# Recommendation Algorithm Based on Refined Knowledge Graphs and Contrastive Learning

**Chapter** · July 2024

DOI: 10.1007/978-981-97-5501-1_16



**6 authors**, including:

Xingpeng Zhang
Southwest Petroleum University

# Recommendation Algorithm Based on Refined Knowledge Graphs and Contrastive Learning

Bing Wang[1], Xiaoling Yang[1], Xingpeng Zhang[1(✉)], Chunlan Zhao[2], Chunhao Wang[1], and Weishan Feng[1]

[1] School of Computer Science and Software Engineering, Southwest Petroleum University, Chengdu, China
{wang_bing,xpzhang}@swpu.edu.cn

[2] School of Science, Southwest Petroleum University, Chengdu, China
zhaochunlan@swpu.edu.cn

**Abstract.** Knowledge graphs are a vital tool for improving recommendation performance and interpretability since they are rich in auxiliary data. Two obstacles must be overcome in the recommendation process, though: 1) Real-world knowledge graphs are frequently noisy and contain connections unrelated to items and entities; 2) Data sparsity is a problem due to the long-tail distribution of user-item interactions. To address the aforementioned issues, a recommendation algorithm based on refined knowledge graphs and contrastive learning (RKGCL) is proposed. Firstly, the algorithm utilizes a graph pruning strategy to trim task-irrelevant knowledge associations, obtaining a high-quality knowledge graph. Secondly, it employs a graph convolutional network based on attention mechanisms to learn item embeddings. Subsequently, a simple yet effective noise-based embedding enhancement is applied for cross-layer contrastive learning, thereby alleviating the data sparsity issue. Finally, a message propagation strategy is employed to obtain user and item embeddings for recommendation prediction. Experimental results on three public datasets - Yelp, amazon-book, and Last-FM - show that our model outperforms other benchmark models on Recall@K and NDCG@K.

**Keywords:** Recommendation Algorithm · Knowledge Graph · Contrastive Learning · Graph Convolutional Networks

## 1 Introduction

Recommendation systems are widely utilized in real-life applications such as ecommerce, online advertising, and social media platforms, providing personalized information services. Collaborative Filtering [9] (CF) stands out as the most extensively applied recommendation algorithm, whose fundamental idea is to leverage the similar behaviors among users to identify those with analogous


C. Cao et al. (Eds.): KSEM 2024, LNAI 14887, pp. 205–217, 2024.
https://doi.org/10.1007/978-981-97-5501-1_16

preferences and recommend potentially interesting items to these similar users. However, they heavily rely on historical user interaction data, making them susceptible to issues like data sparsity and the cold-start problem.

Recently, an increasing number of researchers have started employing knowledge graphs as supplementary information for recommendation systems. This is mainly because knowledge graphs contain rich semantic information, effectively improving the accuracy of feature representation. In addition, it can also alleviate the data sparsity and cold start problems encountered in the recommendation process. The combination of recommendation and knowledge graph is often called knowledge perception recommendation method. Graph neural networks (GNN) can iteratively propagate higher-order information on knowledge graphs and obtain more accurate embedded representations. The knowledge perception recommendation method based on GNN faces the following two problems:

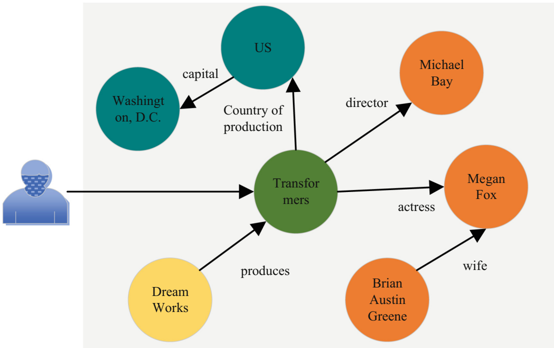


**Fig. 1.** An example of a Film Knowledge Graph.

(1) Knowledge graphs suffer from the issue of noise. Knowledge graphs in real-life scenarios are often sparse, noisy, and contain irrelevant connections between items and entities. Taking Fig. 1 as an example, triplets like (United States, capital, Washington, D.C.) from the movie Transformers and (Brian Austin Green, wife, Megan Fox) are unrelated facts, which is not beneficial for learning high-quality user and item representations. Current knowledge-aware recommendation methods that incorporate knowledge graphs are susceptible to the influence of noise in the knowledge graph.

(2) Data sparsity. However, in practical recommendation scenarios, interaction data between users and items often exhibits sparsity. Most users interact with only a few items, and many items are never noticed by any user. This results in the cold-start problem, making it challenging to provide effective recommendations for items that have not been interacted with by any user before.

To address the aforementioned issues, this paper proposes a knowledge-aware recommendation algorithm based on refined knowledge graphs and contrastive learning. The algorithm starts by employing a graph pruning strategy to trim task-irrelevant knowledge associations, resulting in a high-quality knowledge graph. Subsequently, it utilizes a graph convolutional network based on attention mechanisms to learn item embeddings. Following this, a simple yet effective noise-based embed-ding enhancement is applied for cross-layer contrastive learning to alleviate the data sparsity problem. Finally, a message propagation strategy is employed to obtain user and item embeddings for recommendation prediction. The main contributions of this paper are as follows:

(1) Leveraging knowledge associations within the knowledge graph allows not only the pruning of irrelevant triplets based on a graph pruning strategy but also the aggregation of diverse facts to obtain high-quality knowledge representations.
(2) Introducing a straightforward yet effective noise-based embedding enhancement for cross-layer contrastive learning. By adding uniform noise to the embedding space to create contrastive views, the representations become more uniform, alleviating the issue of data sparsity and enhancing the accuracy and personalization of recommendations.
(3) Proposing a knowledge-aware recommendation algorithm based on refined knowledge graphs and contrastive learning (RKGCL). Experimental evaluations on three publicly datasets demonstrate the significant advantages.

## 2 Related Work

Currently, knowledge graph-based recommendation can be categorized into three types: embedding-based, path-based, and GNN-based recommendation.

Embedding-based recommendation methods primarily leverage knowledge graph embedding techniques such as TransE, TransR, TransH [6,12], etc., to store entities and relations in a low-dimensional space. This process results in vector representations of entities and relations in this feature space, which are then used for recommendation tasks. While these methods effectively preserve structural information within the knowledge graph, they lack the utilization of semantic information concerning the interconnected paths between entities.

Path-based recommendation, such as Hete-MF [17], primarily analyze various connection relationships between entities in the knowledge graph. They select crucial connecting paths and learn semantic information contained within these paths for recommendation purposes. However, the effectiveness heavily relies on manual path construction, leading to poor generalization.

GNN-based methods utilize embedding propagation to iteratively aggregate domain information. By stacking propagation layers, each node can access information from high-order neighbors, making it the prevailing approach in current knowledge-aware recommendation. The KGAT [8] model integrates user-item interaction graph and knowledge graph into a collaborative knowledge graph.

It leverages this collaborative knowledge graph to mine high-order connections between users and items, enriching embedding representations and enhancing recommendation performance. However, the effectiveness of domain information aggregation depends on a high-quality knowledge graph.

# 3 The Proposed Method

## 3.1 Model Description

The proposed RKGCL model, illustrated in Fig. 2, consists of three components:

(1) Knowledge Refinement Aggregation Module: This module adopts pruning strategy for knowledge graph and trims the connection of project entities unrelated to task to obtain high-quality knowledge graph of refined triplet.
(2) Cross-Layer Contrastive Learning Module: This module introduces two different noise sources to enhance item embeddings and applies contrast learning to both enhanced representations. Introduce some differences while retaining most of the information in the original representation.
(3) Prediction Layer: Taking the embeddings of users and items as input, this layer calculates the probability of a user clicking on an item.

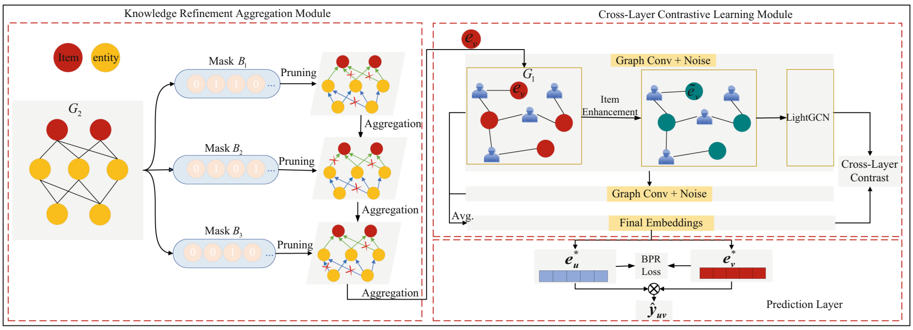


**Fig. 2.** The over framework of RKGCL.

## 3.2 Symbols and Problem Descriptions

The user-item interaction graph is defined as $G_1 = \{(u,v)|u \in U, v \in V, y_{uv} = 1\}$, where $G_1$ is composed of user set $U = \{u_1, u_2, ..., u_M\}$ and item set $V = \{v_1, v_2, ..., v_K\}$. $M$ and $K$ represent the total number of users and items. $y_{uv} = 1$ indicates that there is an interaction between the user and the item, such as clicks, ratings, purchases, etc. $y_{uv} = 0$ means no interaction.

The knowledge graph is defined as $G_2 = \{(h, r, t)|h, t \in \varepsilon, r \in R\}$, where each triplet $(h, r, t) \in T$ represents a relationship $r$ from head entity $h$ to tail entity $t$. $T$, $\varepsilon$, and $R$ denote the sets of triplets, entities, and relations, respectively. For example, $(DetectiveChinatown, Genre, Comedy)$ represents the relationship that the movie "Detective Chinatown" belongs to the genre of comedy.

Given user-item interaction graph $G_1$ and knowledge graph $G_2$, the task is to predict the interaction probability between a user $u$ and an item $v$ that the user has not previously interacted with. Specifically, we aims to learn a prediction function $\hat{y}_{uv} = F(u, v|\Theta)$, where $\hat{y}_{uv}$ represents the probability that user $u$ will interact with item $v$, and $\Theta$ denotes the parameters of the function $F$.

### 3.3 Knowledge Refinement Aggregation Module

**Pruning of Irrelevant Facts.** A large amount of noise and task-irrelevant information in the knowledge graph will degrade the performance. The strategy of manually constructing K-neighborhood subgraphs to constrain the acceptance domain of nodes and randomly dropping some edges for contrast learning depends heavily on the construction quality of the graph, and can not adaptive remove unnecessary edges according to the recommended task. Therefore, this paper adopts a parametric approach to learn pruning strategies. Firstly, a binary mask $b \in \{0, 1\}$ is attached to each triplet in the knowledge graph, indicating whether the triplet should be deleted. Graph pruning can be expressed as:

$$\tilde{T} = \{(h_i, r_i, t_i) \mid b_i = 1\} \tag{1}$$

where $\tilde{T}$ is a subset of $T$. Due to the discrete and non-differentiable nature of the mask set, direct optimization is computationally challenging [3,5]. To address this issue, let $B = (b_1, ..., b_K)$, where each $b_i$ follows a Bernoulli distribution with parameter $\sigma(\alpha_i)$, i.e. $b_i \sim Bern(\sigma(\alpha_i))$. Here, $\sigma(\cdot)$ is the sigmoid function. To jointly optimize the mask with the recommendation task, this paper combines the binary mask with the target loss $L$ and reexpresses it as

$$\tilde{L}(\alpha, \Theta) = \mathbb{E}_{B \sim \Pi_{i=1}^{K} \mathrm{Bern}(b_i; \sigma(\sigma_i))}[L(B, \Theta)] \tag{2}$$

where $\mathbb{E}$ represents the expectation, $\Theta$ denotes the remaining parameters, and $\tilde{L}$ is the Evidence Lower Bound (ELBO) on the target loss $L$ with respect to the parameters $\alpha$. Research shows that existing gradient estimation methods for discrete variables, such as Gumbel-Softmax [5], ARM [15], disarmament [3], etc. are susceptible to high variance or gradient bias. Therefore, this paper adopts the recently proposed Adaptive Reinforcement-based Monte Carlo Search (ARMS) [2], an effective and unbiased low-variance gradient estimator for backpropagating the gradient with respect to the parameter $\alpha$.

The gradient of $\tilde{L}$ with respect to the parameter $\alpha$ is defined as follows:

$$\nabla_\alpha \tilde{L}(\alpha, \Theta) = \sum_i [f(B) - \frac{1}{2}(f(B) + f(\tilde{B}))](\tilde{b}_i - \sigma(\alpha_i)) \tag{3}$$

where $(B, \tilde{B}) = ((b_1, \tilde{b}_1), \ldots, (b_K, \tilde{b}_K))$, the discrete pair $((b_i, \tilde{b}_i) = (1_{1-u_i<\sigma(\alpha_i)}, 1_{i<\sigma(\alpha_i)}))$, and $u \sim U(0,1)$ is sampled from a uniform distribution. When $1 - u_i < \sigma(\alpha_i)$ holds, $b_i = 1$, otherwise, it is 0. The same logic applies to $\tilde{b}_i$. $f(\cdot)$ is the model loss after pruning in the forward pass of RKGCL.

**Composite Knowledge Aggregation.** To obtain a more comprehensive understanding of user preferences, this paper introduces a novel message aggregation mechanism, incorporating noise message pruning and composite knowledge aggregation. Specifically, utilizing $N_h = \{(r,t)|(h,r,t) \in G_2\}$ to denote the domain entities and first-order relationships of item $h$ in the knowledge graph. We propose integrating contextually diverse relations from domain entities to generate the first-order knowledge representation for entity $h$:

$$e_h^{(1)} = e_h^{(0)} + \sum_{r,t \in N_h} \alpha(e_h^{(0)}, e_r, e_t^{(0)}) \cdot b_{h,t}^{(0)} \tag{4}$$

where $e_h^{(0)}$, $e_t^{(0)}$, and $e_r$ represent the initialized embedding representations of the head and tail entities of the triple and the relation between them, respectively. The binary variable $b_{h,t}^{(0)} \in \{0,1\}$ indicates whether the triple $(h,r,t)$ should be pruned. The function $\alpha(e_h, e_r, e_t)$ denotes the estimated attention relevance specific to the entity and relation during the knowledge aggregation process, encoding distinct semantics between items and the relationships among entities.

$$\alpha(e_h, e_r, e_t) = \frac{\exp(\mathrm{LeakyReLU}(e_r^T W[e_h||e_t]))}{\sum_{r,t \in N_h} \exp(\mathrm{LeakyReLU}(e_r^T W[e_h||e_t]))} \tag{5}$$

The matrix $W \in \mathbb{R}^{d \times 2d}$ represents the parameter weight matrix customized based on input items and entity representations. The LeakyReLU activation function is employed for non-linear transformation.

To explore higher-order knowledge associations of the item, additional aggregation layers are stacked. Technically, after $L$ layers, the recursive formulation of the knowledge representation for item $h$ is given by:

$$e_h^{(l)} = e_h^{(l-1)} + \sum_{r,t \in N_h} \alpha(e_h^{(l-1)}, e_r, e_t^{(l-1)}) \cdot b_{h,t}^{(l-1)} \tag{6}$$

### 3.4 Cross-layer Contrastive Learning Module

Contrastive learning aligns well with the need of recommendation systems to address the data sparsity issue by extracting self-supervised signals from raw data. A typical approach of contrast learning in recommendation, such as SGL, enhances the user-item interaction graph with structural perturbation, i.e. random removal of edges/nodes in a certain proportion. However, this random setting may remove key nodes and associated edges, severely damaging the semantic integrity of the original graph. Therefore, this paper adopts the approach from SimGCL, adding uniform noise to the embedding space to create contrasting views. This method preserves most of the information from the original representation while introducing some differences, resulting in data-enhanced representations, mitigating popularity bias and achieving more uniform representations.

By introducing random noise into the embedding representation of item $v$, the enhanced representation of the item is $\widetilde{e}_v = e_v + \text{sign}(e_v) \odot D$. Here, $e_v$ denotes the aggregated embedding of the item after knowledge aggregation, and $D \in \mathbb{R}^d \sim U\{0, 1\}$ is a randomly generated dropout strategy vector.

Under the user-item interaction graph, multiple graph propagation layers are stacked to capture high-order collaborative signals. The enhanced item representation is used as the input feature vector, preserving the semantic information of the knowledge graph. Due to the effectiveness and lightweight architecture of LightGCN [4], its message-passing strategy is adopted to encode collaborative knowledge in the user-item interaction graph $G_1$, as shown below:

$$e_u^{l+1} = \sum_{i \in N_u} \frac{e_v^l}{\sqrt{|N_u||N_v|}}; \quad e_v^{l+1} = \sum_{i \in N_v} \frac{e_u^l}{\sqrt{|N_u||N_v|}} \tag{7}$$

where $\mathrm{e}_u^l$ and $e_v^l$ represent the embedding representations of user $u$ and item $v$ at the $l$-th propagation layer. $N_u$ and $N_v$ denote the sets of items interacted with user $u$ and users interacted with item $v$, respectively.

When capturing high-order collaborative signals at each layer, different-scale random noise is added to the current node embeddings to obtain enhanced representations for users and items. SimGCL chooses different enhancements of the same node and the same layer for comparative learning. However, the similarity of mutual information between these nodes is very high, which easily leads to suboptimal comparison. To address this issue, our approach contrasts enhanced representations from different layers, which share some common information but differ in terms of aggregating neighbors and adding noise. Then we defines the contrastive objective $L_{\text{cl}}$ based on the InfoNCE [1] loss, as follows:

$$L_{cl} = \sum_{i \in N} \left( -\log \frac{\exp(\tilde{z}_i^T \tilde{z}_i^l / \tau)}{\sum_{j \in N} \exp(\tilde{z}_i^T \tilde{z}_j^l / \tau)} \right) \tag{8}$$

Here, $i$ and $j$ represent users/projects sampled from the batch of size $N$, $\tilde{z}_i$ and $\tilde{z}_i^l$ are $L2$-normalized representations with different noise added from different layers for data augmentation learning (i.e., $\tilde{z}_i = \frac{\tilde{e}_i}{||\tilde{e}_i||_2}$). $\tau$ is the temperature that controls the strength of the hard negative sample penalty. The InfoNCE loss promotes consistency between positive samples $\tilde{z}_i$ and $\tilde{z}_i^l$, while minimizing the consistency between negative samples $\tilde{z}_i$ and $\tilde{z}_j^l$.

### 3.5 Model Prediction

Averaging the outputs from multiple layers to get the final embedding.

$$\mathrm{e}_u^* = \frac{1}{L+1}\sum_{l=0}^{L} e_u^l, \quad \mathrm{e}_v^* = \frac{1}{L+1}\sum_{l=0}^{L} e_v^l \tag{9}$$

Then, a prediction function is conducted to estimate the probability of a user $u$ clicking on an item $v$ with which they have not interacted, i.e. $\hat{y}_{uv} = \mathbf{e}_u^{*T} \cdot \mathbf{e}_v^*$.

The loss function employed for predicting click-through rate utilizes the Bayesian Personalized Ranking (BPR) recommendation loss, defined as follows:

$$L_b = \sum_{u \in U} \sum_{v \in N_u} \sum_{v' \notin N_u} -\log \sigma(\hat{y}_{uv} - \hat{y}_{uv'}) \tag{10}$$

where $N_u$ represents the observed set of items interacted with by user $u$, and negative samples are drawn from the non-interactive items $v'$ of user $u$.

In the learning process of RKGCL, the supervised recommendation task serves as the primary task, and the contrastive learning task serves as an auxiliary task. Comprehensive loss optimizes is defined below:

$$L = L_b + \lambda_1 L_{cl} + \lambda_2 \|\Theta|_2^2 \tag{11}$$

where $\lambda_1$ and $\lambda_2$ represent parameters determining the strength of self-supervised signals and regularization for the joint loss function, respectively. $\Theta$ represents the learnable model parameters.

## 4 Experiments

### 4.1 Datasets

We conducted experiments on three publicly collected real-life datasets: Yelp2018, Amazon-Book, and Last-FM. These datasets vary in scale and density. The Yelp2018 dataset consists of user ratings for business places collected from the Yelp platform, where local businesses such as restaurants and bars are considered as items. The Amazon-Book dataset contains user ratings for books collected from the Amazon platform, where books are considered as items. The Last-FM dataset consists of listening records of numerous users collected from the Last.fm music website, where songs are considered as items. Following the work of reference [8], two-hop neighbor entities of items in the knowledge graph were collected to construct the item knowledge graph for each dataset. The datasets were then divided into training, validation, and test sets at a ratio of 8:1:1. Table 1 shows detailed information that used in the experiments.

**Table 1.** Statistics for three datasets.

| | Yelp2018 | Yelp2018 | Last-FM |
|---|---|---|---|
| #Users | 45919 | 70679 | 1872 |
| #Items | 45838 | 24915 | 3846 |
| #Interactions | 1183610 | 846434 | 42346 |
| #Entities | 47472 | 29714 | 9366 |
| #Relations | 42 | 39 | 60 |
| #Triplets | 869603 | 686516 | 15518 |

### 4.2 Contrast Methods

Experiments compared the proposed model with the following baseline models:

CKE [18]: An embedding-based method. It uses TransR to regularize item representations in the knowledge graph and inputs the learned embeddings into the MF framework.

RippleNet [7]: A path-based recommendation method that propagates user preferences along pre-constructed paths in the knowledge graph.

KGAT [8]: A propagation-based recommendation method that recursively propagates embeddings using attention mechanisms.

CKAN [11]: A graph neural network-based recommendation method that employs different aggregation mechanisms for the user-item bipartite graph and the knowledge graph to encode user and item representations.

KGIN [10]: A propagation-based recommendation method that captures distant semantics through a relation-aware aggregation scheme.

KGCL [14]: A propagation-based recommendation method that introduces a knowledge graph enhancement mode to guide the contrastive learning paradigm.

SGL [13]: An advanced self-supervised graph recommendation method that constructs multiple graph and performs contrastive learning for robust learning.

SimGCL [16]: An advanced contrastive learning method that proposes a simple contrastive strategy by adding uniform noise in the embedding space.

### 4.3 Experimental Setting

This experiment employs a full-ranking strategy for performance evaluation [14]. All items not previously interacted with by users are considered as negative samples. Two representative metrics, Recall@N and NDCG@N, are utilized to evaluate the accuracy of recommended items [8,10]. The average evaluation results for all users in the test set are reported, with N set to 20 by default.

The model's code is implemented using PyTorch. To ensure fairness, all baseline models have their parameters set to the optimal values mentioned in their original papers. The common settings include embedding vector dimensions set to 64, batch size set to 1024, and epochs set to 1000.

### 4.4 Results and Analysis

**Comparison Experiment.** Table 2 provides an overall performance evaluation of different models on the dataset.

**Table 2.** Performance Comparison with Baselines on Three Datasets.

| Model | Yelp2018 | | Amazon-book | | Last-FM | |
|---|---|---|---|---|---|---|
| | Recall | NDCG | Recall | NDCG | Recall | NDCG |
| CKE | 0.0686 | 0.0431 | 0.1375 | 0.0685 | 0.0732 | 0.0630 |
| RippleNet | 0.0422 | 0.0251 | 0.1058 | 0.0549 | 0.0814 | 0.0683 |
| KGAT | 0.0675 | 0.0432 | 0.1390 | 0.0739 | 0.0873 | 0.0744 |
| CKAN | 0.0689 | 0.0441 | 0.1380 | 0.0726 | 0.0812 | 0.0660 |
| KGIN | 0.0712 | 0.0462 | 0.1436 | 0.0748 | 0.0920 | 0.0791 |
| KGCL | 0.0756 | 0.0495 | 0.1492 | 0.0785 | 0.0686 | 0.0629 |
| SGL | 0.0719 | 0.0475 | 0.1445 | 0.0766 | 0.0879 | 0.0775 |
| SimGCL | 0.0768 | 0.0520 | 0.1501 | 0.0813 | 0.0824 | 0.0736 |
| **RKGCL** | **0.0794** | **0.0545** | **0.1578** | **0.0836** | **0.0981** | **0.0831** |
| Imp% | 3.39% | 4.81% | 5.13% | 2.83% | 6.63% | 5.06% |

RKGCL consistently demonstrates significant improvements across all datasets compared to the best-performing baselines. NDCG@20 increases by 4.81%, 2.83%, and 5.06% on Yelp2018, Amazon-book, and Last-FM, respectively. These improvements are primarily attributed to the knowledge graph refinement and cross-layer contrastive learning in RKGCL. (1) Knowledge graph refinement: RKGCL uses parameterized binary masks to trim irrelevant facts in the knowledge graph, and then performs aggregation operations to obtain various knowledge related to entities. (2) Cross-layer contrast learning: RKGCL uses cross-layer contrast learning to create contrast views to enhance data representation. Compared with the structural perturbation method of SGL, this method can avoid the loss of key information, reduce the data sparsity and prevalence bias, and produce a more uniform representation.

**Ablation Experiment.** To verify the effectiveness of knowledge refinement and cross-layer contrastive learning, three variants of RKGCL were constructed: (1) RKGCL-r&cl: Knowledge refinement and cross-layer contrastive learning are removed. (2) RKGCL-r: The knowledge refinement part is removed. (3) RKGCL-cl: The cross-layer contrastive learning part is removed.

From Table 3, it is evident that compared to the complete RKGCL model, the performance of RKGCL-r&cl, which lacks knowledge refinement and cross-layer contrastive learning, is significantly reduced. This underscores the necessity of fine-grained knowledge graph modeling and cross-layer contrastive learning. Specifically, RKGCL-r directly aggregates all knowledge associations from the knowledge graph, disregarding the noise in interactions. Consequently, it fails to accurately analyze items and propagate information for learning. While RKGCL-cl retains the knowledge refinement component, its lack of cross-layer contrastive learning results in suboptimal modeling of user preferences.

**Table 3.** Results of Ablation Study.

| Model | Yelp2018 | | Amazon-book | | Last-FM | |
|---|---|---|---|---|---|---|
| | Recall | NDCG | Recall | NDCG | Recall | NDCG |
| RKGCL-r | 0.0772 | 0.0531 | 0.1559 | 0.0828 | 0.0969 | 0.0825 |
| RKGCL-cl | 0.0751 | 0.0525 | 0.1544 | 0.0820 | 0.0944 | 0.0811 |
| RKGCL-r&cl | 0.0724 | 0.0512 | 0.1526 | 0.0814 | 0.0925 | 0.0803 |
| RKGCL | 0.0794 | 0.0545 | 0.1578 | 0.0836 | 0.0981 | 0.0831 |

**Robustness Experiments with Different Noise Ratios.** The experiment constructs training and validation sets with varying noise ratios by randomly replacing the tail entity $t$ with a new tail entity $t'$ in the datasets. The training set remains unchanged during this process. Figure 3 show that RKGCL is more robust than other baseline methods. As the number of noisy triplets increases, RKGCL maintains nearly the same performance, but other methods like KGCL show a significant performance decline.

**Hyperparametric Experiment.** The model utilizes weight coefficient $\lambda_1$ to control the contrastive learning and $\tau$ to control the strength of hard negative sampling. To analyze the impact of $\lambda_1$ and $\tau$, experiments were conducted with $\lambda_1$ set to $\{10^{-1}, 10^{-2}, 10^{-3}\}$ and $\tau$ set to $\{0.1, 0.2, 0.3, 0.4\}$. As shown in Fig. 4, the optimal performance is obtained with $\lambda_1 = 0.1$ and $\tau = 0.2$. This suggests that larger values of $\tau$ may limit the discriminative ability between different negative instances. Additionally, a smaller value of $\lambda_1$ corresponds to a smaller impact of the contrastive loss on the main embedding space.

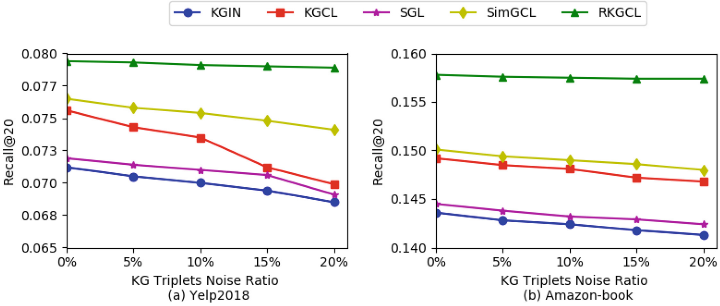


**Fig. 3.** Impact of different ratio of noise in knowledge graph.

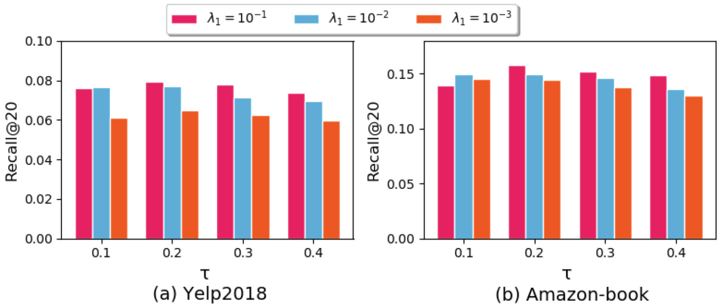


**Fig. 4.** Impact of hyperparameters $\lambda_1$ and $\tau$.

# 5 Conclusion

This paper introduces a new recommendation algorithm designed to solve the problem that existing methods may ignore noise in knowledge graphs, and the problem that long-tail distributions in user-item interactions lead to data sparsity. Our algorithm proposes a graph pruning strategy trims task-irrelevant triplets, enhancing the quality of the knowledge graph. Utilizing a graph convolutional network with attention mechanisms, the algorithm learns item embeddings. Cross-layer contrastive learning within the user-item interaction graph helps alleviate data sparsity. Finally, a message propagation strategy yields user-item embeddings for recommendation predictions. Experimental results on three real datasets demonstrate the superior performance of our algorithm compared to baseline models in recommendation tasks.

# References


1. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: International Conference on Machine Learning, pp. 1597–1607 (2020)
2. Dimitriev, A., Zhou, M.: Arms: Antithetic-reinforce-multi-sample gradient for binary variables. In: International Conference on Machine Learning, pp. 2717–2727 (2021)
3. Dong, Z., Mnih, A., Tucker, G.: Disarm: an antithetic gradient estimator for binary latent variables. Adv. Neural. Inf. Process. Syst. **33**, 18637–18647 (2020)
4. He, X., Deng, K., Wang, X., Li, Y., Zhang, Y., Wang, M.: Lightgcn: simplifying and powering graph convolution network for recommendation. In: International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 639–648 (2020)
5. Jang, E., Gu, S., Poole, B.: Categorical reparameterization with gumbel-softmax. arXiv preprint arXiv:1611.01144 (2016)
6. Lin, Y., Liu, Z., Sun, M., Liu, Y., Zhu, X.: Learning entity and relation embeddings for knowledge graph completion. In: AAAI Conference on Artificial Intelligence, vol. 29 (2015)

7. 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: ACM International Conference on Information and Knowledge Management, pp. 417–426 (2018)
8. Wang, X., He, X., Cao, Y., Liu, M., Chua, T.S.: Kgat: knowledge graph attention network for recommendation. In: ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 950–958 (2019)
9. Wang, X., He, X., Wang, M., Feng, F., Chua, T.S.: Neural graph collaborative filtering. In: International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 165–174 (2019)
10. Wang, X., Huang, T., Wang, D., Yuan, Y., Liu, Z., He, X., Chua, T.S.: Learning intents behind interactions with knowledge graph for recommendation. In: Proceedings of the Web Conference 2021, pp. 878–887 (2021)
11. Wang, Z., Lin, G., Tan, H., Chen, Q., Liu, X.: Ckan: collaborative knowledge-aware attentive network for recommender systems. In: International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 219–228 (2020)
12. Wang, Z., Zhang, J., Feng, J., Chen, Z.: Knowledge graph embedding by translating on hyperplanes. In: AAAI Conference on Artificial Intelligence, vol. 28 (2014)
13. Wu, J., et al.: Self-supervised graph learning for recommendation. In: International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 726–735 (2021)
14. Yang, Y., Huang, C., Xia, L., Li, C.: Knowledge graph contrastive learning for recommendation. In: International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1434–1443 (2022)
15. Yin, M., Zhou, M.: Arm: Augment-reinforce-merge gradient for stochastic binary networks. arXiv preprint arXiv:1807.11143 (2018)
16. Yu, J., Yin, H., Xia, X., Chen, T., Cui, L., Nguyen, Q.V.H.: Are graph augmentations necessary? simple graph contrastive learning for recommendation. In: International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1294–1303 (2022)
17. Yu, X., Ren, X., Gu, Q., Sun, Y., Han, J.: Collaborative filtering with entity similarity regularization in heterogeneous information networks. IJCAI HINA **27** (2013)
18. Zhang, F., Yuan, N.J., Lian, D., Xie, X., Ma, W.Y.: Collaborative knowledge base embedding for recommender systems. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 353–362 (2016)