electronics

MDPI

Article

KGCFRec: Improving Collaborative Filtering Recommendation with Knowledge Graph

Jiquan Peng [1,2], Jibing Gong [1,2], Chao Zhou [1,2], Qian Zang [1,2], Xiaohan Fang [1,2], Kailun Yang [1] and Jing Yu [1,*]

[1] School of Information Science and Engineering, Yanshan University, Qinhuangdao 066004, China; pengjiquan@stumail.ysu.edu.cn (J.P.); gongjibing@ysu.edu.cn (J.G.); chaozhou@stumail.ysu.edu.cn (C.Z.); ysuzangqian@stumail.ysu.edu.cn (Q.Z.); fangxiaohan98@stumail.ysu.edu.cn (X.F.); yangkailun@stumail.ysu.edu.cn (K.Y.)

[2] The Key Laboratory for Computer Virtual Technology and System Integration of Hebei Province, Qinhuangdao 066004, China

* Correspondence: xyyj@ysu.edu.cn

Abstract: Traditional collaborative filtering (CF)-based recommendation systems are often challenged by data sparsity. The recent research has recognized the potential of integrating new information sources, such as knowledge graphs, to address this issue. However, a common drawback is the neglect of the interplay between user–item interaction data and knowledge graph information, resulting in insufficient model performance due to coarse-grained feature fusion. To bridge this gap, in this paper, we propose a novel graph neural network (GNN) model called KGCFRec, which leverages both **K**nowledge **G**raph and user–item **C**ollaborative **F**iltering information for an enhanced **Rec**ommender system. KGCFRec employs a dual-channel information propagation and aggregation mechanism to generate distinct representations for the collaborative knowledge graph and the user–item interaction graph. This is followed by an attention mechanism that adaptively fuses the knowledge graph with collaborative information, thereby refining the representations and narrowing the gap between them. The experiments conducted on three real-world datasets demonstrate that KGCFRec outperforms state-of-the-art methods. These promising results underscore the capability of KGCFRec to enhance recommendation accuracy by integrating knowledge graph information.

Keywords: recommendation systems; graph neural network; collaborative filtering; knowledge graph; feature fusion

Citation: Peng, J.; Gong, J.; Zhou, C.; Zang, Q.; Fang, X.; Yang, K.; Yu, J. KGCFRec: Improving Collaborative Filtering Recommendation with Knowledge Graph. *Electronics* **2024**, *13*, 1927. https://doi.org/10.3390/electronics13101927

Academic Editor: Stefano Ferilli

Received: 15 April 2024
Revised: 7 May 2024
Accepted: 10 May 2024
Published: 15 May 2024

1. Introduction

Recommendation systems (RSs) [1] have become widely used in multimedia applications [2–4], social networks [5,6], and e-commerce platforms [7–9] with the growth of internet services. Great success has been achieved by collaborative filtering (CF) recommendation, which takes into account the users' previous interaction behavior and makes recommendations based on potential similar preferences of users who share similar interests. The CF-based RS, however, generally suffers the data sparsity problem. In order to overcome the problem, some recent research [10] has focused on using side information, such as user attribution and item description. The representation of relationships between users and items can be enhanced and complemented by the side information of the knowledge graph (KG) [11,12]. From certain perspectives, KGs can partially mitigate the issue of the data sparsity problem.

However, the recently proposed KG-based RS models have shortcomings in information modeling and information fusion, resulting in sub-optimal recommendation performance. The existing methods, which independently model knowledge graphs and collaborative information, are faced with problems such as insufficient capability for long-distance information modeling [13,14] and complex task-oriented design [15,16]. In the feature fusion stage, these models tend to ignore the gap between the representation

distribution and combine the knowledge graph and the collaborative information in a coarse-grained way [17,18]. How to perform adaptive feature fusion from the knowledge graph side and the collaborative information side in an end-to-end manner to improve the recommendation performance is still a common challenge facing the current field.

More recently, creating end-to-end models based on a graph neural network (GNN) has been a popular technical approach. The key idea is to utilize the message passing and aggregation mechanism, which can effectively integrate the KG and collaborative information into the embedding space. Motivated by this, we created a novel framework we called KGCFRec, which is a kind of two-channel information propagation and aggregation method aimed to produce the corresponding representations of collaborative knowledge graphs and user–item interaction graphs.

KGCFRec consists of two components to address the challenge in the appropriate way: (1) **Knowledge and Collaborative-aware representation learning.** We predefine the knowledge graph, the user–item interactions graph, and the collaborative-knowledge graph. Then, we design a novel aggregation mechanism of GNN for knowledge- and collaborative-aware representation learning. (2) **Adaptive feature fusion.** In order to achieve adaptively fine-grained fusion of the two types of features, an attention mechanism is designed to learn the weights between nodes on the two graphs throughout the propagation process. At last, the collaborative knowledge graph representation for Top-N recommendation is generated.

In summary, our work makes the following contributions:

- The knowledge graph and user–item interactions graph are combined into a single collaborative-knowledge graph in order to address the data sparsity problem.
- KGCFRec, an end-to-end model, is proposed that fuses knowledge and collaboration features for recommendation via a novel attention mechanism.
- Experiments conducted on three real-world datasets demonstrate that our proposals perform significantly better than baselines.

The organization of our paper is shown as follows: we first present related work in Section 2 and then provide the problem formalization in Section 3. Section 4 describes the knowledge- and collaborative-aware representation learning module and the adaptive feature fusion module. Section 5 describes the experimental setup. The experiment results and discussion are provided in Section 6. In the end, the conclusion and future work are shown in Section 7.

2. Related Work

There are three primary types of current study on knowledge-graph-enhanced recommendation: embedding-based, path-based, and GNN-based.

2.1. Embedding-Based Methods

Embedding-based methods primarily extract the data from the first-order neighboring nodes in the knowledge graph (KG) and the user–item interaction graph [19]. First of all, these approaches frequently use KG embedding techniques like TransE [20] and TransH [21] to improve the entity embeddings. These entity embeddings are then used as item semantic characteristics for supplying the recommendation model. For instance, TransE is utilized by collective knowledge embedding (CKE) [13] to extract entity embeddings from the KG and effectively incorporates them into matrix factorization (MF). Knowledge-aware transferable user preference (KTUP) [14] constructs a multi-task framework that applies TransH to both user–item interactions and KG triplets concurrently, aiming to learn user preferences and accomplish KG completion in unison. However, instead of directly recommending, KG embedding algorithms usually concentrate on modeling strict semantic relationships, which are better suited for link prediction tasks. Furthermore, embedding-based methods frequently ignore higher-order connectivity and lack end-to-end training regimes. This oversight prevents them from capturing the extensive semantics or sequential dependencies along the paths between nodes, thereby constraining their capability to reveal the deeper, underlying user–item relationships.

2.2. Path-Based Methods

Path-based methods consider the relational patterns linking users and items through KG entities, often referred to as meta-paths [22–24], to enhance the predictive capabilities of recommendation systems. In order to extract paths that convey valuable higher-order information, researchers have resorted to either deploying algorithms that pinpoint significant paths or crafting predefined meta-path templates to limit the scope of potential paths. These identified paths are subsequently harnessed to anticipate user preferences through the use of recurrent neural networks and memory networks. A case in point is RippleNet [15], which concentrates on the propagation of user interests and identifies potential paths from items interacted with by users to other items, without the need for predefining meta-paths or meta-graphs. In order to improve top-N recommendations, MCRec [16] has built a deep neural network with a co-attention mechanism that takes advantage of the rich context provided by meta-paths within various information networks. Path-based techniques rely heavily on manually generated meta-paths/meta-graphs, which might complicate performance tweaking in real-world scenarios, even though they can more organically incorporate KG insights.

2.3. GNN-Based Methods

GNN-based methods exploit the information aggregation prowess of GNNs to derive node representations. The key of GNN-based methods is to develop powerful node representations [25–27]. They achieve this by integrating information from the immediate neighboring nodes to refine the representations of the central nodes. Through iterative application of this process, information from nodes several hops away can be incorporated into the representations. This enables these methods to capture extended connectivity patterns. To enhance the vector representations of items, knowledge graph convolutional networks (KGCN) [28] utilize an end-to-end design, a biased neighbor aggregation strategy, and a graph convolution sampling technique to capture distant inter-item interactions. KGNN-LS [29], on the other hand, computes the embedding of each item node by aggregating feature information within the node's local network neighborhood. By designing unique aggregation algorithms for each, CKAN [18] distributes and aggregates data throughout the KG and the user–item interactions graph, respectively. KGAT [17] integrates user–item interactions with the KG into a unified heterogeneous graph and employs an attention mechanism to iteratively propagate embeddings from neighboring nodes—be they users, items, or attributes—to further refine each node's embedding.

While numerous studies have integrated KG and GNN techniques into recommender systems, the fusion of KG information with collaborative data has not been extensively explored. KGCFRec addresses this gap by learning dual information weights through an attention-based fusion mechanism, enhancing the accuracy and robustness of recommendation. Compared with embedding-based methods such as MF [13], KGCFRec is good at modeling long-distance information. Compared with path-based methods such as CKE [30], the task oriented complex design is eliminated; Compared to GNN-based methods [17,18,29,31–33], efficient modeling knowledge and the collaborative perception of dual-channel information can be adaptively fused. The substantial benefits of information fusion are also demonstrated by the end-to-end framework architecture.

3. Preliminaries

The aim of this work is to study KG-augmented recommendation, where the recommender system uses knowledge graph information to improve performance and make recommendations for items to users. We begin by describing the underlying structural data, including the knowledge graph and the user–item interactions graph, followed by a formulation of our task.

Knowledge Graph: The power of the KG approach lies in its ability to store structured auxiliary data—such as item attributes, taxonomies, or external common-sense knowledge—in the form of a directed graph, denoted as $KG =< E, R, T >$. Here, E and T

represent the sets of entities, while R signifies the set of relations. We formally organize this side information as a knowledge graph, defined as $G_{KG} = \{(h, r, t) \mid h, t \in \epsilon, r \in R\}$, where ϵ encompasses real-world entities, and R comprises the set of relations. For instance, the tuple (Tom Hanks, ActorOf, Forrest Gump) indicates that Tom Hanks is an actor in the film Forrest Gump. By mapping items to KG entities ($I \subset \epsilon$), item i can be correlated with an entity h, t within the KG. Consequently, the KG can provide detailed profiles of items and offer supplementary information to the interaction data.

User-Item Interaction Graph: In this work, we focus on implicit feedback within recommendations, where users' historical interactions, indicative of their preferences, are indirectly expressed through actions such as viewing, clicking, and purchasing. We conceptualize these interaction data as a user–item bipartite graph, denoted as G_{UI}, which is formalized as $\{(u, d_{ui}, i) \mid u \in U, i \in I\}$. Here, $U = \{u_1, ..., u_N\}$ represents the set of users, and $I = \{i_1, ..., i_M\}$ represents the set of items. A link with $d_{ui} = 1$ signifies an interaction between user u and item i, whereas $d_{ui} = 0$ indicates the absence of such interaction.

Task Description: Hence, for the KG-based recommendation approach we are introducing, we formalize the recommendation task as follows:

$$\mathbb{F}(G \mid \Theta_{\mathbb{F}}) \rightarrow \mathbb{F}(G_{KG}, G_{UI} \mid \Theta_{\mathbb{F}}) \rightarrow \hat{y} \tag{1}$$

where $\Theta_{\mathbb{F}}$ represents the parameters of the model that need to be optimized.

Input: A collaborative knowledge graph G, which encompasses the knowledge graph G_{KG} and the user–item interactions graph G_{UI}.

Output: A prediction score $\hat{y}_{ui}$, representing the likelihood of user u engaging with item i.

4. The Proposed Methodology

We now introduce our model, a GNN-based framework designed for KG-augmented recommendation that operates on a collaborative knowledge graph. As depicted in Figure 1, KGCFRec comprises three principal components: (a) knowledge-aware representation learning, (b) collaborative-aware representation learning, and (c) adaptive feature fusion.

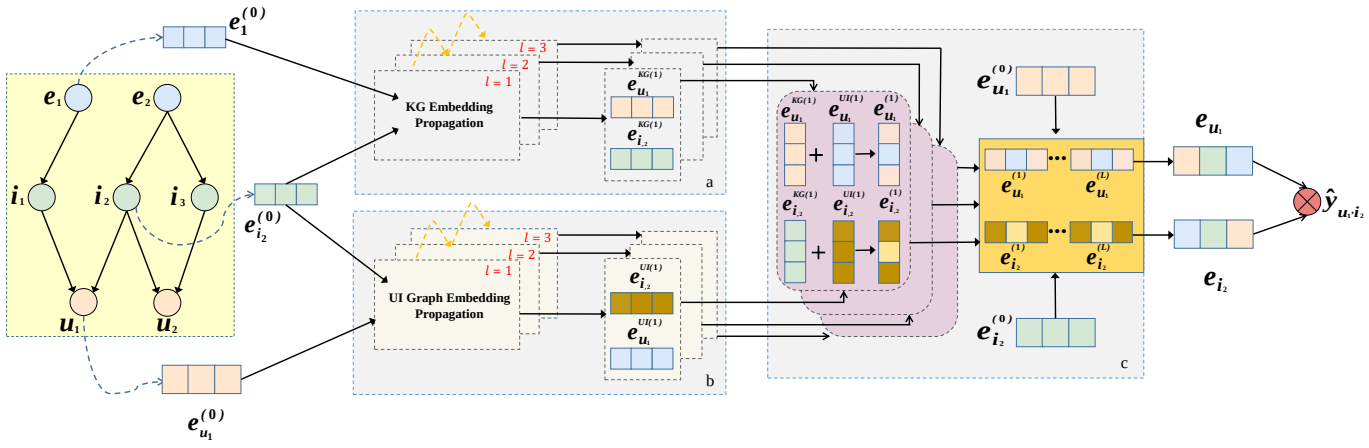

Figure 1. The overall framework of KGCFRec. (**a**) KGCFRec initiates the process by encoding the knowledge information of user u and item i within the KG using varying numbers of GNN layers for each triple (h, r, t) in the KG. (**b**) Subsequently, for each observed interaction (u, i) in the UI graph, KGCFRec similarly encodes the collaborative information of users and items employing GNN layers. (**c**) Following this, KGCFRec leverages an attention mechanism to integrate the user and item knowledge with their collaborative information. Ultimately, the framework computes the similarity score between the user and the item, facilitating the recommendation process.

GNNs progressively enrich the representation of a node by integrating information from its neighboring nodes. As a result, the node's representation after k iterations encapsulates the structural nuances and attributes of nodes within its k-hop vicinity. This approach mitigates the challenges posed by data sparsity. The pivotal aspect of GNN-based recommender systems lies in the design of the neighborhood aggregation strategy. Specifically, the representation vector of an entity is refined by iteratively aggregating and transforming the representations of its multi-hop neighbors.

However, it is challenging to accurately model user and item representations by directly fusing graph and collaborative features in a coarse-grained manner without considering the distributional gap between them. We leverage GNN layers to extract user and item representation from both the knowledge graph and collaborative information channels. Additionally, we introduce a novel attention mechanism to adaptively refine these representations, thereby enhancing the performance of recommendation tasks.

4.1. Knowledge-Aware Representation Learning

Knowledge-aware representation learning is depicted in Figure 2. We define $G_{KG} = \{(h, r, t) \mid h, t \in \epsilon, r \in R\}$, where r denotes the relation from entity h to entity t. Given that entities are connected through one or more varying relations, we contend that relation modeling is as crucial as entity modeling. GNNs excel at modeling entities and relationships, prompting us to adopt a GNN-based approach for the representation learning of entities and relations. An entity can be characterized by different relations across multiple KG triplets, illustrating the content similarity among items from various perspectives. For instance, the movie "Forrest Gump" can be characterized by its director, Robert Zemeckis, or its star, Tom Hanks. The process of information aggregation within the knowledge graph, as facilitated by our model, is defined in Equation (2):

$$e_i^{KG(k+1)} = f_{\mathrm{KG}}\left(\left\{\left(e_i^{KG(k)}, e_r, e_v^{KG(k)}\right) \mid (r, v) \in \mathcal{N}_i^{KG}\right\}\right) \tag{2}$$

where $f_{\mathrm{KG}}(.)$ is an aggregation function that aggregates the features of neighbors of item i, $\mathcal{N}_i^{KG}$ is a set of the neighbor of item i in the knowledge graph, which consists of some tuples, including relation and entity. Here, we implement $f_{\mathrm{KG}}(\cdot)$ as Equation (3):

$$e_i^{KG(k+1)} = \frac{1}{\|\mathcal{N}_i^{KG}\|} \sum_{(r,v)\in\mathcal{N}_i^{KG}} e_r \odot e_v^{KG(k)} \tag{3}$$

where $e_v^{KG(k)}$ is the embedding including knowledge information of entity v of k layer, e_r is the embedding of relation r, $\odot$ is the element-wise product. The reason of design relational message $e_r \odot e_v^{KG(k)}$ is to obtain new $e_i^{KG(k+1)}$, including different meanings of e_r and $e_v^{KG(k)}$.

Additionally, modeling the user's knowledge graph embedding is treated consistently, which is defined as Equation (4):

$$\mathbf{e}_u^{KG(k+1)} = \frac{1}{\|\mathcal{N}_u^{KG}\|} \sum_{i\in\mathcal{N}_u^{KG}} \mathbf{e}_i^{KG(k)} \tag{4}$$

where $\mathcal{N}_u^{KG}$ is a set of the neighbor of user u in the knowledge graph.

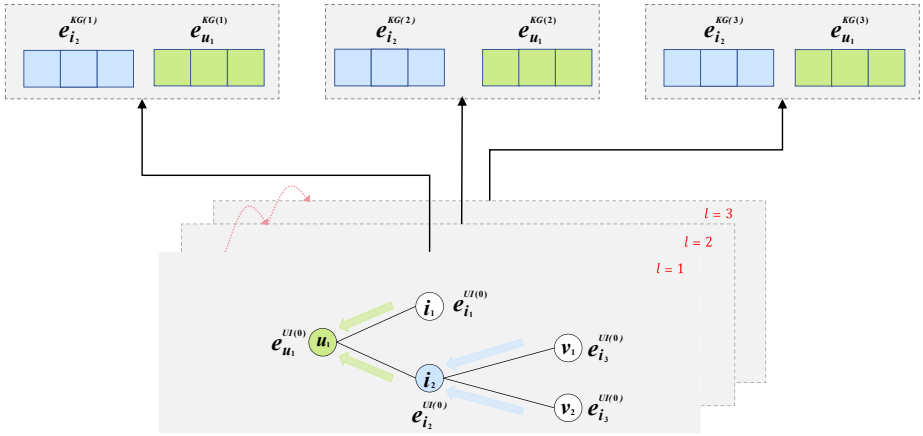

Figure 2. Knowledge-aware representation learning. To capture the knowledge embedded within the graph, we employ the function f_{KG} to aggregate information from the k-hop neighbors of both users and items, resulting in distinct embeddings across K layers.

4.2. Collaborative-Aware Representation Learning

Collaborative-aware representation learning is depicted in Figure 3. The collaborative information [17] holds significant value in characterizing user preferences, assuming that users with similar behaviors tend to share similar preferences for items. Consequently, we refine this collaborative information from the user–item interactions (UI) graph. In Equations (5) and (6), we leverage a GNN with K layers to harmonize the information within the UI graph. Specifically, for the k-th layer, we update the user representation $e_u^{UI(k)}$ and the item representation $e_i^{UI(k)}$ as follows:

$$e_u^{UI(k+1)} = f_{\text{UI}}\left(e_i^{UI(k)}, \left\{e_i^{UI(k)} : i \in \mathcal{N}_u^{UI}\right\}\right) \tag{5}$$

$$e_i^{UI(k+1)} = f_{\text{UI}}\left(e_u^{UI(k)}, \left\{e_u^{UI(k)} : u \in \mathcal{N}_i^{UI}\right\}\right) \tag{6}$$

where f_{UI} serves as an aggregation mechanism that compiles the features of neighboring nodes within the user–item interactions graph. $e_u^{UI(k)}$ and $e_i^{UI(k)}$ represent the k-th layer's representation, which incorporates the collaborative information of user u and item i, respectively. $\mathcal{N}_u^{UI}$ denotes the set of item neighbors for user u, while $\mathcal{N}_i^{UI}$ signifies the set of user neighbors for item i.

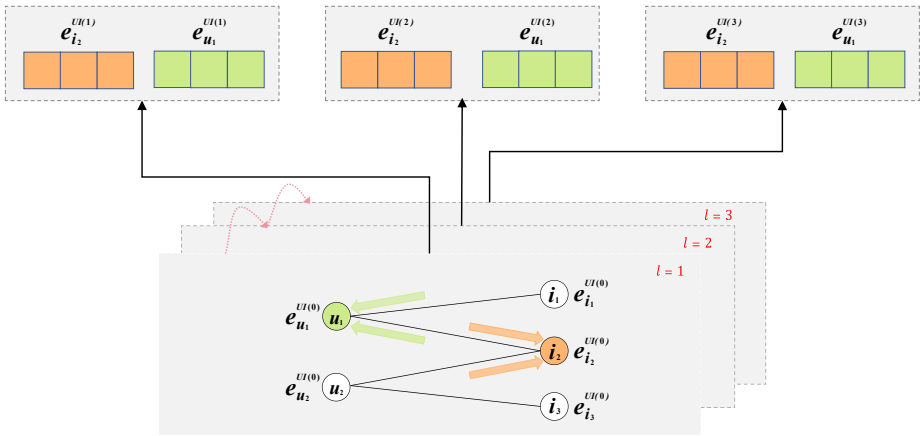

Figure 3. Collaborative-aware representation learning. We employ the aggregation function f_{UI} to consolidate the collaborative information from the k-hop neighbors of both users and items within the user–item interactions graph, yielding distinct embeddings across K layers.

The essence of graph convolutional networks (GCNs) lies in the design of the f_{UI} function. In contrast to the prevalent approaches that integrate feature transformation or nonlinear activation within the f_{UI} function, we streamline the GCN design to enhance its conciseness and suitability for recommendation tasks, like LightGCN [34]. The graph convolution operation is delineated in Equations (7) and (8):

$$\mathrm{e}_u^{UI(k+1)} = \frac{1}{\|\mathcal{N}_u^{UI}\|} \sum_{i \in \mathcal{N}_u^{UI}} \mathrm{e}_i^{UI(k)} \tag{7}$$

$$e_i^{UI(k+1)} = \frac{1}{\|\mathcal{N}_i^{UI}\|} \sum_{u \in \mathcal{N}_i^{UI}} e_u^{UI(k)} \tag{8}$$

It is important to highlight that our aggregation process exclusively considers the neighboring nodes that are directly connected and does not incorporate the target node itself. This is because we aim to generate a comprehensive embedding representation by combining the representations from each layer. Following the application of the f_{UI} function, the user and item representations, which encapsulate collaborative information, can be derived.

4.3. Adaptive Feature Fusion

It is crucial to harmoniously integrate the representations that encapsulate both knowledge and collaborative information to derive a unified final representation for users and items. In scenarios where user–item interactions are plentiful, the model can benefit from learning a wealth of collaborative insights. Conversely, in cases where interactions are scarce, the model should prioritize capturing the knowledge-rich information within the graph.

To fuse the knowledge graph and collaborative information, we define a fusion function f, shown in Equations (9) and (10).

$$e_i^{(k)} = f_i\left(e_i^{KG(k)}, e_i^{UI(k)}\right) \tag{9}$$

$$e_u^{(k)} = f_u\left(e_u^{KG(k)}, e_u^{UI(k)}\right) \tag{10}$$

Specifically, in this paper, we implement the fusion function f as Equations (11) and (12):

$$e_i^{(k)} = Att\left(e_i^{KG(k)}, e_i^{(k-1)}\right) e_i^{KG(k)} + Att\left(e_i^{UI(k)}, e_i^{(k-1)}\right) e_i^{UI(k)} \tag{11}$$

$$e_u^{(k)} = Att\left(e_u^{KG(k)}, e_u^{(k-1)}\right) e_u^{KG(k)} + Att\left(e_u^{UI(k)}, e_u^{(k-1)}\right) e_u^{UI(k)} \tag{12}$$

Hence, we introduce an attention score $Att(\cdot)$ to distinguish the importance of knowledge and collaborative information in Equations (13)–(16):

$$Att\left(e_i^{KG(k)}, e_i^{(k-1)}\right) = \frac{\exp\left(e_i^{(k-1)} \cdot e_i^{KG(k)}\right)}{e_i^{(k)}} \tag{13}$$

$$Att\left(e_i^{UI(k)}, e_i^{(k-1)}\right) = \frac{\exp\left(e_i^{(k-1)} \cdot e_i^{UI(k)}\right)}{e_i^{(k)}} \tag{14}$$

$$Att\left(e_u^{KG(k)}, e_u^{(k-1)}\right) = \frac{\exp\left(e_u^{(k-1)} \cdot e_u^{KG(k)}\right)}{e_u^{(k)}} \tag{15}$$

$$Att\left(e_u^{UI(k)}, e_u^{(k-1)}\right) = \frac{\exp\left(e_u^{(k-1)} \cdot e_u^{UI(k)}\right)}{e_u^{(k)}} \tag{16}$$

Specifically, we define $e_i^{(k)}$ and $e_u^{(k)}$ in Equations (17) and (18):

$$e_i^{(k)} = \exp\left(e_i^{(k-1)} \cdot e_i^{KG(k)}\right) + \exp\left(e_i^{(k-1)} \cdot e_i^{UI(k)}\right) \tag{17}$$

$$e_u^{(k)} = \exp\left(e_u^{(k-1)} \cdot e_u^{KG(k)}\right) + \exp\left(e_u^{(k-1)} \cdot e_u^{UI(k)}\right) \tag{18}$$

Upon completing L layers of representation learning, we acquire the user u and item i representations across layers 0 to L. These representations are then aggregated to construct the ultimate user and item embeddings. Specifically, we aggregate the user (item) embeddings from each layer to derive the comprehensive final representation. This aggregation is achieved by summing the embeddings from all layers:

$$e_i^* = \frac{1}{L+1} \sum_{j=0}^{L} e_i^{(j)} \tag{19}$$

$$e_u^* = \frac{1}{L+1} \sum_{j=0}^{L} e_u^{(j)} \tag{20}$$

The ultimate representations encapsulate various semantics from each layer, as they are derived through the layer combination defined by Equations (19) and (20).

4.4. Model Optimization

Following the integration of the layers, the model's predictive function is formulated as the dot product between the user's and item's final representations, as outlined in Equation (21):

$$\hat{y}_{ui} = e_u^{*T} e_i^* \tag{21}$$

$\hat{y}_{ui}$ is employed to estimate the probability of a user adopting an item, which serves as the ranking metric for generating recommendations.

To optimize the model, we utilize the Bayesian personalized ranking (BPR) loss [35], a pairwise loss function that encourages the model to rank observed entries higher than unobserved ones, as depicted in Equation (22):

$$\mathcal{L}_{\text{BPR}} = \sum_{(u,i,j) \in O} -\ln \sigma\left(\hat{y}_{ui} - \hat{y}_{uj}\right) \tag{22}$$

where $O = \{(u,i,j) \mid (u,i) \in O^+, (u,j) \in O^-\}$ represents the training dataset composed of observed interactions O^+ and unobserved counterparts O^-; $\sigma(\cdot)$ denotes the sigmoid function. Prior to the commencement of training, the items with which the user has interacted are chosen to form a positive sample with the user. For negative samples, we randomly select items that the user has not interacted with, maintaining a ratio of 1:1 to the positive samples.

To mitigate overfitting, we minimize the objective function outlined in Equation (23) to train the model's parameters effectively.

$$\mathcal{L}_{\text{KGCFRec}} = \mathcal{L}_{\text{BPR}} + \lambda_\Theta \|\Theta\|^2 \tag{23}$$

where $\lambda_\Theta \|\Theta\|^2$ is the regularization term. Upon the computation of the loss, the parameters of the KGCFRec model are updated via the gradient descent algorithm Adam [36]. Training ceases after a predefined number of iterations or when the loss does not decrease for a specified number of iterations.

Once training is complete, the process of item recommendation for a user unfolds as follows: the user's embedding vector and the embedding vectors of all items with which the user has not interacted are input into the prediction function $\hat{y}_{ui}$ in pairs. The preference scores for all uninteracted items are then determined, and the top N items with the highest scores are selected for recommendation to the user after ranking.

5. Experimental Setup

In this section, we delineate the dataset utilized for our experiments, the baseline approach and the evaluation metrics employed for comparison, and provide an overview of the experimental setup.

5.1. Datasets

To validate the efficacy of the model presented in this paper, we employ three widely recognized recommendation datasets across distinct domains. These datasets include Last-FM, a music recommendation dataset; Amazon-book, a book recommendation dataset; and Alibaba-iFashion, a fashion clothing recommendation dataset. A comprehensive description of the fundamental aspects of these datasets is provided below.

Amazon-book is a subset of the Amazon-review dataset, a well-known product recommendation dataset published by Amazon.com. It pertains specifically to book recommendations, where the items are books and the user ratings represent the interaction actions. In this study, we treat high user ratings as positive samples, thereby transforming display feedback into implicit feedback. To ensure the quality of the dataset, we employ a preprocessing method to eliminate user and item data with fewer than 10 interactions. Amazon-book is openly available on Github at https://nijianmo.github.io/amazon/index.html, accessed on 15 May 2023.

Last-FM [37] is derived from the Last.fm online music system and represents music listening habits. Each music piece in the dataset is considered an item, and users' music listening patterns constitute the interactions. The data used in this study span the period from January 2015 to June 2015. Last-FM is openly available from the Last.fm website, http://www.lastfm.com, accessed on 18 May 2023.

Alibaba-iFashion is a fashion apparel recommendation dataset sourced from Alibaba's e-commerce platform. The various fashion outfits are treated as items, and user behaviors such as purchases and clicks are considered interactions. The knowledge graph for this dataset is constructed using the attributes of the fashion outfits. Alibaba-iFashion is openly available on Github at https://github.com/renesemela/lastfm-dataset-2020, accessed on 20 May 2023.

The three datasets used in this paper are open source and can be accessed by visiting their official links. A detailed statistical comparison of these three datasets is presented in Table 1. This comparison reveals that the datasets vary in terms of various data metrics, which will facilitate the assessment of the model's effectiveness proposed in this paper.

Table 1. Statistics of the datasets.

Dataset	#Users	#Items	#Interactions	#Entities	#Relations	#Triplets
Amazon-book	70,679	24,915	847,733	88,572	39	2,557,746
Last-FM	23,566	48,123	3,034,796	58,266	9	464,567
Alibaba-iFashion	114,737	30,040	1,781,093	59,156	51	279,155

In the context of each dataset, we randomly select 80% of the historical interaction data for each user to constitute the training set. The remaining 20% is designated as the test set. Within the training set, we further segregate 10% as the validation set, which aids in refining the model's hyper-parameters by evaluating their impact on performance.

5.2. Baseline Methods

This subsection outlines the baseline methods that are used for comparative evaluation against KGCFRec. In this study, these baselines are categorized into three distinct groups: solely utilizing user–item interaction data without incorporating knowledge graph information (MF); embedding-based methods (CKE); and graph-neural-network-based methods (KGAT, KGNN-LS, CKAN, R-GCN, Simple-HGN, KGCN). The specific details of these baseline methods are detailed below:

- **MF [13]** is a conventional matrix factorization approach that exclusively relies on user–item interaction data for recommendations. MF projects users and items into a latent vector space and refines their vector representations by decomposing the co-occurrence matrix, thereby capturing the collaborative signals between user–item interactions.
- **CKE [30]** is a notable example of an embedding-based knowledge graph recommendation. It learns the embedding vector representation of items within the knowledge graph through the knowledge graph embedding algorithm TransR and integrates these item embeddings into the matrix factorization framework.
- **KGAT [17]** is a state-of-the-art knowledge graph recommendation algorithm that leverages graph neural network technology. It implements a neighbor aggregation mechanism based on an attention mechanism, enabling the selective propagation of node information. KGAT operates on a collaborative knowledge graph that includes a user–item interaction graph and a knowledge graph, outperforming other methods in this setting.
- **KGNN-LS [29]** is another knowledge graph recommendation algorithm based on graph neural networks. It differs from KGAT in that it does not operate on a collaborative knowledge graph but rather uses the transformation of knowledge graph information into a user–item interaction graph. KGNN-LS takes into account the propagation of user preferences across the knowledge graph.
- **CKAN [18]** is an extension of the KGNN-LS model. It differs from KGNN-LS in that it propagates and aggregates information on the user–item interaction graph and the knowledge graph separately, and it designs two distinct aggregation mechanisms.
- **R-GCN [32]** is a graph neural network framework. It treats the relationships in the knowledge graph as distinct information dissemination channels and utilizes different channels for aggregating node information based on their relationships.
- **Simple-HGN [33]** is a novel, simple, and effective method for modeling heterogeneous graphs. The model is applicable to knowledge-aware recommendation and has shown promising results.
- **KGCN [31]** is a recommender algorithm that leverages knowledge reasoning optimization. It involves two key steps: knowledge graph completion, which uses a reasoning algorithm to enrich the graph with better user interaction and semantic information, and the application of KGCN to capture higher-order features for improved personalized recommendations through embedding and aggregation of the completed knowledge graph.

5.3. Evaluation Metrics

The model proposed in this study is assessed as a top-n recommendation model. During the evaluation phase, two prevalent evaluation metrics in contemporary recommendation systems are employed: sample evaluation metrics [29] and all-ranking evaluation metrics [38]. Sample evaluation entails presenting a user with a recommendation list and selecting a predefined number of negative samples for final evaluation, rather than predicting all potential negative samples. Conversely, the all-ranking evaluation approach involves predicting all unadopted items by a user and then providing a top-n list of recommendations for evaluation. While sample evaluation is more expedient, all-ranking evaluation offers a more thorough and precise assessment. In this paper, we opt for the all-ranking evaluation method.

To evaluate the efficacy of the model, two commonly employed evaluation metrics, recall [38] and normalized discounted cumulative gain (NDCG) [39], are employed. Recall is defined as shown in Equation (24).

$$\text{Recall} = \frac{\#TP}{\#TP + \#FN} \tag{24}$$

The recall metric is typically assessed by setting a fixed number of items, denoted as K, that each user is recommended. This is represented as Recall@K. Recall is a prevalent metric for assessing the performance of a ranking model, quantifying the proportion of correct recommendations to the total number of correct items available.

The discounted cumulative gain (DCG) metric is introduced to define the normalized discounted cumulative gain (NDCG), as depicted in Equation (25).

$$\text{DCG@K} = \sum_{i}^{K} \frac{2^{r(i)} - 1}{\log_2(i+1)} \tag{25}$$

where $r(i)$ represents the relevance score of the item at position r in the recommendation list. In this study, we define $r(i)$ as 1 if the recommended item has been interacted with by the user; otherwise, $r(i)$ is set to 0. Subsequently, the normalized discounted cumulative gain (NDCG) is formulated as in Equation (26).

$$\text{NDCG@ K} = \frac{DCG@K}{IDCG@K} \tag{26}$$

The ideal discounted cumulative gain (IDCG@K) represents the optimal discounted cumulative gain scenario, where the items in the recommendation list are those that the users have interacted with. NDCG places greater emphasis on the sequential ordering of correctly classified samples, ensuring that they are ranked higher than the recall metric.

5.4. Parameter Settings

We developed our KGCFRec model using PyTorch. Training was conducted on a Nvidia GeForce RTX 3060 with Adam as the optimizer, with a fixed batch size of 1024. To ensure a level playing field with the baseline methods, we standardized the embedding dimension (d) at 64. The model parameters were initialized using Xavier [40]. We use the default model configuration for all baseline methods, keeping the training strategy consistent. At the same time, the grid search traversed the hyper parameters as the parameter of the comparison method to ensure a fair comparison. A grid search was conducted to optimize hyper-parameters: the learning rate (σ) was varied within $\{5 \times 10^{-2}, 5 \times 10^{-3}, 5 \times 10^{-4}\}$, and the L_2 normalization coefficient λ_Θ was explored in $\{10^{-3}, 10^{-4}, 10^{-5}\}$. To enhance the model's generalization capabilities, we employed dropout techniques, specifically node dropout and edge dropout, with their respective ratios adjusted in $\{0.0, 0.1, 0.3, 0.5, 0.7\}$. Each dataset underwent 600 epochs of training, and the parameters yielding the best performance are tabulated in Table 2. The impact of the depth of the graph neural network layers is discussed in Section 6.

Table 2. The parameters of best result.

Dataset	d	δ	λ_1	Node Dropout Rate	Edge Dropout Rate	#Triplets
Amazon-book	64	5×10^{-4}	10^{-4}	0.5	0.1	2,557,746
Last-FM	64	5×10^{-4}	10^{-4}	0.5	0.1	464,567
Alibaba-iFashion	64	5×10^{-4}	10^{-4}	0.3	0.3	279,155

6. Results

In this section, we present the results of our experiments. Initially, we compare the performance of our model against that of the baseline methods. Next, we conduct an

ablation study to illustrate the efficacy of our model. Finally, we conduct hyper-parameter studies to reveal the sensitivity of each module of the model.

6.1. Recommendation Performance

We report the results of the baseline methods and our model in Table 3.

Table 3. Overall performance comparison.

Model	Amazon-Book		Last-FM		Alibaba-iFashion	
	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
MF [13]	0.13	0.0678	0.0724	0.0617	0.1095	0.067
CKE [30]	0.1342	0.0698	0.0732	0.063	0.1103	0.0676
KGAT [17]	0.1487	0.0799	0.0873	0.0744	0.103	0.0627
KGNN-LS [29]	0.1362	0.056	0.088	0.0642	0.1039	0.0557
R-GCN [32]	0.122	0.0646	0.0743	0.0631	0.086	0.0515
CKAN [18]	0.1442	0.0698	0.0812	0.066	0.097	0.0509
Simple-HGN [33]	0.1587	0.0854	0.0917	0.0797	0.0952	0.0566
KGCN [31]	0.1592	0.0864	0.0919	0.0807	0.0971	0.0574
KGCFRec	**0.1646**	**0.0880**	**0.0933**	**0.0826**	**0.1231**	**0.0769**
(%) Imp.	3.39	1.85	1.52	2.35	11.6	13.76

In Table 3, the symbol Imp. stands for the relative improvement of the top-performing method (bolded) compared to the most robust baseline methods (underlined). Table 3 reveals the following key insights:

- KGCFRec surpassed the baseline model in all evaluation metrics across all datasets. When contrasted with the traditional collaborative filtering model MF, which relies solely on collaborative data without leveraging knowledge graph information, KGCFRec's superior performance underscores the significance of integrating knowledge graph data. In comparison with graph neural network models like KGAT and KGNN-LS, which are more focused on encoding knowledge graph information, KGCFRec demonstrates its strength in incorporating collaborative data. Against the CKAN model, which also models both knowledge graph and collaborative data, KGCFRec achieves superior results due to its attention-based information fusion model.
- The experimental findings reveal that MF, which solely relies on user–item interaction data without knowledge graph information, outperforms models like KGAT, KGNN-LS, R-GCN, and CKAN on the Alibaba-iFashion dataset in terms of evaluation metrics. This observation suggests that models that predominantly rely on knowledge graph information without integrating collaborative data may not achieve the expected performance. In contrast, KGCFRec consistently outperforms MF, underscoring the importance of incorporating both collaborative and knowledge information.
- Analyzing the relative gains of KGCFRec over the best baseline across the Amazon-book, Alibaba-iFashion, and Last-FM datasets, the largest gains were observed on the Amazon-book dataset, with 3.39% (Recall@20) and 1.85% (NDCG@20). On the Last-FM dataset, the gains were 1.52% (Recall@20) and 2.35% (NDCG@20). On the Alibaba-iFashion dataset, the gains were 11.6% (Recall@20) and 13.76% (NDCG@20). These differences could be attributed to the varying quality and distribution of the datasets, which may explain the varying gains KGCFRec achieves in fusing collaborative and knowledge information.

6.2. Ablation Study

To further corroborate the efficacy of the attention-based fusion design for fusing knowledge and collaborative information, as presented in Section 4 of this paper, this subsection performs ablation studies on the proposed design. Specifically, the performance of the two models is contrasted across three datasets: Alibaba-iFashion, Last-FM, and

Amazon-book. This comparison is achieved by substituting the attention-mechanism-based fusion with a straightforward summation-based fusion approach. In this paper, the altered KGCFRec model is referred to as the KGCFRec-a model. The experimental parameters of the KGCFRec-a model are tuned identically to those of the KGCFRec model, and the experimental outcomes are presented in Table 4.

Table 4. Impact of the attention mechanism

Model	Amazon-Book		Last-FM		Alibaba-iFashion	
	Recall@20	NDCG@20	Recall@20	NDCG@20	Recall@20	NDCG@20
KGCFRec	0.1646	0.0880	0.0933	0.0826	0.1231	0.0769
KGCFRec-a	0.1583	0.0859	0.0887	0.0786	0.1188	0.074

6.3. Hyper-Parameter Studies

In this subsection, we conduct an empirical investigation into the impact of three hyper-parameters of KGCFRec: the embedding dimension, the number of graph convolutional layers, and the dropout ratio.

Given that KGCFRec represents users and items as embedding vectors, the size of these embeddings significantly influences the model's expressive capabilities. To examine this influence, we varied the embedding size of KGCFRec across the Amazon-book dataset, setting it to 8, 16, 32, and 64, respectively, while the other parameters were determined through grid search. The final experimental outcomes for each embedding size are illustrated in Figure 4.

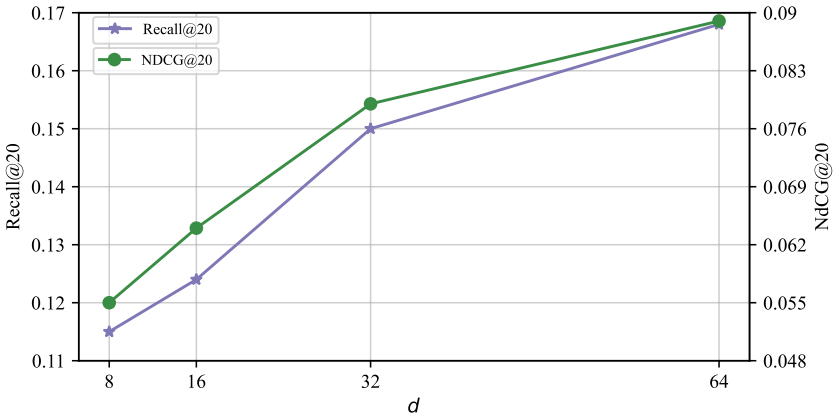

Figure 4. Performance of different embedding sizes (d) on Amazon-book.

From Figure 4, it is evident that as the embedding size increases, the recommendation performance of KGCFRec improves, albeit with diminishing returns. This suggests that a judicious increase in embedding size can enhance the model's expressiveness. However, an excessive increase in size may also complicate the model's convergence, underscoring the importance of selecting an appropriate embedding size based on the specific characteristics of the data in practical applications.

As KGCFRec is a graph-neural-network-based knowledge graph recommendation algorithm, the graph convolutional layers play a pivotal role in aggregating information. Consequently, the number of these layers is a crucial hyper-parameter that influences the recommendation efficacy. In the experiments conducted in this section, we varied the number of graph convolutional layers in KGCFRec to 1, 2, and 3, respectively, and identified the optimal combination of other parameters through grid search. The experimental results on the Amazon-book and Alibaba-iFashion datasets are presented in Figures 5 and 6.

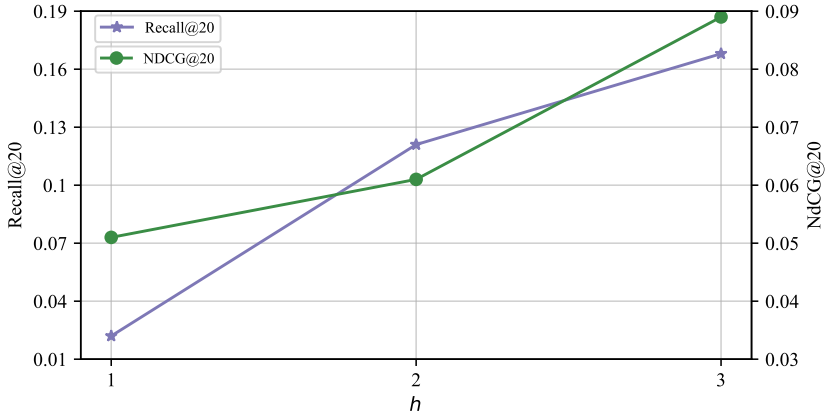

Figure 5. Performance of different graph convolution layers (h) on Amazon-book.

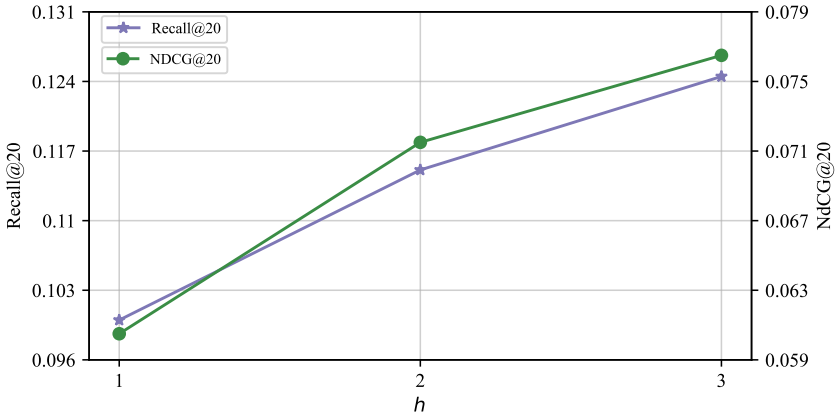

Figure 6. Performance of different graph convolution layers (h) on Alibaba-iFashion.

Figures 5 and 6 indicate that both the Amazon-book and Alibaba-iFashion datasets achieve optimal performance with three graph convolutional layers in terms of Recall@20 and NDCG@20. The general trend suggests that the recommendation performance improves with an increase in the number of convolutional layers, underscoring the soundness of the graph neural network design at the core of KGCFRec. By employing multiple layers of graph convolution, KGCFRec can effectively gather information from more distant nodes, facilitating the learning of a more comprehensive embedding representation for users and items.

Dropout [41] is a technique used to mitigate the risk of overfitting in neural network models. This subsection explores the impact of varying dropout rates on the model's performance using the Last-FM dataset. The results are presented in Figure 7.

Figure 7 suggests that the dropout ratio has a significant influence on the model's performance. For node dropout, increasing the dropout ratio leads to a notable improvement in model performance. Conversely, for edge dropout, increasing the dropout ratio has a detrimental effect on the model's performance, possibly due to the disruption of connections between nodes, which can hinder the model's convergence. These findings indicate that the judicious application of dropout can effectively reduce the training complexity of the model, aiding in the prevention of overfitting and enhancing the model's generalization capabilities.

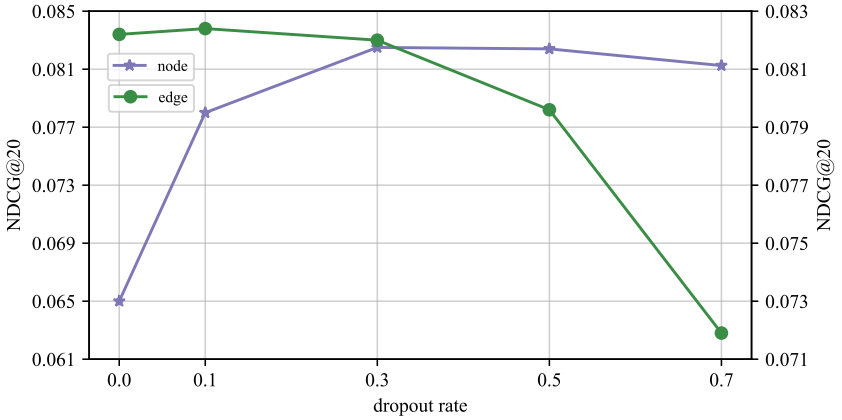

Figure 7. Performance of different dropout rates.

7. Conclusions

In this paper, we have introduced a method for personalized recommendation that fuses collaborative and knowledge information from a knowledge graph. To achieve this, we devised two graph neural networks to collect collaborative information from the user–item interactions graph and knowledge information from the knowledge graph. This design enables the differentiation of the two types of information, thereby facilitating the creation of more robust user and item representations. We then employed an attention mechanism to merge the collaborative and knowledge information embeddings, resulting in a more judicious weighted combination. Extensive experiments conducted on three real-world datasets validate the rationality and efficacy of KGCFRec. The experiment proves that knowledge graph and collaborative information can enhance each other to improve recommendation performance, and further confirms that it is feasible to use side information to improve recommendation performance. Ablation and hyper-parameter studies further corroborate the soundness of the model's design. In future work, we intend to investigate the further leveraging of the background knowledge of large language models to improve KGCFRec performance while implementing interpretable recommendations.

Author Contributions: Study design and writing, J.P. and J.G.; literature search, C.Z. and Q.Z.; figures, X.F. and K.Y.; supervision, J.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by resources provided by Hebei Natural Science Foundation of China Grant F2022203072, CCF-Zhipu AI Large Model Fund (CCF-Zhipu202307), CIPSC-SMP-Zhipu, AI Large Model Cross-Disciplinary Fund and Innovation Capability Improvement Plan Project of Hebei Province (22567626H).

Data Availability Statement: Datasets used in this paper are open source and publicly available. Amazon-book are openly available in Github at https://nijianmo.github.io/amazon/index.html, accessed on 15 May 2023. Last-FM are openly available in Last.fm website, http://www.lastfm.com, accessed on 18 May 2023. Alibaba-iFashion are openly available in Github at https://github.com/renesemela/lastfm-dataset-2020, accessed on 20 May 2023. The three datasets used in this paper are open source and can be accessed by visiting their official links.

Acknowledgments: The authors are thankful to the anonymous reviewers and editors for their valuable comments and suggestions.

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

References

1. Zhao, Y.; Li, C.; Peng, J.; Fang, X.; Huang, F.; Wang, S.; Xie, X.; Gong, J. Beyond the overlapping users: Cross-domain recommendation via adaptive anchor link learning. In Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, Taipei Taiwan, 23–27 July 2023; pp. 1488–1497.
2. Gong, J.; Wang, S.; Wang, J.; Feng, W.; Peng, H.; Tang, J.; Yu, P.S. Attentional graph convolutional networks for knowledge concept recommendation in moocs in a heterogeneous view. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Xi'an, China, 25–30 July 2020; pp. 79–88.
3. Covington, P.; Adams, J.; Sargin, E. Deep neural networks for youtube recommendations. In Proceedings of the 10th ACM Conference on Recommender System, New York, NY, USA, 15–19 September 2016; pp. 191–198.
4. Peng, J.; Li, C.; Zhao, Y.; Lin, Y.; Fang, X.; Gong, J. Improving Vision Transformers with Nested Multi-head Attentions. In Proceedings of the 2023 IEEE International Conference on Multimedia and Expo (ICME), Brisbane, Australia, 10–14 July 2023; pp. 1925–1930.
5. He, X.; Pan, J.; Jin, O.; Xu, T.; Liu, B.; Xu, T.; Shi, Y.; Atallah, A.; Herbrich, R.; Bowers, S.; et al. Practical lessons from predicting clicks on ads at facebook. In Proceedings of the Eighth International Workshop on Data Mining for Online Advertising, New York, NY, USA, 24–27 August 2014; pp. 1–9.
6. Tang, H.; Liu, J.; Zhao, M.; Gong, X. Progressive layered extraction (ple): A novel multi-task learning (mtl) model for personalized recommendations. In Proceedings of the Fourteenth ACM Conference on Recommender Systems, Virtual Event, Brazil, 22–26 September 2020; pp. 269–278.
7. Wang, J.; Huang, P.; Zhao, H.; Zhang, Z.; Zhao, B.; Lee, D.L. Billion-scale commodity embedding for e-commerce recommendation in alibaba. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 839–848.
8. Liu, H.; Lu, J.; Yang, H.; Zhao, X.; Xu, S.; Peng, H.; Zhang, Z.; Niu, W.; Zhu, X.; Bao, Y.; et al. Category-Specific CNN for Visual-aware CTR Prediction at JD. com. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Virtual Event, CA, USA, 6–10 July 2020; pp. 2686–2696.
9. Grbovic, M.; Cheng, H. Real-time personalization using embeddings for search ranking at airbnb. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 311–320.
10. 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]
11. 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]
12. Gong, J.; Fang, X.; Peng, J.; Zhao, Y.; Zhao, J.; Wang, C.; Li, Y.; Zhang, J.; Drew, S. MORE: Toward Improving Author Name Disambiguation in Academic Knowledge Graphs. *Int. J. Mach. Learn. Cybern.* **2024**, *15*, 37–50. [CrossRef]
13. Koren, Y.; Bell, R.; Volinsky, C. Matrix factorization techniques for recommender systems. *Computer* **2009**, *42*, 30–37. [CrossRef]
14. 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 World Wide Web Conference, San Francisco, CA, USA, 13–17 May 2019; pp. 151–161.
15. 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.
16. Hu, B.; Shi, C.; Zhao, W.X.; Yu, P.S. Leveraging meta-path based context for top-n recommendation with a neural co-attention model. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 1531–1540.
17. Wang, X.; He, X.; Cao, Y.; Liu, M.; Chua, T.S. Kgat: Knowledge graph attention network for recommendation. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 4–8 August 2019; pp. 950–958.
18. 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.
19. Ferrara, A.; Anelli, V.W.; Mancino, A.C.M.; Di Noia, T.; Di Sciascio, E. Kgflex: Efficient recommendation with sparse feature factorization and knowledge graphs. *ACM Trans. Recomm. Syst.* **2023**, *1*, 1–30. [CrossRef]
20. Bordes, A.; Usunier, N.; Garcia-Duran, A.; Weston, J.; Yakhnenko, O. Translating embeddings for modeling multi-relational data. *Adv. Neural Inf. Process. Syst.* **2013**, *26*.
21. 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; Volume 28.
22. Du, Y.; Zhu, X.; Chen, L.; Fang, Z.; Gao, Y. Metakg: Meta-learning on knowledge graph for cold-start recommendation. *IEEE Trans. Knowl. Data Eng.* **2022**. [CrossRef]
23. Pan, H.; Yang, X. Intelligent recommendation method integrating knowledge graph and Bayesian network. *Soft Comput.* **2023**, *27*, 483–492. [CrossRef]

24. Zhao, N.; Long, Z.; Wang, J.; Zhao, Z.D. AGRE: A knowledge graph recommendation algorithm based on multiple paths embeddings RNN encoder. *Knowl.-Based Syst.* **2023**, *259*, 110078. [CrossRef]
25. Yang, Y.; Huang, C.; Xia, L.; Li, C. Knowledge graph contrastive learning for recommendation. In Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, 11–15 July 2022; pp. 1434–1443.
26. Li, D.; Qu, H.; Wang, J. A survey on knowledge graph-based recommender systems. In Proceedings of the 2023 China Automation Congress (CAC), Chongqing, China, 17–19 November 2023; pp. 2925–2930.
27. Wang, F.; Zheng, Z.; Zhang, Y.; Li, Y.; Yang, K.; Zhu, C. To see further: Knowledge graph-aware deep graph convolutional network for recommender systems. *Inf. Sci.* **2023**, *647*, 119465. [CrossRef]
28. Wang, H.; Zhao, M.; Xie, X.; Li, W.; Guo, M. Knowledge graph convolutional networks for recommender systems. In Proceedings of the World Wide Web Conference, San Francisco, CA, USA, 13–17 May 2019; pp. 3307–3313.
29. 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.
30. 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.
31. Liu, T.; Cheng, S. A Recommender Algorithm Based on Knowledge Graph Convolutional Network and Knowledge Reasoning Optimization. In Proceedings of the 2023 26th International Conference on Computer Supported Cooperative Work in Design (CSCWD), Rio de Janeiro, Brazil, 24–26 May 2023; pp. 1287–1292.
32. Schlichtkrull, M.; Kipf, T.N.; Bloem, P.; Berg, R.v.d.; Titov, I.; Welling, M. Modeling relational data with graph convolutional networks. In Proceedings of the European Semantic Web Conference, Heraklion, Crete, Greece, 3–7 June 2018; pp. 593–607.
33. Lv, Q.; Ding, M.; Liu, Q.; Chen, Y.; Feng, W.; He, S.; Zhou, C.; Jiang, J.; Dong, Y.; Tang, J. Are we really making much progress? Revisiting, benchmarking and refining heterogeneous graph neural networks. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, Virtual Event, Singapore, 14–18 August 2021; pp. 1150–1160.
34. He, X.; Deng, K.; Wang, X.; Li, Y.; Zhang, Y.; Wang, M. Lightgcn: Simplifying and powering graph convolution network for recommendation. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, China, 25–30 July 2020; pp. 639–648.
35. Rendle, S.; Freudenthaler, C.; Gantner, Z.; Schmidt-Thieme, L. BPR: Bayesian personalized ranking from implicit feedback. *arXiv* **2012**, arXiv:1205.2618.
36. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
37. Cantador, I.; Brusilovsky, P.; Kuflik, T. 2nd Workshop on Information Heterogeneity and Fusion in Recommender Systems (HetRec 2011). In Proceedings of the 5th ACM Conference on Recommender Systems, New York, NY, USA, 23–27 October 2011; RecSys 2011.
38. Krichene, W.; Rendle, S. On sampled metrics for item recommendation. *Commun. ACM* **2022**, *65*, 75–83. [CrossRef]
39. Järvelin, K.; Kekäläinen, J. Cumulated gain-based evaluation of IR techniques. *Acm Trans. Inf. Syst. (TOIS)* **2002**, *20*, 422–446. [CrossRef]
40. Glorot, X.; Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics. JMLR Workshop and Conference Proceedings, Sardinia, Italy, 13–15 May 2010; pp. 249–256.
41. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.