit, leading to the loss of original semantic information during the compression process. Consequently, the deep semantic information of the original data is not adequately captured. The third problem involves the insufficient acquisition of potential multi-relation neighborhood feature information in the KGE module. MKR solely relies on the MLP in KG to capture entity features, thereby failing to obtain the complex and significant multi-relation features in the surrounding neighborhood of a given entity, which leads to the knowledge graph completion problem. As shown in Figure 1, "*Stargate*" was viewed by the user, which is a science fiction movie directed by Roland Emmerich. By using knowledge connections in KGs, the system suggests "*Moonfall*" to the user. However, it is unclear whether the recommendation is based on genre or on the director. The importance of each relationship to the user is not clearly defined. Therefore, it is crucial to model the relationships in KGs.

To address the shortcomings of the original MKR work, we propose MDAR, a multi-task learning method for DeepFM with an attention mechanism (DeepAFM) and a relation-fused multi-head graph attention network (RMGAT). There are three novelties when compared with the original MKR. Firstly, in the RS modules, MDAR utilizes the Deep-AFM to capture both low-level and high-level features of users and items. The attention mechanism in DeepAFM assigns higher weights to features that are more relevant to the target, helping the recommendation system accurately identify users' historical behavior preferences by extracting important low-order linear feature interactions. Secondly, in the KGE modules, MDAR applies the multi-head graph attention network with relation fusion to capture important neighborhood features. By considering the significance of different relationships, MDAR can effectively extract deep multi-relation neighborhood features in the knowledge graph, enhancing the recommendation process by completing the knowledge graph information. Thirdly, we introduce deep neural networks (DNNs) into the cross-compression unit to extract the deep semantic information between the item in the RS module and the entity in the KGE module to solve the problem of missing original data. With the above improvements, each part of MDAR has been redesigned compared with the original MKR, and the novelties and contributions can be summarized as follows:

- We develop a new framework, MDAR, a multi-task learning approach for the DeepFM with an attention mechanism and a multi-head graph attention network with relation-fusion-enhanced recommendation. MDAR has three improvements over the original MKR;

- We construct a DeepAFM in the RS module, which mainly assigns different assigned weights to different cross features through the attention mechanism to mine the user's historical behavior preferences more accurately and to make up the feature contribution consistency problem in the original recommender system;
- We design an RMGAT in the KGE module, where a graph attention network with relation fusion is based on multi-head attention mechanisms, to learn the representation of different relationships for different entities to adequately obtain the entities' aggregate features in KG, which solves the problem of information complementation in the original knowledge graph;
- We introduce a deep neural network (DNN) into the cross-compression unit to extract high-dimensional features from the items in the recommendation system (RS) module and the entities in the knowledge graph embedding (KGE) module. This enables us to share information between the modules and enhance the overall generalization ability of our MDAR and solve the problem of data loss in the data compression process in the original MKR.
- We conduct experiments on four public datasets, and the results demonstrate that MDAR outperforms state-of-the-art baselines in terms of click-through rate (CTR) prediction and top-K recommendation. Furthermore, MDAR maintains comparable performance even in scenarios with sparse data.

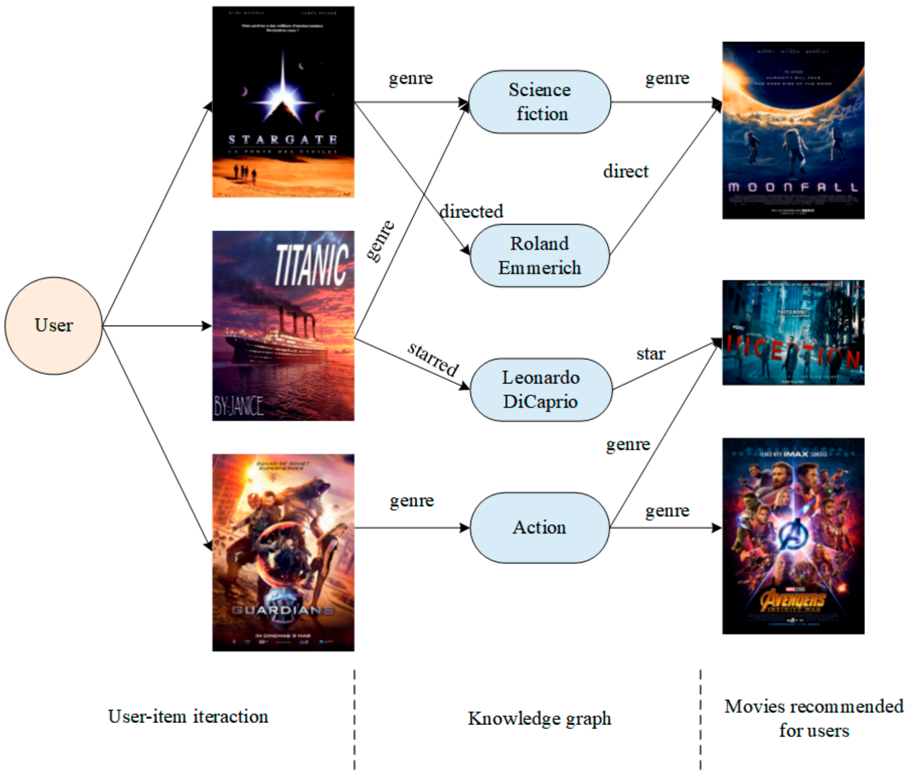


Figure 1. The movie recommendation system based on a knowledge graph.

2. Related Works

The utilization of knowledge graphs as side information in recommender systems is becoming increasingly crucial. Relationships within the knowledge graph also play a significant role in this context. Some recent studies have focused on disseminating multi-layered relational information within the knowledge graph to enhance recommendations. For instance, researchers have employed relation-aware graph neural networks [18,22–25,31–36] to capture neighborhood features. RippleNet [18] naturally integrates the KGE method into the recommendation system through the preference propagation method, thus continuously and automatically excavating the user's potential hierarchical interest. KGCN [22] and KGNN-LS [23] utilize graph convolutional networks (GCNs) to analyze various relationships between entities and understand user interests for better recommendations. CKAN [24] tackles the cold start problem by using knowledge-aware collaborative networks to improve recommendation accuracy. FairSR [25] applies a fairness-aware preference graph embedding method to incorporate the knowledge of users' and items' attributes and their correlation into entity representations and alleviate the unfair distributions of user attributes on items. DFM-GCN [31] and DeepFM_GCN [32] employ the DeepFM and GCNs to learn the entity representation in the RS and the KGE, to excavate the semantic feature information between the two modules and improve the accuracy of recommendation. EMKR [33] proposes to utilize the attention mechanism to aggregate users' historical behavior and utilize the relation-aware graph convolutional neural network to fully capture the deep multi-relation neighborhood features for more accurate mining preferences. Multi-Rec [34] proposes to employ the knowledge graph feature learning and knowledge graph structure learning to mine users' higher-order interests from collective behaviors with the assistance of KG reasoning. AKNEI [35] proposes an attention knowledge network combining explicit and implicit information to effectively capture and exactly describe the correlation between entities in the knowledge graph. MHRS [36] combines deep neural networks with 5G networks and applies them to the field of medicine to contribute to smart healthcare.

Other researchers have utilized graph attention networks (GATs) to aggregate important features from neighboring entities [26,37–42]. For example, the KGAT [26] proposes using the knowledge graph attention network to obtain higher-order relationships in KG and to learn users' historical interests through attention mechanisms. The CKGAT [37] leverages the KGAT to learn higher-order entity representations by extracting the topology of entities in a multi-hop ripple set, resulting in more accurate embeddings. CKSR [38] proposes to utilize a knowledge-aware attention mechanism to model a cross-session knowledge graph, which is performed to capture the complicated transition pattern among interacted items. SMRGAT [39] uses a multi-head attention mechanism to focus on herbs' different effects on symptoms and incorporate the symptoms' semantic information and external knowledge of herbs to analyze information about the symptoms. The KDGN [40] proposes the MedRec model, which introduces a medical knowledge graph and a medicine attribute graph to learn the embedding representations of symptoms and medicine which are used for medicine recommendations. CGRS [41] and CRKG [42] propose the collaborative knowledge propagation graph attention network for recipe recommendations, which employs a knowledge-aware attention graph convolutional neural network to capture the semantic associations between users and recipes on the collaborative knowledge graph and learns the users' requirements in both preference and health by fusing the losses of these two learning tasks.

Although the GAT has been used in different fields through different improvements, the GAT is not suitable for neighborhood accumulation in KGs, however, because it only focuses on information aggregation between nodes and ignores the important relation features. In this paper, we propose a multi-head graph attention network with relation fusion (RMGAT) to model relations and entities in the knowledge graph, thereby enhancing recommendations. Additionally, to capture deeper features, we introduce DeepAFM in this paper.

3. Methods

In this section, we first introduce the problem defined by the recommendation in the paper and then describe the specifically enhanced individual modules of MDAR. Lastly, we analyze the learning algorithm for MDAR.

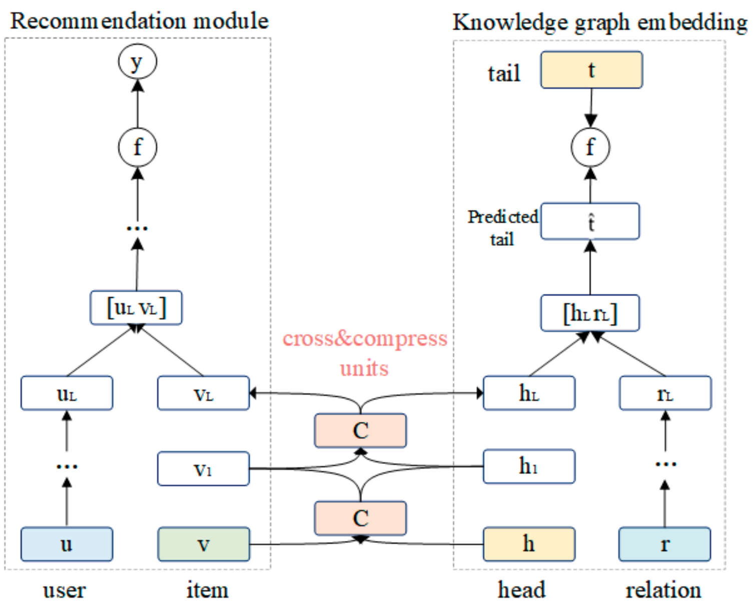
3.1. Problem Formulation

In order to make the recommendation formalization, we establish sets $U = \{u_1, u_2, u_3 \ldots\ldots u_M\}$ and $V = \{v_1, v_2, v_3 \ldots\ldots v_N\}$ to represent the users and items, respectively, where M is the users' number and N is the items' number, respectively. The items could refer to movies, books, and music. The interaction between users and items is captured in the matrix $Y \in R^{M \times N}$, where Y indicates the user's implicit feedback on the item. If $y_{uv} = 1$, this denotes the interaction of the user and the item. If $y_{uv} = 0$, this denotes that the user did not interact with the item. To assist in the recommendation process, the KGE is a triple $G = \{(h, r, t) | h, t \in \varepsilon, r \in R\}$, where h and t are entities and r represents the relationship between them and where ε represents the set of all entities and R represents the set of all relations.

In order to calculate the user–item interaction matrix Y, we need to input data about the users and items into our model. Our ultimate goal is to analyze the preferences of users to items that they have not yet interacted with, but may potentially like. To achieve this, we use a prediction function $\hat{y}_{uv} = F(u, v | \Theta, Y, G)$, where $\hat{y}_{uv}$ represents the probability of user u interacting with item v. The prediction function takes into account various hyperparameters denoted by Θ.

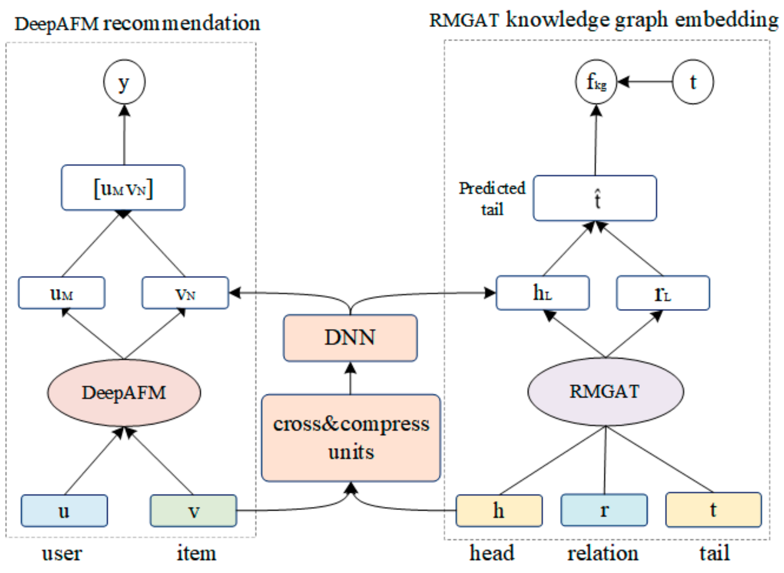
3.2. Model Framework

In this section, compared to the base model MKR, as shown in Figure 2a, we introduce MDAR in detail, which is a multi-task learning method for DeepAFM and RMGAT enhanced recommendation, as shown in Figure 2b. MDAR comprises three main components: the DeepAFM recommendation module, the RMGAT knowledge graph embedding module, and DNN cross and compress units. On the left side is our DeepAFM recommendation module, which utilizes the AFM layers to extract the low-order features and the DNN layers to extract the high-order features from the interactions between users and items. On the right side, RMGAT is employed to aggregate the neighborhood information of items, enabling a comprehensive representation of their characteristics for better recommendations. The middle part of the system consists of DNN cross and compress units, which connect the DeepAFM and RMGAT modules. The units capture higher-order interaction features between items in the RS and entities in the KGE. By enhancing the utilization of the data through the overall learning of the three components, we significantly improve the performance of recommendation systems.



(a)

Figure 2. *Cont.*



(**b**)

Figure 2. (**a**) The basic model of the MKR. (**b**) The strengthened model of MDAR.

3.3. DeepAFM Recommendation Module

In order to address the problem of users' historical behavior preferences, we introduced the DeepAFM framework into our model. This framework is structured in two parts: the AFM layers and the DNN layers. Firstly, the FM with the attention mechanism (AFM) can extract first-order and second-order linear interactive features, which play an important role in target prediction, through giving different assigned weights to different intersection features by the attention network. Secondly, the DNN can be used to extract the higher-order features, as shown in Figure 3.

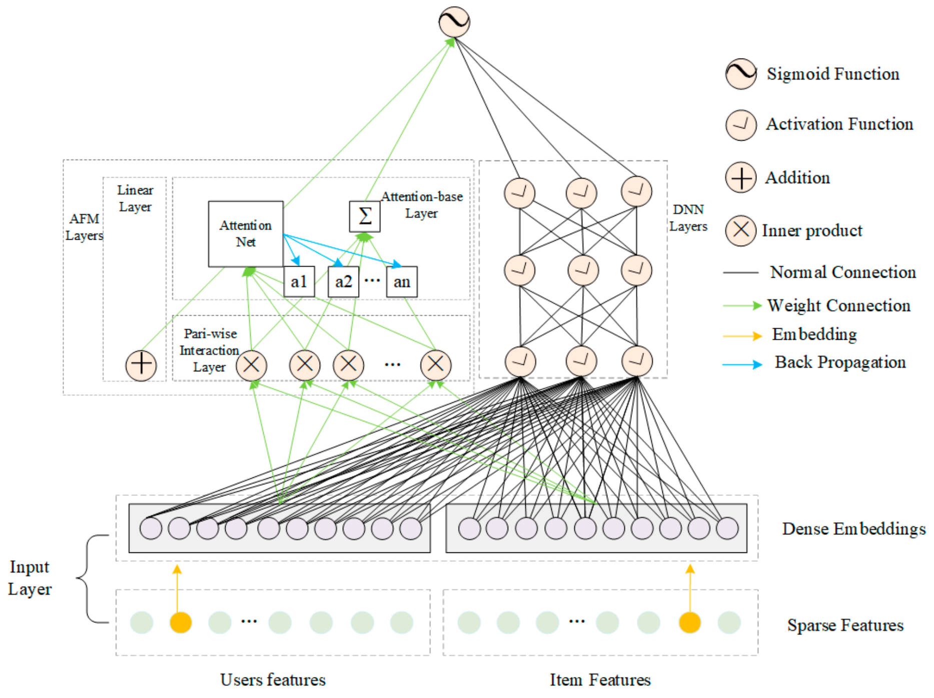


Figure 3. The DeepAFM framework.

Firstly, the input layer of the AFM takes the user and the item features as the input. For the category-based features, the AFM layer utilizes one-hot encoding to handle sparse feature vectors. Then, the sparse feature vector is processed into a dense feature by the embedding layer. For example, given a real feature vector $x \in R^n$, where n denotes the number of users and items, the features of users and items extracted by the FM is as follows:

$$y_{FM}(x) = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=j+1}^{n} \alpha_{ij} \langle v_i, v_j \rangle x_i x_j \quad (1)$$

where w_0 represents global deviation, w_i denotes the i-th weight of the feature, x_i denotes the i-th value of the feature, v_i denotes an implicit vector to the i-th dimensional feature, x_i,v_j denote an implicit vector to the j-th dimensional feature, x_j, and $\langle , \rangle$ denotes the product of elements of two implicit vectors. Through the paired interaction layer, the FM learns the low-order linear combination features between the feature vectors.

In order to better obtain important interaction features, we designed an attention-based pooling layer, which applies the attention mechanism to distribute higher weights to more important features. This process takes the second-order cross-feature vectors as the input in the attention network and then calculates the attention weights of each second-order cross-feature vector. Then, the summation pooling operation is performed by multiplying the attention weights and the second-order cross vector. The resulting weights represent the importance of each combination of features, and the weight function is represented by Equations (2) and (3):

$$\alpha'_{ij} = W' \sigma(W \langle v_i, v_j \rangle x_i x_j + b) \quad (2)$$

$$\alpha_{ij} = \frac{\exp\left(\alpha'_{ij}\right)}{\sum_{i,j \in R_x} \exp\left(\alpha'_{ij}\right)} \quad (3)$$

where $W, W\prime \in R^{t \times k}$, and $b \in R^t$ are the hyperparameters and σ is the nonlinear activation function, such as ReLU.

To summarize, the final output of the AFM is produced by weighting and summing the first-order and second-order linear feature vectors, as shown in Equation (4). The structure of the AFM is shown in Figure 4:

$$y_{AFM}(x) = w_0 + \sum_{i=1}^{n} w_i x_i + P^T \sum_{i=1}^{n} \sum_{j=j+1}^{n} \alpha_{ij} \langle v_i, v_j \rangle x_i x_j \quad (4)$$

where α_{ij} has been defined in Equation (3).

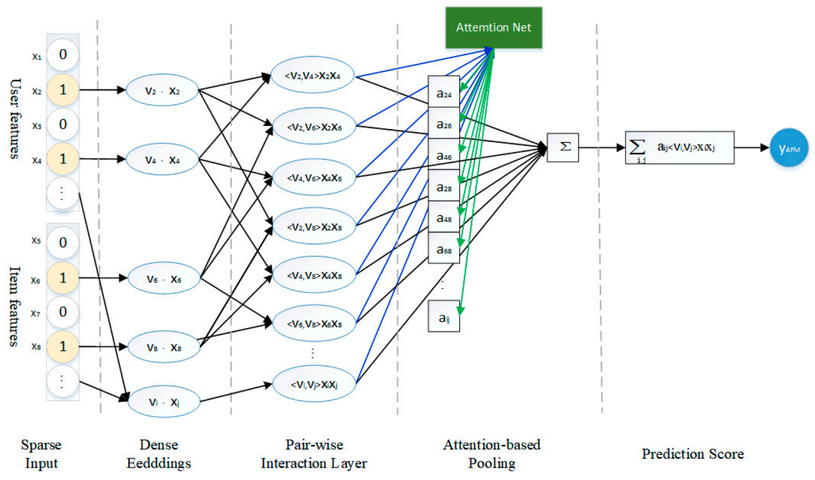


Figure 4. The AFM layers.

The DNN layers in the DeepAFM are used to extract the higher-order features of the user and the item, as shown in Equation (5):

$$a^{(l+1)} = \mathrm{relu}\left(w^{(l)}a^{(l)} + b^{(l)}\right) \tag{5}$$

where indicates the number of hidden layers, $a^{(l+1)}$ indicates the i the input vector of the hidden layers, $w^{(l)}$ is the weights of the hidden layers, and $b^{(l)}$ is the deviation vector of the hidden layers. The final output of the DNN is:

$$y_{DNN} = \mathrm{sigmoid}\left(W^{H+1}a^{H} + b^{H+1}\right) \tag{6}$$

where H is the quantity of hidden layers.

Finally, through the above process, the output results of the AFM and the DNN are combined.

$$\hat{y} = \mathrm{sigmoid}(y_{AFM} + y_{DNN}) \tag{7}$$

The deep feature vectors for both the user u_M and the item v_N will be captured using the calculation above. Finally, the two paths are combined using a prediction function f_{RS}, as shown is Equation (8):

$$\hat{y}_{uv} = \mathrm{sigmoid}(f_{RS}(u_M, v_N)) \tag{8}$$

3.4. Embedding of the RMGAT Knowledge Graph Module

To capture the significant semantic relationships and address the issue of inaccurate recommendations due to disregarding the varying importance of user interactions with different items, we propose the RMGAT network. This network combines the graph convolutional network and the multi-head attention networks to extract the weight information of nodes in the knowledge graph. The weight coefficient is used to propagate and aggregate the node information, resulting in a refined node representation that enhances recommendation accuracy, as shown in Figure 5.

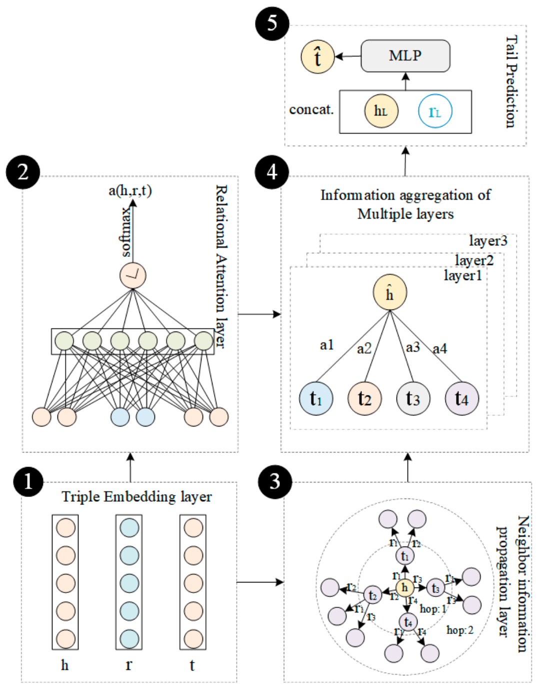


Figure 5. The multi-head graph attention network with relation fusion, where r_1, r_2, r_3, r_4 are multiple neighbor relationships for a given entity h, a_1, a_2, a_3, a_4 are different weights and t_1, t_2, t_3, t_4 are the tail entities, corresponding to different relations.

3.4.1. Embedding Layer

First, we initialize the triples of users, items, and relationships as the embedding vectors through a score function $g : R^d \times R^d \rightarrow R$ to calculate the significance score between relation r and entity h as:

$$\pi(h, r, t) = g(h, r, t) \tag{9}$$

The entity h, t, and relation r are represented by the embedding vectors in the form of $h \in R^d, r \in R^d, t \in R^d$, and d represents the dimension of the vector.

3.4.2. Relation Attention Layer

We learn relational attention weights through the multi-headed attention mechanism during the second RMGAT step, as shown in Figure 5. To better distribute the weights, we use the SoftMax function to normalize the weight coefficients, shown as follows:

$$a(h, r, t) = \text{softmax}(\pi(h, r, t)) = \frac{\exp(\pi(h, r, t))}{\sum_{j=1}^{N_h} \pi(h, r, t)} \tag{10}$$

where a (h, r, t) is the normalized weight coefficient, and N_h indicates the aggregation of neighboring nodes h. The original KG is an unweighted graph that cannot show the weight value. The unweighted graph is converted to a weighted graph by the weight coefficient.

3.4.3. Neighbor Information Propagation Layer

In the KG, an entity has many neighborhood triples and contains different relationships. The different relationships have different contributions to entities and different semantic information. We use $N_h^{(1)} = \{(h, r_1, t_1), (h, r_2, t_2), \cdots (h, r_N, t_N)\}$ to denote the set of triples during the third RMGAT step, as shown in Figure 5.

3.4.4. Information Aggregation with Multiple Layers

By iteratively transmitting information from neighboring nodes with various connections, we can compute the feature vector of the entity's adjacent nodes. This process involves assigning weights to the nodes and aggregating them, resulting in a combined representation of the surrounding neighborhood, $\hat{h}$, which can be mathematically expressed as follows:

$$\hat{h} = \sum_{j=1}^{N_h} a_j(h, r, t) t_j \tag{11}$$

Finally, by aggregating the features of h and its neighborhood features, a first-order features of h can be obtained, during the four RMGAT steps, as shown in Figure 5.

To enhance the understanding of the relationships between nodes, we employ a multi-layer graph convolutional network to capture different relationship features. We employ a sum aggregator to combine the entity's feature vector h with its neighboring feature vector h in order to obtain the final feature vector.

$$e_h = \text{aggsum} = \sigma\,(W\,(h + \hat{h})) + b) \tag{12}$$

where $W \in R^{d \cdot d}$ and $b \in R^d$ are trainable weight matrices and σ is the ReLU activation function.

3.4.5. Tail Prediction

Through the above steps, we obtain the entity's features $h^{(hop)}$ via hop layers. Then, we extract e_h of the knowledge graph and v of the unit through the MLP. Additionally, we utilize the MLP to extract deep features of relation r. Moreover, the h and the r are concatenated, and an H-layer MLP is applied to predict the tail entity $\hat{t}$.

$$h_L = E_{V \sim S(h)}\left[C^L(v, e_h)[e]\right] \tag{13}$$

$$r_L = M^L[(r)] \tag{14}$$

$$\hat{t} = MLP^H(concat(h_L, r_L))] \tag{15}$$

where $\hat{t}$ denotes the predicted tail. We score the triples by calculating the similarity between the predicted tail entity $\hat{t}$ and the actual tail entity t through the scoring function f_{kg}. The f_{kg} can be the inner product of the predicted tail entity $\hat{t}$ and the actual tail entity t, after taking the inner product of the sigmoid.

$$score(h, r, t) = f_{kg}(t, \hat{t}) \tag{16}$$

3.5. DNN Cross and Compress Units

To enhance the representation of user and item features, we introduce a deep neural network into the cross-compression unit, as shown in Figure 6. This helps to address the sparsity of the recommendation module and the knowledge graph module's data. In this process, we create an interaction feature matrix C for each item and head entity. The matrix C is as follow:

$$C_l = v_l h_l^T = \begin{bmatrix} v_l^{(1)} h_l^{(1)} & \cdots & v_l^{(1)} h_l^{(d)} \\ \vdots & \ddots & \vdots \\ v_l^{(d)} h_l^{(1)} & \cdots & v_l^{(d)} h_l^{(d)} \end{bmatrix} \tag{17}$$

$$C_l^T = h_l v_l^T = \begin{bmatrix} h_l^{(1)} v_l^{(1)} & \cdots & h_l^{(1)} v_l^{(d)} \\ \vdots & \ddots & \vdots \\ h_l^{(d)} v_l^{(1)} & \cdots & h_l^{(d)} v_l^{(d)} \end{bmatrix} \tag{18}$$

where $v_l \in R^d, h_l \in R^d$, $C_l \in R^{d \times d}$, and l indicate the number of hidden layers and d denotes the dimension of the hidden layers. This process is called a cross-operation because the interaction feature matrix C_l displays and models each feature interaction $v_l^i h_l^j$ between the item and its corresponding entity. The item feature vector of v_{l+1} and the entity feature vector of e_{l+1} in the next layer are obtained from the feature matrix C_l by compressing it, as shown in Equations (19) and (20):

$$v_{l+1} = DNN\left(W_l^{VV} C_l + W_l^{EV} C_l^T + b_l^V\right) \tag{19}$$

$$e_{l+1} = DNN\left(W_l^{VE} C_l + W_l^{EE} C_l^T + b_l^E\right) \tag{20}$$

where $W_l \in R^d$ represents the l-layer weight vector of the cross-compression unit, $b_l \in R^d$ represents the l-layer deviation vector of the cross-compression unit. This transforms the d $\times$ d dimensional feature matrix into a d dimensional feature vector. This process is a compress operation. And then, we extract higher-order features through a DNN. The DNN process was:

$$DNN\left(i^H\right) = relu(W^{(H)} i^{H-1} + b^{(H)}) \tag{21}$$

where H is the depth of the layer, $DNN\left(i^H\right)$, $W^{(H)}$ and $b^{(H)}$ are the H-th layers' output, model weight, and the bias, respectively.

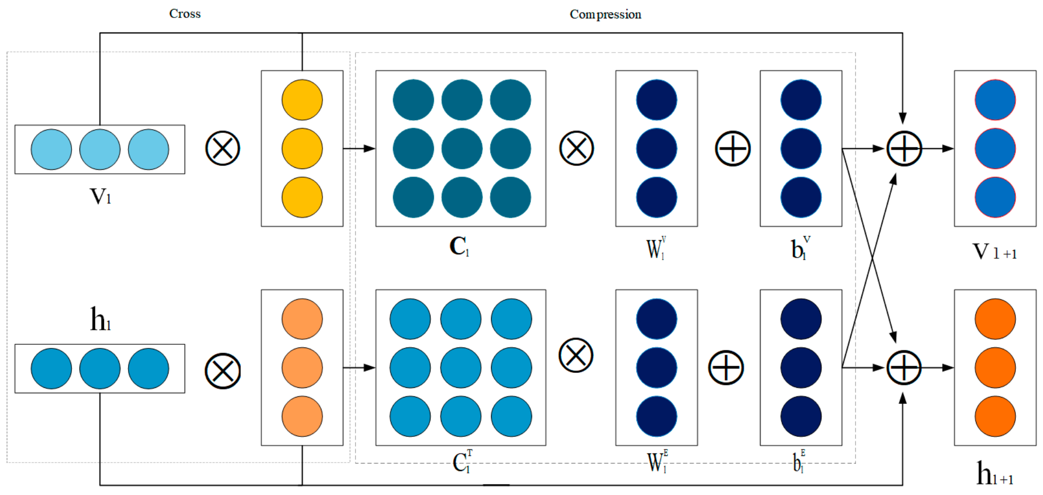


Figure 6. Cross and compress units.

3.6. Learning Algorithm

As mentioned in the previous introduction, the multi-task learning algorithm in this paper contains two modules: a recommendation system and a knowledge graph. Therefore, the loss function includes the RS module loss and the KGE module loss. At the same time, to prevent over-fitting of the model, L2 regularization is added to the above loss function, and the loss function is:

$$\begin{aligned}\mathscr{L} &= \mathrm{L_{RS}} + \mathrm{L_{KGE}} + \mathrm{L_{REG}} \\ &= \sum_{\mu \in U, \nu \in V} f(\hat{y}_{\mu\nu}, y_{\mu\nu}) \\ &\quad - \lambda_1 \Big(\sum_{(\mathrm{h,r,t}) \in G} \mathrm{score}(\mathrm{h,r,t}) - \sum_{(\mathrm{h}', \mathrm{r}, \mathrm{t}') \notin G} \mathrm{score}(\mathrm{h}', \mathrm{r}, \mathrm{t}') + \lambda_2 \|\mathrm{W}\|^2 \Big)\end{aligned} \tag{22}$$

where u ∈ U indicates the user vector, v ∈ V indicates the item vector, $\hat{y}_{\mu\nu}$ is the prediction result, and f is the cross-entropy function. In order to improve the performance of the algorithm, negative sampling is introduced into knowledge graph learning. The loss function of the knowledge graph module is obtained by replacing h, t in the triple (h, r, t) with h′ and t′ at random, keeping the relation r constant. Our goal is to optimize the model by enhancing the score for all accurate triples and by decreasing the score for all incorrect triples. λ_1 is the learning rate between two modules. λ_2 is the L_2 regularization term to prevent over-fitting.

4. Experiment

4.1. Public Datasets

- MovieLens-1M: It is a widely used dataset for movie recommendations and consists of approximately 1,000,000 explicit feeds (rated from 1 to 5) on movie websites;
- MovieLens-20M: It is the most used dataset in the movie recommendation field and contains the rating data (rated from 1 to 5) of approximately 20 million users;
- BookCrossing: It records over 1 million valid reviews of books from around 300,000 readers and consists of 1,149,780 explicit feeds (rated from 0 to 10) from the BookCrossing community;
- Last.FM: It is a record of over 2000 listeners on a popular global online music platform and contains the listening information of 2000 users.

Before conducting the experiment, we performed data preprocessing on the datasets mentioned above. In the case of the MovieLens-1M and MovieLens-20M datasets, we considered scores that are greater than or equal to 5 as positive feedback. For the BookCrossing and Last.FM datasets, we considered the score information itself as positive feedback due to the sparse nature of their data. You can find more information about these datasets in Table 1.

Table 1. Details of the four public datasets.

	MovieLens-1M	**MovieLens-20M**	**BookCrossing**	**Last.FM**
Users	6036	138,159	17,860	1872
Items	2445	16,954	14,910	3846
Interactions	753,772	13,501,622	139,746	42,364
Entities	182,011	102,569	24,039	9366
Relations	12	32	18	60
KG triples	20,782	499,474	19,793	15,518
Sparsity level	0.9489	0.9947	0.9994	0.9941

4.2. Evaluation Metrics

This paper makes use of the AUC score, ACC score, precision rate, and recall rate as evaluation metrics to measure the CTR prediction performance and top-K recommendation of users for items. As shown in Table 2, all samples are divided into four combinations according to their actual classification and model prediction results. Namely, the true positive example (TP), the true negative example (TN), the false positive example (FP), and the false negative example (FN). Where TP represents the positive samples that users are interested in; items with which they have not interacted can be correctly predicted by the model. TN represents the negative samples that users are not interested in; items with which they have not interacted can be correctly predicted by the model. FP means that the model mistakenly predicts the negative samples that users are not interested in as the positive samples that users are interested in. FN means that the model mistakenly predicts the positive samples that users are actually interested in as the negative samples that users are not interested in. The confusion matrix is shown in Table 2.

Table 2. The confusion matrix.

	Clicked	**Not Clicked**
Recommended	True-Positive(TP)	False-Positive(FP)
Not Recommended	False-Negative(FN)	True-Negative(TN)

AUC is defined as the area under the ROC curve. The abscissa of the ROC curve is the false-positive rate (FPR) and the ordinate is the true-positive rate (TPR). The larger the AUC, the better the recommendation effect of the model. The AUC index is shown in Equation (23).

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}},\ \mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{TN}} \tag{23}$$

Accuracy is the percentage of all predictions that are correctly predicted. The ACC index is shown in Equation (24).

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

The precision rate is used to indicate how many of the items that the model predicts users may be interested in are items that the user will actually be interested in when the users have not interacted with items. The formula of precision is as follows in Equation (25).

$$\mathrm{Precision} \ = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{25}$$

The recall rate is used to indicate how many of the items that the user is actually interested in are correctly predicted by the model. The formula of recall is as follows in Equation (26).

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{26}$$

4.3. Baseline Model

To verify the effectiveness of the MDAR model we proposed, we compared the model in this paper with the following seven models for comparable tests:

1. CKE [10]: It combines knowledge graphs and collaborative filtering algorithms, a typical embedding recommendation method;
2. DKN [12]: It is a news recommendation network that uses knowledge graphs to mine the hidden messages in the news;
3. KGCN [22]: It uses the knowledge graph to enrich the user's interests and increase the diversity of items by learning the neighborhood information of items based on the GCN in KG;
4. MKR [14]: It is a multi-task recommendation model that incorporates knowledge graphs, enriches recommendation data, and uses an alternate learning approach to train the model;
5. KGAT [26]: It is a propagation-based recommendation method that applies graph attention networks to collaborative knowledge graphs to learn user feedback and item representations;
6. CKAN [24]: It is an end-to-end propagation recommendation using collaborative signals and knowledge associations combined to enrich the embedding representation of users and items;
7. EMKR [33]: It uses the attention mechanism to distinguish the importance of different relationships to congregate historical behavior and improve the accuracy of recommendations.

4.4. Experiment Settings

In this article, the processed data are divided into a training set, a validation set, and a test set, with a ratio of 6:2:2. Each experiment is repeated five times and the average is taken as the final result. In two experimental environments, we assessed our approach: (1) CTR predictive performance. We utilized AUC and ACC indicators to evaluate the CTR prediction performance of each model. (2) Top-K recommendation. We chose the K items with the highest predicted click-through rate to test the probability of each user intensively and chose Precision@K and Recall@K to examine the effectiveness of the model's recommendations. The details of the experimental setting hyperparameters for four datasets are shown in Table 3.

Table 3. Details of experimental setting hyperparameters for the four datasets.

	MovieLens-1M	**MovieLens-20M**	**BookCrossing**	**Last.FM**
dim	8	8	16	4
L	1	1	2	2
H	3	3	1	1
hop	2	2	3	2
head	8	8	12	12
Batch_size	4096	4096	32	256
ℓ_2	1×10^{-6}	1×10^{-6}	1×10^{-6}	1×10^{-6}
epochs	30	30	30	30
Adam_rs	0.02	0.02	2×10^{-4}	2×10^{-4}
Adam_kge	0.01	0.01	2×10^{-5}	2×10^{-5}

4.5. Results

4.5.1. CTR Prediction Results and Analysis

The CTR prediction results are shown in Table 4. The MDAR model proposed in this chapter was compared with seven baseline models in terms of predicting recommendation algorithms. From Table 4, MDAR performs better than the other compared models on two indicators. In detail, on the MovieLens-1M datasets, compared with the MKR baseline model, MDAR achieves an improvement of approximately 1.2% in AUC and 1.40% in accuracy (ACC) on the test dataset. Similarly, on the MovieLens-20M datasets, MDAR surpasses the basic MKR with an increase of around 1.50% in AUC and 1.2% in accuracy (ACC) on the test dataset. This proves that MDAR is an effective enhanced recommendation of MKR, which can more effectively improve the utilization of the data in RS. In addition, on the BookCrossing dataset, MDAR exhibits notable enhancement, with an improvement of approximately 1.90% in AUC and 1.30% in accuracy (ACC) compared to the MKR baseline model on the test dataset. Similarly, on the Last.FM dataset, MDAR demonstrates a significant improvement, with an approximately 1.90% increase in AUC and a 1.30% increase in accuracy (ACC) on the test dataset when compared to the MKR baseline model. This is because, in a sparse environment, the user's behavior preferences can be more effectively understood by using the user's history of interaction records. Moreover, we compared the CTR prediction of MDAR with KGCN, KGAT, CKAN, and EMKR. MDAR performed well in prediction, thus indicating the added relation-fused multi-head graph attention network can effectively extract the deep relationships in KG, thus helping with recommendations.

Table 4. The results of AUC and ACC in CTR prediction.

	MovieLens-1M		**MovieLens-20M**		**BookCrossing**		**Last.FM**	
	AUC	**ACC**	**AUC**	**ACC**	**AUC**	**ACC**	**AUC**	**ACC**
CKE	0.801	0.742	0.924	0.897	0.671	0.633	0.744	0.673
DKN	0.655	0.589	0.801	0.757	0.622	0.598	0.601	0.581
KGCN	0.901	0.824	0.974	0.922	0.677	0.622	0.801	0.729
MKR	0.914	0.840	0.962	0.907	0.734	0.704	0.791	0.742
KGAT	-		0.976	0.925	0.731	0.669	0.829	0.740
CKAN	0.887	0.811	0.976	**0.929**	0.739	0.665	**0.842**	0.745
EMKR	0.923	0.848	-		0.746	0.705	0.808	0.753
MDAR	**0.926**	**0.854**	**0.977**	0.919	**0.753**	**0.717**	0.810	**0.755**

4.5.2. TopK Results and Analysis

As shown in Figures 7 and 8, the MDAR model is better than the CKAN and EMKR models with the best results in two indicators: Precision@K and Recall@K, which indicates that the AFM is used to extract user cross features. After the relation-fused multi-head graph attention network is used to process the head entity features and relationship features of the knowledge graph, the recommendation effect of the model is further improved. In terms of specific values, compared with the CKAN model, the MDAR model has the indicators of Precision@K increased by about 2.13%and Recall@K increased by about 1.24% in the best effect.

On the two relatively sparse datasets, BookCrossing and Last.FM, it was proven that MDAR can handle sparse data well. In contrast to KGCN and MKR, MDAR shows a significant improvement in Precision@K and Recall@K, which demonstrates the importance and effectiveness of introducing multi-head attention mechanisms into relational modeling. MDAR also shows good performance compared to KGAT, due to the fact that MDAR uses a multi-headed attention mechanism to reconstruct the relationships in the knowledge graph, thus learning a richer representation for each entity and enhancing the user's representation of the item. In terms of specific values, the best effect of the MDAR model is higher than that of the CKAN model, the index of Precision@K is increased by 2.75%, and the best effect of the MDAR model is higher than that of the CKAN model; the index of Recall@K is increased by 2.58%.

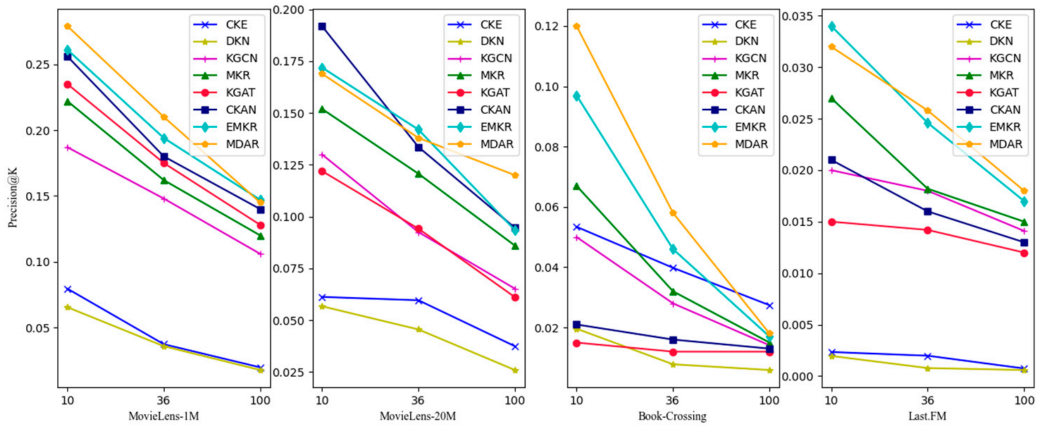


Figure 7. The Precision@K results in the top-K recommendation.

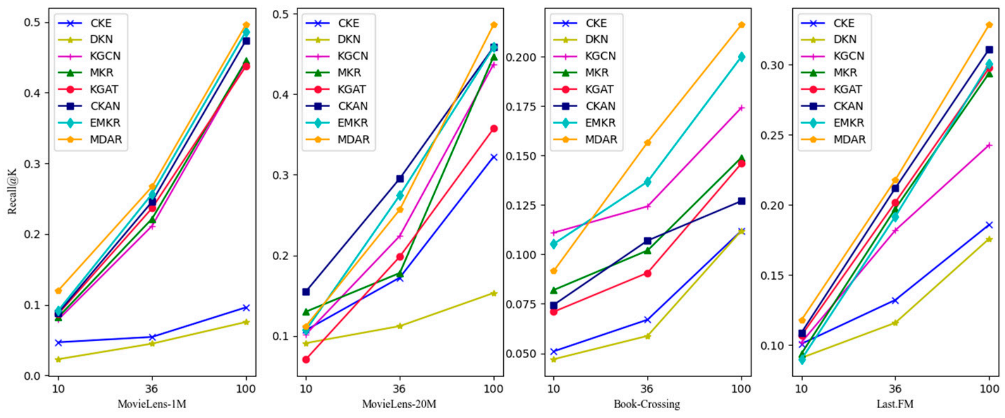


Figure 8. The Recall@K results in the top-K recommendation.

4.6. Ablation Experiment

4.6.1. Impact of Different Components on the Model

In order to evaluate the impact of each component on the overall performance of the model, we performed a quantitative analysis using ablation experiments. Specifically, we examined three factors: the AFM module, the DNN network, and the RMGAT module. The findings of these experiments are reported in Table 5.

Table 5. Impact of different components on AUC.

	MovieLens-1M	MovieLens-20M	BookCrossing	Last.FM
MKR (none)	0.914	0.963	0.734	0.791
AFM_MKR	0.919	0.967	0.737	0.782
DNN_MKR	0.917	0.965	0.738	0.775
DeepAFM_MKR	0.923	0.969	0.742	0.801
MKR_RMGAT	0.925	0.973	0.744	0.804
MDAR	**0.927**	**0.975**	**0.753**	**0.810**

By conducting experiments using different components and datasets, we discovered that incorporating both DeepAFM and RMGAT into the recommendation system led to improved performance. However, we found that the RMGAT module outperformed the DeepAFM module. In detail, by adding the attention mechanism to the DeepAFM module, we observed an increase in the AUC by 0.9%, 0.6%, 0.8%, and 1.0% across the four datasets. This highlights the importance of distinguishing different features through the attention network and leveraging FM to extract linear interaction features for accurate target prediction, ultimately enhancing recommendation performance. Additionally, the inclusion of the RMGAT model also yielded favorable results. On the MovieLens-1M dataset, the AUC increased by approximately 1.10%. On the MovieLens-20M dataset, the AUC increased by approximately 1.00%. On the BookCrossing dataset, the AUC increased by approximately 1.00%. Lastly, on the Last.FM dataset, the AUC increased by approximately 1.30%. These findings demonstrate that the designed relation-fused multi-head attention network can extract more detailed feature information during the analysis of users and items, resulting in more accurate prediction outcomes and significantly improving recommendation performance, especially for datasets with high sparsity.

4.6.2. Impact of Different Networks on the Model

We compared the AUC by adding CNN, GCN, GNN, and the relational fusion of multi-headed attention networks that we proposed to the knowledge graph. From Table 6, we can see that MDAR shows good performance on the movie datasets, and the model also performs better than adding other networks on the sparse datasets, which indicates that neighborhood relations play a crucial role in the establishment of the knowledge graph, and also proves the effectiveness of the proposed method in this paper.

Table 6. Impact of different networks on the AUC.

	MovieLens-1M	MovieLens-20M	BookCrossing	Last.FM
MKR_KG	0.914	0.963	0.734	0.791
MKR_CNN	0.899	0.959	0.689	0.786
MKR_GCN	0.919	0.967	0.737	0.782
MKR_GNN	0.925	0.973	0.744	0.804
MDAR	**0.927**	**0.975**	**0.753**	**0.810**

4.7. Parameter Sensitivity

4.7.1. Impact of the Embedding's Dimension Dim

The results in Table 7 are intuitive in that different scenarios have different dim values, such as 4, 8, 12, and 16 in the movie, book, and music datasets. From Table 7, we can observe that the best result is 8 on the movie dataset. The best result is 16 dim on book dataset and 4 dim on the music dataset. It is clear that a dim that is too small will cause missing information, but a dim that is too large will experience overfitting.

Table 7. Impact of the embedding dimension dim on the AUC.

dim	4	8	12	16
MovieLens-1M	0.918	**0.926**	0.922	0.921
MovieLens-20M	0.967	**0.977**	0.965	0.963
BookCrossing	0.742	0.745	0.751	**0.753**
Last.FM	**0.810**	0.807	0.805	0.804

4.7.2. Impact of L Low-Level Layer and H High-Level Layer Numbers

As Tables 8 and 9 show, for MovieLens-1M and MovieLens-20M, the AUC decreases as the number of low-level layers rises, increases as the number of low-level layers rises, and decreases after reaching a certain peak. For BookCrossing and Last.FM, the AUC peaks at 2 of L and shows good performance at 1 of H. This indicates that not much information needs to be exchanged at the low level when user–item interaction enriches the data, while longer fitting is required at the high level. For datasets with sparse data, it becomes important to mine and share information at the low level.

Table 8. Impact of L low-level layer numbers on the AUC.

L	**1**	**2**	**3**	**4**
MovieLens-1M	**0.926**	0.924	0.922	0.921
MovieLens-20M	**0.977**	0.967	0.965	0.963
BookCrossing	0.746	**0.753**	0.742	0.740
Last.FM	0.807	**0.810**	0.805	0.804

Table 9. Impact of H high-level layer numbers on the AUC.

H	**1**	**2**	**3**	**4**
MovieLens-1M	0.922	0.924	**0.926**	0.920
MovieLens-20M	0.963	0.965	**0.975**	0.967
BookCrossing	**0.746**	0.744	0.742	0.739
Last.FM	**0.810**	0.807	0.805	0.804

4.7.3. Impact of Knowledge Propagation Layers and Hop

In the experiments, the impact of neighborhood information on the performance of the algorithm in this paper was analyzed by varying the number of hops in the knowledge graph. As shown in Figure 9, the AUC peaks when the number of layers on MovieLens-1M, MovieLens-20M, BookCrossing, and Last.FM are 2, 2, 3, and 2, respectively. This is due to the fact that the more hops away from the user's historical click items in the knowledge graph, the lower the relevance of the entity to the user. However, when the number of hops is too small, it is difficult to obtain additional information from the knowledge graph.

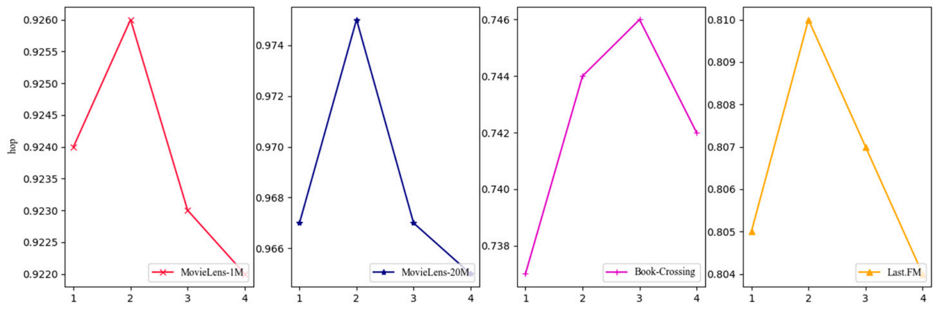


Figure 9. Impact of layers and hops on the AUC.

4.7.4. Impact of the Number of Attention Heads

To assess how the number of attention heads affects the performance of the model, we conducted experiments on four different datasets. As shown in Figure 10, on the movie datasets, the AUC peaks when the number of attention heads is 8, while on the book and music datasets, the AUC peaks at 12 attention heads. The number of attention heads is different for different datasets, but there is an improvement in performance for all of them. This is because multiple attention heads can focus on the importance of different features, which not only stabilizes the learning process but also prevents overfitting. When the number of attention heads is 8 or 12, the experimental results are optimal. However, as the number of attention heads increases, redundant information is introduced, which degrades the performance of the model.

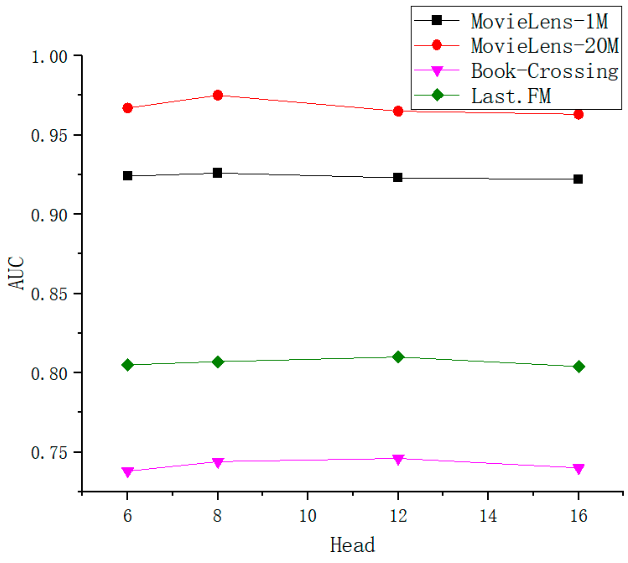


Figure 10. Impact of attention heads on the AUC.

4.7.5. Impact of the Different Ratios r of the Test Set

As Table 10 and Figure 11 show, we conducted experiments on sparsity to evaluate the performance of our model. The results indicate that there is a consistent increase in the AUC as the ratio increases, with the highest value achieved at a ratio of 1. Specifically, when the ratio is increased from 0.2 to 1.0, our model shows an average improvement in the AUC of 2.5%, 2.2%, 4.7%, and 3.40%, surpassing the baseline MKR. These findings demonstrate that the MDAR model performs well even in situations with limited user–item interactions.

Table 10. Impact of the different test set ratios r on the AUC.

r	20%	40%	60%	80%	100%
MovieLens-1M	0.901	0.907	0.917	0.921	**0.926**
MovieLens-20M	0.955	0.961	0.965	0.967	**0.977**
BookCrossing	0.706	0.716	0.732	0.746	**0.753**
Last.FM	0.776	0.784	0.792	0.804	**0.810**

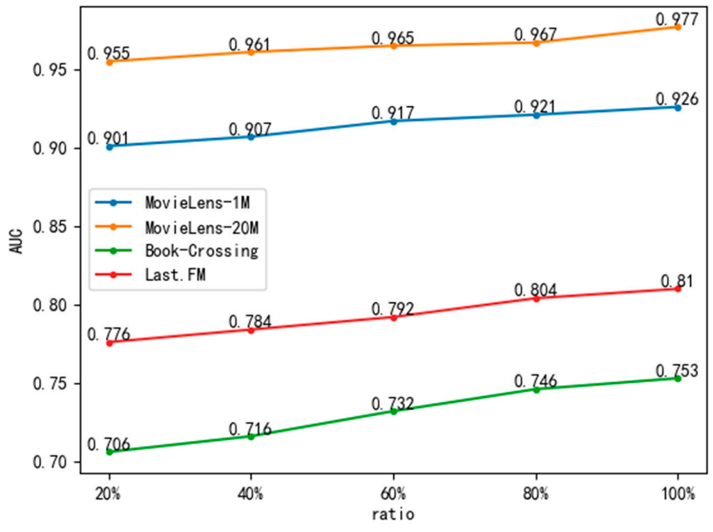


Figure 11. Impact of the sparsity of the test set on the AUC.

5. Discussion

In this study, we focused on the issue of underutilizing data and made improvements to three components in the original MKR. In contrast to existing methods for enhancing recommendations, we introduced, for the first time, the integration of attention mechanisms, factorization machines, and deep learning networks into the MKR model, which allowed us to capture users' behavior preferences. The experimental results demonstrated the effectiveness of our approach. Additionally, we applied the graph attention network (GAT) to the knowledge graph. While previous methods have only used graph convolutional networks (GCN) to capture neighborhood features based on node information in the knowledge graph, we introduced GAT to incorporate both node information and relationship information in the graph. This enhancement enabled us to better learn the integrity of entities.

Although this study has revealed important discoveries, our study has some limitations. In the future, we can explore advanced methods to further improve the performance of our recommendations.

6. Conclusions

In this paper, we propose a method called MDAR, the multi-head relation-fused graph attention network and DeepFM with an attention mechanism, to address the issue of data underutilization in original KGRSs. Firstly, we leverage a DeepFM with an attention mechanism (DeepAFM) in the RS module to distinguish the importance of different feature interactions, extracting low-order linear interaction features crucial for target prediction. Secondly, we employ deep neural networks (DNNs) in the cross-compression unit to extract complex interaction features, facilitating the exchange of deep semantic information between the RS modules and KGE modules. Finally, in the KGE module, we utilize a relation-fused multi-head graph attention network (RMGAT), aggregating multi-relation neighborhood features of a given entity into the KG network to learn a comprehensive entity representation, effectively addressing the problem of knowledge graph completion. The experimental results on real-world datasets demonstrate that MDAR outperforms state-of-the-art baselines in terms of recommendation performance and model efficiency.

In future research, we intend to evaluate the model on more datasets to validate the applicability of MDAR. Meanwhile, we also plan to enhance MDAR in the following aspects:

- Incorporating reinforcement learning and meta-learning into recommendation systems [43,44] to improve the adaptive capability of our system;
- Exploring different attention mechanisms [45,46] in the sharing units to better distinguish feature contributions, advancing research in the recommendation field;
- Building temporal knowledge graphs (TKGs) by adding timestamps to the KG [47–49] to complement the knowledge graph information, offering significant insights into knowledge graph enhancement.

Overall, we aim to utilize various learning methods to continually enhance our recommendation system's performance and broaden its application to diverse fields.

Author Contributions: Conceptualization, Q.X.; methodology, Q.X.; software, Q.X.; validation, Q.X.; formal analysis, S.L.; investigation, P.W. and S.L.; resources, P.W.; writing—original draft, Q.X.; writing—review and editing, S.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Key Research and Development Program of China under Grant No. 2021YFB1714400.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The datasets generated and analyzed during the current study are available from the corresponding author upon reasonable request.

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

References

1. He, Z.; Hui, B.; Zhang, S.; Xiao, C.; Zhong, T.; Zhou, F. Exploring indirect entity relations for knowledge graph enhanced recommender system. *Expert Syst. Appl.* **2023**, *213*, 118984. [CrossRef]
2. Yongheng, M.; Yun, W. Multimodal Movie Recommendation System Using Deep Learning. *Mathematics* **2023**, *11*, 895.
3. 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]
4. Wang, F.; Li, Y.; Zhang, Y.; Wei, D. KLGCN: Knowledge graph-aware light graph convolutional network for recommender systems. *Expert Syst. Appl.* **2022**, *195*, 116513. [CrossRef]
5. Guo, X.; Lin, W.; Li, Y.; Liu, Z.; Yang, L.; Zhao, S.; Zhu, Z. DKEN: Deep knowledge-enhanced network for recommender systems. *Inf. Sci.* **2020**, *540*, 263–277. [CrossRef]
6. Yang, Z.; Dong, S. HAGERec: Hierarchical Attention Graph Convolutional Network Incorporating Knowledge Graph for Explainable Recommendation. *Knowl. Based Syst.* **2020**, *204*, 106194. [CrossRef]
7. Palumbo, E.; Monti, D.; Rizzo, G.; Troncy, R.; Baralis, E. entity2rec: Property-specific knowledge graph embeddings for item recommendation. *Expert Syst. Appl.* **2020**, *151*, 113235. [CrossRef]
8. Khan, N.; Ma, Z.; Ullah, A.; Polat, K. Categorization of knowledge graph based recommendation methods and benchmark datasets from the perspectives of application scenarios: A comprehensive survey. *Expert Syst. Appl.* **2022**, *206*, 117737. [CrossRef]
9. Yan, Q.; Fan, J.; Li, M.; Qu, G.; Xiao, Y. A Survey on Knowledge Graph Embedding. In Proceedings of the 7th IEEE International Conference on Data Science in Cyberspace (DSC), Guilin, China, 11–13 July 2022; pp. 576–583. [CrossRef]
10. 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. [CrossRef]
11. Lin, Y.; Liu, Z.; Sun, M.; Liu, Y.; Zhu, X. Learning entity and relation embeddings for knowledge graph completion. In Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI'15), Austin, TX, USA, 25–30 January 2015; AAAI Press: New York, NY, USA, 2015; pp. 2181–2187.
12. Wang, H.; Zhang, F.; Guo, M. DKN: Deep Knowledge-Aware Network for News Recommendation. In Proceedings of the World Wide Web Conference (WWW '18), Lyon, France, 23–27 April 2018; International World Wide Web Conferences Steering Committee: Geneva, Switzerland, 2018; pp. 1835–1844. [CrossRef]
13. 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, Beijing, China, 26–31 July 2015; pp. 687–696.
14. Wang, H.; Zhang, F.; Zhao, M.; Li, W.; Xie, X.; Guo, M. Multi-task feature learning for knowledge graph enhanced recommendation. In *The World Wide Web Conference*; Association for Computing Machinery: New York, NY, USA, 2019; pp. 2000–2010.
15. Rendle, S. Factorization Machines. In Proceedings of the IEEE International Conference on Data Mining, Sydney, NSW, Australia, 14 December 2010; pp. 995–1000. [CrossRef]
16. Xiao, J.; Ye, H.; He, X.; Zhang, H.; Wu, F.; Chua, T.-S. Attentional factorization machines: Learning the weight of feature interactions via attention networks. In Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI'17), Melbourne, VIC, Australia, 19–25 August 2017; AAAI Press: New York, NY, USA, 2017; pp. 3119–3125.
17. Guo, H.; Tang, R.; Ye, Y.; Li, Z.; He, X. DeepFM: A factorization-machine based neural network for CTR prediction. In Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI'17), Melbourne, VIC, Australia, 19–25 August 2017; AAAI Press: New York, NY, USA, 2017; pp. 1725–1731.
18. 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 Information and Knowledge Management, Torino, Italy, 22–26 October 2018; pp. 417–426.
19. Yang, Y.; Jaechoon, J.; Lim, H. Unifying user preference and item knowledge-based similarity models for top-N recommendation. *Pers. Ubiquitous Comput.* **2019**, *26*, 407–416. [CrossRef]
20. Fan, H.; Zhong, Y.; Zeng, G.; Ge, C. Improving recommender system via knowledge graph based exploring user preference. *Appl. Intell.* **2022**, *52*, 10032–10044. [CrossRef]
21. Wang, J.; Shi, Y.; Yu, H.; Yan, Z.; Li, H.; Chen, Z. A novel KG-based recommendation model via relation-aware attentional GCN. *Knowl. Based Syst.* **2023**, *275*, 110702. [CrossRef]
22. Wang, H.; Zhao, M.; Xie, X.; Li, W.; Guo, M. Knowledge graph convolutional networks for recommender systems. In *The World Wide Web Conference*; Association for Computing Machinery: New York, NY, USA, 2019; pp. 3307–3313. [CrossRef]
23. 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 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Anchorage, AK, USA, 4–8 August 2019; pp. 968–977.

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 (SIGIR '20), Virtual, 25–30 July 2020; Association for Computing Machinery: New York, NY, USA, 2020; pp. 219–228. [CrossRef]
25. Li, C.-T.; Hsu, C.; Zhang, Y. FairSR: Fairness-aware Sequential Recommendation through Multi-Task Learning with Preference Graph Embeddings. *ACM Trans. Intell. Syst. Technol.* **2022**, *13*, 1–21. [CrossRef]
26. 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.
27. Duan, H.; Liu, P.; Ding, Q. RFAN: Relation-fused multi-head attention network for knowledge graph enhanced recommendation. *Appl. Intell.* **2022**, *53*, 1068–1083. [CrossRef]
28. Dai, Q.; Wu, X.-M.; Fan, L.; Li, Q.; Liu, H.; Zhang, X.; Wang, D.; Lin, G.; Yang, K. Personalized Knowledge-Aware Recommendation with Collaborative and Attentive Graph Convolutional Networks. *Pattern Recognit.* **2022**, *128*, 108628. [CrossRef]
29. Wang, Q.; Cui, H.; Zhang, J.; Du, Y.; Zhou, Y.; Lu, X. Neighbor-Augmented Knowledge Graph Attention Network for Recommendation. *Neural Process. Lett.* **2023**, *7*, 1–17. [CrossRef]
30. He, Q.; Liu, S.; Liu, Y. Optimal Recommendation Models Based on Knowledge Representation Learning and Graph Attention Networks. *IEEE Access* **2023**, *11*, 19809–19818. [CrossRef]
31. Xiao, Y.; Li, C.; Liu, V. DFM-GCN: A Multi-Task Learning Recommendation Based on a Deep Graph Neural Network. *Mathematics* **2022**, *10*, 721. [CrossRef]
32. Chen, L.; Bi, X.; Fan, G.; Sun, H. A multitask recommendation algorithm based on DeepFM and Graph Convolutional Network. *Concurr. Comput. Pract. Exp.* **2022**, *35*, e7498. [CrossRef]
33. Gao, M.; Li, J.-Y.; Chen, C.-H.; Li, Y.; Zhang, J.; Zhan, Z.-H. Enhanced Multi-Task Learning and Knowledge Graph-Based Recommender System. *IEEE Trans. Knowl. Data Eng.* **2023**, *1*, 1–14. [CrossRef]
34. Shu, H.; Huang, J. Multi-task feature and structure learning for user-preference based knowledge-aware recommendation. *Neurocomputing* **2023**, *532*, 43–55. [CrossRef]
35. Deng, S.; Qin, J.; Wang, X.; Wang, R. Attention Knowledge Network Combining Explicit and Implicit Information. *Mathematics* **2023**, *11*, 724. [CrossRef]
36. Liu, W.; Yin, L.; Wang, C.; Liu, F.; Ni, Z. Multitask Healthcare Management Recommendation System Leveraging Knowledge Graph. *J. Healthc. Eng.* **2021**, *2021*, 1233483. [CrossRef]
37. Xu, Z.; Liu, H.; Li, J.; Zhang, Q.; Tang, Y. CKGAT: Collaborative Knowledge-Aware Graph Attention Network for Top-N Recommendation. *Appl. Sci.* **2022**, *12*, 1669. [CrossRef]
38. Zhang, X.; Ma, H.; Gao, Z.; Li, Z.; Chang, L. Exploiting cross-session information for knowledge-aware session-based recommendation via graph attention networks. *Int. J. Intell. Syst.* **2022**, *37*, 7614–7637. [CrossRef]
39. Zhang, Y.; Wu, X.; Fang, Q.; Qian, S.; Xu, C. Knowledge-Enhanced Attributed Multi-Task Learning for Medicine Recommendation. *ACM Trans. Inf. Syst.* **2023**, *41*, 1–24. [CrossRef]
40. Yang, X.; Ding, C. SMRGAT: A traditional Chinese herb recommendation model based on a multi-graph residual attention network and semantic knowledge fusion. *J. Ethnopharmacol.* **2023**, *315*, 116693. [CrossRef] [PubMed]
41. Zhang, S.; Lin, X.; Bai, Z.; Li, P.; Fan, H. CGRS: Collaborative Knowledge Propagation Graph Attention Network for Recipes Recommendation. *Connect. Sci.* **2023**, *35*, 2212883. [CrossRef]
42. Chen, Y.; Guo, Y.; Fan, Q.; Zhang, Q.; Dong, Y. Health-Aware Food Recommendation Based on Knowledge Graph and Multi-Task Learning. *Foods* **2023**, *12*, 2079. [CrossRef]
43. Tao, S.; Qiu, R.; Cao, Y.; Xue, G.; Ping, Y. Path-guided intelligent switching over knowledge graphs with deep reinforcement learning for recommendation. *Complex Intell. Syst.* **2023**, *6*, 2198–6053. [CrossRef]
44. Chang, Y.; Zhou, W.; Cai, H.; Fan, W.; Hu, L.; Wen, J. Meta-relation assisted knowledge-aware coupled graph neural network for recommendation. *Inf. Process. Manag.* **2023**, *60*, 103353. [CrossRef]
45. Wang, J.; Xie, H.; Wang, F.L.; Lee, L.K. A Transformer–Convolution Model for Enhanced Session-Based Recommendation. *Neurocomputing* **2023**, *531*, 21–33. [CrossRef]
46. Gao, L.; Lan, Y.; Chen, Q.; Liu, Y. A Personalized Paper Recommendation Method Based on Knowledge Graph and Transformer Encoder with a Self-Attention Mechanism. *SSRN Electron. J.* **2023**. [CrossRef]
47. Xu, Y.; Ou, J.; Xu, H.; Fu, L. Temporal Knowledge Graph Reasoning with Historical Contrastive Learning. *Proc. AAAI Conf. Artif. Intell.* **2023**, *37*, 4765–4773. [CrossRef]
48. Nie, H.; Zhao, X.; Yao, X.; Jiang, Q.; Bi, X.; Ma, Y.; Sun, Y. Temporal-structural importance weighted graph convolutional network for temporal knowledge graph completion. *Future Gener. Comput. Syst. Int. J. Escience* **2023**, *143*, 30–39. [CrossRef]
49. Bai, L.; Ma, X.; Meng, X.; Ren, X.; Ke, Y. RoAN: A relation-oriented attention network for temporal knowledge graph completion. *Eng. Appl. Artif. Intell.* **2023**, *123*, 106308. [CrossRef]