ISSN: 1813-4890

MicroGCL: A Micro Graph Contrastive Learning Method for Recommendation System

Junwei Cao [1], Junxiang Li [1,2], Chen Liu [1,*], Jinrui Zhu [1]

[1]Business School, University of Shanghai for Science and Technology, Shanghai 200093, China;

[2]School of Intelligent Emergency Management, University of Shanghai for Science and Technology, Shanghai 200093, China

* Corresponding Author

Abstract

Graph Neural Networks (GNNs) have been widely adopted to alleviate data sparsity in recommendation systems by leveraging their powerful capacity for graph structure modeling. Recommendation models based on graph contrastive learning (GCL) typically construct contrasting views through node dropping or edge perturbation, yet these strategies may compromise graph integrity and lead to information loss. To address this issue, this paper proposes a lightweight graph contrastive learning method named MicroGCL. By introducing a multi-view subgraph generation approach based on random node subsets and probabilistic filtering, MicroGCL achieves effective representation learning without relying on complex network architectures. Extensive experiments on multiple sparse recommendation datasets demonstrate that MicroGCL yields substantial improvements over existing GCL baselines, notably achieving gains of 3%–6% on Recall@20 and NDCG@20, confirming its strong generalization capability in data-sparse scenarios.

Keywords

Deep learning; recommendation algorithm; graph neural network; graph contrastive learning.

1. Introduction

In the era of big data, efficiently mining content that aligns with user interests from vast amounts of information has become a critical research topic in information retrieval and personalized recommendation. Recommender systems, which model user behavior data to provide personalized item suggestions, have been widely deployed across various domains, including e-commerce, social media, online education, and content platforms. However, real-world recommender systems face numerous challenges, such as interaction sparsity, cold-start problems, and interest drift, which significantly hinder recommendation performance. In recent years, Graph Neural Networks (GNNs) have been introduced into recommender systems due to their powerful capability in handling non-Euclidean structured data, emerging as a crucial tool for addressing data sparsity and structural modeling issues.

In recommendation tasks, user-item interactions can be naturally represented as a bipartite graph, where nodes correspond to users and items, and edges denote interaction records. GNNs propagate information through graph structures, enabling the aggregation of high-order neighbors, thereby mitigating the representational limitations of traditional methods[1]. For instance, PinSage[2] incorporates convolutional operations into user-item graphs, while LightGCN[3] demonstrates that removing nonlinear activations and weight matrices still yields superior recommendation performance, highlighting the importance of structural propagation for modeling user interests. Other models, such as DGCF[4], disentangle semantic factors to

capture diverse preferences; HyRec[5] leverages hyperbolic space to enhance hierarchical representation; and MHCN[6] integrates social and attribute information via multi-channel hypergraphs, further extending the applicability of GNNs in diverse recommendation scenarios. Moreover, in practical applications where user-item interactions are extremely sparse or new nodes frequently appear, GNNs leverage neighbor information to assist modeling, demonstrating robustness in sparse data and cold-start scenarios, thereby broadening their application boundaries in recommender systems.

Graph-based recommendation methods have achieved significant performance improvements on multiple public datasets, primarily due to their ability to exploit high-order correlations within graph structures, thereby enhancing the expressiveness of user and item representations. Building on this, researchers have further explored methods to improve node representations in unsupervised settings, reducing reliance on labeled data and improving model generalization. Graph Contrastive Learning (GCL), an emerging paradigm for unsupervised graph representation learning, has been widely adopted to enhance the expressiveness of node embeddings. The core idea of GCL is to construct different views of data and maximize the consistency of node representations across these views, enabling the capture of latent semantic information and learning more discriminative representations without labels[7]. This concept originates from the success of contrastive learning in computer vision and self-supervised learning, as demonstrated in models like SimCLR[8] and MoCo[9].

Graph contrastive learning has been rapidly developed and extended in recommender systems. Current mainstream methods typically generate multiple views by perturbing the original graph structure, such as randomly removing edges, subsampling nodes, or applying feature masking. For example, SGL[10] constructs positive and negative sample pairs by generating two subgraphs through neighbor sampling, while SimGCL[11] introduces Gaussian noise to simulate graph representations under different structures and designs simple yet effective regularization terms to enhance training stability. Although these methods achieve strong results on multiple benchmark datasets, they also reveal critical limitations. First, structural or feature perturbations inevitably discard information from the original graph, which is particularly problematic in recommendation scenarios where user-item interactions are inherently sparse—further perturbations often lead to information loss, degrading representation quality. Second, some methods rely on complex model designs or extensive parameter tuning, increasing training costs and deployment complexity.

To address these issues, this paper proposes MicroGCL, a lightweight graph contrastive learning method. Here, "GCL" stands for Graph Contrastive Learning, while "Micro" signifies the use of an extremely compact and lightweight network architecture. The core innovation of this method lies in constructing a random view generation strategy that does not require structural perturbations. By relying solely on node sampling, it generates view pairs with sufficient semantic variation while preserving structural integrity. During training, MicroGCL employs a contrastive learning objective to maximize the consistency of node representations between the original graph and the sampled graph, thereby learning robust and efficient representations. Compared to existing methods, MicroGCL is more concise in design, introduces almost no additional computational overhead, and exhibits enhanced stability and generalization in sparse recommendation scenarios.

The main contributions of this paper are summarized as follows:

1. We propose MicroGCL, a graph contrastive learning-based recommendation model that eliminates the need for edge perturbations or node masking, and validate its effectiveness on multiple sparse recommendation datasets.

ISSN: 1813-4890

2. We introduce a stochastic view generation method for MicroGCL, which constructs contrastive views through a generative strategy, minimizing information loss while preserving original content.

3. We design MicroLoss, a contrastive learning loss function specifically tailored for MicroGCL, to better optimize representation learning on sparse graph data. This loss function integrates structural consistency and semantic preservation constraints, significantly enhancing the model's robustness and generalization capability.

2. Related Work

In recommender systems, graph neural networks and contrastive learning have emerged as important methods for improving recommendation performance. This section will introduce graph neural network-based recommendation methods and the application of graph contrastive learning in recommender systems, providing theoretical and technical background support for the approach proposed in this paper.

2.1. Garph Neural Networks

Graph Neural Networks (GNNs) constitute a class of deep learning models specifically designed for processing graph-structured data, which have garnered extensive attention and in-depth research across numerous fields in recent years. Compared to traditional neural networks, GNNs can effectively model the relational structures between nodes and integrate the features of nodes and their neighbors during information propagation, achieving efficient representations of graph-structured data.

As a type of non-Euclidean structure, graphs are prevalent in the real world, such as user relationship graphs in social networks, user-item interaction graphs on e-commerce platforms, protein-protein interaction networks in bioinformatics, and road networks in transportation systems. Traditional Convolutional Neural Networks (CNNs) have limitations when processing such data. GNNs, however, employ a message-passing mechanism that enables nodes to aggregate information from their neighbors layer by layer, thereby obtaining more expressive node embeddings. One of the earliest representative graph convolution models is GCN[12], which simplifies spectral graph convolution to a local first-order approximation. This allows for feature aggregation operations on each node and its immediate neighbors, significantly enhancing the representational capacity for tasks like node classification on graph data, and establishing GNNs as a foundational method for graph learning. However, due to its full-graph training approach, this model suffers from poor scalability and over-smoothing issues, limiting its application on large-scale graphs. To alleviate the computational bottlenecks of GCN in large-graph scenarios, GraphSAGE[13] introduces neighbor sampling and learnable aggregation functions, enabling inductive learning on graph structures and supporting the generation of representations for unseen nodes. It also systematically analyzes the impact of different aggregation functions on graph representations, laying the groundwork for subsequent large-scale graph learning. GAT computes weights for adjacent nodes using a self-attention mechanism, dynamically capturing the importance of different neighbors, which effectively enhances the model's expressive flexibility.

Furthermore, to improve training efficiency, FastGCN[14] employs importance sampling on node neighbors, transforming the overall training process into mini-batch processing and significantly reducing memory overhead. GraphSAINT[16] adopts a random walk-based subgraph sampling method, preserving the graph's structural information while controlling training complexity. Cluster-GCN[17] utilizes graph clustering algorithms to partition a large graph into smaller, denser clusters for training, further improving the model's efficiency and accuracy on large-scale graph data.

ISSN: 1813-4890

In the context of recommender system applications, PinSage[2] combines the ideas of GraphSAGE [13][13] with random walk mechanisms to construct high-order connections between users and items. It employs an edge-weighting mechanism to model user preferences, becoming a paradigm for GNNs in industrial-scale recommender systems. LightGCN[3], by removing nonlinear activations and feature transformation operations and retaining only adjacency matrix propagation and weighted aggregation of node embeddings, demonstrates that a simplified structure actually improves recommendation performance while effectively reducing model complexity, indicating that overfitting is a key issue in deep GNNs.

However, while graph neural networks have provided a novel modeling paradigm for recommender systems, enabling deep mining of user-item relationships through effective aggregation of node and neighbor information and thereby advancing the development of recommendation algorithms, their performance gains have gradually plateaued. It has become increasingly difficult to achieve significant performance improvements by merely deepening network structures or designing more complex aggregation functions. To further exploit the information within graph structures and enhance model generalization capabilities, researchers have begun to introduce the concept of contrastive learning into graph neural networks.

2.2. Contrastive Learning

The initial introduction of contrastive learning, serving as an unsupervised alternative to established supervised learning paradigms, catalyzed significant advancements in the field of computer vision. Research in self-supervised learning for representation learning has since progressed rapidly, with numerous contrastive learning-based methods being proposed, substantially boosting the performance of unsupervised representation learning.

InstDisc[18] is a contrastive learning method based on instance discrimination. It pioneeringly achieves self-supervised learning by formulating an instance discrimination task where each image is treated as a distinct class of its own. CPC[19] introduces a mechanism for predicting future information, learning meaningful representations by maximizing mutual information between different data segments (e.g., context and future predictions), demonstrating high representational power. BYOL[20] proposes an innovative, non-contrastive learning strategy that employs a dual-network architecture with an asymmetric structure (target and online networks) for representation learning. It achieves strong performance without requiring negative samples or a momentum encoder. SimSiam[21] takes this further by discarding negative samples and momentum mechanisms entirely, using a stop-gradient operation within a simple architectural design to achieve efficient self-supervised representation learning. MoCo v3[22], as the latest method in the MoCo series, integrates Transformer architectures, extending self-supervised learning to a wider variety of network structures.

Operating within the framework of self-supervised contrastive learning, these methods employ diverse strategies to enhance the quality of unsupervised representations, thereby laying a solid foundation for image representation learning. Subsequently, a growing body of research has progressively extended contrastive learning methods to recommendation algorithms, where they have also yielded promising results. CL4SRec[23] designs a sequential recommendation model that models preference changes within user behavior sequences through contrastive learning and data augmentation, thereby improving the capture of dynamic user interests. SCL[24] proposes a session-based recommendation model that captures users' short-term interest preferences by generating different session views and applying contrastive learning. LightGCL[25] simplifies the graph contrastive learning structure by employing Singular Value Decomposition (SVD) to generate views, mitigating the susceptibility to noise perturbations often caused by node-level augmentations.

ISSN: 1813-4890

Despite the effectiveness demonstrated by existing graph contrastive learning methods in recommender systems, issues such as information loss and model complexity persist. To address these challenges, this paper proposes a novel, lightweight graph contrastive learning method called MicroGCL. It does not rely on structural perturbations; instead, it constructs contrastive views through a random view generation mechanism, thereby reducing information loss and eliminating strategies based on edge perturbation or node deletion.

Furthermore, MicroGCL requires no additional complex network architectures. By utilizing a simple view sampling mechanism, it achieves excellent performance in recommendation tasks.

3. Methodology

In this section, we provide a detailed introduction to the proposed MicroGCL framework. This framework constitutes a lightweight graph contrastive learning paradigm, which is formally defined in Figure 1. We first present the algorithm employed for view generation, followed by a description of the framework used for graph contrastive learning. Finally, we introduce MicroLoss, the contrastive loss function specifically designed for this framework.

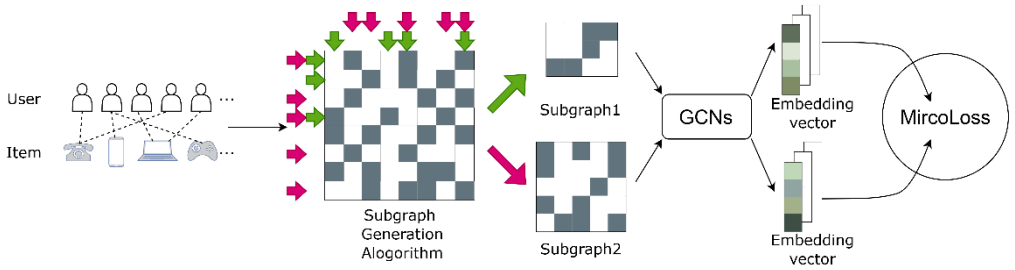

Fig.1 Overall structure of MicroGCL

3.1. Multi-View Subgraph Generation Algorithm

In recommender system research, the interactions between users and items are typically represented by a sparse matrix $Y \in \mathbb{R}^{J\times K}$, where J and K denote the number of users and items, respectively. The matrix Y contains only binary entries, i.e., $Y_{j,k} \in \{0,1\}$, where a value of 1 indicates an interaction between user j and item k, and 0 indicates no interaction. To construct local subgraphs for subsequent graph contrastive learning, this paper proposes a submatrix generation method based on random sampling.

Specifically, we first sequentially select the first $n+1$ rows from the matrix Y, starting from the first row, where n is a natural number. Let the index set of these selected rows be $\{1,2,\dots,n+1\}$. Within this set of rows, we perform random sampling on all column indices that have a value of 1, with a sampling probability p, to obtain a column index set $H \subseteq \{1,2,\dots,K\}$. Next, from the selected column index set H, we further identify all row indices that have a value of 1 in these columns and perform random sampling with a probability q, resulting in a row index set $F \subseteq \{1,2,\dots,J\}$. Finally, using the index sets F and H, we extract the corresponding submatrix $Y_{1,1}(F_1, H_1)$ from the original matrix Y, where F_1 and H_1 represent the sets of row and column indices of the first sampled submatrix, respectively.

After constructing the first submatrix $Y_{1,1}$, we repeat the above process to generate a second submatrix $Y_{2,1}(P_1, Q_1)$ in the same manner, where P_1 and Q_1 denote the row and column index sets of this new submatrix, respectively. This procedure continues iteratively until all rows of the original matrix Y have been completely traversed. Consequently, we generate a total of $\lceil J/(n+1) \rceil$ pairs of subgraphs, where $\lceil x \rceil$ denotes the smallest integer not less than x. The pair of subgraphs generated at the t-th iteration is denoted as $Y_{1,t}(F_t, H_t)$ and $Y_{2,t}(F_t, H_t)$.

ISSN: 1813-4890

3.2. Micro Contrastive Learning Framework

3.2.1. User-Item Embedding Layer

Given a dataset, the model initializes by generating user and item embeddings based on the dataset size. Let the user set U contain M users and the item set V contain N items. We then define:

$$E_u \in \mathbb{R}^{M\times d}, E_v \in \mathbb{R}^{N\times d}$$

where E_u and E_v represent the embedding matrices for users and items, respectively, and d denotes the embedding dimension.

3.2.2. GCN Layer

A two-layer GCN is adopted for information propagation on the graph structure:

$$H^{(l+1)} = \left(\widetilde{D}^{-\frac{1}{2}}\widetilde{A}\widetilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right) \quad (1)$$

where $\widetilde{A} = A + I$ is the adjacency matrix with self-loops; $\widetilde{D}$ is the degree matrix of $\widetilde{A}$; $H^{(l)}$ denotes the node representation at the l -th layer, with the initial representation $H^{(0)}$ obtained by concatenating the user and item embeddings, i.e., $H^{(0)} = \begin{bmatrix} E_u \\ E_v \end{bmatrix}$;$W^{(l)} \in \mathbb{R}^{d\times d}$ is the trainable weight matrix at the l-th layer; is the non-linear activation function, which in this case is ReLU.

3.3. MicroLoss

The expression for the row (user) loss function:

$$L_{row} = -\frac{1}{N_r}\sum_{i=1}^{N_r}\log\left(\frac{\exp\left(\frac{z_{1,row}^{(i)}\cdot z_{2,row}^{(i)}}{\tau}\right)}{\sum_{j=1}^{N_r}\exp\left(\frac{z_{1,row}^{(i)}\cdot z_{2,row}^{(i)}}{\tau}\right)}\right) \quad (2)$$

The expression for the colum (item) loss function:

$$L_{col} = -\frac{1}{N_c}\sum_{i=1}^{N_c}\log\left(\frac{\exp\left(\frac{z_{1,col}^{(i)}\cdot z_{2,col}^{(i)}}{\tau}\right)}{\sum_{j=1}^{N_c}\exp\left(\frac{z_{1,col}^{(i)}\cdot z_{2,col}^{(i)}}{\tau}\right)}\right) \quad (3)$$

$$L = L_{row} + L_{col} \quad (4)$$

where

N_r denotes the number of user nodes (rows) participating in the contrastive learning;

N_c denotes the number of item nodes (columns) participating in the contrastive learning;

$z_{1,row}^{(i)}$ represents the embedding of the i-th user in the first view;

$z_{2,row}^{(i)}$ represents the embedding of the i-th user in the second view;

$z_{1,col}^{(i)}$ represents the embedding of the i-th item in the first view;

$z_{2,col}^{(i)}$ represents the embedding of the i-th item in the second view;

τ is the temperature coefficient, which controls the smoothness of the softmax distribution;

$\cdot$ denotes the dot product.

ISSN: 1813-4890

4. Evaluation

This section primarily presents the experimental setup and results analysis conducted to validate the effectiveness of the proposed MicroGCL in recommender systems. We first describe the experimental setup, including the datasets, baseline methods, and experimental parameters. Subsequently, we present a comparative performance analysis and further investigate the performance of MicroGCL under conditions of data sparsity and varying hyperparameters.

4.1. Experimental Settings

4.1.1. Datasets And Evaluation Protocols

We evaluate the performance of MicroGCL on multiple datasets, which are selected to cover varying levels of data sparsity and scales. The statistical information of these datasets is presented in the following table.

Table1 Datasets

Dataset	Yelp	Gowalla	Amazon
#Users	29601	50821	78578
#Items	24734	57440	77801
#Interactions	1069128	1172425	2240156

4.1.2. Baseline Methods

To verify the performance of MicroGCL on recommendation tasks, we select the following classic and high-performing methods for comparison:

(1) DGCF (Dynamic Graph Collaborative Filtering) [4]: This method models user-item interactions through a dynamic graph neural network to enhance recommendation accuracy.

(2) HyREC (Hyperbolic Graph Collaborative Filtering) [5]: This approach utilizes hyperbolic space for recommendation, enabling better capture of user-item relationships, particularly those with hierarchical structures.

(3) LightGCN (Light Graph Convolutional Network) [3]: A lightweight graph neural network model that retains only neighbor information aggregation, simplifying the design while improving recommendation effectiveness.

(4) MHCN (Multi-channel Hypergraph Convolutional Network) [6]: By introducing a multi-channel hypergraph structure, this method achieves the effective fusion of high-order relationships in social recommendation scenarios.

4.1.3. Metrics

We employ the following common recommendation evaluation metrics for performance comparison:

(1) R@K (Recall@K): This metric measures the proportion of relevant items successfully retrieved within the top-K recommended list from the test set.

(2) NDCG@K (Normalized Discounted Cumulative Gain@K): This metric considers the ranking positions of relevant items, assigning higher scores to items that appear earlier in the list.

In this experiments, we primarily report the results for R@20, R@40, NDCG@20, and NDCG@40. The specific formulas are as follows:

Recall is calculated as:

$$\text{Recall} = \frac{\text{Relevant items retrieved}}{\text{Total relevant items}} \tag{5}$$

The Discounted Cumulative Gain (DCG) is calculated as:

ISSN: 1813-4890

$$\mathrm{DCG@k} = \sum_{i=1}^{k} \frac{2^{\mathrm{rel(i)}} - 1}{\log_2(i+1)} \tag{6}$$

where rel(i) represents the relevance score of the item at rank i (typically 0 or 1 for binary relevance; different values can be used for graded relevance). The term $\log_2(i+1)$ provides a discount based on the rank, with lower ranks (higher positions) receiving less discount.

The Ideal DCG (IDCG) is calculated as:

$$\mathrm{IDCG@k} = \sum_{i=1}^{k} \frac{2^{\mathrm{rel}^*_{(i)}} - 1}{\log_2(i+1)} \tag{7}$$

where $\mathrm{rel}^*_{(i)}$ denotes the relevance score at position i in an ideal ranking (i.e., relevance scores sorted in descending order).

Finally, the Normalized Discounted Cumulative Gain (NDCG) is calculated as:

$$\mathrm{NDCG@k} = \frac{\mathrm{DCG@k}}{\mathrm{IDCG@k}} \tag{8}$$

4.2. Experimental Performance

Table2 Performance comparison with baselines on datasets.

Data	Metric	DGCF	HyRec	LightGCN	MHCN	MicroGCL	impr.
Yelp	R@20	0.0466	0.0472	0.0482	0.0503	0.0559	3%
	N@20	0.0395	0.0395	0.0409	0.0424	0.0459	4%
	R@40	0.0774	0.0791	0.0803	0.0826	0.0851	3%
	N@40	0.0511	0.0522	0.0527	0.0544	0.0571	5%
Gowalla	R@20	0.0944	0.0901	0.0985	0.0955	0.1044	6%
	N@20	0.0522	0.0498	0.0593	0.0574	0.0628	6%
	R@40	0.1401	0.1356	0.1431	0.1393	0.1516	6%
	N@40	0.0671	0.0660	0.0710	0.0689	0.0759	7%
Amazon	R@20	0.0211	0.0302	0.0319	0.0296	0.0335	5%
	N@20	0.0154	0.0225	0.0236	0.0219	0.0253	5%
	R@40	0.0351	0.0432	0.0499	0.0489	0.0519	4%
	N@40	0.0201	0.0246	0.0290	0.0284	0.0304	5%

4.3. Ablation Study

To empirically validate the specific contribution of the proposed random node sampling strategy to model performance, we design an ablation study in which this module is removed from the overall framework and the resulting variant is compared against the full model. Specifically, in the ablated version, we eliminate the random node sampling during training while keeping all other graph contrastive learning components unchanged. This allows us to investigate whether the sampling strategy plays a critical role in generating effective representations.

ISSN: 1813-4890

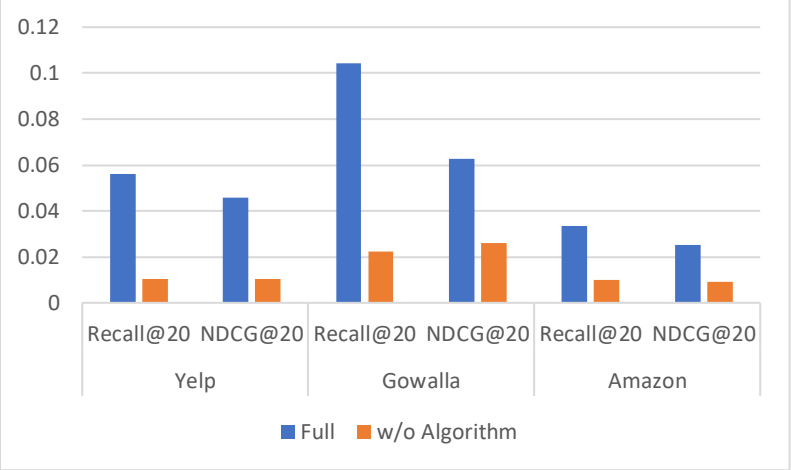

Fig.2 Ablation study on MicroGCL

The experimental results presented in Figure 2 demonstrate that removing the random node sampling strategy leads to a decline in all evaluation metrics (Recall@20, NDCG@20) across different datasets. This performance degradation is particularly pronounced on datasets with higher sparsity. These findings indicate that the random node sampling strategy effectively enhances the disparity between graph structural views, thereby promoting the learning of more discriminative representations.

Further analysis suggests that random sampling introduces a beneficial regularization effect during training, which helps mitigate potential homogeneity bias inherent in the graph structure. Consequently, this enables the model to exhibit stronger generalization capabilities when confronted with complex and diverse recommendation scenarios. These results conclusively demonstrate that random node sampling, as a crucial component of our proposed method, plays a positive and significant role in improving recommendation performance.

4.4. Hyperparamenter Analysis

In this study, we conduct an analysis of the key hyperparameters of the graph contrastive learning model to evaluate their impact on recommendation performance. The primary hyperparameters investigated include batch size n, hidden dimension d, row sampling rate p, column sampling rate q, temperature parameter τ, and learning rate α. Using a controlled variable approach, we adjust one hyperparameter at a time while keeping others fixed, and observe the resulting model performance.

First, we conduct experiments with different batch sizes (1, 3, 10, 20). The batch size affects both the precision of gradient estimation and computational efficiency. Experimental results indicate that a very small batch size (e.g., 1) leads to unstable gradient estimation, hindering model convergence, while an excessively large batch size (e.g., 20) increases computational overhead and reduces the ability to capture personalized user characteristics. Ultimately, we select a batch size of 3 as the optimal choice, balancing computational efficiency and model performance.

Second, the temperature parameter is a critical factor in the contrastive learning loss function, controlling the sensitivity of similarity calculations. We conduct experiments with temperature values of 0.1, 0.2, 0.5, and 1.0. The experimental results show that setting the temperature to 0.2 yields the best performance in terms of metrics such as Recall and NDCG. This suggests that a moderate temperature parameter helps enhance the model's discriminative ability, whereas values that are too low or too high may lead to gradients that are either too large or too small, adversely affecting model optimization.

Furthermore, we investigate the impact of the hidden dimension on the model. Experiments are conducted comparing four hidden dimensions: 32, 64, 128, and 256. The results

ISSN: 1813-4890

demonstrate that when the hidden dimension is too small (e.g., 32), the model's expressive capacity is limited, leading to suboptimal recommendation performance. Conversely, increasing the hidden dimension to 256 makes the model prone to overfitting during training. Consequently, we select 64 as the optimal hidden dimension, achieving a balance between model complexity and generalization ability.

Finally, we analyze the influence of the learning rate on the model. We train and evaluate the model using four different learning rates: 0.0001, 0.0005, 0.001, and 0.005. The experimental results reveal that a learning rate that is too low (e.g., 0.0001) results in slow convergence, while a learning rate that is too high (e.g., 0.005) can lead to unstable training. Ultimately, we select 0.001 as the optimal learning rate, ensuring stable convergence and achieving strong recommendation performance.

Table3 Performance of Yelp among different groups of parameters

n	d	p	q	τ	α	R@20
3	32	0.8	0.8	0.3	0.001	0.0545
3	32	0.8	0.7	0.3	0.002	0.0531
10	32	0.6	0.6	0.2	0.001	0.0536
10	64	0.6	0.5	0.2	0.001	0.0529
20	32	0.8	0.8	0.3	0.001	0.0537
20	64	0.8	0.7	0.3	0.001	0.0535
3	64	0.9	0.8	0.2	0.001	0.0559
20	64	0.6	0.5	0.2	0.002	0.0542

In summary, the appropriate selection of hyperparameters significantly impacts the performance of the GCL model. Through experimental analysis, this study has identified an optimal combination of hyperparameters: a batch size of 3, a row sampling rate of 0.9, a column sampling rate of 0.8, a temperature parameter of 0.2, a hidden dimension d of 256, and a learning rate α of 0.001. Building upon these findings, future research can further explore methods for adaptive hyperparameter tuning to enhance the robustness and generalization capability of the model.

4.5. MicroLoss Funtion Variation Analysis

In this experiment, the loss function did not decrease smoothly but instead exhibited persistent fluctuations, as shown in Figure 3. This is a normal phenomenon, primarily attributable to the stochastic nature of data sampling and the dynamic updates of model parameters. Firstly, due to the use of mini-batch stochastic gradient descent during training, different users and items are randomly sampled in each training batch. Consequently, the computation of the loss function depends on different subsets of data, introducing a certain degree of variance and causing fluctuations in the loss across different batches. Furthermore, within the contrastive learning framework, the loss function typically depends on the selection of contrastive samples, and the inherent uncertainty in sampling further exacerbates the volatility of the loss.

Secondly, during the training process of graph neural network-based models, the embedding representations of users and items continuously evolve as parameters are updated. Since the direction of the gradient update is influenced by the data distribution of the current batch, the optimization path may vary in each training iteration, leading to fluctuations in the loss value within a certain range. This is particularly noticeable in the early stages of training, where the model parameters have not yet fully converged, often resulting in larger fluctuations in the loss. As training progresses, although the overall loss value exhibits a downward trend, minor fluctuations persist due to the impact of random sampling. This is a normal consequence of the dynamic variations in data distribution.

ISSN: 1813-4890

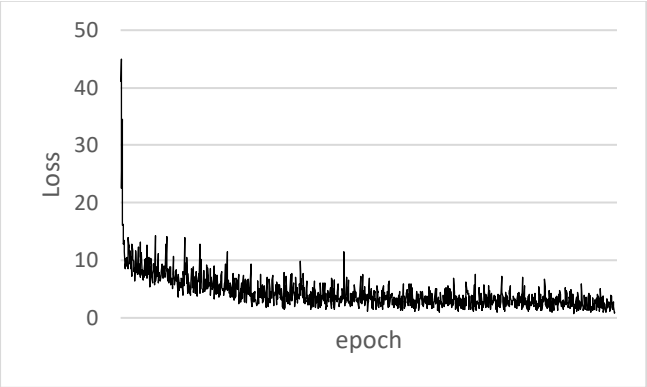

Fig.3 MicroLoss of Yelp during training

5. Conclusion

This paper proposes a novel lightweight graph contrastive learning method: MicroGCL, which replaces traditional edge perturbation and node deletion techniques with a view generation algorithm, effectively mitigating information loss. Through training, the model achieves effective representation learning and attains excellent performance in downstream recommendation tasks. Experimental results on multiple datasets demonstrate that MicroGCL exhibits strong generalization capabilities in data-sparse environments and surpasses existing state-of-the-art methods on certain tasks. In the future, we will explore more efficient contrastive learning strategies to further enhance recommendation performance.

Acknowledgments

The work is supported by Innovation Fund Project for Undergraduate Student in Shanghai (No. SH2025080).

References

[1] Wu Z, Pan S, Chen F, et al. A comprehensive survey on graph neural networks[J]. IEEE transactions on neural networks and learning systems, 2020, 32(1): 4-24.

[2] 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.

[3] 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.

[4] Wang X, Jin H, Zhang A, et al. Disentangled graph collaborative filtering[C]//Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval. 2020: 1001-1010.

[5] Wang J, Ding K, Hong L, et al. Next-item recommendation with sequential hypergraphs[C]//Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval. 2020: 1101-1110.

[6] Yu J, Yin H, Li J, et al. Self-supervised multi-channel hypergraph convolutional network for social recommendation[C]//Proceedings of the web conference 2021. 2021: 413-424.

[7] You Y, Chen T, Sui Y, et al. Graph contrastive learning with augmentations[J]. Advances in neural information processing systems, 2020, 33: 5812-5823.

[8] Chen T, Kornblith S, Norouzi M, et al. A simple framework for contrastive learning of visual representations[C]//International conference on machine learning. PmLR, 2020: 1597-1607.

[9] He K, Fan H, Wu Y, et al. Momentum contrast for unsupervised visual representation learning[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2020: 9729-9738.

[10] 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.

[11] 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.

[12] Kipf T N, Welling M. Semi-supervised classification with graph convolutional networks[J]. arXiv preprint arXiv:1609.02907, 2016.

[13] Hamilton W, Ying Z, Leskovec J. Inductive representation learning on large graphs[J]. Advances in neural information processing systems, 2017, 30.

[14] Velickovic P, Cucurull G, Casanova A, et al. GRAPH ATTENTION NETWORKS[J]. stat, 2018, 1050: 4.

[15] Chen J, Ma T, Xiao C. Fastgcn: fast learning with graph convolutional networks via importance sampling[J]. arXiv preprint arXiv:1801.10247, 2018.

[16] Zeng H, Zhou H, Srivastava A, et al. Graphsaint: Graph sampling based inductive learning method[J]. arXiv preprint arXiv:1907.04931, 2019.

[17] Chiang W L, Liu X, Si S, et al. Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks[C]//Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining. 2019: 257-266.

[18] Wu Z, Xiong Y, Yu S X, et al. Unsupervised feature learning via non-parametric instance discrimination[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 3733-3742.

[19] Oord A, Li Y, Vinyals O. Representation learning with contrastive predictive coding[J]. arXiv preprint arXiv:1807.03748, 2018.

[20] Grill J B, Strub F, Altché F, et al. Bootstrap your own latent-a new approach to self-supervised learning[J]. Advances in neural information processing systems, 2020, 33: 21271-21284.

[21] Chen X, He K. Exploring simple siamese representation learning[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2021: 15750-15758.

[22] Chen X, Xie S, He K. An empirical study of training self-supervised vision transformers[C]// Proceedings of the IEEE/CVF international conference on computer vision. 2021: 9640-9649.

[23] Xie X, Sun F, Liu Z, et al. Contrastive learning for sequential recommendation[C]//2022 IEEE 38th international conference on data engineering (ICDE). IEEE, 2022: 1259-1273.

[24] Shi Z, Wang X, Lipani A. Self contrastive learning for session-based recommendation[C]//European Conference on Information Retrieval. Cham: Springer Nature Switzerland, 2024: 3-20.

[25] Cai X, Huang C, Xia L, et al. LightGCL: Simple yet effective graph contrastive learning for recommendation[J]. arXiv preprint arXiv:2302.08191, 2023.