Applied Soft Computing Journal 177 (2025) 113220

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Recommender system based on noise enhancement and multi-view graph contrastive learning

Duantengchuan Li [a,b,c], Jiayao Lu [d], Zhihao Wang [d], Jingxiong Wang [d], Xiaoguang Wang [a,b,c,*], Fobo Shi [a,b,**], Yu Liu [d]

[a] School of Information Management, Wuhan University, Wuhan 430072, China
[b] Intelligent Computing Laboratory for Cultural Heritage, Wuhan University, Wuhan 430072, China
[c] Big Data Institute, Wuhan University, Wuhan 430072, China
[d] School of Computer Science, Wuhan University, Wuhan 430072, China

ARTICLE INFO

Keywords:
Knowledge graph
Recommender system
Contrastive learning
Graph representation learning

ABSTRACT

Graph neural network has become the mainstream model of Knowledge-aware Recommendation because of its ability to capture higher-order information. Contrastive learning, due to its self-supervised learning paradigm, has been successfully employed to alleviate the sparse supervision signal problem in Knowledge-aware Recommendation models based on graph neural network. While graph neural networks and contrastive learning have advanced knowledge-aware recommendation, two critical limitations persist: (1) indistinguishable node representations due to data sparsity exacerbate the long-tail problem, and (2) suboptimal contrastive performance caused by non-uniform node distributions in embedding space. To mitigate the aforementioned issues, we propose the novel recommender system based on Noise Enhancement and Multi-View Graph Contrastive Learning (MNCL), the first framework that jointly optimizes node distribution uniformity and multi-view semantic alignment. Specifically, we inject uncertainty-aware noise into some graphs to achieve topology-invariant uniform representations and construct structural diversity-driven view for contrastive preference disentanglement. Unlike existing GNN-based contrastive methods that rely on heuristic augmentation or single-view alignment, MNCL dynamically regulates distribution entropy via noise intensity and preserves heterogeneous graph semantics through multi-view sampling. Extensive experiments on three public datasets show that our model achieves better results compared with the mainstream methods.

1. Introduction

As an effective information filtering technology, recommendation systems are widely applied in areas such as service recommendation [1], social relationship prediction [2,3], third-party library recommendation [4], and review-based e-commerce recommendation [5,6]. Traditional personalized recommendation algorithms (e.g., collaborative filtering algorithms) rely on users' historical behavioral data (e.g., ratings, clicks, purchases, etc.) to discover similarities among users or items. However, due to the presence of difficulties such as cold start, data scarcity [7], and long-tail dispersion [8,9], these traditional personalized recommendation algorithms are not able to perform well based on limited interaction data. In order to alleviate these problems, many of the recommendation algorithm using neural network modeling ability in high dimensional feature [10] incorporate various auxiliary information to improve the accuracy of recommendation [3,11,12]. Since knowledge graphs (KGs) contain a wealth of external knowledge information [13], they offer a useful method for simulating users and items more precisely [14].

Early recommendation systems based on KG (also known as knowledge-aware recommendation) mainly include recommendation models based on KG feature representation and models that integrate multi-source information. The former primarily focuses on utilizing entity relationships and attribute information in the KG to construct representations. These algorithms typically utilize embedding techniques, such as TransE [15], SHGNet [16], SDFormer [17], ConvHLE [18], etc., to map entities from the KG to low-dimensional vector spaces, after which the similarity between users and items is computed using these vectors. The latter combines information from different data sources to construct richer and more accurate representations with the intention of making recommendations perform better. However, some of the above methods overlook the modeling of global information in

* Corresponding author at: School of Information Management, Wuhan University, Wuhan 430072, China.
** Corresponding author.
E-mail addresses: dtclee1222@whu.edu.cn (D. Li), wxguang@whu.edu.cn (X. Wang), foboshi99@whu.edu.cn (F. Shi).

https://doi.org/10.1016/j.asoc.2025.113220
Received 22 June 2024; Received in revised form 14 April 2025; Accepted 18 April 2025
Available online 12 May 2025

the KG, specifically the higher-order connectivity information among entities [19]. With the emergence and development of graph neural networks (GNNs) [20] and graph contrastive learning [21], their application in recommendation algorithms and KG has become increasingly widespread. GNN-based recommendation models can effectively utilize the graph structure information in KG, thereby making it possible to extract higher-order connectivity information from graph data and produce more precise, tailored recommendations.

However, GNN-based recommendation models face the issue of extremely sparse supervision signals because of the extreme sparsity of interaction data in recommendation systems. The main challenge in mitigating the supervised signal sparsity problem is the lack of training labels. Liu et al. [22] presented a contrastive learning technique that allows model training without the need for explicit labeling. It maximizes the distance between negative samples and minimizes the distance between positive samples, learning discriminative embedding representations from unlabeled sample data. In addition, KHDNN [23] and MKRec [24] enhance knowledge-aware recommendations by leveraging hyperbolic and multi-space learning to model diverse relations, high-order signals, and geometric structures for improved representation learning.

Since contrastive learning does not rely on labeled data, it has a significant advantage in addressing data sparsity issues, leading many researchers to attempt its application in knowledge graph-based recommendation tasks. The KGCL [25] model utilizes contrastive learning methods to extract useful signals for predicting user preferences from the KG, hence lessening the recommendation system's exposure to noisy information. Additionally, Zou et al. proposed the MCCLK [26] model to conduct contrastive learning in a hierarchical manner, taking into account the entity graph's structural information. To alleviate the problem of the dominant user–item view during training, the KACL [27] model makes advantage of contrastive learning to increase the expressive capacity of the model by appending a learnable embedding vector to every relationship in the KG. These recommendation models utilize contrastive learning to mitigate the noise impact from external data in the KG and the problem of sparse interaction data, ultimately improving the learning of user–item vectors. However, existing GNN-based recommendation models integrating contrastive learning still have the following two limitations: Firstly, due to data sparsity, in the process of GNN information aggregation, the learning representations of all items tend to converge towards popular items, resulting in the issue of identical node feature representations degrading [28], which is detrimental to long-tail items. Secondly, the performance of contrastive learning methods is affected by the quality of samples, and the uniform distribution of samples in the embedding space also affects the performance of contrastive learning models [29].

Considering the issues mentioned, we proposes a multi-view noise-enhanced knowledge-aware recommendation method based on graph contrastive learning, named MNCL. Firstly, this article innovatively employs three views, user–item, user–item–entity, and item–entity, for multi-context feature extraction. These various contextual features provide high-quality positive and negative samples for subsequent contrastive learning, significantly enhancing its effectiveness. Secondly, a lightweight Light-GCN is utilized to reduce computational complexity. To address the inherent issue of homogeneity in learned embeddings caused by GCN, which makes user or item representations difficult to distinguish, this article proposes a simple noise augmentation method that effectively mitigates this problem, thereby improving GCN's feature learning capability. Furthermore, this enhances the quality of contrastive learning samples. Finally, the empirical outcomes on various publicly accessible datasets also substantiate the potency of MNCL.[1] The main contributions of our work are concluded as follows:

[1] Source code will be available at https://github.com/dacilab/MNCL.

- We introduces a new recommendation model, MNCL, which utilizes multi-view and noise enhancement to generate high-quality positive and negative samples for contrastive learning, ultimately employing multi-negative instance contrastive learning methods to fully learn information across different views.
- Considering the importance of uniform distribution of samples during the information aggregation process, we propose noise enhancement methods in this process to optimize the uniformity of sample distribution.
- Multifaceted experiments: Numerous experiments were carried out on Movielens, Book-Crossing, and Last.FM. The outcomes demonstrate the dominance of MNCL in representation learning, demonstrating the potency of optimized contrastive learning methods.

In the subsequent chapters, Section 2 will showcase related works relevant to this investigation. Sections 3 and 4 will provide detailed explanations of the components and tasks of the MNCL model. Section 5 will cover experimental settings, results, and analysis, including ablation experiments and hyperparameter experiments. The final two Section will summarize the content of this article and discuss its practical significance.

2. Related work

This section focuses on two main topics: knowledge-aware recommendation systems and contrastive learning-based recommendation techniques.

2.1. Knowledge-aware recommendation

According to the representation forms of KG, recommendations based on KG are mainly divided into those based on KG embedding methods and path-based recommendation methods.

KG embedding methods [15,17,18,34,35] are commonly employ knowledge embeddings as bearers of knowledge for entities and relationships within the KG, which are then utilized directly to enrich the information in recommendation systems, capturing the structural information of the KG. As the respective embedding-based method, DKN [30] uses entity features and title features in news recommendation, specifically using TransD to obtain the embedding of entities and relations in the KG and calculating the attention coefficients of candidate items for users to form the final news representation. Conversely, CKE [31] integrates entity, text, and image features into the item's feature representation, employing collaborative filtering for recommendations.

Path-based recommendation models construct user and item historical interactions into heterogeneous information networks (HINs), where users and items serve as nodes within the network, and the connections among different nodes represent the information about the possible feature between users and items. Path-based recommendation models, which depend on artificially crafted meta-paths, which are used through leveraging the interconnected properties of HINs, face challenges in real-world recommendation scenarios and are difficult to deploy on a large scale [36–38]. With advancements in GNNs, GNN-based recommendation models have become more efficient at leveraging the auxiliary information in KG. The KGCN [39] model focuses on user preferences for relationships within the KG, aggregating multi-hop neighbors of items associated with the KG using GNN-based information aggregation. KGAT [32] constructs a collaborative KG and utilizes GNNs along with attention mechanisms to iteratively propagate across the collaborative KG, mining the latent features of users and items. In order to generate detailed representations of users and items, KGIN [40] iteratively combines neighbors on the entity graph with an attention method. However, GNN-based recommendation methods face the challenge of sparse supervision signals when targeting tail data in long-tail datasets, as constructing accurate user and item representations is difficult with extremely sparse training labels.

Table 1
Comparison of mainstream recommendation methods based on knowledge graph.

Reference	Scenario	Method	Advances	Drawbacks
DKN [30]	News Recommendation	Embeds knowledge graph into news recommendation using CNN	Integrates both semantic and knowledge-based features	Limited scalability, struggles with long-tail data
CKE [31]	Collaborative Filtering	Integrates collaborative filtering with knowledge embeddings	Combines text, image, and structure-based information	Relies heavily on feature engineering
KGAT [32]	General Recommendation	Uses GNN with attention mechanisms	Efficient aggregation of user–item interactions	Performance deteriorates with sparse data
SimGCL [33]	Contrastive Learning for Recommendation	Applies contrastive learning	Reduces bias, improves representation uniformity	High computational complexity
Our Method	Multi-view Recommendation	Uses multi-view contrastive learning with noise augmentation	Enhanced user–item embeddings through multi-view learning and noise augmentation	Computationally expensive in large-scale datasets

2.2. Contrastive learning-based recommendation

Contrastive learning methods derive node representations by contrasting positive and negative instances. The self-supervised graph learning (SGL) [41] model utilizes three distinct graph augmentation strategies: node dropping, edge dropping, and random walks, to create various views of the same node, facilitating deeper exploration of node information. Lin et al. [42] integrate underlying architectural and meaningful neighboring nodes into contrastive learning to obtain the architectural and meaningful information of higher-order neighbors. SEPT [43] suggests an approach that utilizes users' social information to enrich data views, facilitating socially aware contrastive learning. Considering that in the context of recommendation systems, the contrastive loss function is more critical to performance than data augmentation. The XSimGCL model [29] does not rely on graph data augmentation to improve model accuracy but focuses on optimizing the contrastive loss function to reduce bias in representation distribution. MKGCL [44] enhances contrastive learning-based recommendation by integrating knowledge-augmented views and mixed-curvature spaces, effectively addressing data sparsity and improving graph structure scalability for better recommendation accuracy. However, these contrastive learning models overlook the importance of ensuring a uniform distribution of samples in the representation space, which directly impacts the final outcome of the contrastive learning process.

Our method, MNCL, is a significant departure from traditional methods in terms of its approach to learning from multiple views and applying noise augmentation during the graph convolution process. The use of multi-view learning allows for a more comprehensive understanding of the relationships between users, items, and entities. Additionally, the introduction of noise augmentation helps mitigate degradation in the model's performance, ensuring more uniform distribution of node features in the embedding space. This is a distinct innovation compared to other methods which either focus on single view learning or rely on noise addition without sufficient optimization. Table 1 shows the differences between our proposed approach and other related approaches in scenario, method, advances, and drawbacks.

3. Problem formulation

In this section, two basic categories of structured data are introduced: the user–item interaction matrix and the knowledge graph. Subsequently, we describe the tasks for contrastive learning that make use of the KG.

User–item interaction data: In typical recommendation tasks, the sets of users and items are defined as $\mathcal{U} = \{u_1, u_2, \dots, u_M\}$ and $\mathcal{V} = \{v_1, v_2, \dots, v_N\}$, respectively, where M is the number of users and N is the number of items. The user–item interaction matrix $\mathbf{R} \in \mathbb{R}^{M\times N}$ is outlined based on user implicit feedback, where $y_{uv} = 1$ illustrates that a user has interacted with an item, such as selecting or making a purchase; otherwise, $y_{uv} = 0$.

Knowledge Graph: In terms of KG, besides considering historical interactions, information related to items (such as item attributes) is stored in the graph as a heterogeneous KG. Let $\mathcal{G} = \{(h, r, t) | h, t \in \mathcal{E}, r \in \mathcal{R}\}$ represent the KG, where h, t, and r respectively stand for the head, tail, and relation of knowledge triples; $\mathcal{E}$ and $\mathcal{R}$ refer to the sets of entities and relations within $\mathcal{G}$. In many recommendation scenarios, items $v \in \mathcal{V}$ matching entities $e \in \mathcal{E}$ in the KG. By aligning items with entities in the KG, the graph can provide supplemental information for interaction data and descriptions of the items.

Knowledge graph-based contrastive learning recommendation: Given $\mathbf{R}$ and $\mathcal{G}$, the goal is learning a function $\hat{y}_{uv} = f(u, v)$ that aims to predict how likely a user would interact with a specific item.

4. Methodology

This section introduces the Multi-View with Noise-Enhanced Graph Contrastive Learning Knowledge-Aware Recommendation Model (MNCL). Fig. 1 shows the flowchart of the MNCL model. The multi-view mechanism enables the model to capture features from different contexts, providing higher-quality positive and negative samples for contrastive learning. Specifically, it constructs multiple views, including user–item interaction graphs, item–entity views, and user–item–entity views. Various data augmentation strategies are applied to enhance the diversity of contrastive samples. The user–item view optimizes the uniformity of data distribution through noise-enhanced Light-GCN [45], while the user–item–entity view employs random edge and node dropping for structural enrichment. Different graph encoders are then used to learn contextualized node representations. The user–item view utilizes noise-enhanced Light-GCN to mitigate the issue of embedding homogeneity, ensuring more distinguishable user and item embeddings, thereby improving the quality of contrastive samples. The user–item–entity view leverages a Path-aware GNN to capture structured user–item relationships, and the item–entity view employs a Relation-aware GNN to extract richer item representations from the knowledge graph. The following sections will provide a detailed explanation of these components.

4.1. Noise-enhanced light-GCN

For graph network aggregation, this work utilizes the mainstream Light-GCN [45] for node feature aggregation. Light-GCN, known for its straightforward message passing and aggregation mechanisms, requires no feature transformations or nonlinear activations, making it not only efficient but also cost-effective in terms of computation. Node aggregation in this model is primarily executed recursively. At each Lth layer of aggregation, the process can be articulated as follows:

$$\mathbf{e}_u^{(l+1)} = \sum_{i\in\mathcal{N}_u} \frac{1}{\sqrt{|\mathcal{N}_u|\,|\mathcal{N}_i|}} \mathbf{e}_i^{(l)}, \tag{1}$$

$$\mathbf{e}_i^{(l+1)} = \sum_{u\in\mathcal{N}_i} \frac{1}{\sqrt{|\mathcal{N}_u|\,|\mathcal{N}_i|}} \mathbf{e}_u^{(l)}, \tag{2}$$

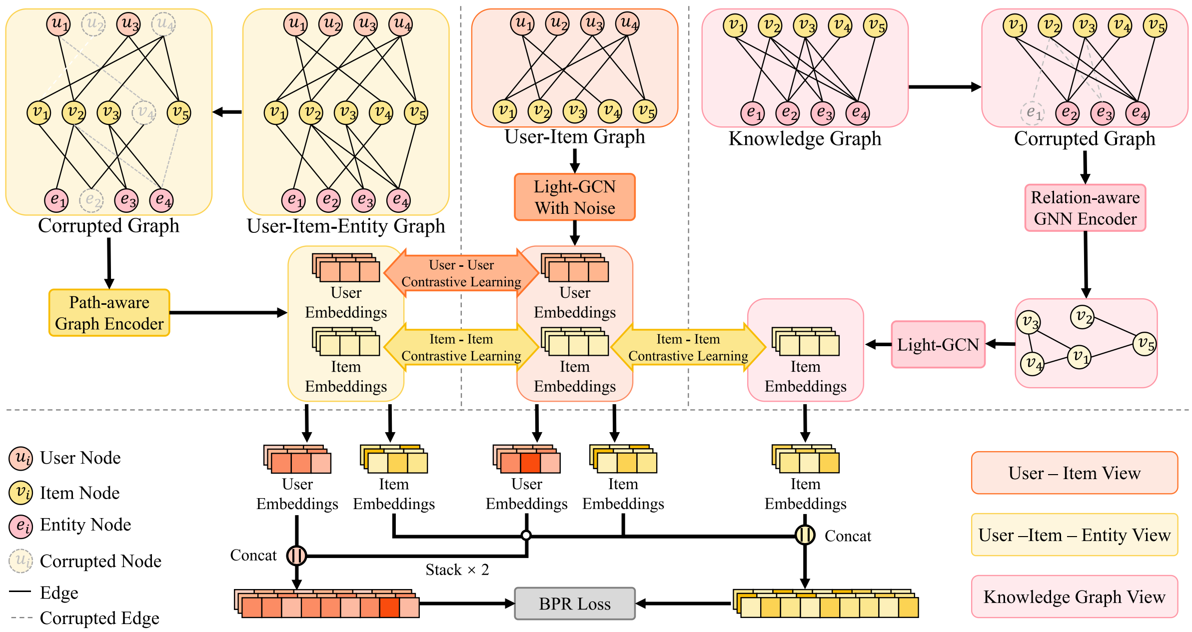

Fig. 1. MNCL model diagram.

where $\mathbf{e}_u^{(l)}$ and $\mathbf{e}_i^{(l)}$ indicate the embeddings of user u and item i at the lth layer, while $\mathcal{N}_u$ and $\mathcal{N}_i$ denote the neighbors of user u and item i, respectively. For the convenience of implementation of Light-GCN, we provide the matrix form of Eqs. (1) and (2) as follows:

$$\mathbf{e}^{(l+1)} = \widetilde{\mathbf{A}}\mathbf{e}^{(l)}, \tag{3}$$

where $\widetilde{\mathbf{A}}$ is the normalized sparse graph adjacency matrix, $\widetilde{\mathbf{A}} = (\mathbf{D})^{-\frac{1}{2}}\mathbf{A}(\mathbf{D})^{-\frac{1}{2}}$, $\mathbf{A} = \begin{pmatrix} 0 & \mathbf{R} \\ \mathbf{R}^{\mathrm{T}} & 0 \end{pmatrix}$, and $\mathbf{R} \in \mathbb{R}^{M\times N}$ is the matrix of user interactions. After L layers Light-GCN, the embeddings from various layers are summed to produce the final feature vectors $\mathbf{e}_u$ and $\mathbf{e}_i$, as shown below:

$$\mathbf{e}_u = \sum_{l=0}^{L} a_l \mathbf{e}_u^{(l)}, \tag{4}$$

$$\mathbf{e}_i = \sum_{l=0}^{L} a_l \mathbf{e}_i^{(l)}, \tag{5}$$

where a_l represents the weights of the vector representations across different layers, empirically set to $\frac{1}{L}+1$ according to Ref. [45]. To mitigate the degradation issue caused by multi-layer aggregation and to increase sample diversity, noise enhancement φ is introduced in each layer of Light-GCN message aggregation. The expression for φ at the layer l has the following expression:

$$\varphi = \mathrm{sign}\left(\mathbf{e}^{(l)}\right) \odot \gamma \odot \beta, \tag{6}$$

where $\mathbf{e}_{(l)}$ indicates the input vector matrix at the layer l. γ is a normalized random matrix with the same dimensions as $\mathbf{e}_{(l)}$, and β is the noise enhancement coefficient. After adding noise enhancement φ, the Light-GCN aggregation process at the lth layer can be described as below:

$$\mathbf{e}^{(l+1)} = \widetilde{\mathbf{A}}\mathbf{e}^{(l)} + \varphi. \tag{7}$$

By applying simple noise enhancement, the issue of overly homogeneous distributions caused by GCN can be effectively mitigated, thereby improving feature distinguishability and quality. After summing and averaging the matrix representations aggregated over L layers of Light-GCN, the following is how the final embedding is obtained:

$$\begin{aligned}\mathbf{e} &= \frac{\mathbf{e}^{(0)} + \mathbf{e}^{(1)} + \cdots + \mathbf{e}^{(L)}}{L} \\ &= \frac{\mathbf{e}^{(0)} + \left(\widetilde{\mathbf{A}}\mathbf{e}^{(0)} + \varphi\right) + \cdots + \left(\widetilde{\mathbf{A}}^{L}\mathbf{e}^{(0)} + \widetilde{\mathbf{A}}^{L-1}\varphi + \cdots + \widetilde{\mathbf{A}}^{2}\mathbf{e}^{(0)} + \widetilde{\mathbf{A}}\varphi + \varphi\right)}{L}.\end{aligned} \tag{8}$$

The final feature vector $\mathbf{e}$ encompasses the ultimate user feature vector $\mathbf{e}_u^s$ from the user–item view and the final item feature vector $\mathbf{e}_i^s$.

4.2. Relation-aware GNN

KG not only include information about entities but also possess extensive relational information between these entities. To fully leverage the auxiliary information from KG and address the data sparsity in the original interaction graph, this chapter introduces a relation-aware GNN module. This model, in its message aggregation process, maintains the relational information between these entities in addition to aggregating data from nearby entities. Specifically, the model recursively learns item representations from the KG $\mathcal{G}$ over K iterations, where the formula for the kth iteration of relation-aware aggregation is described below:

$$\mathbf{e}_i^{(k+1)} = \frac{1}{|\mathcal{N}_i|} \sum_{(r,j)\in\mathcal{N}_i} \mathbf{e}_r \odot \mathbf{e}_j^{(k)}, \tag{9}$$

$$\mathbf{e}_i^{(k+1)} = \frac{1}{|\mathcal{N}_j|} \left(\sum_{(r,j)\in\mathcal{N}_i} \mathbf{e}_r \odot \mathbf{e}_j^{(k)} + \sum_{(r,i)\in\mathcal{N}_j} \mathbf{e}_r \odot \mathbf{e}_i^{(k)} \right), \tag{10}$$

where $\mathbf{e}_i^{(k)}$ and $\mathbf{e}_j^{(k)}$ respectively represent the representations of items i and entities j, which store relational signals propagated from their $(k-1)$ hop neighbors. After the item–entity view undergoes relation-aware GNN aggregation, it employs Light-GCN for kth order aggregation to enhance item representations. The message passing and aggregation process at the kth level in each layer can be explained below:

$$\mathbf{e}_i^{(k+1)} = \sum_{j\in\mathcal{N}(i)} \tilde{\mathbf{S}}\mathbf{e}_i^{(k)}, \tag{11}$$

where $\mathcal{N}(i)$ represents neighboring items, $\tilde{\mathbf{S}}$ is the normalized sparse graph adjacency matrix within the formula, and $\mathbf{e}_i^{(k)}$ indicates the

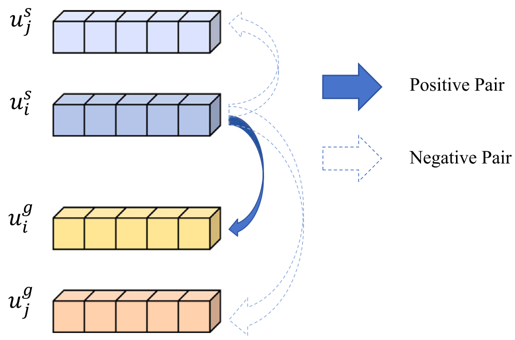

Fig. 2. Multi-negative example Contrastive Learning.

representation of item after the kth message aggregation in each layer of Light-GCN. Subsequently, by summing the representations of item i from the k aggregation operations, the final feature vector $\mathbf{e}_i^g$ of the item–entity view is derived.

KG often contain excessive noise interference; adding noise enhancement could introduce additional disruptive information, potentially reducing the model's performance. This is also evident in how adding noise enhancement in the user–item view can impact the final outcome of message aggregation. However, appropriate levels of noise enhancement can optimize the uniform distribution of samples.

4.3. Path-aware GNN

Considering that solely using relation-aware GNNs for extracting information from knowledge graphs leads to knowledge modeling during training, but research indicates low utilization of this information during training, therefore, this study adopts a path-aware GNN [26], which learns and aggregates information from neighbor nodes h times while also preserving path information, specifically the user–item–entity connections. The aggregation process for each hth iteration at every layer can be represented as follows:

$$\mathbf{e}_u^{(h+1)} = \frac{1}{|\mathcal{N}_u|} \sum_{i \in \mathcal{N}_u} \mathbf{e}_i^{(h)}, \tag{12}$$

$$\mathbf{e}_i^{(h+1)} = \frac{1}{|\mathcal{N}_i|} \sum_{(r,u) \in \mathcal{N}_i} \delta(i,r,u)\mathbf{e}_r \odot \mathbf{e}_j^{(h)}, \tag{13}$$

where $\mathbf{e}_i^{(h)}$ and $\mathbf{e}_j^{(h)}$ respectively represent the representations of items and entities, storing relational signals propagated from $(h-1)$ neighbor aggregations, thus capturing global information from multi-hop paths. To apply weights to each relation and entity, attention weights $\delta(i,r,u)$ are calculated in the manner described below:

$$\begin{aligned}\delta(i,r,u) = &\ \mathrm{softmax}\left(\left(\mathbf{e}_i \parallel \mathbf{e}_r\right)^T \cdot \left(\mathbf{e}_u \parallel \mathbf{e}_r\right)\right) \\ = &\ \frac{\exp\left(\left(\mathbf{e}_i \parallel \mathbf{e}_r\right)^T \cdot \left(\mathbf{e}_u \parallel \mathbf{e}_r\right)\right)}{\sum_{(u,r)\in\widehat{\mathcal{N}}(i)} \exp\left(\left(\mathbf{e}_i \parallel \mathbf{e}_r\right)^T \cdot \left(\mathbf{e}_u \parallel \mathbf{e}_r\right)\right)},\end{aligned} \tag{14}$$

where $\parallel$ indicates concatenate operation, $\widehat{\mathcal{N}}(i)$ denotes the set of neighboring entities $\mathcal{N}(i)$ and item i itself. Then all layers' representations are sum up to obtain the global representations $\mathbf{e}_u^{(m)}$ and $\mathbf{e}_i^{(m)}$.

4.4. Multi-negative sample contrastive learning

The aforementioned process results in the embedding vectors for both the user–item view and the item–entity view. In the user–item view, the final feature vector for the user is $\mathbf{e}_u^{(s)}$, and for the item is $\mathbf{e}_i^{(s)}$. In the item–entity view, the item's final feature vector is $\mathbf{e}_i^{(g)}$, and in the user–item–entity view, the final feature vector of the user is $\mathbf{e}_u^{(m)}$, and the final feature vector of the item is $\mathbf{e}_i^{(m)}$. All vectors above are then mapped to a space for computing contrastive loss, where positive and negative samples are subsequently defined. As demonstrated in Fig. 2, for any user node $\mathbf{u}_i^{(s)}$ in view S, its positive sample for comparison is the same node embedding $\mathbf{u}_i^{(g)}$ learned in another view G. In both views, embeddings of nodes other than $\mathbf{u}_i^{(s)}$, specifically $\mathbf{u}_j^{(s)}$ and $\mathbf{u}_j^{(g)}$, are considered negative samples. It is posited in this study that the weights of negative samples differ between views, with the weight in the original view defined as ω. Having defined the positive and negative instances, this chapter delineates the contrastive loss into global contrastive loss $\mathcal{L}^G$ and KG contrastive loss $\mathcal{L}^{KG}$, illustrated by the items's loss, with the specific formula as follows:

$$\mathcal{L}_i^G = -\log \frac{\exp\left(s\left(\mathbf{e}_i^s, \mathbf{e}_i^m\right)/\tau\right)}{\exp\left(s\left(\mathbf{e}_i^s, \mathbf{e}_i^m\right)/\tau\right) + \omega \sum_{i \neq j} \exp\left(s\left(\mathbf{e}_i^s, \mathbf{e}_j^s\right)/\tau\right) + \sum_{i \neq j} \exp\left(s\left(\mathbf{e}_i^s, \mathbf{e}_j^m\right)/\tau\right)}, \tag{15}$$

$$\mathcal{L}_i^{KG} = -\log \frac{\exp\left(s\left(\mathbf{e}_i^s, \mathbf{e}_i^g\right)/\tau\right)}{\exp\left(s\left(\mathbf{e}_i^s, \mathbf{e}_i^g\right)/\tau\right) + \omega \sum_{i \neq j} \exp\left(s\left(\mathbf{e}_i^s, \mathbf{e}_j^s\right)/\tau\right) + \sum_{i \neq j} \exp\left(s\left(\mathbf{e}_i^s, \mathbf{e}_j^g\right)/\tau\right)}, \tag{16}$$

where τ represents the temperature coefficient. $s(\cdot)$ denotes the calculation of cosine similarity. j represents nodes other than i. The contrastive loss $\mathcal{L}_u^G$ for computing user feature vectors is similar to that of $\mathcal{L}_i^G$, simply replace i with u in the formula. The overall contrastive loss is:

$$\mathcal{L}_{CL} = \mathcal{L}_i^G + \mathcal{L}_u^G + \mathcal{L}_i^{KG}. \tag{17}$$

4.5. Model prediction

After multi-layer aggregation and contrastive learning optimization of the user–item view, item–entity view, and user–item-relationship view, multiple embeddings for users, $\mathbf{e}_u^{(s)}$ and $\mathbf{e}_u^{(m)}$, and multiple embeddings for items, $\mathbf{e}_i^{(s)}$, $\mathbf{e}_i^{(m)}$, and $\mathbf{e}_i^{(g)}$, are obtained. These optimized feature vectors are then combined to generate the ultimate feature vectors for users and items, and the final prediction scores are obtained through a prediction function, as shown in the calculation process below:

$$\mathbf{e}_u = \mathbf{e}_u^s \parallel \mathbf{e}_u^s \parallel \mathbf{e}_u^m, \tag{18}$$

$$\mathbf{e}_i = \mathbf{e}_i^s \parallel \mathbf{e}_i^m \parallel \mathbf{e}_i^g, \tag{19}$$

$$\hat{y}(u,i) = \mathbf{e}_u^{\mathrm{T}} \mathbf{e}_i. \tag{20}$$

The multi-task training approach is what we use to optimize the overall model. In order to reconstruct historical data and encourage higher prediction scores for user's historical items than for unobserved items, therefore, we uses BPR loss to build the loss function:

$$\mathcal{L}_{BPR} = \sum -\log \sigma\left(\hat{y}\left(u, i^+\right) - \hat{y}\left(u, i^-\right)\right), \tag{21}$$

where i^+ represents the items that have interacted with user u observed during training, and i^- represents the items that were not observed during training. By integrating BPR loss in contrastive loss, the ultimate goal function can be expressed as follow:

$$\mathcal{L}_{MNCL} = \mathcal{L}_{BPR} + \alpha\mathcal{L}_{CL} + \lambda\|\Theta\|_2^2, \tag{22}$$

where Θ represents the model parameters, while α and λ are hyperparameters employed to counterbalance the L_2 regularization terms and contrastive loss.

5. Experimental settings

5.1. Experimental datasets

The datasets include the MovieLens-1M, the Book-Crossing, and the Last.FM, detailed as follows:

(1) **MovieLens-1M**: This dataset contains 6036 users and 2445 items, with users rating items from 1 to 5.

Table 2
Basic statistical data of three public datasets.

	Movilens-1M	Book-Crossing	Last.FM
# Users	6036	17,860	1872
# Items	2445	14,967	3846
# Interactions	753,772	139,746	42,364
# Entities	182,011	77,903	9366
# Relations	12	25	60
# Triples	1241,996	151,500	15,518

(2) **Book-Crossing**: Originating from the Book Crossing community, this dataset features real ratings by 17,860 users for approximately 15,000 books, with ratings ranging from 0 to 10.
(3) **Last.FM**: This dataset comprises listening information for 3846 songs by 1872 listeners from the Last.FM community, including ratings for music and data on user and music attributes.

Since interactions in MovieLens-1M, Book-Crossing, and Last.FM involve explicit feedback, this chapter adopts the RippleNet method to convert them into implicit feedback. Here, a label of 1 denotes positive user feedback for an item, while a label of 0 denotes negative feedback. For MovieLens-1M, a rating threshold of 4 is set, such that if a user's rating for an item exceeds 4, the sample label is assigned a 1. Considering the sparse data of Last.FM and Book-Crossing, no thresholds be set. Any interaction record (rating behavior) receives a label of 1, while no interaction results in a label of 0. Regarding negative instances, for user, we randomly selects items they have not viewed, with the quantity matching their number of positive samples. This chapter employs FreeBase to construct a KG. Initially, items' IDs in the dataset are paired with IDs in FreeBase, followed by filtering sub-knowledge graphs related to these items from the knowledge base. During the filtering process, items involving multiple entities or those that cannot be matched to any entity are excluded. Subsequently, the items' IDs are matched with the IDs of the head entities in triples within the subgraph, and corresponded triples are chosen. The statistical data for above datasets are concluded in Table 2.

5.2. Evaluation metrics

The following scenario was used to evaluate our model: for click-through rate (CTR) prediction, the trained model is applied to forecast every interaction in the test set. Two commonly used measures are used to evaluate the effectiveness of CTR prediction: F1 and AUC.

AUC stands for area under the ROC curve, which displays the true positive rate (TPR) against the false positive rate (FPR). A larger AUC value indicates superior performance of the model, ranging between 0 and 1. AUC evaluates the classifier's ability to rank samples overall and is effective in handling class imbalance issues.

Recall refers to the proportion of true positives correctly identified by the classifier among all actual positive instances. It is also known as sensitivity. Recall values vary from 0 to 1, with higher values reflecting a better capacity of the model to detect positives. The formula for Recall is as follows:

$$\mathrm{Recall} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}, \tag{23}$$

where TP (True Positives) denotes correctly predicting positive instances as positive, FN (False Negatives) are erroneously predicting positive instances as negative, and Recall represents the proportion of correctly predicted positives in the sample.

Precision is a metric used to assess the correctness of positive predictions made by a classification model. It shows the percentage of true positive samples that are anticipated to be positive. Following is the formula:

$$\mathrm{Precision} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}, \tag{24}$$

where FP (False Positives) indicates predicting negative instances as positive, Precision represents the proportion of truly positive samples among those predicted as positive. The F1-score evaluation metric comprehensively considers the results of Precision and Recall to calculate a measure score, mathematically representing the harmonic mean of both, as shown in the formula below:

$$\mathrm{F}_1 = \frac{2\mathrm{PR}}{\mathrm{P}+\mathrm{R}}, \tag{25}$$

where P and R stand for Precision and Recall, respectively.

5.3. Experimental model comparison

5.3.1. Comparison method

To validate the MNCL model, this work selects 10 models for comparative analysis. These models include recommendation models based on collaborative filtering algorithms(BPRMF), recommendation models based on knowledge graphs(CKE, RippleNet, PER), recommendation models based on GNNs(KGCN, KGNN-LS, KGAT, KGIN), and the latest knowledge-aware recommendation models using contrastive learning methods(MCCLK). (1)~(12) are the comparison methods, and (13)~(16) are the method and variants proposed in this work.

(1) **BPRMF** [46]: It is a conventional approach based on Collaborative Filtering (CF) that only considers interactions between users and items, without involving KG.
(2) **PER** [36]: A common path-based approach that represents connections between users and items using meta-path-based attributes.
(3) **CKE** [31]: This algorithm integrates collaborative filtering with text, structured, and visual information within a recommendation framework.
(4) **RippleNet** [47]: Incorporates KG into the recommendation algorithm, using preference propagation methods to automatically model user interest features.
(5) **KGCN** [39]: Applies graph convolutional networks to extract high-level features in KG and distinguishes the importance of different entity features.
(6) **KGNN – LS** [19]: Utilizes label propagation algorithms to introduce new supervisory signals as a loss function to prevent overfitting.
(7) **KGAT** [32]: An approach that integrates neighbors on the user–item–entity graph with an attention mechanism to derive user/item embeddings based on GNN.
(8) **KGIN** [40]: The model represents user intent graphs by using interactions between users and items, enhancing recommendation performance by mining the semantics of long-range connections in the graph and capturing the interaction features between users and items.
(9) **MCCLK** [26]: A sophisticated GNN-based method that performs contrastive learning in a layered manner, fully keeping in mind the user–item–entity graph's architectural features.
(10) **Multi – Rec** [48]: Utilizing graph recommendation models by integrating hybrid curvature manifolds and graph convolutional networks effectively leverages the geometric structure of KG to enhance recommendation performance.
(11) **CurvRec** [49]: The knowledge-aware recommendation model, by integrating hybrid curvature manifolds and graph convolutional networks, effectively leverages the geometric structure of KG to enhance recommendation performance.
(12) **KGIE** [50]: A model founded on interaction embeddings in graph neural networks integrates knowledge graphs and user–item interaction matrices to optimize neighborhood aggregation methods, aiming to improve the precision and context sensitivity of recommendation systems.

Table 3
Parameter settings for the MNCL model.

Dataset	α	K	L	β	ω
MovieLens-1M	0.1	2	4	1.5	0.7
Book-Crossing	0.1	2	4	1.5	0.8
Last.FM	0.1	2	4	1.5	0.8

(13) **SimGCL** [33]: SimGCL generates contrast views by adding noise to embedded vectors without graph enhancement, simplifying the model structure and improving recommendations by learning a more uniform representation.

(14) **XSimGCL** [29]: XSimGCL maintains the same input and adjacency matrix as the SimGCL method, but changes the contrast learning of the last layer to cross-layer contrast, making the model more lightweight while still performing better than SimGCL.

(15) **MCL**: The MCL model is a multi-view contrastive learning knowledge-aware recommendation model without the Path-aware GNN module.

(16) **MNCL**$^{N}_{w/o}$: Variants of the MNCL. To investigate the impact of noise enhancement, a variant without noise enhancement has been designed.

(17) **MNCL**$^{G}_{w/o}$: Variants of the MNCL. To investigate how multi-view affects the accuracy of models, we designed a variant model that removes the user–item–entity view module.

(18) **MNCL**: The model presented in this article thoroughly exploits features and structural information across three distinct views. Noise augmentation is applied within the graph convolutional neural network to enhance the learning of discriminative embedding representations.

5.3.2. Parameter settings and complexity analysis

In the experimental section, MNCL and all baseline models were implemented using PyTorch, with careful adjustment of key parameters. The embedding size of all models in this chapter is fixed to 64, and the Xavier approach is implemented to initialize the embedding settings. Adam optimizer is utilized in this chapter to optimize the algorithm, with a batch size set to 4096. Grid search was used to verify the ideal parameter configurations, adjusting the learning rate η_1 among $\{0.001, 0.003, 0.01, 0.03\}$ and the L_2 regularization term η_2 between $\{10^{-7}, 10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}\}$. Additional hyperparameter configurations are offered in Table 3.

α: weight of the contrastive loss, K: number of layers in the item–entity view Light-GCN, L: number of layers in the noise-enhanced Light-GCN, β: optimization parameter, ω: weight of the original view.

The complexity calculation of the MNCL algorithm involves several key components. Firstly, the complexity of the GCN is $O(L \times |E| \times D)$, where L is the number of GCN layers, $|E|$ is the number of non-zero elements in the user–item interaction matrix, and D is the embedding dimension of each node. Secondly, the complexity of the contrastive learning loss is $O(N^2 + N \times D)$, where N is the number of nodes, and D is the dimension of each node. The complexity of the BPR loss is $O(B \times D)$, where B is the batch size during training. In summary, the total complexity of the MNCL algorithm is $O(L \times |E| \times D + N^2 + N \times D + B \times D)$.

5.4. Experimental results and analysis

5.4.1. Comprehensive comparison of experimental results

Table 4 displays our comparison of the MNCL model's results on benchmark datasets with the previously described comparable experimental models. To guarantee the validity of the experimental findings, we conducted multiple independent randomized trials for all models and ultimately reported the best-performing results.

Table 4
Performance comparison of all methods on three public datasets.

Model	MovieLens-1M		Book-Crossing		Last.FM	
	AUC	F1	AUC	F1	AUC	F1
BPRMF [46]	0.8923	0.7894	0.6586	0.6117	0.7562	0.7014
LibFM [51]	0.9015	0.8213	0.6615	0.6182	0.7759	0.7103
PER [36]	0.7126	0.6670	0.6052	0.5732	0.6408	0.6042
CKE [31]	0.9072	0.8026	0.6752	0.6098	0.7423	0.6715
RippleNet [47]	0.9195	0.8420	0.7209	0.6472	0.7765	0.7021
KGCN [39]	0.9095	0.8375	0.6842	0.6321	0.8025	0.7095
KGNN-LS [19]	0.9152	0.8413	0.6765	0.6269	0.8053	0.7219
KGAT [32]	0.9146	0.8433	0.7318	0.6524	0.8148	0.7426
KGIN [40]	0.9162	0.8435	0.7264	0.6608	0.8492	0.7608
MCCLK [26]	0.9351	0.8631	0.7625	0.6777	0.8763	0.7964
Multi-Rec [48]	0.9326	0.8598	0.7546	0.6812	0.8465	0.7554
CurvRec [49]	0.9349	0.8619	0.7607	0.6726	0.8530	0.7658
KGIE [50]	0.9338	0.8621	0.7590	0.6736	0.8682	0.7925
SimGCL [33]	0.9234	0.8489	0.6462	0.5901	0.9213	0.7975
XSimGCL [29]	0.9167	0.8406	0.6606	0.5942	0.9230	0.8041
MCL (*ours*)	0.9360	0.8632	0.7629	0.6781	0.8751	0.7934
MNCL (*ours*)	**0.9396**	**0.8700**	**0.7736**	**0.6892**	**0.8951**	**0.8095**

Table 5
The impact of MNCL and its ablation variants.

Model	MovieLens-1M		Book-Crossing		Last.FM	
	AUC	F1	AUC	F1	AUC	F1
MNCL$^{N}_{w/o}$	0.9351	0.8655	0.7624	0.6798	0.8837	0.7992
MNCL$^{G}_{w/o}$	0.9367	0.8659	0.7656	0.6813	0.8829	0.7986
MNCL	**0.9396**	**0.8700**	**0.7736**	**0.6892**	**0.8951**	**0.8095**

Comparing CKE and RMF, it is known that the importance of introducing KG as auxiliary features. After integrating the auxiliary features embedded in the KG into MF, the performance of CKE is superior to MF, whether it is on the sparse MovieLens-1M or the dense Last.FM dataset, which proves that introducing KG as auxiliary information can effectively improve the recommendation performance. Furthermore, comparing KGCN and KGAT with CKE, it can be found that GNN has strong node representation capability. Most GNN-based methods perform better than methods based on knowledge graph embedding (CKE) and path-based methods (such as PER), indicating that GNN can enhance the ability of high-order information modeling of graph nodes.

Comparing the dual-view model MCL with the triple-view model MNCL, it is evident that the addition of appropriate views allows for further exploration of user and item embeddings. Simultaneously, learning more comprehensive and richer samples from different perspectives for contrastive learning can help exploring the potential of contrastive learning. In addition, compared with the graph contrast learning recommendation methods SimGCL and XSimGCL based on fusion noise, this work further improves the accuracy of recommendation due to the use of three views and multiple negative cases contrast learning methods.

On all three datasets, the MNCL model presented in this article has good performance. Compared to the latest MCCLK model, it performs better on all three datasets. Experimental results prove that adding a certain degree of noise enhancement during the aggregation process of graph neural networks does not reduce the model's performance; instead, it improves it, proving the effectiveness of noise enhancement, which effectively alleviates the problem of node degradation. Furthermore, optimizing the uniform distribution of nodes in the embedding space is beneficial to the performance of contrastive learning, thereby learning more accurate node representations, and also alleviating the long-tail problem, thereby enhancing the effectiveness of the model.

5.4.2. Ablation studies

To verify the influence of each module, this section conducts ablation experiments. It compares several variants of the MNCL model and

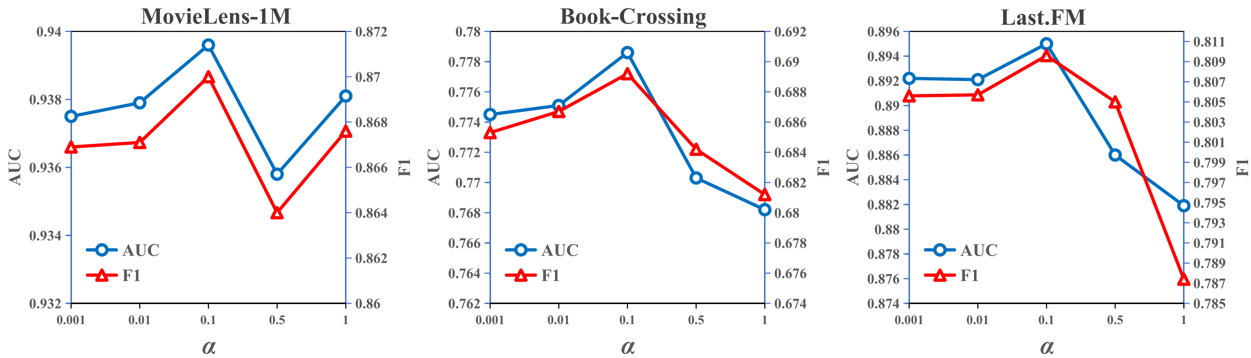

Fig. 3. Impact of Contrastive Learning Weight α on model performance.

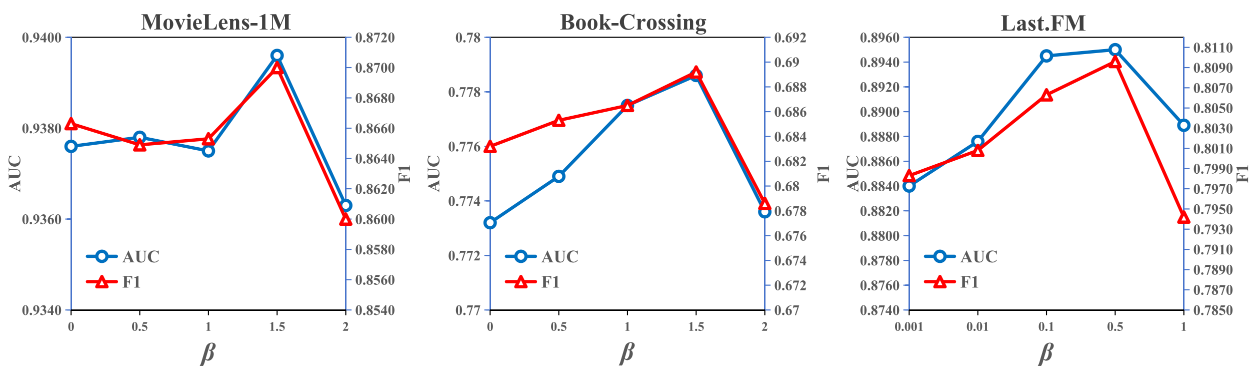

Fig. 4. The influence of noise enhancement coefficient on model performance.

analyzes the effects of these modules. $\mathbf{MNCL}_{w/o}^{N}$ denotes replacing the noise-enhanced Light-GCN module with a regular Light-GCN module. $\mathbf{MNCL}_{w/o}^{G}$ denotes removing the global view to evaluate the impact of multiple views. The outcomes are listed in Table 5.

The ablation experiment outcomes in Table 5 point out that various modules contribute to the MNCL model. The following conclusions can be drawn: Removing noise enhancement results in a noticeable decline in model recommendation performance, particularly evident in sparse datasets like MovieLens-1M and Book-Crossing. The severe node degradation issue after message aggregation on sparse datasets underscores the efficacy of our suggested data augmentation approach in mitigating degradation caused by indistinguishable nodes. Optimizing node uniformity in the embedding space benefits long-tail items, thus enhancing recommendation accuracy. The decline in model performance upon removing the user–item–entity view module underscores the contribution of each view to more accurate recommendations. Users and items can enrich the final vector representation by learning different perspectives of information from diverse views. Moreover, a comprehensive and diverse sample set facilitates the contrastive learning ability. In addition, by injecting noise into the node embeddings, we increase the diversity of the node representations, which helps differentiate between items, especially in sparse scenarios. This process ensures that the model does not produce overly similar embeddings for all nodes, even those that are rarely interacted with, which is a common issue in alleviate the long-tail problem.

5.4.3. Hyperparameter analysis

(a) Sensitivity Analysis of Contrastive Learning Weight α. This chapter employs a multi-task training strategy to train the entire model. This section describes a parameter tuning experiment based on the contrastive learning weight ratio α, which varies in $\{0.001, 0.01, 0.1, 0.5, 1\}$, to explore the impact of contrastive learning in the multi-task learning process. The experimental results are depicted in Fig. 3. Across three public benchmark datasets, the model performs best when is set to 0.1; increasing further leads to a sharp decrease in performance, suggesting that prioritizing the recommendation task in a multi-task environment enhances the model's effectiveness.

(b) Sensitivity analysis of noise enhancement coefficient β. Although noise enhancement can alleviate the long-tail problem, excessively strong or weak noise may adversely affect the feature extraction of GNN. To investigate the impact of the noise enhancement coefficient β on model performance, this section varies β between [0, 0.1, 0.5, 1, 1.5, 2]. Experimental outcomes are displayed in Fig. 4, indicating that the MNCL performs best on the MovieLens-1M, Book-Crossing, and Last.FM datasets when $\beta = 1.5$ With the increase of β, significant performance improvements are observed on both the dense Last.FM dataset and the extremely sparse MovieLens-1M dataset. The model obtains optimal performance when β reaches 1.5, demonstrating the generalizability of the data augmentation approach. However, with continuous growth of β, the performance significantly declines. It may be because larger values of β introduce excessive noise interference, affecting the final recommendation performance. This also explains why we did not incorporate noise enhancement into GNNs in the KG to learn the final representations. Since there is already considerable noise in the KG, adopting noise enhancement would instead impede the learning of the final vector of items in the KG.

(c) Sensitivity analysis of the number of layers L in noise-enhanced Light-GCN. For the user–item view in noise-enhanced Light-GCN, we conducted a sensitivity analysis of the number of layers L and its effect on the efficiency of the model. This section adjusted the range of L in [1, 2, 3, 4, 5]. Fig. 5 displays the outcomes of the experiment. It demonstrates that the model obtains optimal results on the three public datasets when $L = 4$. Owing to the extremely low amount of data on user–item interactions, learning more accurate embedding representations often requires deeper graph neural networks, especially after adding noise enhancement, where node distributions become more uniform, highlighting the advantages of deeper networks. However, the model performs considerably worse after more than four network levels. The analysis suggests that this result may be due to the increasing

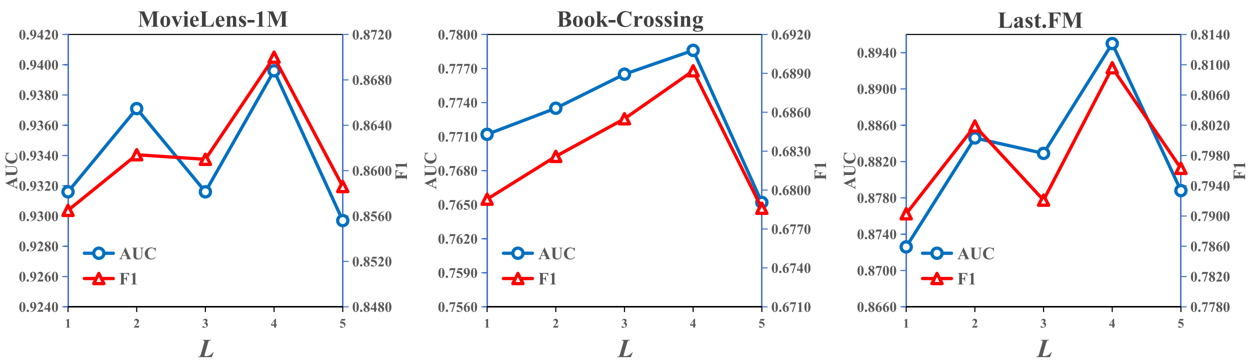

Fig. 5. The influence of the number of layers in noise-enhanced Light-GCN on model performance.

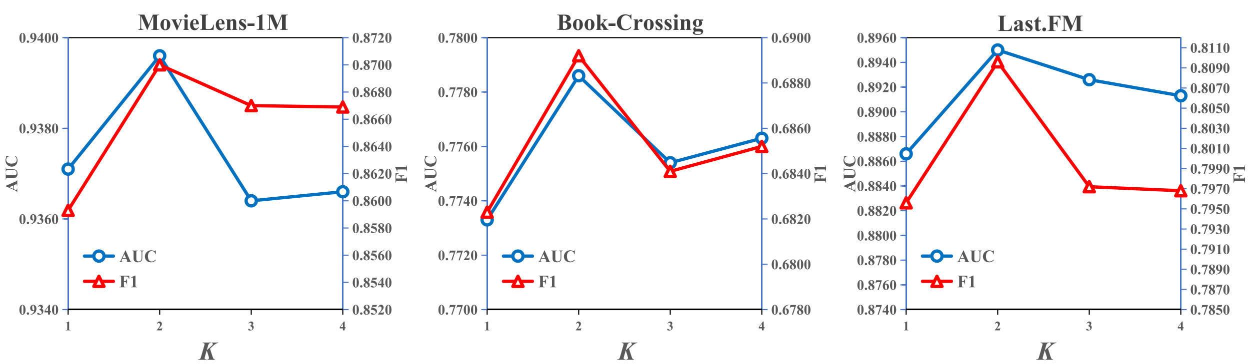

Fig. 6. The influence of the number of layers in the item–entity view Light-GCN on model performance.

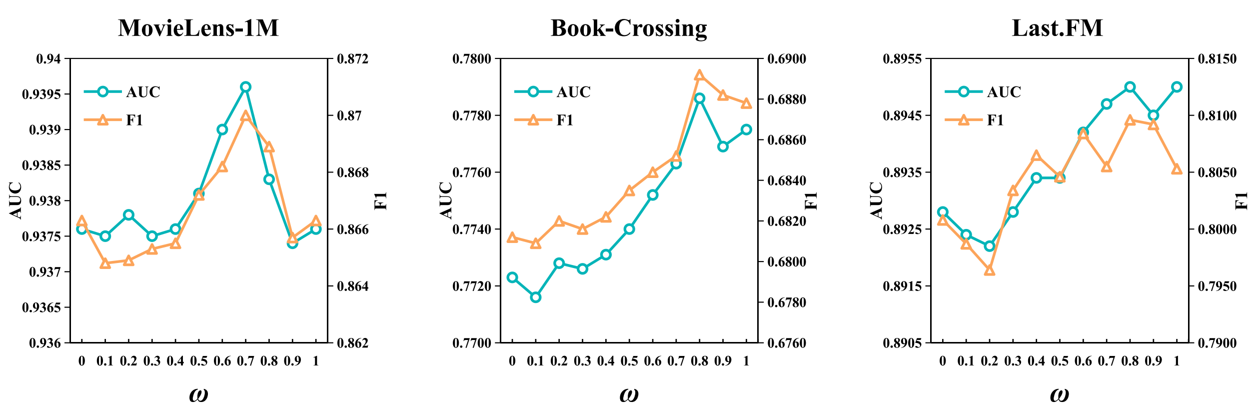

Fig. 7. The influence of the weight ω for contrastive learning in the original view on model performance.

difficulty in distinguishing node representations as embedding vectors delve deeper into multi-layer graph neural networks, consequently affecting node representation learning and reducing model prediction accuracy. The above analysis reflects the ongoing degradation issue in GNN-based recommendation models. Our proposed noise enhancement method effectively alleviates this problem under appropriate circumstances.

(d) Sensitivity analysis of the number of layers K in the item-view Light-GCN. Fig. 6 illustrates the effect of the number of layers K in the item–entity view module of Light-GCN. The model performs best on all three datasets when $K = 2$. Hence, it can be inferred that one to two hops are sufficient for integrating neighbor information in this view. As the number of layers in the Light-GCN increases in the item–entity view, performance noticeably declines. The primary reason is the presence of a significant amount of noisy information in the KG, which is encoded into the embedding vectors, thereby affecting the final rating prediction. Hence, we did not consider adding noise enhancement to Light-GCN in the item–entity view, as this might cause degradation of performance.

(e) Sensitivity analysis of the weight ω for contrastive learning in the original view. We proposes a multi-negative-instance contrastive learning method and analyzes the impact of negative instance weights in the original view on the final performance of contrastive learning. Fig. 7 shows that the contrastive learning performance is optimal at $\omega = 0.8$ on the Last.FM dataset and $\omega = 0.7$ on the MovieLens-1M and Book-Crossing datasets. The weight explains the influence of samples from different views on the performance of contrastive learning. In classical contrastive learning methods, most models only utilize negative samples from the contrastive view, often overlooking other potential information in the positive sample view. Our proposed multi-negative-instance contrastive learning method aligns with the idea of multi-view, both aiming to learn more comprehensive and richer user and item vector representations from different perspectives, thereby enhancing recommendation accuracy.

6. Conclusion and future work

This work emphasizes the importance of applying contrastive learning in recommendation models and effectively addresses the issue of indistinct node differentiation due to sparse interaction data. However, real-world recommendation scenarios are complex, and the research methods may not necessarily be applicable. Our proposed MNCL model requires constructing different views, which is computationally costly, and in real-world scenarios, time is also a criterion for evaluating models. To enhance the applicability and robustness of the MNCL, future work should consider optimizing the structure of the model and designing more suitable contrastive learning methods in conjunction with actual factors. In addition, neural estimation [52] and self-contrastive learning [53] methods are considered in the future to improve the performance of contrastive learning. Of course, exploring how to apply multi-perspective techniques [54,55] to knowledge graph-based recommendation will also be the focus of our next work.

CRediT authorship contribution statement

Duantengchuan Li: Writing – review & editing, Writing – original draft, Validation, Software, Resources, Methodology, Formal analysis, Data curation, Conceptualization. **Jiayao Lu:** Writing – original draft, Software, Data curation. **Zhihao Wang:** Writing – review & editing, Visualization. **Jingxiong Wang:** Writing – review & editing, Visualization, Software, Conceptualization. **Xiaoguang Wang:** Writing – review & editing, Supervision, Methodology. **Fobo Shi:** Writing – review & editing, Supervision, Methodology, Conceptualization. **Yu Liu:** Writing – review & editing, Supervision, Resources.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work is supported by Major Project of the National Social Science Fund, China (No. 21&ZD334).

Data availability

Data will be made available on request.

References

[1] W. Lin, M. Zhu, X. Zhou, R. Zhang, X. Zhao, S. Shen, L. Sun, A deep neural collaborative filtering based service recommendation method with multi-source data for smart cloud-edge collaboration applications, Tsinghua Sci. Technol. 29 (3) (2024) 897–910.
[2] H. Liu, L. Qi, S. Shen, A.A. Khan, S. Meng, Q. Li, Microservice-driven privacy-aware cross-platform social relationship prediction based on sequential information, Softw.: Pr. Exp. 54 (1) (2024) 85–105.
[3] H. Liu, C. Zheng, D. Li, Z. Zhang, K. Lin, X. Shen, N.N. Xiong, J. Wang, Multi-perspective social recommendation method with graph representation learning, Neurocomputing 468 (2022) 469–481.
[4] D. Li, Y. Gao, Z. Wang, H. Qiu, P. Liu, Z. Xiong, Z. Zhang, Homogeneous graph neural networks for third-party library recommendation, Inf. Process. Manage. 61 (6) (2024) 103831, http://dx.doi.org/10.1016/j.ipm.2024.103831.
[5] H. Liu, C. Zheng, D. Li, X. Shen, K. Lin, J. Wang, Z. Zhang, Z. Zhang, N.N. Xiong, EDMF: Efficient deep matrix factorization with review feature learning for industrial recommender system, IEEE Trans. Ind. Inform. 18 (7) (2022) 4361–4371.
[6] D. Li, H. Liu, Z. Zhang, K. Lin, S. Fang, Z. Li, N.N. Xiong, CARM: Confidence-aware recommender model via review representation learning and historical rating behavior in the online platforms, Neurocomputing 455 (2021) 283–296.
[7] L. Wang, X. Zhao, N. Liu, Z. Shen, C. Zou, Cognitive process-driven model design: A deep learning recommendation model with textual review and context, Decis. Support Syst. 176 (2024) 114062.
[8] Y. Koren, Factorization meets the neighborhood: a multifaceted collaborative filtering model, in: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2008, pp. 426–434.
[9] L. Zheng, V. Noroozi, P.S. Yu, Joint deep modeling of users and items using reviews for recommendation, in: Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, 2017, pp. 425–434.
[10] S. Feng, L. Zhao, H. Shi, M. Wang, S. Shen, W. Wang, One-dimensional VGGNet for high-dimensional data, Appl. Soft Comput. 135 (2023) 110035.
[11] D. Li, C. Deng, X. Wang, Z. Li, C. Zheng, J. Wang, B. Li, Joint inter-word and inter-sentence multi-relation modeling for summary-based recommender system, Inf. Process. Manage. 61 (3) (2024) 103631.
[12] B. Liu, D. Li, J. Wang, Z. Wang, B. Li, C. Zeng, Integrating user short-term intentions and long-term preferences in heterogeneous hypergraph networks for sequential recommendation, Inf. Process. Manage. 61 (3) (2024) 103680.
[13] F. Shi, D. Li, X. Wang, B. Li, X. Wu, TGformer: A graph transformer framework for knowledge graph embedding, IEEE Trans. Knowl. Data Eng. 37 (1) (2025) 526–541.
[14] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, T.-S. Chua, Neural collaborative filtering, in: Proceedings of the 26th International Conference on World Wide Web, 2017, pp. 173–182.
[15] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, O. Yakhnenko, Translating embeddings for modeling multi-relational data, in: Advances in Neural Information Processing Systems, vol. 26, 2013.
[16] Z. Li, Q. Zhang, F. Zhu, D. Li, C. Zheng, Y. Zhang, Knowledge graph representation learning with simplifying hierarchical feature propagation, Inf. Process. Manage. 60 (4) (2023) 103348.
[17] D. Li, T. Xia, J. Wang, F. Shi, Q. Zhang, B. Li, Y. Xiong, SDFormer: A shallow-to-deep feature interaction for knowledge graph embedding, Knowl.-Based Syst. 284 (2024) 111253.
[18] J. Wang, Q. Zhang, F. Shi, D. Li, Y. Cai, J. Wang, B. Li, X. Wang, Z. Zhang, C. Zheng, Knowledge graph embedding model with attention-based high-low level features interaction convolutional network, Inf. Process. Manage. 60 (4) (2023) 103350.
[19] H. Wang, F. Zhang, M. Zhang, J. Leskovec, M. Zhao, W. Li, Z. Wang, 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, 2019, pp. 968–977.
[20] F. Wu, D. Li, K. Lin, H. Zhang, Efficient nodes representation learning with residual feature propagation, in: Advances in Knowledge Discovery and Data Mining, 2021, pp. 156–167.
[21] Z. Wang, D. Yu, S. Shen, S. Zhang, H. Liu, S. Yao, M. Guo, Select your own counterparts: Self-supervised graph contrastive learning with positive sampling, IEEE Trans. Neural Netw. Learn. Syst. (2024) 1–15.
[22] X. Liu, F. Zhang, Z. Hou, L. Mian, Z. Wang, J. Zhang, J. Tang, Self-supervised learning: Generative or contrastive, IEEE Trans. Knowl. Data Eng. 35 (1) (2023) 857–876.
[23] Y. Zhang, K. Li, J. Zhu, M. Yuan, Y. Huang, X. Li, Leveraging hyperbolic dynamic neural networks for knowledge-aware recommendation, IEEE Trans. Comput. Soc. Syst. 11 (3) (2024) 4396–4411.
[24] K. Li, Y. Zhang, J. Zhu, X. Li, X. Wang, Multi-space interaction learning for disentangled knowledge-aware recommendation, Expert Syst. Appl. 254 (2024) 124458.
[25] Y. Yang, C. Huang, L. Xia, C. Li, Knowledge graph contrastive learning for recommendation, in: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2022, pp. 1434–1443.
[26] D. Zou, W. Wei, X.-L. Mao, Z. Wang, M. Qiu, F. Zhu, X. Cao, Multi-level cross-view contrastive learning for knowledge-aware recommender system, in: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2022, pp. 1358–1368.
[27] H. Wang, Y. Xu, C. Yang, C. Shi, X. Li, N. Guo, Z. Liu, Knowledge-adaptive contrastive learning for recommendation, in: Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining, 2023, pp. 535–543.
[28] J. Gao, D. He, X. Tan, T. Qin, L. Wang, T. Liu, Representation degeneration problem in training natural language generation models, in: International Conference on Learning Representations, 2019.
[29] J. Yu, X. Xia, T. Chen, L. Cui, N.Q.V. Hung, H. Yin, XSimGCL: Towards extremely simple graph contrastive learning for recommendation, IEEE Trans. Knowl. Data Eng. 36 (2) (2023) 913–926.
[30] H. Wang, F. Zhang, X. Xie, M. Guo, DKN: Deep knowledge-aware network for news recommendation, in: The World Wide Web Conference, 2018, pp. 23–27.
[31] F. Zhang, N.J. Yuan, D. Lian, X. Xie, W.-Y. Ma, Collaborative knowledge base embedding for recommender systems, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 353–362.
[32] X. Wang, X. He, Y. Cao, M. Liu, T.-S. Chua, KGAT: Knowledge graph attention network for recommendation, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 950–958.

[33] J. Yu, H. Yin, X. Xia, T. Chen, L. Cui, Q.V.H. Nguyen, Are graph augmentations necessary? Simple graph contrastive learning for recommendation, in: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2022, pp. 1294–1303.
[34] G. Ji, S. He, L. Xu, K. Liu, J. Zhao, Knowledge graph embedding via dynamic mapping matrix, in: Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics, 2015, pp. 687–696.
[35] D. Li, F. Shi, X. Wang, C. Zheng, Y. Cai, B. Li, Multi-perspective knowledge graph completion with global and interaction features, Inform. Sci. 666 (2024) 120438.
[36] X. Yu, X. Ren, Y. Sun, Q. Gu, B. Sturt, U. Khandelwal, B. Norick, J. Han, Personalized entity recommendation: a heterogeneous information network approach, in: Proceedings of the 7th ACM International Conference on Web Search and Data Mining, 2014, pp. 283–292.
[37] X. Yu, X. Ren, Q. Gu, Y. Sun, J. Han, Collaborative filtering with entity similarity regularization in heterogeneous information networks, in: IJCAI HINA, vol. 27, 2013.
[38] H. Zhao, Q. Yao, J. Li, Y. Song, D.L. Lee, Meta-graph based recommendation fusion over heterogeneous information networks, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017, pp. 635–644.
[39] H. Wang, M. Zhao, X. Xie, W. Li, M. Guo, Knowledge graph convolutional networks for recommender systems, in: The World Wide Web Conference, 2019, pp. 3307–3313.
[40] X. Wang, T. Huang, D. Wang, Y. Yuan, Z. Liu, X. He, T.-S. Chua, Learning intents behind interactions with knowledge graph for recommendation, in: Proceedings of the Web Conference 2021, 2021, pp. 878–887.
[41] J. Wu, X. Wang, F. Feng, X. He, L. Chen, J. Lian, X. Xie, Self-supervised graph learning for recommendation, in: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2021, pp. 726–735.
[42] Z. Lin, C. Tian, Y. Hou, W.X. Zhao, Improving graph collaborative filtering with neighborhood-enriched contrastive learning, in: Proceedings of the ACM Web Conference 2022, 2022, pp. 2320–2329.
[43] J. Yu, H. Yin, M. Gao, X. Xia, X. Zhang, N.Q. Viet Hung, Socially-aware self-supervised tri-training for recommendation, in: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021, pp. 2084–2092.
[44] Y. Zhang, J. Zhu, R. Chen, W. Liao, Y. Wang, W. Zhou, Mixed-curvature knowledge-enhanced graph contrastive learning for recommendation, Expert Syst. Appl. 237 (2024) 121569.
[45] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, M. Wang, 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, 2020, pp. 639–648.
[46] S. Rendle, C. Freudenthaler, Z. Gantner, L. Schmidt-Thieme, BPR: Bayesian personalized ranking from implicit feedback, in: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, 2009, pp. 452–461.
[47] H. Wang, F. Zhang, J. Wang, M. Zhao, W. Li, X. Xie, M. Guo, RippleNet: Propagating user preferences on the knowledge graph for recommender systems, in: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, 2018, pp. 417–426.
[48] H. Shu, J. Huang, Multi-task feature and structure learning for user-preference based knowledge-aware recommendation, Neurocomputing 532 (2023) 43–55.
[49] J. Wang, Y. Shi, H. Yu, X. Wang, Z. Yan, F. Kong, Mixed-curvature manifolds interaction learning for knowledge graph-aware recommendation, in: Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2023, pp. 372–382.
[50] M. Li, W. Ma, Z. Chu, KGIE: Knowledge graph convolutional network for recommender system with interactive embedding, Knowl.-Based Syst. (2024) 111813.
[51] S. Rendle, Factorization machines with libFM, ACM Trans. Intell. Syst. Technol. 3 (3) (2012).
[52] Y. Ma, X. Zhang, P. Zhang, K. Zhan, Entropy neural estimation for graph contrastive learning, in: Proceedings of the 31st ACM International Conference on Multimedia, 2023, pp. 435–443.
[53] Y. Ma, K. Zhan, Self-contrastive graph diffusion network, in: Proceedings of the 31st ACM International Conference on Multimedia, 2023, pp. 3857–3865.
[54] C. Xia, C. Niu, K. Zhan, Hierarchical consensus network for multiview feature learning, 2025, arXiv:2502.01961.
[55] C. Liu, Z. Liao, Y. Ma, K. Zhan, Stationary diffusion state neural estimation for multiview clustering, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36, (no. 7) 2022, pp. 7542–7549.