International Journal on Semantic Web and Information Systems
Volume 19 • Issue 1

A Lightweight Method of Knowledge Graph Convolution Network for Collaborative Filtering

Xin Zhang, School of Artificial Intelligence and Big data, Hefei University, China
Shaohua Kuang, School of Artificial Intelligence and Big data, Hefei University, China*
https://orcid.org/0009-0005-7523-4375

ABSTRACT

In recent years, knowledge-aware recommendation systems have gained popularity as a solution to address the challenges of data sparsity and cold start in collaborative filtering. However, traditional knowledge graph convolutional networks impose significant computational burdens during training, demanding substantial resources and increasing the cost of recommendations. To address this issue, this article proposes a lightweight knowledge graph convolutional network for collaborative filtering (LKGCF). LKGCF eliminates the feature transformation and nonlinear activation components, by focusing on essential elements such as neighborhood aggregation and layer combination. LKGCF captures the user's long-distance personalized interests on the knowledge graph by sampling from neighborhood information and constructing a weighted sum of item embeddings. Experimental results demonstrate that the proposed model is easy to train and implement due to its coherence and simplicity. Furthermore, notable improvements in recommendation performance are observed compared to strong baselines.

KEYWORDS

collaborative filtering, graph convolution network, knowledge graph

INTRODUCTION

The advancement of social technology, specifically the widespread integration of the mobile Internet into people's daily lives, has resulted in individuals being exposed to an extensive range of information daily. The overwhelming quantity of data has given rise to information overload, causing individuals to experience feelings of being overwhelmed. Recommendation systems have emerged to alleviate the issue of information overload. The main objective of recommendation systems is to assist individuals in navigating through the extensive data and identifying content that may be relevant or of personal interest.

Recommendation systems are widely used in diverse domains, including e-commerce, short videos, healthcare services, and education (George & Lal, 2021; Salloum & Tekli, 2021; Xiao et al., 2022). For example, a general approach in recommendation systems is ranking, where items are rated according to popularity, and highly popular items are recommended to users. However, this recommendation method may need more attention focused on user preferences and personalized

DOI: 10.4018/IJSWIS.327353 *Corresponding Author

This article published as an Open Access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0/) which permits unrestricted use, distribution, and production in any medium, provided the author of the original work and original publication source are properly credited.

needs. Collaborative filtering is a conventional method that leverages historical user-item interactions to generate personalized recommendations. Extensive literature has demonstrated the significant advantages of collaborative filtering in improving recommendation performance (He et al., 2017; Herlocker et al., 2004). However, collaborative filtering may encounter challenges, such as data sparsity and cold start, in certain recommendation scenarios (Wei & He, 2022).

Knowledge graph is a knowledge database representing the objective world in a graphical form. It is currently widely utilized in various applications (Ji et al., 2021), such as human-computer interaction and intelligent search. Higher-level structures and semantic information extracted from the given entities can effectively alleviate the data sparsity and cold-start issues encountered in traditional recommendation (Li et al., 2022). Several studies have demonstrated the substantial benefits of incorporating knowledge graphs into collaborative filtering (H. Wang et al., 2019; Zhang et al., 2016). Currently, the predominant approach involves constructing knowledge-aware recommendations using graph neural networks, of which knowledge graph convolutional networks (KGCN) (H. Wang et al., 2019) and knowledge graph attention networks (X. Wang et al., 2019a) are two common methods.

Despite the effectiveness of these models based on graph neural networks in enhancing recommendation performance, several challenges still need to be addressed. For instance, many of these models inherit the steps from traditional graph neural networks. Nonetheless, feature transformation and nonlinear activation are ineffective in knowledge-aware recommendation systems and may impede recommendation performance in collaborative filtering. Furthermore, this leads to a substantial increase in the complexity of the recommendation system, thereby complicating the training procedure.

Table 1 presents a summary of the main acronyms used in the paper. It provides a quick reference for readers to understand the abbreviations employed in this paper.

Background Example

NGCF is a conventional model based on GCN for the recommendation, which constructs a user-item graph by incorporating the interaction history between users and items. It leverages graph neural networks to uncover the personalized interests of users. NGCF employs GCN-based iterative aggregation to discover higher-order latent information about items or users. Figure 1(a) visually illustrates the iterative aggregation process for item embeddings in NGCF.

LightGCN builds upon NGCF with a lightweight framework and surpasses NGCF in terms of recommendation accuracy. Figure 1(b) visually represents the iterative aggregation process for item embeddings in LightGCN.

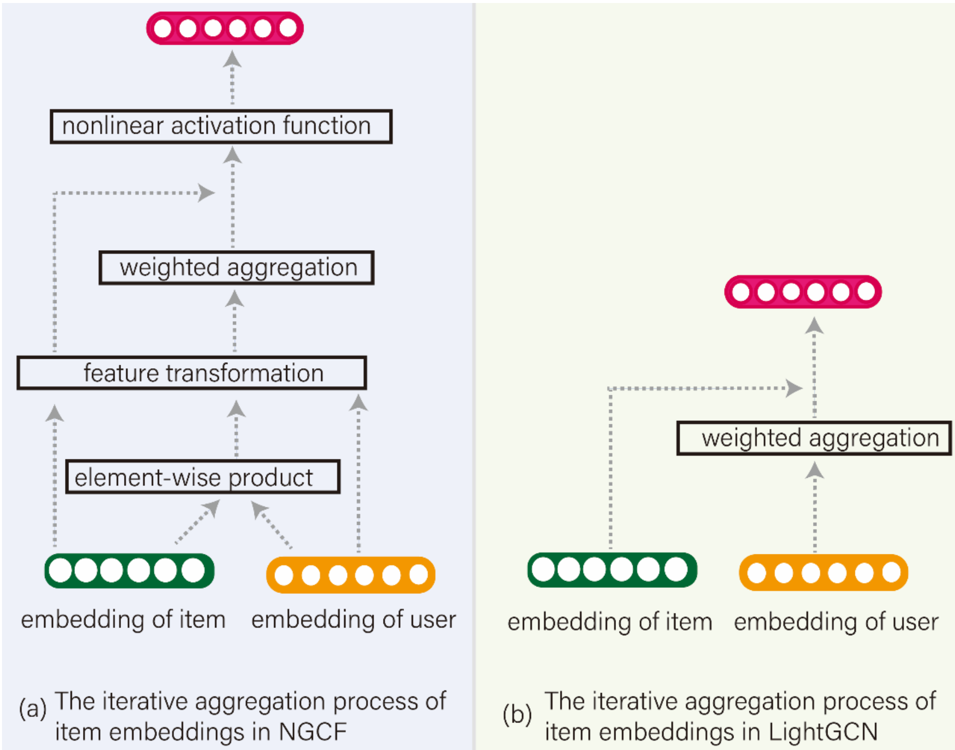
It can be observed that LightGCN removes the non-linear activation function and feature transformation matrix from NGCF. Inspired by this, it is believed the design of a lightweight model is crucial in scenarios where knowledge graphs and graph neural networks are combined for the recommendation. This belief stems from the fact that the node representations in knowledge graphs and user-item graphs exhibit a striking similarity, as both are represented by a single identity document (ID) to denote the nodes in the graph. Therefore, integrating knowledge graphs and graph neural networks in recommendation systems calls for a lightweight approach.

Table 1. List of major acronyms

Acronyms	Full Name
KGCN	Knowledge Graph Convolutional Network (H. Wang et al., 2019)
GCN	Graph convolutional neural network (Bruna et al., 2013)
LightGCN	Light Graph Convolution Network (He et al., 2020)
GCL	Graph contrastive learning
NGCF	Neural graph collaborative filtering (X. Wang et al., 2019b)

Figure 1. The iterative aggregation process of item embeddings in both NGCF and LKGCN models

Note: The red vector signifies the item vector post a single round of aggregation.



Main Contributions

In this paper, the data structure of the knowledge graph in collaborative filtering is analyzed. It is observed that one-hot ID represents each node (i.e., entity) in the knowledge graph without any concrete semantics, which differs from the data processed by traditional graph neural networks. Drawing inspiration from LightGCN, LKGCF, a collaborative filtering model based on knowledge graphs, is raised in this paper. LKGCF preserves the embedding layer and neighborhood aggregation solely and introduces an additional layer combination to mitigate over-smoothing. The key contributions of this paper are outlined as follows.

- A coherent and lightweight knowledge graph convolution network is proposed in this paper. Compared to KGCN, LKGCF significantly reduces complexity by consisting of only essential components, mitigating training difficulties. Additionally, the training efficiency and resources cost are improved by applying the raised LKGCF method.
- Ablation experiments are conducted to enhance the interpretability of the proposed method from a technical perspective. This demonstrates that feature transformation and nonlinear activation are ineffective in KGCN for collaborative filtering. Thus, it is feasible to design a concise framework for KGCN.
- The experiments conducted on public datasets demonstrate the significant superiority of the proposed method over strong baselines. In this paper, two public datasets about movies and music recommendations are adopted to validate the recommendation effectiveness of the proposed method. The two public datasets exhibit different characteristics both in quantities and densities,

which creates excellent challenges for the LKGCF approach. The experimental results show distinct advantages over strong baselines in various evaluation metrics, including area under curve (AUC), Recall@N, F1-score (F1), and RelaImpr.

RELATED WORKS

This section presents the related research in graph neural networks and knowledge graph-based recommendation systems.

Graph Convolutional Neural Network

Graph neural networks were initially introduced by leveraging recurrent neural networks to process diverse graph-structured data (Gori et al., 2005). Bruna et al. (2013) introduced the GCN, which utilizes frequency domain convolution and necessitates access to the entire graph for frequency domain computation. Kipf and Welling (2016) proposed a spatial domain-based convolutional network that utilizes the degree matrix to determine the distribution of node weights via convolution operations. To facilitate the distributed training of GCN on large-scale graph data, Hamilton et al. (2017) introduced a graph sampling neural network that employs a node-centric neighbor sampling strategy. X. Wang et al. (2019a) proposed graph attention networks, which integrated the attention mechanism with GCN to improve recommendation performance. The GCL method combined contrastive learning with GCN to address the issue of graph structural imbalance (Wang et al., 2022).

However, it is important to note that the data processed by these neural networks is typically high dimensional. In other words, the data processed by GCN often takes the form of an attribute graph, where each node carries abundant semantic information. This differs significantly from the data format used in knowledge graphs for collaborative filtering, where an item is identified by a single ID, and its additional information is commonly represented as triplets. Therefore, this situation should be carefully considered when trying to use GCN for the knowledge graph. Motivated by this observation, LKGCF is raised by simplifying KGCN by exclusively preserving the essential neighborhood aggregation algorithm while discarding feature transformation with bias term and nonlinear activation.

Knowledge-Aware Recommendation

Knowledge-aware recommendation systems are classified into three categories: embedding-based methods, path-based methods, and unified methods (Guo et al., 2020).

Generally, embedding-based methods improve user and item representations by employing knowledge graph embedding techniques. Considering the heterogeneity of nodes and relationships, Zhang et al. (2016) introduced collaborative knowledge based on embedding, where they learned the latent vector by extracting a structural representation for a specific item. H. Wang et al. (2018a) presented a deep knowledge-aware network for news recommendation, which employs attention to match candidate news and integrates the user's historical interests as user embeddings with varying weights. A unified graph-based recommendation model was proposed by Zhao et al. in 2021, which effectively leveraged the undirected co-occurrence and directed knowledge information in a graph to discover user preferences. However, knowledge graph embedding methods need to pay more attention to the abundant structural information of a graph by solely focusing on the nodes without leveraging the connections between entities.

Path-based methods apply knowledge graphs more naturally by discovering the connections between entities in knowledge graphs through paths.

X. Wang et al. (2019c) proposed an explainable reasoning mechanism over knowledge graphs for the recommendation, which utilized long short-term memory to encode paths in the graph and designed a weight pooling to distinguish the importance of different paths. The method of leveraging meta-path based on the context for the top-N recommendation was proposed by Hu et al. (2018), in which a new attention mechanism was raised to improve the representation of the context, users, and items based on different paths. Chen et al. (2021) modeled the dynamic user-item interactions over time to enhance recommendation performance and interpretability. These approaches tend to have good interpretability, but some expertise in designing meta-paths is required while building the models.

The methods mentioned above only utilize a single aspect of the knowledge graph. Unified methods extract semantic information about entities and relationships while utilizing connectivity information within the knowledge graph. RippleNet continuously and automatically discovers users' potential hierarchical interests by propagating preferences through the knowledge graph (H. Wang et al., 2018b). To capture users' long-distance interests, KGCN was introduced, which utilizes the relationships between items and users to construct the knowledge graph for performing graph convolution. However, many methods based on graph neural networks attempt to emulate the steps of traditional GCN, including feature transformation and nonlinear activation operations. Intuitively, feature transformation is essential for data with high-dimensional semantic information as it enables discovering hidden connections among different dimensions. Considering that each entity in the knowledge graph is represented by a single ID, the operations of feature transformation and nonlinear activation are redundant in knowledge graph recommendation systems. To reduce the training parameters and enhance the recommendation performance, a lightweight KGCN approach for collaborative filtering by discarding the redundant components of feature transformation and nonlinear activation is proposed.

EMPIRICAL EXPLORATIONS ON KGCN

To automatically capture the high-level structure and semantic information, the embedding of a given entity in KGCN is represented by calculating neighborhood information with bias in the neighborhood aggregation procedure, with the bias term added during the feature transformation operation. Formula 1 demonstrates the utilization of feature transformation and nonlinear activation after neighborhood aggregation in KGCN.

$$e_i^{h+1} = \sigma\left(we_i^h + b\right) \tag{1}$$

e_i^{h+1} is the feature representation of item *i* after h loops of aggregation; σ is nonlinear activation function; w is the weight matrix of feature transformation; *b* is the bias term for feature transformation.

Since each node of user-item interaction in collaborative filtering only processes a single attribute (i.e., ID), the operation of feature transformation with bias term and nonlinear activation becomes redundant, as it cannot yield rich features. In certain cases, it may even escalate the training complexity. Thus, ablation experiments are performed using rigorous experimental setups on two authentic datasets, namely MovieLens-20M and Last.FM. The following sections provide comprehensive descriptions of these datasets. This section conducts ablation experiments by eliminating specific components of KGCN, namely nonlinear activation and feature transformation with bias term. Three types of ablation experiments are designed as below.

- KGCN_no_ft. Discards the operation of feature transformation with bias term in KGCN.
- KGCN_no_na. Discards the operation of nonlinear activation in KGCN.
- KGCN_no_na&ft. Both operations feature transformation with bias term and nonlinear activation is discarded from KGCN.

Figures 2 and 3 present the experimental results of ablation. It is important to note that all experiments use the same parameter settings as KGCN, ensuring optimal performance. These figures clearly demonstrate that all three variants of KGCN (KGCN_no_na, KGCN_no_ft, and KGCN_no_na&ft) generally outperform KGCN on both datasets when evaluated using Precision@20 and Recall@20.

The recall of different types of KGCN on Last.FM and MovieLens-20M are plotted in Figure 3. The recommendation effectiveness on Last.FM by applying different KGCN models is illustrated in Figure 3(a). It can be observed that the recall of KGCN_no_ft shows notable advantages when compared with KGCN, which proves that the operation of feature transformation is not helpful in improving the recommendation effectiveness. Although the improvement by applying KGCN_no_na is not notable, a few enhancements can also be made when compared with KGCN. From these results, it is suggested that the operations of feature transformation and nonlinear activation are abundant. In Figure 3(b), KGCN_no_na also performs better than KGCN on the dataset MovieLens-20M. However, it can be observed that KGCN_no_ft behaves worse than KGCN with recall as the evaluation metric. It is believed that in the dataset MovieLens-20M, a large number of items are relevant to the recommendation topic, and then KGCN is more likely to obtain a slightly higher recall compared to KGCN_no_ft.

It can also be observed that on Last.FM, KGCN_no_na brings a few benefits when compared with KGCN, and the recommendation performance is not generally affected as much as KGCN_no_ ft. It is believed that the sparse data in Last.FM results in a more negative impact while applying feature transformation, so the performance is significantly better by removing feature transformation. Surprisingly, the highest recommendation performance on the two datasets is obtained by applying KGCN_no_na&ft, in which neither feature transformation with bias term nor nonlinear activation is included. It can be demonstrated that the two operations in KGCN are redundant, and the recommendation effectiveness can be improved when the two operations are removed from KGCN.

Figure 2. The Top_n recommendation performance of three types of KGCN on public datasets with precision@20 as the evaluation metric

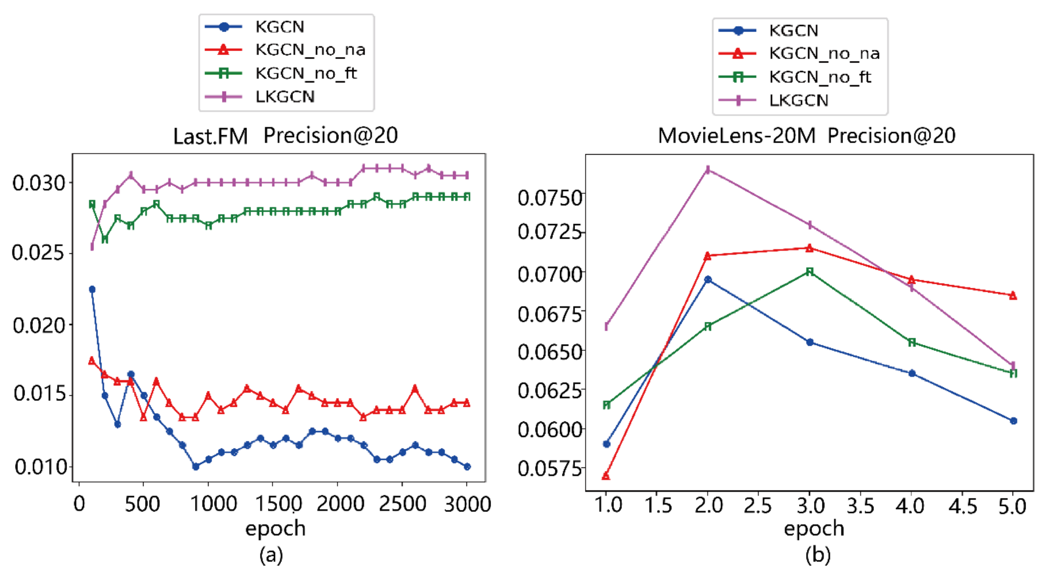
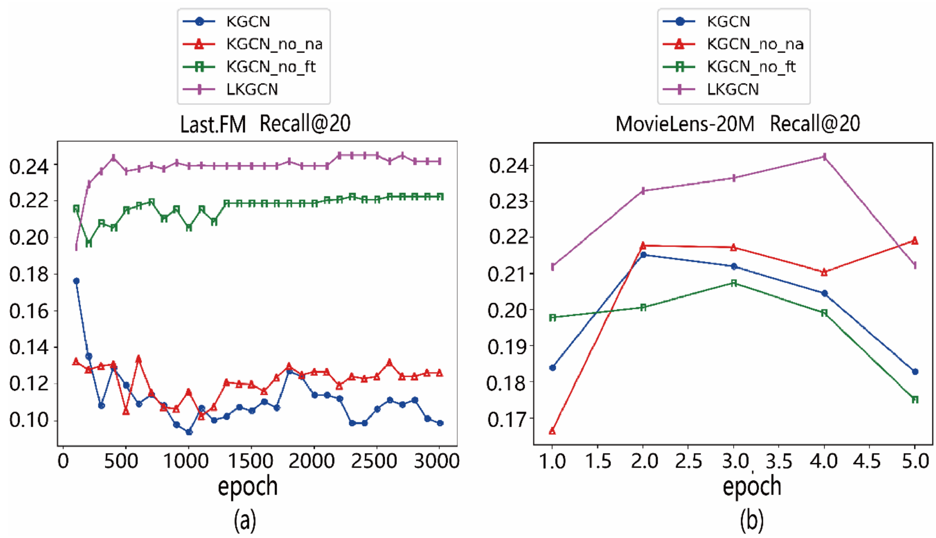


Figure 3. The Top_n recommendation performance of three types of KGCN on public datasets with recall@20 as the evaluation metric



METHODOLOGY

The contribution of feature transformation with bias term and nonlinear activation components to the recommendation performance in collaborative filtering, based on the knowledge graph, is limited. Therefore, a lightweight knowledge graph convolutional network, namely LKGCF, is proposed to reduce the complexity of KGCN and enhance recommendation effectiveness in collaborative filtering. In order to simplify the LKGCF approach and reduce the number of parameters, the proposed method eliminates the redundant components of feature transformation with bias and nonlinear activation functions. Additionally, the proposed LKGCF method incorporates an additional combination layer for item embeddings to enhance the representation of target items and prevent over smoothing. The proposed LKGCF approach is easier to train and interpret due to its simplified design. In this section, an explanation of the preliminaries is provided, followed by the presentation of the LKGCF framework. Model training and prediction are then introduced.

Preliminaries

This section begins with an explanation of the data structure in the user-item interaction matrix and knowledge graph, followed by an introduction to the task of collaborative filtering based on the knowledge graph. The main objective of this paper is to leverage the lightweight LKGCF approach to effectively recommend interesting items to users.

Definition 1 (User-Item Interaction): In a typical recommendation scenario, $U = \{ u_1, u_2, u_3, \ldots\ldots, u_n \}$ is defined a set of M users, and $V = \{ v_1, v_2, v_{3,} v_{4,\ldots\ldots,}, v_{n,} \}$ is a set of N items. Y records the historical interactions between the user and items; the user-item interaction matrix is represented as $Y \in R^{MXN}$, where each element is 0 or 1. And $Y_{uv} = 0$ indicates that the user u has no interactions with the item v, otherwise $Y_{uv} = 1$.

Definition 2 (Knowledge Graphs): Knowledge graphs are introduced as auxiliary information to improve the performance of collaborative filtering. Define *G = {(h, r, t) | h, t E, r R}* as the knowledge graph, where *h*, *r*, and *t* represent the head, relation, and tail of a triple in the knowledge graph, respectively. *E* and *R* indicate the set of entities and relations in knowledge graph *G*. Considering that the information of the real world contained in the knowledge graph is limited, $A = \{(v, e)|v \in V, e \in E\}$ indicates the entities in the knowledge graph. Moreover, each item v in the real world is represented by the entity v in the knowledge graph. In collaborative filtering based on the knowledge graph, the embedding of users and items are represented by using the historical interaction records between users and items in Y, along with many real information-related entities in the knowledge graph.

Definition 3 (Recommended Items): Given the matrix of the user-item interaction Y along with the knowledge graph G, the task of collaborative filtering is to predict whether the user u is interested in the item v. To achieve the goal, LKGCF attempts to learn a function $\hat{y}_{uv} = F(u, v \mid \Theta, Y, G)$ to predict the probability that the user interacts with the given item v, where Θ denotes the model parameters of function F.

Table 2 presents the key symbols used in this paper.

The Proposed Framework of LKGCF

In LKGCF, KGCN is significantly simplified by retaining only the neighborhood aggregation (graph convolution), and embedding layer while eliminating feature transformation and nonlinear activation.

Table 2. List of key symbols

Symbol	Meaning
e_i^h, e_u^h	Embeddings of item i and user u after h rounds of aggregation
σ	Non-linear activation function
$N(u)$, $N(i)$	The set of neighbors for user u and item i
$U=\{ u_1, u_2, u_3, \ldots\ldots, u_n \}$	Set of M users
$V=\{ v_1, v_2, v_{3,} v_{4,\ldots\ldots,}, v_{n,} \}$	Set of N items
Y	User-item interaction matrix
$G=\{(h, r, t) \mid h,t \in E, r \in R\}$	Knowledge graph
E,R	Set of entities and relations in the knowledge graph G
$A= \{(v, e)|v \in V, e \in E\}$	Entities in the knowledge graoh
e_i, u, r	Embeddings of e∈E, u∈U, r∈R respectively
$\hat{y}_{u,v}$	Clicking rate score on the item v for a user
K	Number of neighbors sampled during the neighborhood aggregation
D	Dimension of the embedding
H	Loops of neighborhood aggregations
λ	*L2* regularize weight
Lr	Learning rate

The framework of LKGCF is illustrated in Figure 4. In LKGCF, the input is a knowledge graph consisting of users, items, and their relations. Each user in the recommendation systems is identified with a unique user ID. The entities in the knowledge graph are extracted from the items that are candidates for a given user or from the auxiliary information conveyed in the knowledge graph. A unique relation ID represents the relation between different entities. Then, all the items/relations/users are embedded by utilizing the full connection layer, and the weighted sum of the embeddings learned at all layers is adopted as the final embedding. It can be observed that the most crucial components of LKGCF are neighborhood aggregation and layer combination.

The Embedding Layer

Training a neural network is typically more manageable when using dense high-dimensional data than sparse low-dimensional data. However, acquiring large amounts of high-dimensional data may only sometimes be feasible. Thus, it is a great challenge to effectively leveraging low-dimensional sparse data. One common approach to address this issue is to enrich representation for low-dimensional data through embedding. By incorporating an embedding layer into the original graph neural network, obtaining a high-dimensional representation of the input becomes straightforward. Formulas 2, 3, and 4 outline the method employed to implement the embedding layer,

$$e_i = w_e . id_e \tag{2}$$

$$u = w_u . id_u \tag{3}$$

$$r = w_r . id_r \tag{4}$$

where e_i, u and r is the embedding of $e \in E$, $u \in U$, $r \in R$, respectively. w_e, w_u and w_r are the embedding weight matrices for entities, users, and their relations, respectively. id_e, id_u, id_r are the ids of entities, users, and their relations. Taking these $id\,s$ as the input of the embedding layer in the proposed LKGCF method, the outputs are dense high-dimensional features after the processing of the full connection layer. Figure 5 illustrates the process of embedding in the proposed LKGCF approach.

Figure 4. The framework of the proposed LKGCF method

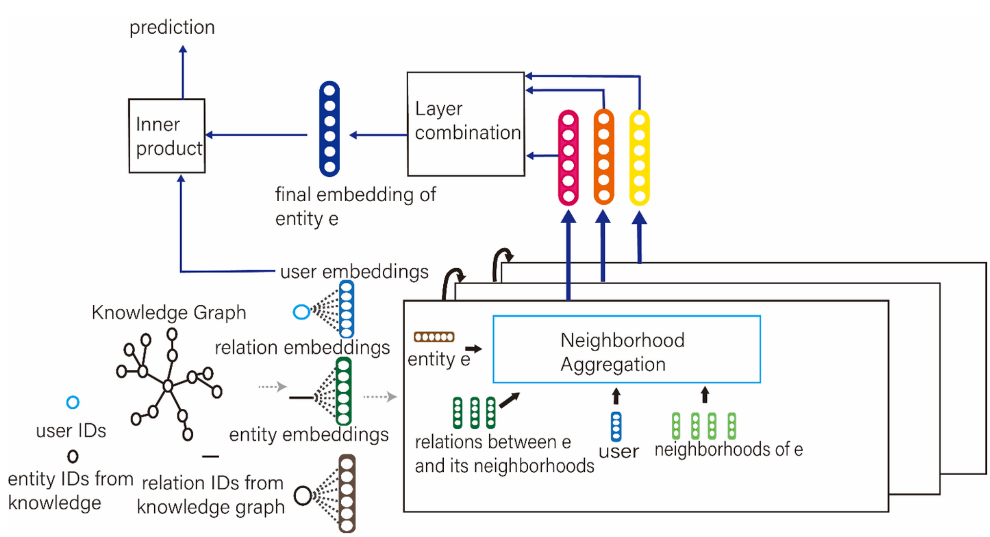


Figure 5. The process of embedding

Note: On the right side of the graph, the brown nodes represent entity embeddings, while the green edges represent relation embeddings.

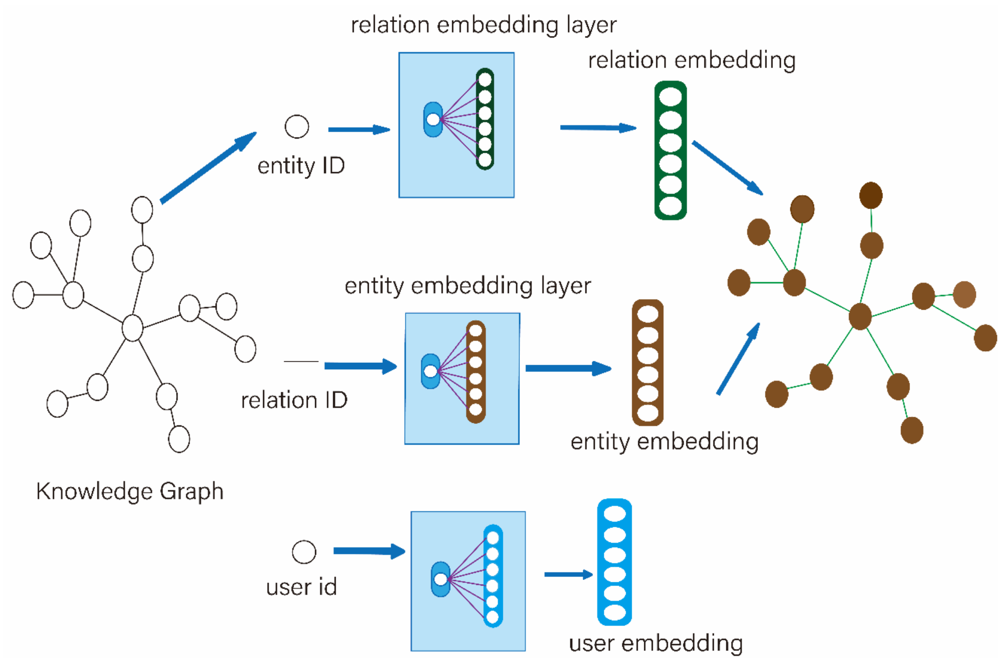
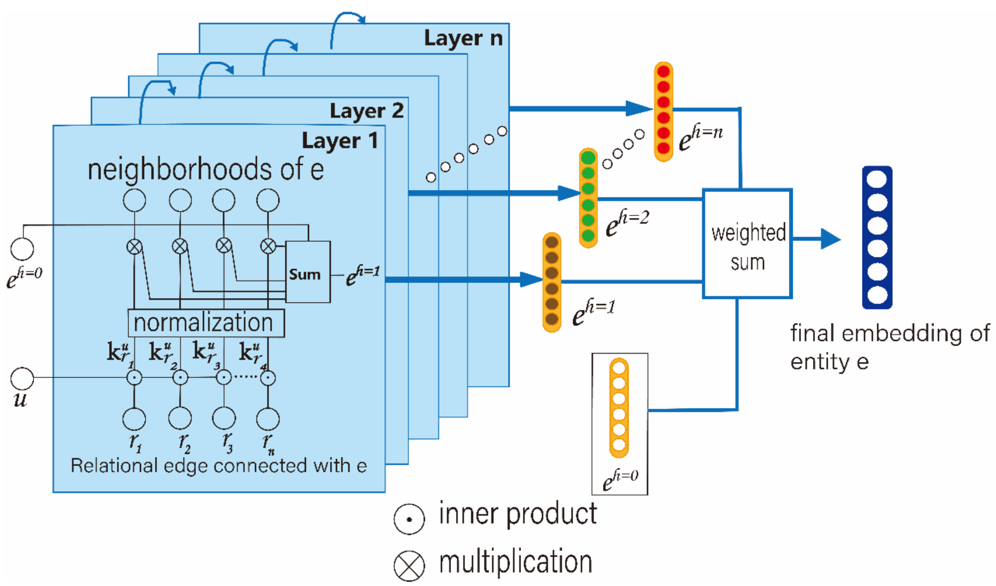


Figure 6. The framework of neighborhood aggregation and layer combination

Note: In this graph, e represents the embedding of an entity; i is the candidate item; u represents the embedding of a user u; $e^{h=0}$ *represents the embedding of e after h loops of neighborhood aggregation, and r represents the embedding of the relationship edge r in the knowledge graph.*



Neighborhood Aggregation and Layer Combination

Neighborhood aggregation plays a crucial role in LKGCF as it updates the embedding of a specific item by capturing the user's preference toward the attributes associated with the item in the knowledge graph. The structure of this component is depicted in Figure 6. Specifically, in a triad where the target entity serves as the head, the tail represents an attribute linked to the target entity. The selection of the neighborhood follows a random sampling strategy. K samples are randomly selected from the neighboring nodes of the target entity, with each sample represented by a single node. These K sampled nodes, together with the target entity and their corresponding relationships, form K triples. It is important to note that the value of K, which determines the number of random samples, is optimized through grid search to obtain the optimal value. The user's preference for the given item is determined using Formula 5.

$$K_r^u = u \cdot r^T \tag{5}$$

Where u and r are the embedding of $u \in$U and $r \in$R, respectively. K_r^u represents the degree of preference between the given user u and the relation r. A higher relation score indicates a stronger user preference for the relationship r. In a knowledge graph, an entity typically has multiple relations with other nodes, and it is necessary to normalize K, as depicted in Formula 6:

$$K\_norm_r^u = \frac{\exp\left(k_{r_{i,e}}^u\right)}{\sum_{e \in N(i)} \exp\left(k_{r_{i,e}}^u\right)} \tag{6}$$

Here, $N\left(i\right)$ denotes the embedding set of the first-order neighbors of item *i* in the knowledge graph, at the same time, $r_{i,e}$ represents the embedding of the relation edge between entity i and its neighboring entity e. Once the relation score is obtained, the conventional aggregation process in KGCN is expressed by Formula 7:

$$e_i^{h+1} = \sigma\left(w\left(e_i^h + \sum_{e \in N(i)} K\_norm_{r_{i,e}}^u e\right) + b\right) \tag{7}$$

Where e_i^h is the embedding of e_i after *h* loops of aggregation, and e_i^{h+1} represents the next aggregation result of e_i^h. LKGCF drops the operations of feature transformation and nonlinear activation function, as shown in Formula 8:

$$e_i^{h+1} = e_i^h + \sum_{e \in N(i)} K\_norm_{r_{i,e}}^u e \tag{8}$$

It is suggested that the knowledge graph contains a wealth of information, and it is not enough to collect only the nodes that are directly connected to the target node. The high-order neighbors of a target node are also believed to contain wealth information (Li et al., 2022). Therefore, the final embedding of a target item carries the implicit long-distance interests of a user. For example, the filming location of the movie "Farewell My Concubine" is Beijing. This movie is illustrated in a knowledge graph like this: "Farewell My Concubine - starring - Leslie Cheung, Farewell My Concubine - filming location – Beijing." If a user cares more about the starring role of a movie, then the relation score between the user and the "starring" obtained by applying Formula 5 is undoubtedly higher.

The aggregated embedding of "Farewell My Concubine" is believed to transfer more information about "Leslie Cheung."

As the number of aggregations increases, the feature representation of the target item becomes smoother (Li et al., 2018). Additionally, the semantic information captured in each aggregation varies. Therefore, in the proposed LKGCF approach, a layer combination is integrated to prevent excessive smoothing of embeddings. Formula 9 represents the final weighted sum of the embeddings obtained through each neighborhood aggregation:

$$e_{final} = \sum_{m=1}^{h} (e_i^m / (h+1)) \tag{9}$$

where e_{final} indicates the final embedding of the entity *e*; h means the loops of neighborhood aggregation; e_i^m is the embedding of *e* after *m* loops of neighborhood aggregation.

Prediction and Learning Algorithm

In LKGCF, the entity ID serves as the input to the embedding layer, which transforms it into high dimensional feature vectors through multiple aggregation iterations and layer combinations. Finally, the entity is represented by the embedding e_{final}. In the proposed LKGCF approach, whether an item is preferred by a user is determined by the click-through rate (CTR). The inner product of e^i and u is treated as the clicking rate score, as shown in Formula 10:

$$\hat{y}_{u,v} = e_{final} \cdot u^T \tag{10}$$

where $\hat{y}_{u,v}$ represents the clicking rate score on item *v* for a user. The likelihood that a user clicks on the item increases as $\hat{y}_{u,v}$ increases. As shown in Formula 11, after getting the clicking score, CTR can be obtained, then *pre* can also get in Formula 12:

$$CTR = sigmoid(\hat{y}_{u,v}) \tag{11}$$

$$pre = \begin{cases} 1 & if \ \ ctr \geq 0.5 \\ 0 & if \ \ ctr < 0.5 \end{cases} \tag{12}$$

The value of *pre* indicates the probability of the user clicking on the target item. A higher value of *pre* indicates that the user is more interested in the target item.

In order to improve the performance of the proposed model, LKGCF needs to learn the features of users and items constantly. The learning technique minimizes the loss function by continuously updating the initial user and item feature vectors, making the prediction more accurate. When calculating the loss of the learned model, the cross-entropy loss combined with the sigmoid function is adopted, and L2 regularization is applied to prevent over-fitting during the training procedure. Considering that the predicted labels are only mutually exclusive, namely 0 and 1, the specific loss function is shown in Formula 13:

$$loss = \sum_{(u,v)\in M} -\left(y_{u,v} lg\left(\frac{1}{1+e^{-\hat{y}_{u,v}}}\right) + (1-y_{u,v}) \lg\left(1-\frac{1}{1+e^{-\hat{y}_{u,v}}}\right)\right) + \lambda \ \ \mathcal{F}_2^2 \tag{13}$$

The first term in Formula 13 is the cross-entropy loss function, where *M* is the user-item interaction matrix. The second term is the L2 regularization that prevents over-fitting. It should be noted that in the whole learning process, only the parameters of the embedding layer in the proposed LKGCF must be updated iteratively.

The algorithm of LKGCF is presented as follows.

Algorithm 1
The Algorithm of LKGCF

Input: Knowledge Graph *G(E,R)*; interaction matrix *Y*; sampling neighbors mapping N: $e \rightarrow 2^E$; dimension of the embedding D; loops of neighborhood aggregations H; $\{\mathrm{u}\}_{u\in U}$, $\{\mathrm{r}\}_{r\in R}$, $\{\mathrm{e}\}_{e\in E}$

Output: Prediction function $F(u,v \mid \Theta, Y, G)$;

while LKGCF not converge **do**
  for (u,v) **in** Y **do**
    $\mathrm{M}[\mathrm{h}] \leftarrow \mathrm{v}$
      for h=H-1, ……., 0 **do**
        M[h] ← M[h+1]
        for e∈M[h+1] **do**
          M[h] ← M[h] ∪ N(e);
    $e^u[0] \leftarrow e, \forall e \in \mathrm{M}[0]$
    for h=1, …., H **do**
      for $e \in \mathrm{M}[\mathrm{h}]$ **do**
        $$e^u[h] = e^u[h-1] + \sum_{e' \in N(e)} K\_norm^u_{r_{e.e'}} e'[h-1]$$
    $$e_{final} = \sum_{m=1}^{\mathrm{H}} (e^u[m] / (\mathrm{H}+1))$$
    $$\hat{y}_{u,v} = e_{final} \cdot u^T$$
  Update parameters by gradient descent;
return F

EXPERIMENTS

Extensive experiments are conducted on two public datasets to evaluate the recommendation performance of the proposed LKGCF method. The next section provides an introduction to the datasets and preprocessing methods in the experiment. The strong baselines are then presented and parameter settings are explained. The experimental results are then presented and analyzed.

Datasets and Preprocessing

To examine the effectiveness of the proposed LKGCF approach, experiments are conducted on the two public datasets about film and music, respectively.

- **MovieLens-20M**. It is a movie ratings dataset widely used in movie recommendations, in which approximately 20 million explicit ratings on the MovieLens website are consistent. In MovieLens-20M, each movie is rated by users from 1 to 5, and the higher the ratings, the more the user likes the movie. Applying a lightweight model with a few training parameters proposed in this paper on such datasets may be a great challenge since the dataset of MovieLens-20M is characterized in large amounts with dense features.

- **Last.FM.** It is a public dataset for music recommendation. About 2000 users and more than 90,000 preferences on the songs are recorded in Last.FM. This is a relatively small dataset with sparse features, which is quite different from MovieLens-20M. The implicit features between users and items are relatively limited, and it needs to exploit further hidden user intention to improve the recommendation effectiveness. Thus, the LKGCF model is anticipated to achieve favorable recommendation results as the raised approach can further enrich auxiliary information by incorporating the knowledge graph. Besides, the neat LKGCF method is suitable for applications without redundant information, just like the dataset Last.FM.

Explicit ratings about the movies and songs can be obtained easily from the two datasets. However, the explicit feedbacks cannot be directly used in the proposed LKGCF approach since the input of the proposed LKGCF method is binary, namely, either positive or negative. Thus, these original explicit feedbacks are converted into 0 or 1, in which 1 represents the entity marked as positive by the users. For the corpora, MovieLens-20M, an effective mechanism of setting the threshold of positive and negative is to judge whether a rating of the target movie is higher than 4, just as KGCN did (H. Wang et al., 2019). In other words, if the rating of a given movie is 4 or 5, then the implicit feedback is transformed as 1; otherwise, the implicit feedback is transformed as 0. Considering the preferences on the songs are quite sparse on the other corpora Last.FM, if the rating of a song is higher than 0, it is marked as positive; otherwise, it is marked as negative. Once the user-item interaction is acquired, the corresponding entities for items can be inferred from the knowledge graph. Subsequently, the abundant semantic information within the knowledge graph can be utilized to construct embeddings for the items. After successfully constructing item embeddings, the items are recommended by interacting with user embeddings.

In order to ensure the fairness of the experimental results, the knowledge graph was constructed by adopting Microsoft Satori, just as KGCN was (H. Wang et al., 2019). Since only a sub-map of Microsoft Satori can be fetched from KGCN, the items not contained in the knowledge graph are eliminated.

After data processing, the user-item interaction matrix and knowledge graph are constructed, and a total of 13,501,622 interactions and 499,474 pairs of triples are generated for MovieLens-20M. And on the dataset Last.FM, 42,346 interactions, along with 15,518 triples, are built. The statistical information of the dataset after preprocessing is shown in Table 3.

Formula 14 to 17 demonstrates the computational process of all evaluation metrics in the experiment. AUC and F1 are taken as the evaluation criterion to measure the recommendation performance. The calculation of AUC is shown as follows:

$$AUC = \frac{\sum_{i \in positiveclass} rank_i - \frac{M \cdot \left(M+1\right)}{2}}{M \cdot N} \tag{14}$$

Table 3. The statistical information of the datasets after preprocessing

	MovieLens-20M	Last.FM
#users	138,159	1872
#items	16954	3846
#interactions	13,501,622	42,346
#entities	102,569	9,366
# relations	32	60
#knowledge graph triples	499,474	15,518

where $rank_i$ is the serial number of the i_{th} sample; *M* and *N* represent the amounts of the positive and negative samples, respectively.

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{15}$$

F_1 is also called the balanced F Score, which is defined as the harmonic average of precision and recall. Besides, Recall@N is taken as the evaluation metric, as shown in Formula 16:

$$Recall@N = \frac{TP}{N} \tag{16}$$

Where TP represents the number of the items recommended by the proposed model that the users are really interested in, and *N* is the total number of recommended items. Additionally, RelaImpr is taken as another evaluation measure to compare the relative improvement among different models, as shown in Formula 17:

$$RelaImpr = \left(\frac{AUC\left(measured\_model\right) - 0.5}{AUC\left(base\_model\right) - 0.5} - 1 \right) * 100\% \tag{17}$$

Where $measured\_model$ is the models applied in the experiments. Note that in the following experimental results, the $base\_model$ refers to KGCN.

Baselines

LKGCF is compared with the following strong baselines to examine how many advantages in recommendation performance can be brought.

- **Singular Value Decomposition (SVD).** This is a traditional collaborative filtering method in which the missing interactions between users and items can be predicted by applying the SVD algorithm (Koren, 2008).
- **Factorization Machine Library (libFM).** This is a model for CTR prediction based on Eigen-decomposition (Rendle, 2012).
- **Factorization Machine Library + Translating Embedding (LibFM+TransE).** This is a combination model of LibFM and TransE (Bordes et al., 2013), which captures the entity representation in the knowledge graph as auxiliary information for recommendation.
- **Personalized Entity Recommendation (PER).** The user preferences are distributed to different meta-paths in the information network to generate potential user and item features. (Yu et al., 2014).
- **Collaborative knowledge base embedding (CKE).** This is a combined recommendation model by integrating knowledge graph and collaborative filtering (Zhang et al., 2016).
- **RippleNet.** This is a recommendation model by propagating user preferences in a knowledge graph (H. Wang et al., 2018b).
- **Knowledge graph convolutional networks (KGCN).** This is a knowledge graph convolutional network that can effectively capture the inter-item relatedness by exploiting their associated attributions on the knowledge graph (H. Wang et al., 2019).

Parameter Settings

Experiments are conducted on TensorFlow (Abadi et al., 2016), and the parameters are strictly tuned through the grid search on different datasets to achieve the best experimental results. The parameters of the model on different datasets are shown in Table 4.

Results

Experimental results are shown in Table 5. It can be observed that on the two public datasets the proposed LKGCF method achieves the best prediction performance. On the MovieLens-20M dataset, the highest AUC is obtained by LKGCF. Compared with PER, the highest improvement of 15% in AUC is achieved on MovieLens-20M. Although the improvement is not notable (an enhancement of 0.4%) when compared with KGCN, notable improvements are obtained against all the other baselines when taking AUC as the evaluation metric. Furthermore, it can be observed that the results of LibFM+TransE are superior to those of LibFM. This highlights the significant advantage of incorporating knowledge graphs into recommendation systems, where the rich auxiliary information in the knowledge graph plays a crucial role in the actual recommendation process. However, neither of the two methods outperforms better than the proposed method LKGCF. It suggests that the usage of graph convolutional network also brings benefits in improving the recommendation performance.

Taking F1 as the evaluation measure, LKGCF outperforms the strong baselines on the dataset MovieLens-20M. An improvement of 1.0% is obtained when comparing LKGCF with KGCN. A surprising enhancement of 19.4% can be observed while taking PER as the baseline. On the other dataset Last.FM, the proposed method LKGCF also shows significant advantages in improving the recommendation performance. Compared with the strongest baseline KGCN, AUC improves by 2.0%, and F1 improves by 1.9% while applying the proposed LKGCF method on Last.FM. The highest advantages are obtained when comparing LKGCF with PER. Interestingly, LKGCF behaves better on the dataset Last.FM. It is believed that this is because the amount of Last.FM is smaller, and it is more suitable for training the proposed model with fewer parameters.

Table 4. The optimized parameters of the proposed model on different datasets

	MovieLens-20M	Last.FM
K	10	10
D	32	16
H	2	1
λ	1e-7	1e-4
Lr	2e-2	5e-4

Table 5. Experimental results of different approaches

Methods	MovieLens-20M AUC RelaImpr F1	Last.FM AUC RelaImpr F1
SVD	0.963 -3.14% 0.919	0.769 -9.12% 0.696
LibFM	0.959 -3.97% 0.906	0.778 -6.08% 0.710
LibFM+ TransE	0.966 -2.51% 0.917	0.777 -6.42% 0.709
PER	0.832 -30.54% 0.788	0.633 -55.07% 0.596
CKE	0.924 -11.30% 0.871	0.744 -17.57% 0.673
RippleNet	0.968 -2.09% 0.912	0.780 -5.41% 0.702
KGCN	0.978 0% 0.932	0.796 0% 0.721
LKGCF	**0.982 0.84% 0.941**	**0.812 5.41% 0.735**

With RelaImpr as the evaluation measure and KGCN as the base model, an enhancement of 0.84% is obtained on MovieLens-20M and the improvement on Last.FM is significant, with an increase of 5.41% while applying LKGCF. The advantages of the proposed LKGCF are even more notable when compared with other baselines with Relalmpr as the evaluation metric.

Furthermore, the recall of different models on the two datasets was investigated. The experimental results of Recall@10, Recall@50, and Recall @100 are illustrated in Figure 7. The figure shows that the highest recall is obtained by applying the proposed LKGCF approach, which is consistent with the results in Table 5. The experimental results demonstrate that the proposed LKGCF brings notable advantages when compared with strong baselines, despite diverse evaluation metrics. Another interesting observation was that PER behaves worse on both of the two datasets. It is analyzed that PER is a path-based model for which it is very difficult to design an effective meta-path.

It can be concluded that the proposed LKGCF method can improve the recommendation performance for different recommendation scenarios. The performance of LKGCF is attributed to the following characteristics. Firstly, LKGCF abandons the redundant structures of the traditional graph neural network in favor of a lightweight graph convolutional network for data training. Secondly, LKGCF uses layer combination to prevent the over-smoothing of the final item embedding and enriches the representation of the original input.

DISCUSSION

LKGCF has a distinct advantage over traditional recommendation systems that rely on GCN and knowledge graphs; it possesses a considerably smaller number of trainable parameters. Thus, the proposed LKGCF method significantly reduces the computational requirements during model training. Additionally, the computational resources are effectively conserved, and the computational burden on hardware is alleviated.

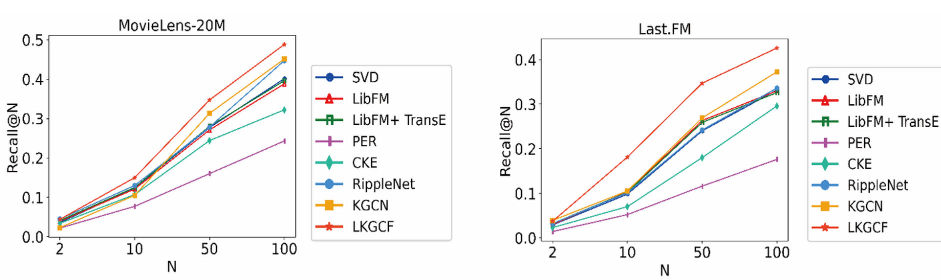
In KGCN, the trainable parameters are the embedding layer parameters, the feature transformation matrix parameters, and bias terms for each layer during the aggregation process at each iteration. The number of parameters in KGCN is presented in Formula 18. Significant numbers of training parameters also lead to growth in time complexity. The time complexity of a single forward propagation in KGCN is shown in Formula 19.

$$p_{KGCN} = P_{kgcn\_e} + P_{kgcn\_agg} = V_e * I * D + H * \left(D^2 + D\right) \tag{18}$$

$$O\left(KGCN\right) = O\left(I * D + H\left(K + D^2\right)\right) \tag{19}$$

P_{kgcn_e} and P_{kgcn_agg} represents the number of parameters in the embedding layer and the total number of parameters involved in the iterative aggregation, respectively. V_e represents the number of

Figure 7. The recall of different methods on public datasets



embeddings types. $O\left(KGCN\right)$ indicates the time complexity of a single forward pass in KGCN. I is the input dimension of KGCN; D denotes the dimension of the embedding, and H is the number of layers in KGCN.

In LKGCF, only the parameters of the embedding layer are involved in the training process, significantly reducing the number of training parameters. The number of parameters in LKGCF is presented in Formula 20. The time complexity of a single forward propagation in LKGCF is shown in Formula 21:

$$p_{LKGCF} = P_{LKGCF\_e} = V_e * I * D \tag{20}$$

$$O\left(LKGCF\right) = O\left(I * D + H * K\right) \tag{21}$$

p_{LKGCF} represents the embedding layer parameters in LKGCF. $O\left(LKGCF\right)$ represents the time complexity of a single forward propagation in LKGCF. It can be clearly observed that the parameter quantity of LKGCF is significantly reduced compared to KGCN. Furthermore, LKGCF also significantly reduces the time complexity. However, it should be noted that this is an approximate calculation, as additional memory usage must be considered. The actual complexity may vary slightly.

CONCLUSION AND FUTURE WORK

The issue of data sparsity encountered in collaborative filtering is alleviated by using the knowledge graph as auxiliary information. This paper puts forward a lightweight knowledge graph convolutional network for collaborative filtering by discarding the operations of feature transformation and nonlinear activation in traditional GCN to simplify the proposed method and decrease the number of training parameters. In this proposed approach, only the parameters of the embedding layer are needed to train, which significantly shortens the training procedure. Moreover, an additional combination layer is incorporated in the raised approach to prevent over-smoothing and enrich the embeddings of the initial input in LKGCF. Experiments on two public datasets about movies and music demonstrate that LKGCF achieves the best recommendation performance.

LKGCF can be deployed on devices with limited computational capabilities due to its small number of trainable parameters and low computational resource requirements. This is significant for reducing the operational costs of recommendation systems. Additionally, due to the essence of LKGCF, which involves graph convolution on knowledge graphs, it can also be applied in various domains that leverage knowledge graphs, such as intelligent search and intelligent question answering. However, it is observed that the utilization of user-item interaction and knowledge graph information is imbalanced, as both of exhibit different densities in quantities. This imbalance can negatively impact experimental results while applying the LKGCF method.

In the future, it is necessary to deeply investigate the latent interests of the users by avoiding representing a user just by simplistic embeddings. Besides, it is essential to enhance the diversity of recommended results by mining the items in the long tails.

AUTHOR NOTE

We have no known conflict of interest to disclose.

This work is supported by the University Natural Sciences Research Project of Anuhi Province (KJ2021A0993).

Correspondence concerning this article should be addressed to Shaohua Kuang, Dept. of School of Artificial Intelligence and Big Data, Hefei University, No. 99, Jinxiu Avenue, Hefei, Anhui Province, China. Email: kuangshaohua123@gmail.com

REFERENCES

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., & Zheng, X. (2016). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*. doi: /arXiv.1603.0446710.48550

Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., & Yakhnenko, O. (2013). Translating embeddings for modeling multi-relational data. *Advances in Neural Information Processing Systems*, *26*, 1478–1488. doi:10.1145/3543507.3583241

Bruna, J., Zaremba, W., Szlam, A., & LeCun, Y. (2013). Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*. doi: /arXiv.1312.620310.48550

Chen, H., Li, Y., Sun, X., Xu, G., & Yin, H. (2021, March). Temporal meta-path guided explainable recommendation. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining* (pp. 1056-1064). ACM. doi:10.1145/3437963.3441762

George, G., & Lal, A. M. (2021). A personalized approach to course recommendation in higher education. *International Journal on Semantic Web and Information Systems*, *17*(2), 100–114. doi:10.4018/IJSWIS.2021040106

Gori, M., Monfardini, G., & Scarselli, F. (2005, July). A new model for learning in graph domains. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks,* 2005. (Vol. 2, pp. 729-734). IEEE. doi:10.1109/IJCNN.2005.1555942

Guo, Q., Zhuang, F., Qin, C., Zhu, H., Xie, X., Xiong, H., & He, Q. (2020). A survey on knowledge graph-based recommender systems. *IEEE Transactions on Knowledge and Data Engineering*, *34*(8), 3549–3568. doi:10.1109/TKDE.2020.3028705

Hamilton, W., Ying, Z., & Leskovec, J. (2017). Inductive representation learning on large graphs. *Advances in Neural Information Processing Systems*, *30*. Advance online publication. doi:10.48550/arXiv.1706.02216

He, X., Deng, K., Wang, X., Li, Y., Zhang, Y., & Wang, M. (2020, July). 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* (pp. 639-648). ACM. doi:10.1145/3397271.3401063

He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T. S. (2017, April). Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web* (pp. 173-182). ACM. doi:10.1145/3038912.3052569

Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, *22*(1), 5–53. doi:10.1145/963770.963772

Hu, B., Shi, C., Zhao, W. X., & Yu, P. S. (2018, July). 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 and Data Mining* (pp. 1531-1540). ACM. doi:10.1145/3219819.3219965

Ji, S., Pan, S., Cambria, E., Marttinen, P., & Philip, S. Y. (2021). A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems*, *33*(2), 494–514. doi:10.1109/TNNLS.2021.3070843 PMID:33900922

Kipf, T. N., & Welling, M. (2016). Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*. https://doi.org//arXiv.1609.0290710.48550

Koren, Y. (2008, August). Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 426-434). ACM. doi:10.1145/1401890.1401944

Li, P., Li, T., Wang, X., Zhang, S., Jiang, Y., & Tang, Y. (2022). Scholar recommendation based on high-order propagation of knowledge graphs. *International Journal on Semantic Web and Information Systems*, *18*(1), 1–19. doi:10.4018/IJSWIS.313190

Li, Q., Han, Z., & Wu, X. M. (2018, April). Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 32, No. 1). AAAI. doi:10.1609/aaai.v32i1.11604

Rendle, S. (2012). Factorization machines with libfm. [TIST]. *ACM Transactions on Intelligent Systems and Technology*, *3*(3), 1–22. doi:10.1145/2168752.2168771

Salloum, G., & Tekli, J. (2021). Automated and personalized nutrition health assessment, recommendation, and progress evaluation using fuzzy reasoning. *International Journal of Human-Computer Studies*, *151*, 102610. doi:10.1016/j.ijhcs.2021.102610

Wang, H., Zhang, F., Wang, J., Zhao, M., Li, W., Xie, X., & Guo, M. (2018b, October). Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management* (pp. 417-426). ACM. doi:10.1145/3269206.3271739

Wang, H., Zhang, F., Xie, X., & Guo, M. (2018a, April). DKN: Deep knowledge-aware network for news recommendation. In *Proceedings of the 2018 World Wide Web Conference* (pp. 1835-1844). ACM. doi:10.1145/3178876.3186175

Wang, H., Zhao, M., Xie, X., Li, W., & Guo, M. (2019, May). Knowledge graph convolutional networks for recommender systems. In *The World Wide Web Conference* (pp. 3307-3313). ACM. doi:10.1145/3308558.3313417

Wang, R., Wang, X., Shi, C., & Song, L. (2022). Uncovering the structural fairness in graph contrastive learning. *Advances in Neural Information Processing Systems*, *35*, 32465–32473. doi:10.48550/arXiv.2210.03011

Wang, X., He, X., Cao, Y., Liu, M., & Chua, T. S. (2019a, July). Kgat: Knowledge graph attention network for recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 950-958). ACM. doi:10.1145/3292500.3330989

Wang, X., He, X., Wang, M., Feng, F., & Chua, T. S. (2019b, July). Neural graph collaborative filtering. In *Proceedings of the 42nd international ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 165-174). ACM. doi:10.1145/3331184.3331267

Wang, X., Wang, D., Xu, C., He, X., Cao, Y., & Chua, T. S. (2019c, July). Explainable reasoning over knowledge graphs for recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 33, No. 01, pp. 5329-5336). ACM. . v33i01.33015329 doi:10.1609/aaai.v33i01.33015329

Wei, T., & He, J. (2022, August). Comprehensive fair meta-learned recommender system. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (pp. 1989-1999). ACM. doi:10.1145/3534678.3539269

Xiao, J., Liu, X., Zeng, J., Cao, Y., & Feng, Z. (2022). Recommendation of healthcare services based on an embedded user profile model. *International Journal on Semantic Web and Information Systems*, *18*(1), 1–21. doi:10.4018/IJSWIS.313198

Yu, X., Ren, X., Sun, Y., Gu, Q., Sturt, B., Khandelwal, U., Norick, B., Han, J., & Han, J. (2014, February). Personalized entity recommendation: A heterogeneous information network approach. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining* (pp. 283-292). ACM. doi:10.1145/2556195.2556259

Zhang, F., Yuan, N. J., Lian, D., Xie, X., & Ma, W. Y. (2016, August). Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 353-362). ACM. doi:10.1145/2939672.2939673

Zhao, X., Cheng, Z., Zhu, L., Zheng, J., & Li, X. (2021, July). UGRec: modeling directed and undirected relations for recommendation. *In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 193-202). ACM. doi:10.1145/3404835.3462835

Xin Zhang received the bachelor's degree from Shandong University and the Ph.D. degree from the University of Chinese Academy of Sciences, China. She is currently an associate professor with the Department of Artificial Intelligence and Big Data, Hefei University. Her research interests include big data information processing and information retrieval.

Shaohua Kuang received a bachelor's degree from Fuyang Normal University and is currently pursuing a master's degree from Hefei University. His research direction include recommendation systems, knowledge graphs, and graph neural networks.