Popularity-Correction Sampling and Improved Contrastive Loss Recommendation

Wei Lu, Xiaodong Cai, Minghui Li
School of Information and Communication, Guilin University of Electronic Technology, Guilin, China

***Abstract*—In recommendation systems, negative sampling strategies are crucial for the calculation of contrastive learning loss. Traditional random negative sampling methods may lead to insufficient quality of negative samples during training, thereby affecting the convergence and performance of the model. In addition, the Bayesian Personalized Ranking (BPR) loss function usually converges slowly and is prone to falling into suboptimal local solutions. To address the above problems, this paper proposes a recommendation algorithm based on popularity-corrected sampling and improved contrastive loss. First, a dynamic negative sampling method with popularity correction is proposed, which reduces the impact of item popularity distribution bias on model training and dynamically screens out negative samples to improve the quality of model recommendations. Second, an improved contrastive loss is proposed, which selects the most challenging negative samples and introduces a boundary threshold to control the sensitivity of the loss, enabling the model to focus more on samples that are difficult to distinguish and further optimize the recommendation effect. Experimental results on the Amazon-Book, Yelp2018, and Gowalla datasets show that the proposed model significantly outperforms mainstream state-of-the-art models in recommendation tasks. Specifically, the Recall metric, which reflects model accuracy, improves by 16.8%, 12.9%, and 5.72% respectively on these three datasets. The NDCG metric, which measures ranking quality, increases by 20.7%, 16.4%, and 7.76% respectively. These results confirm the effectiveness and superiority of the recommendation algorithm across different scenarios. Compared with baseline models, it demonstrates stronger adaptability in complex situations, such as the sparse dataset Gowalla and the long - tail distribution dataset Amazon-Book, with the highest improvement in core metrics exceeding 20%.**

***Keywords*—Recommendation algorithms; contrast loss; difficult negative samples object; popularity bias**

I. INTRODUCTION

Due to the outstanding ability of recommendation systems in alleviating information overload, they have been widely applied in various fields, including video, news, and e-commerce [1, 2]. Collaborative Filtering (CF) is a widely - researched topic in recommendation systems, and the learning of CF models typically relies on three main components, namely interaction encoders, negative sampling, and loss functions [3]. Although many existing studies focus on designing more powerful interaction encoders, the impact of loss functions and negative sampling has not been fully explored.

Traditional negative sampling methods predominantly employ random selection strategies. For instance, Rendle et al. [4] proposed the Bayesian Personalized Ranking, which randomly selects negative samples from items users haven't interacted with and ensures positive samples have higher predicted values to achieve personalized ranking. Despite its renowned simplicity and efficiency, this method faces challenges in scenarios with long-tailed item popularity distributions, where the frequent occurrence of popular items slows down convergence and exacerbates the popularity bias. To address the limitations of traditional negative sampling methods, researchers have proposed various improvements. Steffen et al. [5] introduced a dynamic oversampling strategy, prioritizing uninteracted items with higher model-predicted scores as negative samples to enhance their prediction scores. However, this approach's overreliance on "easily distinguishable samples" makes it difficult to capture fine-grained interaction information in implicit feedback, thus destabilizing the training process. Subsequent studies have incorporated external information and hybrid enhancement techniques to overcome this restriction: Togashi et al. [6] utilized knowledge graph structural information to filter potential positive samples for pseudo-labeling, effectively boosting recommendation performance in cold-start scenarios but relying on knowledge graph construction. Huang et al. [7] adopted a jumping-mixing technique, injecting positive-sample features into candidate negative samples. By aggregating multi-order neighbor information, they generated highly distinctive synthetic negative samples, yet at the cost of high computational complexity. Petrov et al. [8] proposed a sampling method based on temporal importance. Using an exponential decay function to assign higher sampling probabilities to recent interactions, they improved training efficiency and model performance while closely aligning with the ultimate goal of sequential recommendation. However, determining appropriate temporal weights is necessary. Shi et al. [9] injected positive-sample information into negative samples to create synthetic hard negative samples dominated by positive information, avoiding incorrect negative sample selection but risking oversmoothing and overlooking key samples. Xue et al. [10] dynamically adjusted negative-sample difficulty based on positive-sample prediction scores, selecting suitable negative samples for each training stage through ranking candidate sets. This balances model convergence speed and expressiveness but requires a complex dynamic adjustment mechanism.

In terms of loss function design, despite efforts to develop stronger encoders for capturing collaborative signals, recommendation performance remains heavily influenced by the training loss function [11, 12]. Pairwise Loss, which models users' relative preferences between items, has become a

mainstream optimization paradigm. Most existing models adopt the BPR loss based on maximum a posteriori estimation, directly maximizing the prediction difference between positive and negative samples but possibly ignoring complex relationships between samples. The Hinge loss proposed by Chen et al. [13] introduced a margin constraint, requiring the positive-sample prediction to exceed the negative one by a threshold, yet determining the right threshold is challenging. Recently, Mao et al. [14] proposed the Cosine Contrastive Loss, shifting to vector-space optimization. It enhances representation distinctiveness through cosine-similarity contrast under margin constraints but is computationally intensive and requires determining appropriate constraints.

Despite the achievements of the above recommendation algorithms, there are still some pressing issues to be resolved. First, traditional popularity-based negative sampling strategies will intensify the Matthew effect in recommendation results, making popular items more popular and cold items harder to be discovered. Second, existing contrastive loss improvement schemes still rely on processing large numbers of low-quality negative samples, making it difficult to balance efficiency and effectiveness.

To address the issues mentioned above, this paper proposes a recommendation algorithm based on popularity - corrected sampling and improved contrastive loss (PICRec). The model first calculates the interaction frequency of each item among all users to determine its popularity. It then penalizes high - popularity items and assigns higher weights to low - popularity items according to the item popularity distribution in order to alleviate the popularity bias problem. A masking mechanism is used to prevent sample conflicts. Secondly, by adjusting the threshold to regulate the sensitivity between positive and negative samples, the model is better able to learn and distinguish different samples. Moreover, the most challenging hard negative samples are used to accelerate the training process, thereby enhancing the recommendation effect.

The following outlines the structure of the paper: Section II reviews related work. Section III delves into the design of the PICRec model, covering the encoder, popularity - sampling - correction strategy, and enhanced contrastive learning loss. Section IV presents and analyzes experimental results to validate the approach. Section V summarizes the work and explores future research directions.

II. Related Work

A. Negative Sampling

In implicit feedback collaborative filtering systems, negative sampling techniques have emerged as a core methodology to address the severe imbalance between positive and negative samples by constructing high-quality negative sample sets, balancing optimization efficiency and ranking performance [15]. A common strategy involves static sampling based on predefined prior distributions, which generates negative samples through fixed probability distributions. This approach reduces computational complexity by avoiding dynamic parameter adjustments during training [16]. A typical example is uniform random sampling, where negative samples are randomly selected from unobserved user-item pairs under a uniform distribution, serving as a model-agnostic baseline widely adopted in practice [17]. However, this strategy inherently assumes homogeneity (i.e., all unobserved interactions are equally irrelevant), leading to insufficient confidence in distinguishing true negative samples from potential positive ones. Inspired by term frequency sampling in natural language processing [18] and node degree distributions in graph learning [19], recent studies propose popularity-aware non-uniform sampling, where item popularity is leveraged to construct biased sampling distributions. This method increases the likelihood of sampling head items as negatives, effectively alleviating popularity bias in recommendations. Nevertheless, over-penalizing long-tail items during training may exacerbate the Matthew Effect and degrade recommendation diversity [20].

B. Loss Function

The core objective of collaborative filtering is to enable the model to accurately grasp user preferences, and the key to achieving this lies in carefully designing the loss function to guide the model's learning direction. Depending on the differences in learning objectives, collaborative filtering loss functions can be broadly categorized into three types: point-wise loss, pair-wise loss, and list-wise loss [21]. Point-wise loss focuses on the model's independent prediction of a user's preference for a single item, such as predicting click-through rates or specific ratings (e.g., binary cross-entropy [22], mean squared error [23]). Its advantage lies in simplicity and efficiency, but its independent optimization nature leads the model to focus solely on fitting individual user-item pairs, ignoring the relative relationships between items. This "isolated learning" paradigm is highly susceptible to the "popularity bias"—where the model tends to recommend items with high exposure rather than accurately capturing users' true preferences. In order to overcome the above-mentioned defect of point-wise loss, pair-wise loss requires the model to perform relative ranking on a pair of items (a positive sample and a negative sample). For example, BPR loss [4] maximizes the score difference between positive and negative samples, enabling the model to learn that "users prefer positive samples over specific negative samples." This approach shifts from "absolute prediction" to "relative comparison," initially alleviating the popularity bias issue. However, pair-wise loss still has significant limitations: each comparison involves only two items, making it unable to model the user's global ranking intent for all items. It's like trying to infer a player's ranking based solely on scattered match clips, which fails to ensure the overall rationality of the ranking [14]. In order to further break through the local perspective limitation of pair-wise loss, list-wise loss expands the optimization goal to global ranking, requiring the model to place preferred items before all others. The ideal solution, Softmax loss, achieves this through full-item probability normalization, but its computational complexity is linearly related to the number of items, making it impractical in scenarios with millions of items [24]. To address this, researchers have proposed two improvement ideas: one is negative sampling contrastive learning, which randomly selects a small number of negative samples to replace full-item computation; the other is margin constraint, which requires the similarity of positive samples to exceed that of negative samples by a certain threshold. Therefore, how to select

appropriate negative samples to optimize the loss function can be a direction for improving recommendation performance.

III. PICREC 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. 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 PICRec model is shown in Fig. 1. First, the initial embeddings of users and items are obtained based on the user - item bipartite graph. The interaction matrix of users and items is used to perform matrix multiplication with the initial ID embeddings, thereby enhancing the initial embeddings. Next, the interaction frequency of each item among all users is counted to calculate the item popularity, and negative samples are filtered according to the item popularity distribution. Then, through the improved contrastive loss function, the difference in scores between positive and negative samples is maximized. By selecting the negative samples most similar to the user, the model gradually learns the user's true preferences and accelerates the model's training. Finally, the primary task and the auxiliary task are jointly learned to update the user and item embeddings.

In the primary task, model employs the NSE-LightGCN [25] model to propagate information on the user-item interaction graph. It linearly transmits the embedded representations of users and items and aggregates node information to generate the final user and item embeddings. To optimize the model's performance, an improved contrastive loss function is utilized, enabling the model to gradually capture users' true preferences. Meanwhile, the auxiliary task introduces item popularity regularization loss to constrain the learning process of item embeddings. Ultimately, by jointly optimizing the primary and auxiliary tasks, the system can collaboratively update the embedded representations of users and items, thereby enhancing the recommendation effect.

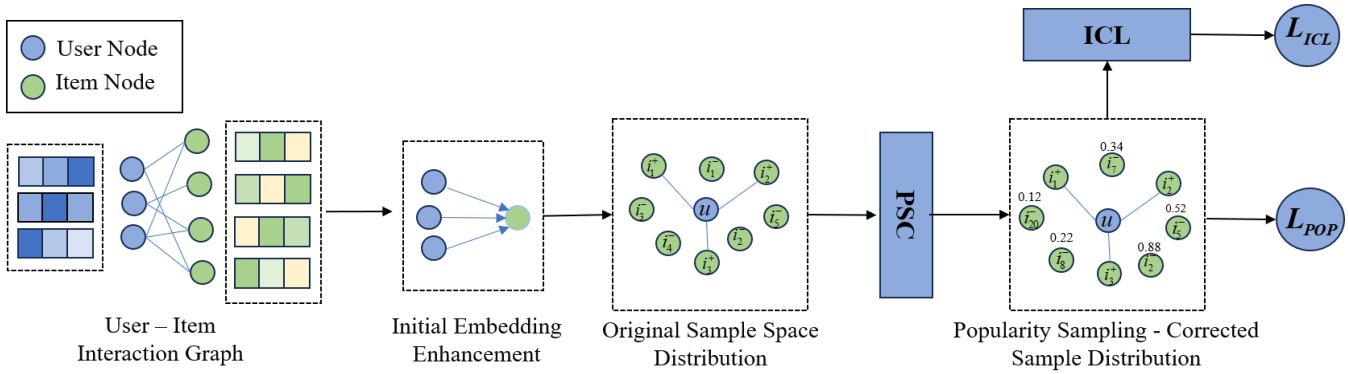

Fig. 1. PICRec overall framework.

C. NSE-LightGCN Graph Encoder

In this paper, the encoder adopts an improved NSE-LGCN [25] with LightGCN [26] as the backbone. Its initial embeddings are fused with the information of first - order neighbors, encoding the local topological information of nodes and can be regarded as a semantic enhancement of the initial ID embeddings. By aggregating the representations of neighboring nodes, it aims to enable the propagated representations to gain a certain structural understanding before message passing, making the implementation more efficient. Specifically, given the node representation matrix and the interaction matrix R, matrix multiplication is performed to obtain the user structural embedding $\hat{Z}_{\mathrm{U}}$ and the item structural embedding $\hat{Z}_{\mathrm{I}}$.

$$\hat{Z}_{\mathrm{U}}=RZ_{\mathrm{I}},\hat{Z}_{\mathrm{I}}=RZ_{\mathrm{U}} \tag{1}$$

The node representations enhanced via NSE already possess certain structural semantic information. In order to capture higher - order connectivity, a canonical message - passing scheme will be employed, and more meaningful node representations will be obtained by stacking multiple convolutional layers. The specific aggregation strategy and the formula for the propagation mechanism are as follows:

$$\hat{z}_u^{(l+1)}=\sum_{i\in N_u}\frac{1}{\sqrt{|N_u||N_i|}}z_i^{(l)} \tag{2}$$

$$\hat{z}_i^{(l+1)}=\sum_{i\in N_i}\frac{1}{\sqrt{|N_u||N_i|}}z_u^{(l)} \tag{3}$$

where l represents the number of convolutional layers, and $z_u^{(l)}$ and $z_i^{(l)}$ represent the user and item embeddings at the l - th layer, respectively. N_u denotes the set of items interacted with by user node u, and N_i denotes the set of users associated with item node i. Given that embeddings at different layers carry distinct semantics, the embeddings from different layers are weighted and combined. The embedding combination strategy is illustrated in Eq. (4) and Eq. (5):

$$z_u=\sum_{l=0}^{L}\gamma_l\hat{z}_u^{(l)} \tag{4}$$

$$z_i = \sum_{l=0}^{L} \gamma_l \hat{z}_i^{(l)} \tag{5}$$

where z_u and z_i denote the user embedding and item embedding in the l-th layer, respectively. γ_l is the weight for each layer, and L is the number of convolutional layers.

D. Popularity Sampling Correction Strategy

Negative sampling strategies have a crucial impact on the training of collaborative filtering models. Traditional methods usually adopt uniform negative sampling based on item popularity, which implicitly assumes that users have an equal negative attitude towards uninteracted items. However, this paradigm has significant flaws: the over - exposure of high - frequency items in negative samples can distort the model's perception of users' true preferences, leading the recommendation system into the dilemma of popularity bias. That is, the model tends to overestimate the negative correlation of popular items while underestimating the positive potential of long - tail items. To reduce popularity bias, inspired by the literature [27, 28], we propose a popularity - corrected negative sampling strategy (PSC). First, the original popularity of items is smoothed to reduce the impact of extreme values on the sampling process. Subsequently, the sampling probability is further optimized through exponential adjustment to ensure the diversity and representativeness of negative samples. By constraining the distribution of negative samples during the training process through item popularity regularization loss, the effectiveness of negative samples is further controlled, effectively reducing popularity bias and enhancing the model's recommendation ability for long - tail items, thereby improving the performance of the overall recommendation system and user satisfaction. The execution process is as shown in Fig. 2.

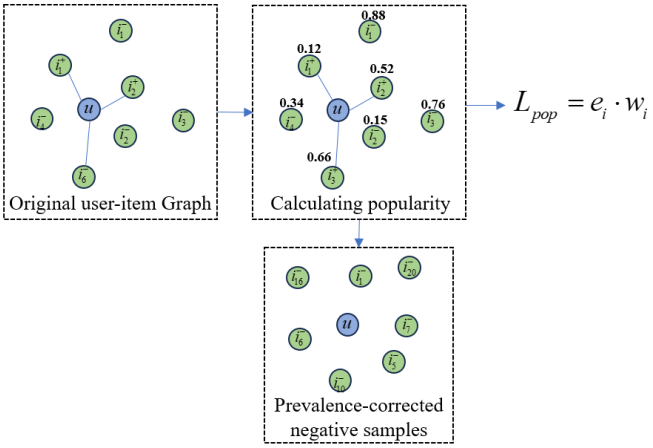

Fig. 2. Popularity Sampling Correction (PSC).

1) *Calculation of popularity*: Popularity is an important metric for measuring the popularity of items, and is typically calculated based on the number of interactions with the item. In this study, the popularity of each item is first calculated, and then the popularity is smoothed through a logarithmic transformation. Specifically, for each item, its popularity $p(i)$ is calculated as the logarithm of the number of interactions with the item. The formula is as follows:

$$p(i) = \log(count(i) + 1) \tag{6}$$

To avoid the impact of items with excessively high popularity on training, an exponential adjustment is made to item popularity, using the following formula:

$$\hat{p}(i) = p(i)^{-\alpha} \tag{7}$$

where α is the popularity adjustment index, which is tuned through experiments to control the impact of popularity on negative sampling. Subsequently, normalization is performed to avoid sampling bias or computational instability issues caused by values that are too large or too small. The formula is as follows:

$$p(i) = \frac{\hat{p}(i)}{p(i)} \tag{8}$$

2) *Conflict sample optimization*: In traditional negative sampling methods, negative samples are usually obtained by randomly selecting items for sampling. However, this approach may lead to conflicts between negative and positive samples, thereby affecting model training. To address this issue, we introduce a masking mechanism to avoid conflicts between negative and positive samples. Specifically, we first select negative samples based on the item popularity distribution using a sampling method. Suppose we select b negative samples from the item set, denoted a $\{i_1, i_2, ..., i_b\}$, with their popularity given by the adjusted values. If a negative sample conflicts with a positive sample ($i_j = i_{pos}$), we then reselect the negative sample through the masking mechanism until all negative samples do not conflict with positive samples. The mathematical expression of this process is as follows:

$$mask(i_j) = \begin{cases} 1, & \text{if } i_j = i_{pos} \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

Check the mask values of all negative samples; if a negative sample is the same as a positive sample and the mask value is 1, then resample the negative sample until there is no conflict. This ensures that negative and positive samples will not conflict, avoiding interference during the training process.

3) *Popularity weight regularization*: In recommendation systems, users and items are typically represented by high - dimensional embedding vectors. However, high - dimensional embedding vectors may lead to model overfitting. Therefore, the core idea of regularization loss is to penalize embedding vectors with large values to prevent the model from over - relying on certain specific features during training. To improve the model's generalization ability, we impose constraints on the embedding vectors of users and items, thereby reducing unnecessary complexity and prompting the embedding vectors to maintain a smaller scale.

To further optimize the performance of recommendation systems, we introduce popularity information to weight the regularization loss. In practice, items with high popularity usually have more interaction records and are followed by more users. Therefore, we impose stronger regularization on the embedding vectors of items with high popularity to avoid overfitting of these high - popularity items. Specifically, we calculate the weighted regularization weight based on item popularity. The weighted regularization weight $w(i)$ of item i can be calculated through its popularity $\hat{p}(i)$, and the weight formula is as follows:

$$w(i) = \frac{\hat{p}(i)}{\sum_j \hat{p}(j)} \tag{10}$$

By increasing the regularization strength of these popular item embedding vectors, the model can pay more attention to other items that may have potential value, thereby improving the diversity and accuracy of recommendations. Finally, the regularization loss function with popularity weighting can be expressed as:

$$L_{pop} = \sum_{i \in I} \| z_i \|^2 \cdot w(i) \tag{11}$$

$$L_{reg} = \eta_u \sum_u \| z_u \|^2 + L_{pop} \tag{12}$$

E. Improved Contrastive Learning Loss

Traditional loss functions, such as Bayesian Personalized Ranking (BPR) loss and Sampling Soft Maximum Cross - Entropy (SSM) loss, have to some extent enhanced the performance of recommendation systems. However, they usually suffer from improper negative sample selection and a slow training process. To better address these challenges, inspired by the literature [29, 30], this paper proposes a new loss function. It aims to accelerate the training process and improve recommendation quality by optimizing the negative sample selection strategy and maximizing the score difference between positive and negative samples. We first calculate the scores of each positive and negative sample based on the embedding vectors of users and items. Then, we define the loss value by maximizing the difference between the scores of positive and negative samples. If the score difference between positive and negative samples is less than the set threshold, the loss will be calculated; otherwise, the loss is zero. The process is as shown in Fig. 3:

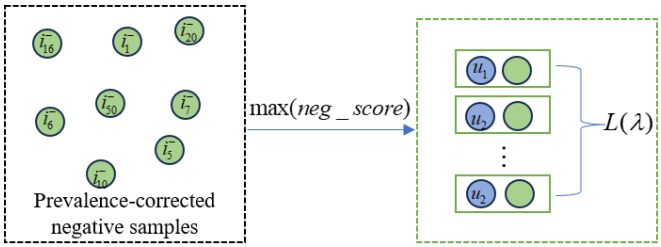

Fig. 3. Improved Contrasting Learning (ICL).

After the message propagation and aggregation mechanism of the GNN encoder, the final user and item embeddings are obtained. For the users and items in the test set, the scores of their interactions are predicted, with higher scores indicating a greater degree of user interest in the item. The calculation process for the positive sample predicted score is as shown in Eq. (13):

$$pos\_score = z_u \cdot z_{i^+} \tag{13}$$

where z_u is the embedding vector of user u , and z_{i^+} is the embedding vector of the positive sample. The result of the inner product represents the match degree between the user and the item.

In order for the model to focus on training the difficult negative samples, the calculation process for the negative sample score involves selecting the negative sample that is most similar to the user embedding from the popularity - weighted negative samples for training. This process helps to increase the training difficulty of the model and improve its generalization ability. The calculation process for the negative sample predicted score is as shown in the formula:

$$neg\_score = \max(z_u \cdot z_{i^-}) \tag{14}$$

where z_{i^-} is the embedding vector of the negative sample.

When calculating the loss, only the most difficult negative samples are considered, while those that are easy to distinguish are ignored. This strategy enables the model to converge more quickly and improve the quality of recommendations. The selection of negative samples and the calculation of their scores are interrelated; only by accurately selecting the most difficult negative samples can the model truly learn to distinguish between positive and negative samples.

To ensure sufficient discriminability between positive and negative samples, we set a margin λ . Specifically, the model is penalized only when the score of the positive sample is lower than that of the negative sample, and the difference between the two is less than the set margin value λ . Otherwise, the loss value is zero. This strategy ensures that the model optimizes only the differences between positive and negative samples that are meaningful, thereby effectively enhancing the model's learning efficiency and ultimately its recommendation performance. The specific loss calculation formula is as follows:

$$\begin{aligned} L_{ICL} &= \max(\lambda - pos\_score + neg\_score, 0) \\ &= \max\{\lambda - e_u \cdot e_{i^+} + \max(e_u \cdot e_{i^+}), 0\} \end{aligned} \tag{15}$$

The loss function L_{ICL} , through the selection of the most difficult negative samples and the optimization of score differences, encourages the model to better learn to distinguish users' preferences for positive and negative items. The total loss for a batch of N samples is the average of the losses of all samples:

$$L_{ICL} = \frac{1}{N} \sum_{n=1}^{N} loss_n \tag{16}$$

where $loss_n$ is the loss value of the n-th sample. By averaging the losses of all samples, the loss function L_{ICL} can balance the contribution of each sample to the final model performance, enabling it to better guide the model in learning the potential differences between positive and negative samples.

F. Pseudo-Code of the Model

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

TABLE I PSEUDO-CODE OF PICREC

Algorithm: PICRec

Input: User - Item Interaction Data .inter File
Output: Predicted Scores of Target Users for Items
While PICRec Not Convergence **do**
  for x in Data **do**
    Count interaction frequency, calculate and smooth and normalize popularity;
    Filter negative samples based on the distribution of item popularity;
    Conflict Sample Processing;
    Generate user final embedding and item final embedding
    Calculate the item popularity regularization loss
    Calculate Improved Contrastive Loss: Select the most difficult negative samples and introduce a threshold;
    Calculate the total loss and optimize the model;
  end for
end while

IV. 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 effectiveness of the recommendation model presented in this paper on datasets with different scenarios, scales, and sparsity levels, experiments were conducted using three publicly - available datasets: Amazon – Books[31], Yelp①, and Gowalla [32]. The ratio of the training set, validation set, and test set is 8:1:1, and the dataset information is shown in Table II.

TABLE II STATISTICS FOR THE DATASETS

Data type	Amazon-Book	Yelp2018	Gowalla
Number of users	58144	45477	29858
Number of items	58051	30708	40988
Interactive data	2517437	1777765	1027464
Data density	0.075%	0.127%	0.084%

① https://www.yelp.com/dataset

3) Evaluation indicators: This study employs Recall@K, NDCG@K, MRR@K, and Hit@K as evaluation metrics for top-K recommendations, with K set to 10.

Recall measures the system's ability to cover users' true interests, particularly suitable for evaluating the exposure effectiveness of long-tail items.

NDCG quantifies ranking quality through position-based discounting, reflecting the practical utility of the recommendation list.

MRR emphasizes the accuracy of the first relevant result, applicable to real-time feedback scenarios such as search engines.

Hit adopts a binary evaluation to assess whether the recommendation list captures user interests, offering an intuitive reflection of basic coverage.

This combination of metrics comprehensively addresses recommendation coverage, ranking quality, real-time responsiveness, and foundational performance. These metrics align closely with the objectives of this study—mitigating popularity bias and optimizing contrastive learning—where higher metric values indicate superior recommendation performance.

4) Baseline modelling and parameter setting: In order to verify the effectiveness and superiority of PICRec, we selected several existing state - of - the - art collaborative filtering models for comparison, namely NSE-LGCN [25], LightGCN [26], MultiGCCF [33], and DGCF [34]. The relevant parameter settings are as follows: the batch size is 4096, Xavier is used as the default initialization method for all parameters, the number of convolutional layers is 3, and the learning rate is 0.001.

B. Results of the Experiment

In order to more intuitively compare the performance of different models, Tables III and IV are presented. PICRec's results are in bold, and the best benchmark performance is underlined. $*$ indicates statistical significance ($p<0.05$) compared to the best baseline. For implemented models, we reuse the results reported in previous work [25].

TABLE III MODEL PERFORMANCE COMPARISON 1

Method	Amazon-Books		Yelp		Gowalla	
	MRR	Hit	MRR	Hit	MRR	Hit
NSE-LGCN	<u>0.087</u>	<u>0.2091</u>	<u>0.0998</u>	<u>0.2246</u>	<u>0.1275</u>	<u>0.2691</u>
PICRec	0.1059*	0.2379*	0.1038*	0.2295*	0.1416*	0.2886*
Improve	21.7%	13.7%	4.01%	2.18%	11.5%	7.24%

TABLE IV MODEL PERFORMANCE COMPARISON 2

Method	Amazon-Books		Yelp		Gowalla	
	Recall	NDCG	Recall	NDCG	Recall	NDCG
MultiGCCF	0.0625	0.0433	0.0646	0.0450	0.1108	0.0791
DGCF	0.0737	0.0521	0.0723	0.0514	0.1252	0.0902
LightGCN	0.0844	0.0603	0.0790	0.0573	0.1344	0.0963
NSE-GCN	0.0885	0.0631	0.0830	0.0610	0.1362	0.0967
PICRec	0.1034*	0.0762*	0.0937*	0.0710*	0.1465*	0.1064*
Improve	16.8%	20.6%	12.9%	16.3%	7.56%	10.0%

Multi - GCCF proposes a method that constructs user - item interaction graphs, user - user graphs, and item - item graphs. By combining different aggregation and transformation functions, it explicitly handles high - order information and similarities between users and items, enhancing the model's embedding space representation capability. DGCF demonstrates that decoupling complex user intents can better model user interest preferences and improve model interpretability. It iteratively optimizes user intents to more efficiently extract relevant information for each intent to train the model. LightGCN linearly propagates user and item embeddings on the user - item bipartite graph interaction data to learn user preference information. It removes feature transformation and nonlinear activation modules to enhance the performance of traditional collaborative filtering models. Building on this, NSE - LGCN utilizes first - order adjacency information to construct structural embeddings. During the propagation process, each node can maintain its own characteristics, effectively distinguishing itself from other nodes and alleviating the over - smoothing problem.

The PICRec proposed in this paper has achieved significant improvements in evaluation metrics on three datasets compared to mainstream recommendation models. The most notable increase was on the least sparse Amazon-Books dataset, where Recall and NDCG increased by 16.8% and 20.7% respectively, indicating that PICRec can effectively address the data sparsity issue. Compared to other models, the advantage of PICRec lies in the proposed PSC module, which uses popularity parameters to weight the sampling of item popularity, reducing the probability of selecting negative samples of popular items. This allows the model to focus more on niche items during training, enhancing the diversity and personalization of recommendations. In addition, the improved contrastive loss function sets a minimum margin between positive and negative samples and selects the most difficult negative samples to maximize the score difference between positive and negative samples. This encourages the model to better learn to distinguish users' preferences for positive and negative items.

C. Ablation Experiments

1) Validation of PSC and ICL component effectiveness: In order to verify the effectiveness of the PSC and ICL components, two variant models, PICRec - PSC and PICRec - ICL, were designed for ablation studies. First, to verify the effect of the PSC component, the variant model PICRec - PSC was designed. This model removes the PSC module and uses random sampling for experiments. Second, the variant model PICRec - ICL was designed, which uses the BPR loss function for training. The Amazon-Books and Yelp-2018 datasets were used for this experiment, and the results are shown in Fig. 4.

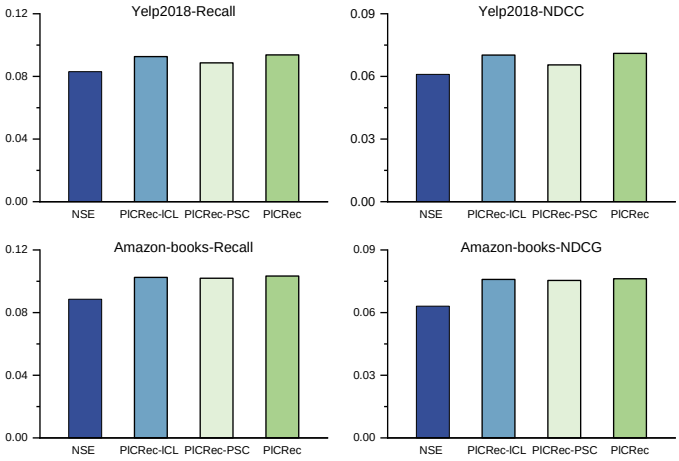

Fig. 4. Effectiveness analysis of PSC and ICL.

As depicted in Fig.4, the PICRec model and its variant models outperform the NSE model across all metrics, demonstrating the necessity of each component. The PICRec model's superior metrics compared to the PICRec-PSC variant highlight the effectiveness of the PSC component. The model can adjust the sampling probability of popular items, reducing their frequency in negative samples, which prompts the recommendation system to better focus on cold items, thereby enhancing diversity and the recommendation performance of long-tail items. Furthermore, the PICRec model's metrics surpass those of the PICRec-ICL variant, indicating that the ICL component can optimize the relative distance between positive and negative samples and sample hard negative samples to prompt the model to more accurately learn the underlying relationships between users and items.

2) Parameter analysis: Performance Comparison Regarding Parameter α . Parameter α , which represents the adjustable item popularity weight, is primarily used in recommendation systems to regulate the distribution of items. By conducting weighted sampling based on item popularity and performing appropriate smoothing, the popularity parameter helps optimize the selection of negative samples and balance the recommendations between popular and niche items. This prevents the model from over - focusing on popular items, enhances the recommendation quality of niche items, improves the model's training process, and thus boosts the overall recommendation performance. The impact of the popularity weight parameter α on the recommendation results is shown in the Fig. 5.

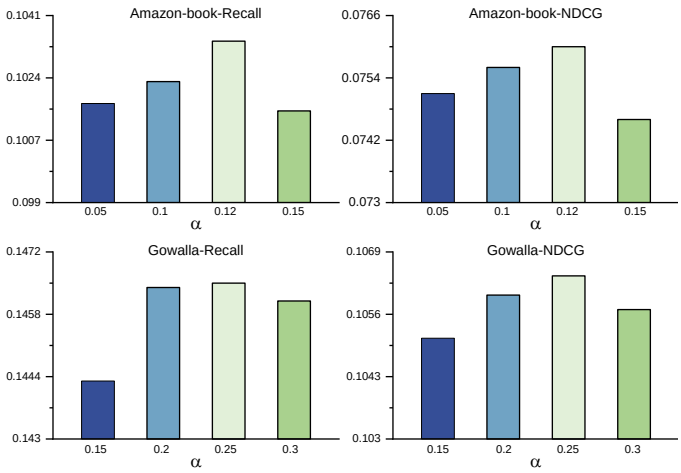

Fig. 5. Effectiveness analysis of α .

As shown in Fig. 5, when the popularity weight is too high, the recommendation system tends to over - recommend items with high popularity. This may lead the recommendation system into an "information cocoon," where only similar popular items are recommended to users, while niche or emerging items are ignored, resulting in a lack of diversity and novelty in recommendations. When the popularity weight is set too low, the recommendation system may overlook the impact of popular items, over - emphasizing the recommendation of niche items and neglecting users' potential interest in most popular items, which may fail to meet users' basic needs. When the popularity weight is set to 0.12 and 0.25, the model demonstrates excellent performance on the Amazon - Books and Gowalla datasets.

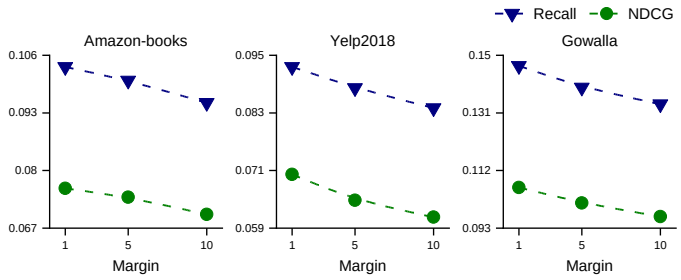

Fig. 6. Effectiveness analysis of margin.

Performance Comparison Regarding Parameter γ, An important hyperparameter in our improved contrastive loss is the boundary threshold γ, which controls the relative importance between positive and negative sample losses. A larger γ would lead to a greater difference between the losses of positive and negative samples. This might cause the model to too "loosely" ignore the contributions of some negative samples, thereby affecting the model's training and final performance. We used three different values [1.0, 5.0, 10.0] of γ on three large - scale datasets to check its effect. The number of experimental epochs was 500, and the experimental results are shown in Fig. 6.

As can be seen from Fig. 6, when the parameter is set to 1, the model demonstrates excellent performance on the two datasets, Amazon - Books, Yelp2018, and Gowalla.

V. Conclusion

This paper proposes a recommendation algorithm based on popularity bias correction sampling and improved contrastive loss (PICRec). The introduced PSC module dynamically adjusts the sampling distribution through logarithmic smoothing and inverse power-law transformation, balancing the exposure rate of long-tail items, and significantly alleviating the popularity bias issue in recommendation systems. Additionally, the ICL module controls the distance between positive and negative samples via a threshold, prompting the model to increase the similarity of positive samples while reducing that of negative samples. It optimizes the training process using the most challenging hard negative samples, enhancing the model's ability to distinguish between positive and negative samples, making the boundary between them clearer and effectively improving the model's capacity to fit user preferences. Experiments on three public datasets demonstrate the effectiveness and advancement of this algorithm.

For future research directions, we plan to explore the following aspects in depth:

Multimodal Information Fusion: The current model primarily utilizes user-item interaction data. In the future, we will explore how to effectively integrate item content features, social network information, and temporal data to design sampling strategies and contrastive learning paradigms specifically for different modal information, further enhancing the model's representational capabilities. Specifically, we will investigate how to incorporate content similarity factors into the PSC module, making the sampling process consider both popularity and content relevance.

Cross-Domain Recommendation Applications: We plan to extend the PICRec model to cross-domain recommendation scenarios, studying how to leverage popularity distribution information from the source domain to assist in designing sampling strategies for the target domain. Particularly in cold-start situations, we will explore how to effectively transfer sampling knowledge from the source domain to accelerate model convergence in the target domain.

Dynamic Threshold Mechanism: The current ICL module uses a fixed threshold to control the distance between positive and negative samples. In future work, we will research and design an adaptive threshold mechanism that dynamically adjusts threshold parameters based on user interaction history and item characteristics to accommodate different user groups and recommendation scenarios in various domains. Specifically, we will explore using user activity level and item popularity as regulatory factors to construct personalized threshold functions.

Through in-depth research in these directions, we expect the PICRec model to be further developed and refined on both theoretical and practical levels, providing more valuable insights and methods for the field of recommender systems research.

REFERENCES

[1] Liu T H, Yang X X, Zhou H, et al. A survey of collaborative filtering recommender algorithms based on graph neural networks [J]. Journal of Integration Technology, 2024, 13(4): 1-15.

[2] Wei T R, Fang Y. Diffusion Models in Recommendation Systems: A Survey[J]. arXiv preprint arXiv: 2501.10548, 2025.

[3] Park S, Yoon M, Park H, et al.Toward a Better Understanding of Loss Functions for Collaborative Filtering[C]//Proceedings of the 32nd ACM International Conference on Information and Knowledge Management.Birmingham:ACM,2023:2034-2043

[4] Rendle S, Freudenthaler C, Gantner Z, et al. BPR: Bayesian personalized ranking from implicit feedback[C]//Proceedings of the 25th conference on uncertainty in artificial intelligence. USA: AUAI Press, 2009:452–461.

[5] Rendle S, Freudenthaler C. Improving pairwise learning for item recommendation from implicit feedback[C]//Proceedings of the 7th ACM international conference on Web search and data mining. USA: ACM, 20014: 832 – 841.

[6] Togashi R, Otani M, Satoh, S. Alleviating cold-start problems in recommendation through pseudo-labelling over knowledge[C]// Proceedings of the 14th ACM International Conference on Web Search and Data Mining. New York: USA, 2021: 931-939.

[7] 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. New York: ACM, 2021: 665-674.

[8] Petrov A, Macdonald C. Effective and Efficient Training for Sequential Recommendation using Recency Sampling[J].ACM Transactions on Recommender Systems, 2025, 3(1):1-32.

[9] Shi K X, Zhang Y, Jing B Y, et al.Soft BPR Loss for Dynamic Hard Negative Sampling in Recommender Systems[J].arXiv preprint arXiv:2211.13912, 2022.

[10] Xue Y, Cai X D, Fang S, et al. Contrastive Learning and Multi-choice Negative Sampling Recommendation[J]. Contrastive Learning and Multi-Choice Negative Sampling Recommendation, 2025:15(5)

[11] Chen H Y, Lai V V , Jin H Y, et al. Towards mitigating dimensional collapse of representations in collaborative filtering[C]// Proceedings of the 17th ACM International Conference on Web Search and Data Mining. New York : ACM, 2024:106–115.

[12] Jin H Y, Han X T, Yang J F, et al. Llm maybe longlm: Self-extend llm context window without tuning[J]. arXiv preprint arXiv:2401.01325, 2024.

[13] Hsieh C K, Yang L Q, Cui Y, et al. Collaborative metric learning[C]//In Proceedings of the 26th international conference on world wide web. Republic and Canton of Geneva: International World Wide Web Conferences Steering Committee. 2017: 193–201.

[14] Mao K L, Zhu J M, Wang J P, et al. SimpleX: A Simple and Strong Baseline for Collaborative Filtering[C]//Proceedings of the 30th ACM International Conference on Information & Knowledge Management. New York: ACM, 2021:1243–1252.

[15] Chen T, Sun Y Z, Shi Y, Hong L J. On Sampling Strategies for Neural Network-based Collaborative Filtering[C]//Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.New York:ACM, 2017: 767–776.

[16] Wu G, Volkovs M, Soon C L, et al. Noise Contrastive Estimation for One-Class Collaborative Filtering[C]//Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval.New York:ACM, 2019: 135–144.

[17] Yu J L, Yin H Z, 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.New York: ACM, 2022:1294–1303.

[18] Grover A, Leskovec J. Node2vec: Scalable Feature Learning for Networks[C]//Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York:ACM, 2016: 855–864.

[19] Mikolov T, Sutskever I, Chen K, et al. Distributed Representations of Words and Phrases and Their Compositionality[C]//Proceedings of the 27th International Conference on Neural Information Processing Systems. Red Hook: Curran Associates Inc, 2013:3111–3119.

[20] Chen J W, Dong H D, Wang X, et al. Bias and Debias in Recommender System: Survey and Future Directions[J].arXiv preprint arXiv:2010.03240, 2020.

[21] Chen H Y,Lai V V, Jin H Y, et al. Towards mitigating dimensional collapse of representations in collaborative filtering[C]//Proceedings of the 17th ACM International Conference on Web Search and Data Mining. New York: ACM, 2024:106–115.

[22] He X N, Liao L Z, Zhang H W, et al. Neural collaborative filtering[C]//Proceedings of the 26th International Conference on World Wide Web. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva: International World Wide Web Conferences Steering Committee, 2017:173–182.

[23] Chen C, Zhang M, Zhang Y F, et al. Efficient Neural Matrix Factorization without Sampling for Recommendation[J].ACM Transactions on Information Systems (TOIS) , 2020, 38(2) :1–28.

[24] Covington P, Adams J, Sargin E. Deep neural networks for youtube recommendations[C]//Proceedings of the 10th ACM conference on recommender systems. New York:ACM, 2016: 191–198.

[25] Jin X Z, Li J T, Xie Y Z, et al. Enhancing Graph Collaborative Filtering via Neighborhood Structure Embedding[C]//The 2023 IEEE International Conference on Data Mining. China:IEEE, 2023: 190-199.

[26] 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. New York: ACM, 2020: 639-648

[27] Wu J C, Wang X, Gao X Y, et al. On the effectiveness of sampled softmax loss for item recommendation[J].ACM Transactions on Information Systems, 2024, 42(4):1-26.

[28] Ma H K, Xie R B, Meng L, et al. Negative Sampling in Recommendation: A Survey and Future Directions[J]. arXiv preprint arXiv:2409.07237, 2024.

[29] Mao A Q, Mohri M, Zhong Y T. Cross-entropy loss functions: Theoretical analysis and applications[C]//Proceedings of the 40th International Conference on Machine Learning. USA: JMLR, 2023:23803-23828.

[30] Yang X D, Chen H Y, Yan Y C. et al. SimCE: Simplifying Cross-Entropy Loss for Collaborative Filtering[J]. arxiv.org/pdf/2406.16170,2024.

[31] He R , McAuley J, Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering[C]//Proceedings of the 25th International Conference on World Wide Web. Republic and Canton of Geneva: International World Wide Web Conferences Steering Committee, 2016:507–517.

[32] Liang D, Charlin L, McInerney J, et al, Modeling user exposure in recommendation[C]//Proceedings of the 25th International Conference on World Wide Web ,Republic and Canton of Geneva: International World Wide Web Conferences Steering Committee, 2016: 951-961.

[33] Sun J N; Zhang Y X; Ma C, et al. Multi-graph convolution collaborative filtering[C]//2019 IEEE International Conference on Data Mining, China :IEEE, 2019: 1306-1311.

[34] Wang X, Jin H Y, Zhang A, et al. Disentangled graph collaborative filtering[C]//The 43rd International ACM SIGIR Conference on Research and Development in Information Retrieva. USA: ACM, 2020, 1001-101.