Contrastive Learning and Multi-Choice Negative Sampling Recommendation

Yun Xue[1], Xiaodong Cai[2], Sheng Fang[3], Li Zhou[4]
School of Information and Communication, Guilin University of Electronic Technology, Guilin, China[1, 2, 3]
Nanning West Bank Fenggu Business Data Co., Ltd, Nanning, China[4]

***Abstract*—Most existing recommendation models that directly model user interests on user-item interaction data usually ignore the natural noise present in the interaction data, leading to bias in the model's learning of user preferences during data propagation and aggregation. In addition, the currently adopted negative sampling strategy does not consider the relationship between the prediction scores of positive samples and the degree of difficulty of negative samples, and is unable to adaptively select a suitable negative sample for each positive sample, leading to a decrease in the model recommendation performance. In order to solve the above problems, this paper proposes a Contrastive Learning and Multi-choice Negative Sampling Recommendation. Firstly, an improved topology-aware pruning strategy is used to process the user-item bipartite graph, which uses the topology information of the graph to remove noise and improve the accuracy of model prediction. In addition, a new multivariate selective negative sampling module is designed, which ensures that each positive sample selects a negative sample of appropriate hardness through two sampling principles, improving the model embedding space representation capability, which in turn leads to improved model recommendation accuracy. Experimental results on the Urban-Book and Yelp2018 datasets show that the proposed algorithm significantly improves all the metrics compared to the state-of-the-art model, which proves the effectiveness and sophistication of the algorithm in different scenarios.**

***Keywords*—Recommendation algorithms; comparative learning; negative sampling; pruning strategies**

I. Introduction

Previous research has focused on modelling interest preferences from users' historical interaction data in order to obtain better recommendation results and provide personalised recommendation services to users to solve the problem of information overload [1]. However, collaborative filtering algorithms recommend poorly when the data lacks explicit user feedback, at which point the quality of negative sampling becomes crucial for improving the performance of recommendation models. Existing collaborative filtering algorithm all choose to train models using implicit feedback (e.g., click, buy, favourite, etc.) by default [2] and set the items of user interest as positive samples, but how to select high-quality negative samples is still a major challenge in the recommendation field. In addition, most models directly take user-item interaction data as the ideal data of user's preference, but due to the influence of external factors such as human error clicks, uncertainty, etc., which results in implicit feedback data containing a lot of natural noise [3], how to deal with the noise in the interaction data and to reduce the impact of noise on the recommendation accuracy is also a worthwhile research problem in the recommendation field.

Yu et al. [4] proposed the DropEdge mechanism to reduce the impact of noise on the node classification task by randomly deleting away the fixed edges in the original graph. However, random deletion has the potential to discard user preference information, resulting in lower recommendation accuracy. Thus, Zhang et al. [5] designed a classification-aware denoising based self-encoder to remove the noise effect by integrating the classification information. Fan et al. [6] removed the noisy data from the user-item interaction matrix by top-K sampling, balanced the number of interactions of all the users, and improved the accuracy of the model.

Rendle et al.[7] allowed the model to extract more feature information from the positive samples by randomly selecting items that users did not interact with as negative samples, and then using a loss function to give higher scores to the user-positive sample pairs while lowering the scores of the user-negative sample pairs. However, the practice of selecting negative samples with equal probability ignores the problem that the items that the user did not interact with are not necessarily items that the user dislikes, and it is possible that the user just did not see them. Ultimately, this leads to poor model predictions. Thus, Chen et al. [8]proposed popularity-based negative sampling, which takes item exposure as an important basis, and if a popular item with high enough exposure is still disliked by users, it means that the item can be used as a negative sample. Meanwhile, Ying et al. [9] proposed PinSage to calculate the node importance score, using difficult negative sample data for training to improve the overall performance of the model. Yang et al. [10] redesigned the sampling distributions of positive and negative samples, gave the calculation of negative sampling probability based on their structural similarity, and concluded that negative and positive samples are equally important. Huang in study [11] used user-item dichotomous graphs and the aggregation process of graph neural networks (GNN) to study negative sampling, and constructed a difficult negative sample candidate set by interpolating and mixing the negative samples to fuse part of the positive sample information, which improved the model training effect. Chen et al. [12] proposed the FairStatic dynamic adaptive negative sampling method, which improves the sampling fairness among groups while taking into account the sampling efficiency to ensure that each group of items can obtain equal recommendation quality. Lai et al. [2] proposed the DENS method, which firstly uses the hierarchical gating module to classify the similarity and dissimilarity of information between

Guangxi Driven Development Project (桂科 *AA20302001*).

positive and negative samples and identifies the negative samples through the factor-aware sampling strategy, so as to allow the difficult negative samples to provide more informative training signals and provide better user representation.

Although the above various negative sampling methods have allowed recommender system models to achieve good results, there are still some problems. Most of the existing negative sampling methods improve the model training effect by constructing difficult negative samples, however, they do not take into account the degree of matching between negative samples and positive samples, and negative samples with too much hardness may lead to the semantic bias between the samples and are not conducive to the final recommendation prediction. In addition, most algorithms remove noise by designing cumbersome components with high model complexity, and some models even omit the interaction data denoising step and use it directly as the positive samples for training, which leads to the model not being able to correctly model users' interest preferences, and the recommendation results are biased.

In order to solve the above problems, this paper proposes a Contrastive Learning and Multi-choice Negative Sampling Recommendation (CLMRec). The model firstly analyses the degree of contribution of edges to nodes by Topology-aware Pruning Strategy (TPS) based on topology, calculates the probability that each edge can be retained, and then removes the noisy data according to the probability to reduce the impact of noise on the node embedding representations in the propagation process. Finally, in the negative sampling stage, a new Multi-Choice Negative Sampling (MCNS) strategy is proposed to adaptively select negative samples of appropriate hardness through two sampling principles to optimize the model training effect and obtain more accurate user embeddings and item embeddings to improve the accuracy of recommendations.

In summary, our contributions are highlighted as follows:

- We propose the TPS denoising framework to remove noise from user-item interaction data, preventing the adverse effects of noise during the information aggregation process.
- We introduce the MCNS negative sampling framework, which enables adaptive selection of negative samples of appropriate difficulty, thereby enhancing the quality of model training.

II. CLMRec Model Design

A. Notation Definition and Description

In this paper, the model input is the user-item interaction data, where $U=\{u_1,u_2,...u_m\}$ is the set of users, and $I=\{i_1,i_2,...i_n\}$ is the set of items, where m is the number of users, and n is the number of items. $\boldsymbol{R}$ is the user-item interaction matrix, and $G=\{U,I,E\}$ is the user-item interaction graph, where E is the set of user-item edges.

B. Overall framework

The overall framework of the CLMRec model is shown in Fig. 1. Firstly, for the interaction data in the user-item dichotomous graph G, the TPS is used to calculate the retention probability of each edge and remove the noise, and then multi-task joint training is constructed, and comparative learning is used as a secondary task to construct comparative views on the interaction data, and potential feature information between different views is extracted by the Infonce loss function [13] to enhance the model's representation learning capability.

The main task uses LightGCN [1] to linearly propagate user embeddings and item embeddings on the interaction graph, aggregating node information to obtain the final user embeddings z_u and the final item embeddings z_i. For item embeddings, the MCNS component adaptively selects negative samples of appropriate hardness for each positive sample, and continuously optimises the positive sample similarity scores and reduces the negative sample prediction scores through the BPR loss function [7]. The prediction score, allowing the model to gradually learn the correct user preferences. Finally, the main and auxiliary tasks are jointly learnt to update the user embeddings and item embeddings.

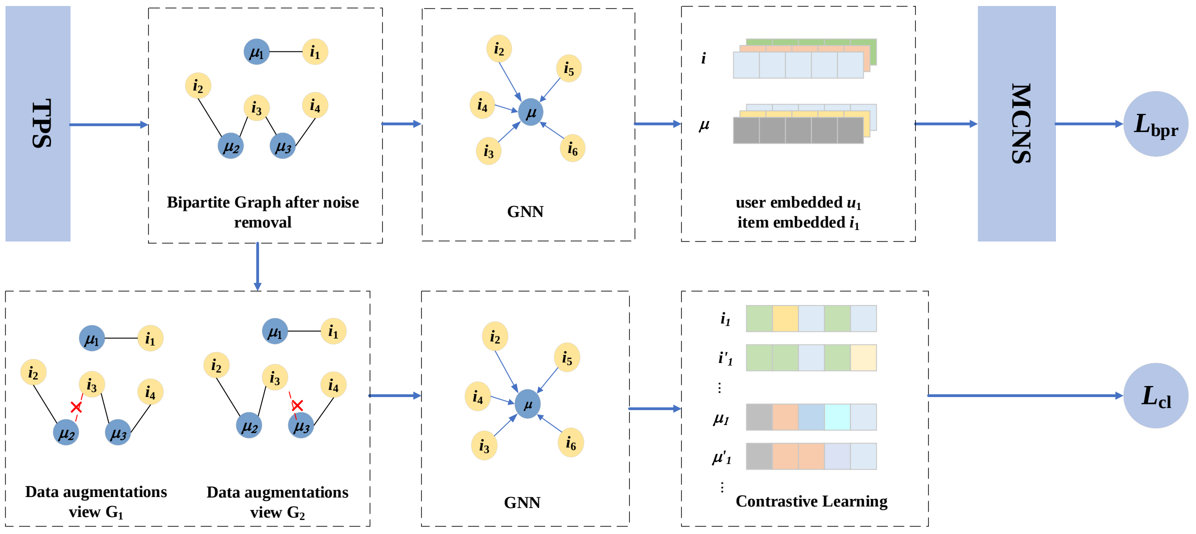

Fig. 1. DFFSM overall framework.

C. Topology-aware Pruning Strategy

In order to avoid too much noise accompanying the interaction data into the model, inspired by the literature [4][14] and following the idea of model sparsification, the natural noise present in the interaction data is handled by removing redundant edges from the graph. In this paper, a topology-aware pruning strategy (TPS) is designed. Firstly, the user-item data is processed into a user-item interaction matrix *R*. The degree of each node is calculated using *R* to obtain the degree matrix *D*. Next, the retention probability of each edge in the graph is calculated. Finally, some edges of the user-item dichotomous graph are removed according to the magnitude of the probability to complete the denoising of the interaction data.

To make the exposition easier, the TPS process is plotted as in Fig. 2:

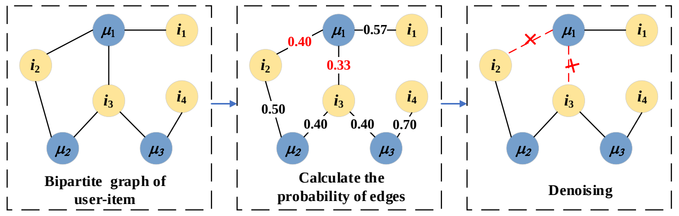

Fig. 2. TPS.

Literature [15] states that the degree of a node is the number of edges directly connected to that node and can be considered as a measure of the importance of the node in the network. For the interaction matrix *R*, the process of computing the degree matrix *D* is as follows:

$$\boldsymbol{D}[k][k] = \sum_{j=1}^{n} \boldsymbol{R}[k][j] \tag{1}$$

$$\boldsymbol{D} = diag\left(\sum_{j=1}^{n} \boldsymbol{R}[1][j], \sum_{j=1}^{n} \boldsymbol{R}[2][j], \ldots, \sum_{j=1}^{n} \boldsymbol{R}[n][j] \right) \tag{2}$$

where, $\boldsymbol{D}[k][k]$ is the element on the *k*th diagonal of the degree matrix *D* and $\boldsymbol{R}[k][j]$ is the element in the kth row and jth column of the interaction matrix *R*. Since *D* is a diagonal matrix and all the positional elements are 0 except those on the diagonal, the degree matrix *D* can be derived from Eq. (2), which $diag(\cdot)$ indicates that a diagonal matrix is constructed by using the elements in parentheses as the elements on the diagonal.

Some papers use randomly discarded edges to reduce the influence of height nodes and to prevent overfitting phenomena, but randomly discarded edges have the potential to destroy important information about the nodes, leading to biased node semantics. The degree of contribution of an edge to a node should be calculated and the natural noise should be removed based on the weights, and the square root of the degree is often used as a factor to adjust the edge weights.

The idea is that nodes with larger degrees have more connections in the network and have a higher probability of noisy data, so the weights of edges connected to them should be reduced to balance the importance of the node. On the contrary, nodes with smaller degree have fewer connections in the network, so the weights of the edges connected to them should be increased to better reflect the importance of the nodes. In addition, the effect of edge discarding on the two connected nodes should be considered, so the degree of both nodes should be included in the formula as follows:

$$p_{(i,j)} = \frac{1}{\sqrt{d_i}\sqrt{d_j}} \tag{3}$$

where, $i \in (1,2,3,\ldots,n), j \in (1,2,3,\ldots,n),$ 且$i \notin j$, *i* and *j* denote the nodes in the bipartite graph of user items, and d_i and d_j represent the degrees of node *i* and node *j*, respectively, as shown in Fig. 2, the retention probability of each edge is calculated by using the TPS, and then the edge between i_3 and u_1 is removed. While these two nodes are equivalent to popular nodes for other nodes with more interaction data, discarding the edges of these two nodes can reduce the influence of popular nodes on low-degree nodes, and also prevent the model from overfitting. For the edges of these two nodes, i_4 and u_3, the retention probability is high because i4 has only one interaction data, which should be fully retained to facilitate the model's learning of i_4 commodity embedding.

D. LightGCN

After removing some of the noise from the interaction data, the model is started to model the user interest. In this paper, LightGCN [1] is used as an encoder. Firstly, user embeddings and item embeddings are randomly initialised, and multiple rounds of embedding propagation are performed through the graph convolution layer, and the embedding vectors of users and items are updated through iterations. Since the algorithm is constructed for joint multi-task learning, divided into main task and auxiliary task, the differences between the two tasks are described below.

In each round of the main task, the user and item embeddings are weighted and summed according to the user-item interaction matrix $\boldsymbol{R}$, and the neighbour node information is aggregated. After multiple propagation and aggregation, the model can extract the user's higher-order interests, and the specific aggregation strategy and propagation mechanism are shown in Eq. (4) and Eq. (5).

$$z_u = \sum_{l=0}^{L} \alpha_l z_u^{(l)} \tag{4}$$

$$z_i^{l+1} = \sum_{u \in N_i} \frac{z_u^l}{\sqrt{|N_u||N_i|}} \tag{5}$$

where, l represents the number of convolutional layers and denote the user embedding and item embedding in the lth layer, respectively.

The pooling of the convolved embeddings is performed to obtain the final user embeddings and item embeddings. Considering that the embeddings of different layers have different semantics, the embeddings of different layers are weighted and combined, and the embedding combination strategy is shown in Eq. (6) and Eq. (7).

$$z_u = \sum_{l=0}^{L} \alpha_l z_u^{(l)} \tag{6}$$

$$z_i = \sum_{l=0}^{L} \alpha_l z_i^{(l)} \tag{7}$$

where, z_u represents the final user embedding, z_i represents the final item embedding, α_l is the weight of each layer, L is the number of convolutional layer layers, and in this paper, we follow the practice of literature [1], and take α_l as the inverse of *L*.

In the auxiliary task, contrast learning is mainly used to alleviate the data sparsity problem. Firstly, data augmentation is performed on the denoised interaction data to obtain augmented view G_1 and augmented view G_2, followed by constructing positive sample pairs and negative sample pairs for the vectors in the two views. Ultimately, the loss function of the model is used to bring the positive pair embeddings closer together and push the negative pair embeddings farther apart, so that the model extracts the unlabelled extra information in the interaction data, learns high-quality embedded representations of the users and items, and improves the accuracy of the recommendations.

E. Multi-Choice Negative Sampling

Positive and negative samples need to be selected after obtaining the user embedding *u* and item embedding *i*. The positive and negative samples are then passed through the BPR loss function [7] to give high prediction scores to the user-positive samples and reduce the prediction scores of the user-negative samples, which facilitates the model to learn the user interest preferences correctly.

Since the interaction data have been removed from the noise before entering the model and the interaction data are the real interest preferences of users, the items interacted in the dichotomous graph are directly selected as the corresponding user-positive samples. However, how to select high-quality negative samples to train the model is a difficult point, and the existing models do not select appropriate negative samples based on the information and prediction scores of the positive samples. For example, in Fig. 3, when the model selects negative samples, if an i_{20} is randomly selected as a negative sample from the items that the user node u_1 has not interacted with, it does not mean that the user does not like the item, and it is possible that the item exposure is too low for the user to see. In addition, some models do not construct difficult negative samples based on the characteristics of positive samples, which causes the problem of high model training cost.

In order to solve the above problems, inspired by the literature [16] [17], multivariate selective negative sampling (MCNS) is proposed. MCNS constrains the selection range of negative samples by two principles: suitable negative samples must be selected based on the characteristics of positive samples; and the hardness of negative samples must be inversely proportional to the prediction scores of positive samples. These two principles ensure that the model adaptively selects negative samples of appropriate hardness for positive samples during negative sampling.

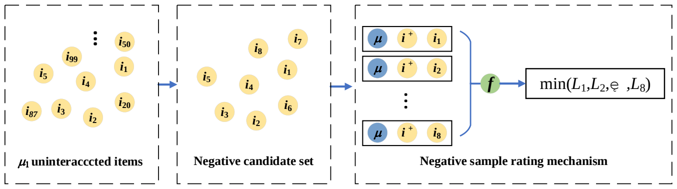

Fig. 3. MCNS.

Firstly, Principle 1 is set to eliminate the uncertainty of user preference brought by randomly selecting negative samples. In the prediction stage, the model calculates the prediction scores of the positive sample embedding and the user embedding, and determines whether to recommend or not based on the high or low prediction scores, inspired by this, this paper decides to take the prediction scores of the positive samples as an important factor in selecting the negative samples, and Eq. (8) denotes the calculation of the prediction scores:

$$score^{+} = z_u^T z_{i^+} \tag{8}$$

$$score^{-} = z_u^T z_{i^-} \tag{9}$$

score$^+$ represents the positive sample prediction score, score$^-$ represents the negative sample prediction score, z_u^T represents the transpose matrix of the user matrix, z_{i^+} and z_{i^-} represents the positive sample matrix and negative sample matrix respectively. The higher the prediction score, the closer the two embeddings are in space, i.e., the more interested the user is in the item.

The negative sampling process is shown in Fig. 3, where a specified number of items are randomly selected from all the data that the user has not interacted with to construct a candidate set of negative samples, and then, for all the items in the candidate set, the association level of the items is calculated using the rating function, and then, the appropriate items are selected as negative samples. For positive samples with high prediction scores, it indicates that the model has learnt sufficiently well for that sample, and picking simple negative samples can reduce the training cost. Selecting negative samples with a high degree of difficulty will cause the model's performance to degrade in the process of classifying positive and negative samples, affecting the final prediction accuracy. Conversely, for positive samples with low prediction scores, it indicates that the model is not yet able to adequately capture similar user interests, at which point difficult negative samples should be constructed to allow the model to learn deeper features and improve the model's representational ability. This determines principle two, where the difficulty of the negative sample is inversely proportional to the prediction score of the positive sample. Eq. (10) is the rating function, which is used to determine the level of negative samples selected.

$$L_n = \left| z_u^T z_{i_n} - \left(z_u^T z_i + \alpha \right)^{p+1} \right| \tag{10}$$

where, L_n is smaller, the greater the probability that the sample will be a negative sample. p is less than -1, and when considering L_n and α are equal, negative sample hardness $h\left(i^-\right)$ is defined as the ratio of positive and negative prediction scores, as defined in Eq. (11):

$$h\left(i^-\right)=\frac{z_u^T z_i}{z_u^T z_{i^-}}=\left(z_u^T z_i+\alpha\right)^p \tag{11}$$

Negative correlation was verified using Eq. (12). First, the negative correlation is transformed into a derivation problem by using the negative sample difficulty level to derive the positive sample prediction scores.

$$\frac{\partial\left(h\left(i^-\right)\right)}{\partial\left(z_u^T z_i\right)}=\frac{\partial\left(\left(score^+ +\alpha\right)^{p+1}\right)}{\partial\left(score^+\right)}=p\cdot\left(score^+ +\alpha\right)^p \tag{12}$$

where, p is a hyperparameter less than 0. Obviously, the derivative is negative, indicating that the function is monotonically decreasing, proving that the difficulty of negative samples is negatively correlated with the prediction scores of positive samples.

F. Model prediction and Training

After the message propagation and aggregation mechanism of the GNN encoder, the final user embedding and item embedding , and enter the prediction stage to predict the user's interest in the item according to Eq. (13).

$$\hat{y}_{u,i}=z_u^T z_i \tag{13}$$

Then, the main task adaptively selects negative samples of appropriate hardness for each positive sample through MCNS and calculates the loss so that the model learns more accurate user preferences and item characteristics from the training data, adopting the method of literature [18], and using the BPR loss as the loss function of the recommendation task to measure the difference between the prediction results and the real labels as shown in Eq. (14):

$$L_{bpr}=\sum_{u\in U}\sum_{i\in I}\sum_{j\in I^-}\ln\sigma\left(\hat{y}_{u,i}-\hat{y}_{u,i^-}\right) \tag{14}$$

In the auxiliary task, Infonce is used as a comparative learning loss function to maximise the mutual information between the same sample views and minimise the information of different sample views, and by comparing the differences between different views, the model can extract the extra unlabelled information in the interaction data as a way to improve the representation of the embedding space and the model performance. The Infonce loss is as shown in Eq. (15):

$$L_{cl}=\sum_{n\in G}\frac{\exp\left(s\left(z_n^1,z_n^2\right)/\tau\right)}{\sum_{n'\in G,n'\neq n}\exp\left(s\left(z_n^1,z_{n'}^2\right)/\tau\right)} \tag{15}$$

where, G is a user-item bipartite graph, n and n' represent different nodes in G respectively, $s(\cdot)$ is the cosine function, and τ is the temperature parameter.

Finally, the main task loss and auxiliary task loss are combined to construct the model multi-task learning framework, and the total model loss is shown in Eq. (16):

$$L=L_{bpr}+\lambda_1 L_{cl}+\lambda_2\left\|\Theta\right\|_2^2 \tag{16}$$

where, L_{bpr} denotes the main task loss, L_{cl} denotes the auxiliary task comparison learning loss, and denotes the regularisation parameters, and denotes the learnable model parameters.

G. Pseudo-code of the Model

In order to give the reader a clearer understanding of the execution process of the CLMRec model, the pseudo-code of the model is given, as shown in Table I:

TABLE I. PSEUDO-CODE OF CLMREC

```
Algorithm: CLMRec
 1: Input: User-Item bipartite graph G, training dataset $X$
 2: Output: Sst of recommended items
 3: While CLMRec Not Convergence do
 4:     for x in Dataloader( $X$ ) do
 5:         Calculate the degree matrix D from the interaction matrix
            R;
 6:         Calculate the retention probability P;
 7:         Noise removal according to P;
 8:     Generate user final embedding and item final embedding;
 9:         Adaptive selection of suitable negative samples
10:         Generate comparison views $G_1$ and $G_2$;
11:         Calculate BPR loss $L_{\text{bpr}}$;
12:         Calculate contrastive learning loss $L_{\text{CL}}$;
13:         Calculate total loss $L$;
14:     end for
15: end while
```

III. EXPERIMENTAL RESULTS AND ANALYSIS

A. Experimental Setup

1) Experimental environment: The experimental environment is set up as follows: the graphics card configuration is NVIDIA GeForce RTX 2080Ti, the operating system is Ubuntu 18.04, the programming language Python, and the deep learning framework is Pytorch.

2) Datasets: In order to verify the effect of the algorithm proposed in this paper on datasets with different sparsity levels and its performance in different scenarios, two publicly available datasets, Douban-Book and Yelp2018, are used for experiments. The dataset information is shown in Table II.

TABLE II. STATISTICS FOR THE DATASETS

Datasets information	Douban-book	Yelp2018
Number of users	12638	31668
Number of iems	22222	38048
Interactive data	478730	1237259
Data density	0.1704%	0.1026%

3) Evaluation indicators: Following the practice of literature [18], Recall and Normalised Discount Cumulative Gain (NDCG), which are commonly used in recommender systems, are used as the evaluation metrics in this experiment. The higher the Recall metric, the more items the recommender system can find that the user is interested in, and the NDCG measures the accuracy and sorting information of the recommended items, this paper uses the two metrics to comprehensively evaluate the recommender system performance. The number of users, the number of items, interaction data, and data density vary between two datasets, resulting in different recommendation accuracies. Higher data density in a dataset allows the model to learn more accurate user preferences, leading to higher recommendation accuracy.

4) Baseline modelling and parameter setting: In order to verify the effectiveness of CLMRec, four representative models are selected for comparison, the baseline model LightGCN [1] based on graph neural network, the models SGL [19] and SimGCL [18] based on comparative learning, and MixGCF [11] based on hybrid technology to generate negative samples for comparison. After hyper-parameter tuning, the batch size is set to 2048, the number of convolutional layers is set to 3, the temperature parameter is set to 0.2, and the learning rate is set to 0.001.

- LightGCN is a state-of-the-art GCN-based recommendation method which simplifies the convolution operations during the message passing among users and items.
- SGL introduces self-supervised learning to enhance recommendation. We focus on exploring self-supervised learning (SSL) in recommendation, to solve the foregoing limitations. Though being prevalent in computer vision (CV) and natural language processing (NLP).
- MixGCF designs the hop mixing technique to synthesize hard negatives for graph collaborative filtering by embedding
- Interpolation and Introduce the idea of synthesizing negative samples rather than directly sampling negatives from the data for improving GNN-based recommender systems.
- SimGCL proposed a simple yet effective graph-augmentation-free CL method for recommendation that can regulate the uniformity in a smooth way. It can be an ideal alternative of cumbersome graph augmentation-based CL methods.

B. Results of the Experiment

The experimental results of the CLMRec model and each baseline model in the Recall@20 and NDCG@20 evaluation metrics are shown in Table III, with the best performance of the comparison models underlined, and the experimental results of this paper's model shown in bold font.

All of the above models transform user interaction data into graph-structured data, and through graph convolutional models, aggregate neighbour node information to capture the user's interest preferences. As seen from the data in Table III, LightGCN effectively improves recommendation accuracy by simply iteratively aggregating neighbour features into target node embedding representations, removing the redundancy of feature transformations and non-linear activation components. Inspired by the natural language and image processing domains, SGL introduces contrast learning into the recommendation domain, proposes three methods for constructing a contrasted view, and achieves better recommendation results than LightGCN SimGCL, based on SGL, proposes a more concise and effective data enhancement method to solve the problem of possibly deleting the important information of nodes in SGL, and speeds up the process of constructing positive and negative contrast views, which is a simple and efficient model. MixGCF Changes the traditional negative sampling strategy by mixing the positive sample information with the negative sample information to construct difficult negative samples for training, which improves the overall performance and proves that high-quality negative samples allow the model to learn more accurate user embeddings and item embeddings.

TABLE III. SMODEL PERFORMANCE COMPARISON

Method	Douban-Book		Yelp2018	
	Recall	NDCG	Recall	NDCG
LightGCN	0.1494	0.1217	0.0642	0.0537
SGL	0.1730	0.1549	0.0678	0.0558
MixGCF	0.1732	0.1553	0.0710	0.0588
SimGCL	<u>0.1778</u>	<u>0.1585</u>	<u>0.0721</u>	<u>0.0596</u>
CLMRec	**0.1899**	**0.1706**	**0.0736**	**0.0604**

The CLMRec model proposed in this paper significantly improves Recall and NDCG on both datasets compared to all comparison models. Among them, CLMRec improves 6.80% and 7.63% on the Urban-Book dataset and 1.34% and 2.14% on the yelp dataset, respectively, compared to SimGCL, which is the best performer, demonstrating the validity and sophistication of this paper's model in different scenarios. Compared with other models, the advantage of CLMRec is that the removal of noise is completed when the interaction data enters the model before, which effectively avoids the negative impact of noise on other nodes in the graph convolution process. In addition, the combination of contrast learning and LightGCN's encoder extracts extra information from the samples, which improves the model training effect, and finally the MCNS adaptively selects negative samples with appropriate hardness, which significantly improves the accuracy of recommendation.

C. Ablation Experiments

1) Verifying the effect of the number of samples on recommendation accuracy.

In order to verify the impact of the number of samples in the negative sampling candidate set on the recommendation results, this paper chooses to conduct experiments on the Urban-Book, by choosing a different number of samples to construct the negative sampling candidate set, the specific experimental results are shown in Table IV:

TABLE IV. SMODEL PERFORMANCE COMPARISON

Number of samples	Recall	NDCG
4	0.1856	0.1663
8	0.18569	0.1671
16	0.1877	0.1676
32	0.1889	0.1687
64	0.1895	0.1697
128	**0.1899**	**0.1706**
256	0.1900	0.1706

As can be seen from Table IV, the higher the number of samples in the candidate set and the higher the number of negative samples available, the better the recommendation performance. However, the more the number of samples, the more time is needed to calculate the rating function Ln, and the recommendation effect is not obviously improved when the number of samples is too large. Measuring the efficiency problem, this paper takes 128 negative samples to construct the candidate set.

2) Validation of TPS and MCNS component effectiveness: In order to validate the effectiveness of the method proposed in this paper, variant models are designed for denoising experiments. Firstly, in order to demonstrate the denoising effect of TPS, the variant model CLMRec-TPS is designed to omit the denoising step by removing the TPS module and taking the user-item interaction data as input directly. Secondly, the variant model CLMRec-MCNS is designed, which discards the strategy of adaptively selecting negative samples of appropriate hardness by randomly selecting items that the user did not interact with as negative samples. The performance of these variant models on the two datasets is shown in Fig. 4.

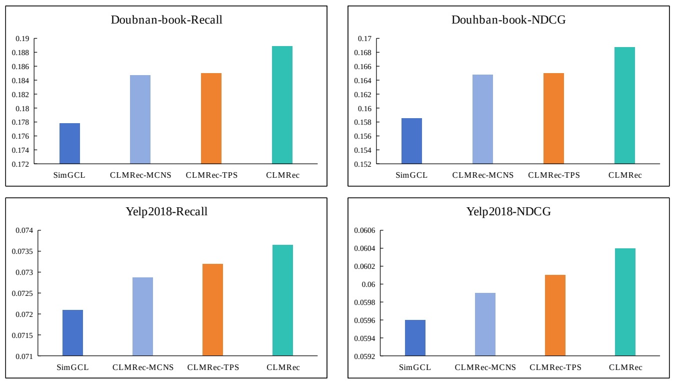

Fig. 4. Effectiveness analysis of TPS and MCNS.

As can be seen from Fig. 4, all the metrics of the CLMRec model are higher than those of the variant models, proving the necessity of each component. The metrics of the CLMRec model are higher than those of the CLMRec-TPS model, which indicates that the model can effectively remove the noise in the interaction data, avoiding the noise from affecting the accuracy of the recommendation in the process of propagation. In addition, the indicators of CLMRec-MCNS are lower than CLMRec, which proves the effectiveness of the MCNS component, i.e., when negative sampling, it is necessary to choose negative samples with different hardnesses according to the characteristics of positive samples.

IV. CONCLUSION

In this paper, a recommendation model CLMRec based on comparative learning and multivariate selection of negative sampling is proposed. The pruning strategy component based on topology perception, with low time complexity and high compatibility, is suitable for denoising of graph neural networks, and it can effectively reduce the impact of noise on prediction accuracy. In addition, the multivariate selection negative sampling component is proposed to comprehensively consider the relationship between the prediction scores of the positive samples and the hardness of the negative samples, adaptively select the negative samples with appropriate hardness, which enhances the embedding representation ability of the model and provides a new research idea for the negative sampling technique of graphs. Experiments on two publicly available datasets show that the present model has a significant improvement in recommendation performance compared with the current state-of-the-art models, and has good practical application value.

Considering the differences in users' long and short-term interests, future work will explore how to capture users' interests in different periods to improve the recommendation accuracy.

ACKNOWLEDGMENT

We would like to express our deepest gratitude to Guangxi Driven Development Project (桂科 AA20302001) for their financial support, without which this research would not have been possible.

Special thanks are due to Qing Ye and Yanyan Zhang for their collaboration and assistance in various aspects of this research project. Their contributions have significantly contributed to the success of this study.

REFERENCES

[1] He X, Deng K, Wang X, et al. Lightgcn: Simplifying and powering graph convolution network for recommendation[C]//Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval. 2020: 639-648.

[2] Lai R, Chen L, Zhao Y, et al. Disentangled negative sampling for collaborative filtering[C]//Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining. 2023: 96-104.

[3] Yera R, Castro J, Martínez L. A fuzzy model for managing natural noise in recommender systems[J]. Applied Soft Computing, 2016, 40: 187-198.

[4] Rong Y, Huang W, Xu T, et al. Dropedge: Towards deep graph convolutional networks on node classification[J]. arXiv preprint arXiv:1907.10903, 2019.

[5] Zhang C, Li T, Ren Z, et al. Taxonomy-aware collaborative denoising autoencoder for personalized recommendation[J]. Applied Intelligence, 2019, 49: 2101-2118.

[6] Fan Z, Xu K, Dong Z, et al. Graph collaborative signals denoising and augmentation for recommendation[C]//Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2023: 2037-2041.

[7] Rendle S, Freudenthaler C, Gantner Z, et al. BPR: Bayesian personalized ranking from implicit feedback[J]. arXiv preprint arXiv:1205.2618, 2012.

[8] Chen T, Sun Y, Shi Y, et al. On sampling strategies for neural network-based collaborative filtering[C]//Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2017: 767-776.

[9] Ying R, He R, Chen K, et al. Graph convolutional neural networks for web-scale recommender systems[C]//Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining. 2018: 974-983.

[10] Yang Z, Ding M, Zhou C, et al. Understanding negative sampling in graph representation learning[C]//Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining. 2020: 1666-1676.

[11] Huang T, Dong Y, Ding M, et al. Mixgcf: An improved training method for graph neural network-based recommender systems[C]//Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining. 2021: 665-674.

[12] Chen X, Fan W, Chen J, et al. Fairly adaptive negative sampling for recommendations[C]//Proceedings of the ACM Web Conference 2023. 2023: 3723-3733.

[13] Wu C, Wu F, Huang Y. Rethinking infonce: How many negative samples do you need?[J]. arXiv preprint arXiv:2105.13003, 2021.

[14] Zhou X, Lin D, Liu Y, et al. Layer-refined graph convolutional networks for recommendation[C]//2023 IEEE 39th International Conference on Data Engineering (ICDE). IEEE, 2023: 1247-1259.

[15] Arul S M, Senthil G, Jayasudha S, et al. Graph Theory and Algorithms for Network Analysis[C]//E3S Web of Conferences. EDP Sciences, 2023, 399: 08002.

[16] Chen J, Lian D, Jin B, et al. Learning recommenders for implicit feedback with importance resampling[C]//Proceedings of the ACM Web Conference 2022. 2022: 1997-2005.

[17] Lai R, Chen R, Han Q, et al. Adaptive hardness negative sampling for collaborative filtering[J]. arXiv preprint arXiv:2401.05191, 2024.

[18] Yu J, Yin H, Xia X, et al. Are graph augmentations necessary? simple graph contrastive learning for recommendation[C]//Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval. 2022: 1294-1303.

[19] Wu J, Wang X, Feng F, et al. Self-supervised graph learning for recommendation[C]//Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval. 2021: 726-735.