

Received 11 December 2023, accepted 2 January 2024, date of publication 10 January 2024,  
date of current version 18 January 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3352043

## RESEARCH ARTICLE

# Course Recommendation Model Based on Layer Dropout Graph Differential Contrastive Learning

YONG OUYANG<sup>1</sup>, HAO LONG<sup>1</sup>, RONG GAO<sup>1,2</sup>, AND JINGHANG LIU<sup>1</sup>

<sup>1</sup>School of Computer Science, Hubei University of Technology, Wuhan 430068, China

<sup>2</sup>State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China

Corresponding author: Rong Gao (gaorong@hbut.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 62106070.


**ABSTRACT** At present, the course recommendation model of graph collaborative filtering mainly uses bipartite graph modeling to obtain user-course cooperative relationship. However, the bipartite graph lacks the acquisition of user-user and course-course relationship information. In addition, due to the inherent defects of graph convolution, multi-layer graph convolution will cause overfitting problems. Moreover, the existing graph contrastive learning methods to solve the sparsity of recommendation data simply divide nodes into positive and negative pairs, without taking into account that users who have chosen the same course in the recommendation are similar. In contrastive learning, the feature similarity distance of these users should be different. To solve these problems, a course recommendation model based on layer dropout graph differential contrastive learning (DGDCL) is proposed. Specifically, a hybrid graph convolution network of fusion graph and hypergraph is used to obtain both low-order and high-order information. Then, using the layer dropout method to alleviate overfitting in neural network, the multi-layer feature embeddings of graph nodes are randomly dropout. Finally, two different layer drops are used to generate the contrastive views to reduce the additional noise and computational overhead of generating the contrastive views. The prior similarity of users and courses is used to adjust the calculation of the contrastive loss function, and differentiated contrastive learning of graph nodes is realized to make the contrastive learning more suitable for the recommendation model. The experimental results of XuetaangX and MOOCCube datasets show that the proposed model is better than the existing model.

**INDEX TERMS** Course recommendation, layer dropout, graph differential contrastive learning.

## I. INTRODUCTION

With the popularization of the Internet and the development of online education platforms, more and more people choose online learning to improve their skills and knowledge, such as the MOOC online learning platform with a large number of quality course resources, which provides people with courses from top universities around the world. However, with the increasing number of courses in the learning platform, the learning platform provides users with abundant learning resources and at the same time produces the problem of “information overload” [1]. In the face of massive learning resources, it is often

difficult for users to find suitable courses for themselves, and it is difficult for learning platforms to provide users with learning resources they are interested in according to their learning characteristics. This not only wastes the user’s time and energy, but also seriously affects the user’s learning effectiveness and learning experience on the learning platform. Therefore, how to provide users with personalized and targeted courses has important research and practice value. Course recommendation, as the main technology to solve information overload, can filter and screen massive course resources, and provide users with appropriate courses through data analysis of users’ learning preferences. Accurate recommendation results can save users’ time in searching for courses and improve their learning efficiency and interest.

The associate editor coordinating the review of this manuscript and approving it for publication was Yin Zhang .

At present, some course recommendation models have been proposed successively. The earliest course recommendation model was recommended by calculating the feature similarity of courses selected among users. Later, some deep learning-based models were proposed successively [2], [3]. In recent years, the recommendation model based on graph neural network (GNN) has become the latest research direction in recommendation system because of its advantages of processing structured data and mining structured information. For example, the simplifying and powering graph convolution network for recommendation (LightGCN) [4] represents the user-course interactive relationship through bipartite graph, and uses multi-layer graph convolution to carry out feature embeddings learning on nodes to obtain multi-layer feature embeddings of users and courses.

However, there are some shortcomings in the current graph collaborative filtering recommendation model. 1) Firstly, the bipartite graph lacks mining high-order interactive information of users and courses. In a bipartite graph, the information propagation of graph convolution comes from neighboring nodes, and all adjacent nodes of user (course) nodes are course (user) nodes. Therefore, bipartite graph can obtain low-order information such as user-course well, but it is difficult to obtain high-order information such as user-user, which leads to missing information obtained by the model. 2) On the other hand, multi-layer graph convolution stacking can cause overfitting problems, and too many layers can lead to too many aggregation times for each node's surrounding neighbors. This approach will lead to increasingly similar values for all vertices, ultimately converging to the same value, making it difficult to distinguish the personality characteristics of each node. 3) In addition, there are problems of sparse data and prevalence bias in recommendation data. The data is sparse because there is very limited information about user-course interactions. The popularity bias is due to the fact that a small number of popular courses are selected by a large number of users, while some unpopular courses are often selected by a small number of users. In this case, the recommendation performance of the course recommendation model will decline, because it is difficult for the model to learn enough information from the data and cannot accurately predict the user's preference for unknown courses.

In recent years, some related research efforts have attempted to address the above three issues. Dual channel hypergraph collaborative filtering (DHCF) [5] generates hypergraph over user-item bipartite graph, proving that hypergraph can effectively capture high-order correlations in data. DenseGCN and ResGCN [6] apply the residual connection and dense connection methods in convolutional network models (CNN) to graph convolution, effectively alleviating the overfitting problem of multi-layer graph convolution. Contrastive learning [7] has been used to solve the problem of data sparsity, as it can extract additional self supervised signals from the original data. However, the currently used contrastive learning methods in recommendation models treat different users and courses as positive and negative sample

relationships, and in reality, different types of users are not completely different, The feature similarity between users who have chosen the same course and users who have not chosen the same course is different.

Inspired by the above research work, this paper proposes a course recommendation model based on layer dropout graph differential contrastive learning (DGDCL). First, based on the bipartite graph, a high order correlation model of users (courses) is established, and interactive information is further mined by generating user (courses) hypergraphs, and multi-layer information propagation is carried out by a hybrid method of graph convolution and hypergraph convolution. Second, drawing on the dropout method in neural networks, a layer dropout method is proposed for graph convolution, which randomly dropout the multi-layer feature embeddings obtained from graph convolution to alleviate the overfitting problem of multi-layer graph convolution stacking. On this basis, the existing contrastive learning methods are improved by using improved graph contrastive learning to solve the problem of data sparsity. Two different user (course) layers dropout feature embeddings as two contrastive views, and a differentiated contrastive loss function is used for loss calculation. This allows users (course) with high feature similarity to have similar features closer to each other, rather than traditional contrastive learning that pulls all sample features farther apart, make it more in line with the characteristics of the recommended data. Experiments on two course recommendation datasets show that the proposed method outperforms existing course recommendation models. The main contributions of this paper are as follows:

1. This paper generates a hypergraph in the user course interaction graph constructed from a bipartite graph, and utilizes a hybrid graph convolutional network for node feature embeddings learning, fully mining low-order and high-order information in the graph.

2. Propose a layer dropout graph differential contrastive learning method. Layer dropout can replace the cumbersome methods of graph enhancement and feature enhancement to generate contrastive views. The differentiated contrastive learning of graph improves the contrast loss function by using the prior feature similarity of users (courses), and allows the feature similarity of some graph nodes to be close to each other instead of uniformly pulling apart the features of different nodes, so that contrastive learning is more suitable for recommendation.

3. A comprehensive experimental study was conducted on two course datasets, and the experimental results showed that the DGDCL has significant improvements compared to existing models, effectively improving the accuracy of course recommendations.

## II. RELATED WORK

This section mainly reviews the research work related to DGDCL, including three main parts: graph and hypergraph convolution networks, graph contrastive learning, and course recommendations.

## A. GRAPH AND HYPERGRAPH CONVOLUTION NETWORKS

Inspired by CNN, graph convolutional networks (GCNs) use spectral theory for graph convolution operations. The theoretical basis of graph convolution is that convolution in the time domain is equivalent to dot product in the spectral domain. Firstly, the Fourier transform is used to transform the graph signal in the time and spectral domains, and then the convolution operation is performed through parameterized convolution kernels. The widely used GCN [8] utilizes first-order Chebyshev polynomials as graph convolution kernels to simplify calculations. However, adding more layers in GCN can lead to overfitting and the problem of gradual disappearance. This means that backpropagation through these networks will lead to super smoothing, ultimately leading to the convergence of the features of the graph nodes to the same value. Due to these limitations, most state-of-the-art GCNs do not exceed 4 layers. Li et al. [6] proposed a new method for successfully training deep GCN. Drawing on the concepts of convolutional networks, especially residual, dense connections, and dilated convolutions, and adapting them to the GCN architecture.

In recommendation, the interaction between the user and the item is represented as a binary graph, which can be used for node feature extraction through GCN, such as NGCF [9] and LightGCN. NGCF applies GCN to recommendation tasks, which is divided into two parts: information construction and information aggregation. LightGCN simplifies the method of graph convolution and achieves better performance in recommendations.

As a generalization of graphs, hypergraphs have also attracted widespread attention in recent years. In a hypergraph, hyperedges can connect any number of nodes and simulate higher-order correlations between nodes. Inspired by the GCN method, the hypergraph convolutional network (HGCN) [10] utilizes the unique structured information of hypergraphs to perform hypergraph convolution in the spectral domain. Compared to the graph, the hypergraph successfully models higher-order data associations for downstream tasks (except for paired connections). DHCF first applies the hypergraph convolution method to the collaborative filtering recommendation model, generating hypergraphs on the user-item interaction matrix. However, it is limited by complex information dissemination frameworks and intuitive judgments that define hypergraph convolutional kernels. Yang et al. [11] proposed improving hypergraph convolutional networks through feature crossover and contrastive learning to enhance recommendation performance. HGCN is combined with feature crossover networks in a parallel manner to achieve a balance between feature crossover and excessive smoothing. Li et al. [12] proposed an effective hybrid graph and hypergraph convolutional network (EHGCN) recommendation model that uses graphs and hypergraphs to model the correlation between nodes in interaction graphs, achieving multi-level learning. EHGCN

also adopts the DenseGCN training framework to optimize the graph convolution strategy from the perspective of graph signal processing.

## B. COMPARATIVE LEARNING OF FIGURES

Comparative learning is a form of Self Supervised Learning (SSL) that trains models by maximizing the consistency between two enhanced views (i.e., pairwise) of an instance. This approach does not require manual annotation of data, thus improving data utilization efficiency [13]. Comparative learning is widely applied in various deep learning fields due to its powerful self supervised learning ability. In terms of image processing, images are enhanced through cropping, flipping, color transformation, and other methods. Two data from the same image are enhanced as positive samples, and two data from different images are enhanced as negative samples. The goal of comparative learning optimization is to make the features of positive samples close while those of negative samples far apart. In the graph comparison learning recommendation model, Wu et al. [14] first proposed the self supervised graph learning recommendation model (SGL) and designed three methods to generate graph contrastive views, namely node dropout, edge dropout, and random walking, which change the structure of the graph in different ways. Subsequently, Yu et al. [15] proposed the Simple Graph contrastive Recommendation Model (SimGCL) and found that in the recommendation model of contrastive learning, contrastive learning can learn more uniformly distributed user/item representations, which can implicitly reduce popular bias. Meanwhile, research has shown that graph augmentation is considered necessary and only plays a negligible role. Chen et al. [16] proposed a minimalist contrastive learning recommendation model (XSimGCL), which is an improvement on SimGCL. This method is based on SimGCL's noise based enhancement method, and achieves the same effect as SimGCL by unifying recommendation tasks and contrastive tasks, while also reducing model time complexity.

## C. COURSE RECOMMENDATION RESEARCH

Course recommendation is the application of recommendation algorithms in the field of education, with a focus on providing personalized courses for users. By analyzing users' learning behavior and preferences, the most suitable courses are recommended to improve their learning efficiency. Jing et al. [17] designed a CF based algorithm to learn user interests under the premise of course relationships, providing a CF based method for elective courses and using grades as a basic parameter to provide relevant courses. Recently, many studies have also focused on using neural networks and deep learning techniques for data preprocessing and recommendation. For example, Jiang et al. [18] used an improved recursive neural network to obtain students' course selection information and utilized all available information to obtain personalized student preferences. Zhang et al. [2]

proposed a hierarchical reinforcement learning method based on students' course history to modify user information and recommend courses. Sheng et al. [19] utilized the Heterogeneous Information Network (HIN) in the MOOC platform to design multiple meta paths for MOOC resources, which can make better recommendations. Lin et al. [20] proposed dynamic attention and hierarchical reinforcement learning to capture users' dynamic interests. Although the methods of the above courses have achieved significant results, they mainly focus on exploring the correlation between courses, and there has not been sufficient research on the high-level relationship similarity between users and courses.

The DGDCL model is different from the above methods as follows:

1. DGDCL uses the mixed method of bipartite graph and hypergraph to learn node feature representation, which can more effectively capture low-order association information and high-order association information, and enrich the mining of association information in data.

2. DGDCL uses the dropout method of neural networks for reference. In the training process, the layer dropout strategy is adopted to randomly dropout the multi-layer feature of the graph convolution, so as to alleviate the overfitting problem of the graph convolution network and enhance the node feature learning ability of the graph convolution network.

3. DGDCL adopts graph differential contrastive learning. By calculating the prior similarity of users (courses), the feature distance between different nodes is controlled differently in graph contrastive learning, so that the feature distance between some similar nodes is not indiscriminately far away, and the feature distance between these similar nodes is allowed to be close together, which alleviates the problem that contrastive learning will hinder model training.

### III. METHOD

In response to the shortcomings of existing bipartite graph course recommendation models, inspired by SimGCL and EHGCN, this paper proposes a course recommendation model based on layer dropout graph differential contrastive learning. Firstly, construct a hypergraph relationship on the existing user and course bipartite graph, and use graph convolution and hypergraph convolution to construct a hybrid graph convolution information passing method. Secondly, perform multi-layer graph convolution on the hybrid graph to obtain multi-layer feature embeddings of users and courses. Then, three sets of user course feature embeddings are obtained through three layers dropout, one of which is used for the course recommendation task, and the other two groups are used to construct a contrastive learning view. When calculating contrastive loss, the cosine similarity between users and courses is calculated using the original user-course interaction matrix, and a comparative loss mask is generated to achieve differentiated comparative learning. Finally, perform joint loss optimization on the two tasks

to obtain the final recommendation result. The overall architecture of the model is shown in Figure 1.

#### A. BIPARTITE GRAPH AND HYPERGRAPH GENERATION

This section introduces the construction methods of user-course bipartite graph, user hypergraph, and course hypergraph, as well as the mode of information propagation on them.

##### 1) USER-COURSE BIPARTITE GRAPH

In the user-course bipartite graph, users and courses are represented as two types of nodes. Similar nodes have no adjacency relationship. When users have selected courses, the corresponding user nodes and course nodes have adjacency relationship. The adjacency matrix of user-course bipartite graph is defined as equation 1:

$$A_g = \begin{pmatrix} 0 & H \\ H^T & 0 \end{pmatrix}, \quad (1)$$

where  $H \in R^{M \times N}$  is user-course interaction matrix,  $M$  and  $N$  respectively represent the number of users and courses. for each item  $H_{ui}$  in  $H$ , if user  $u$  has selected course  $i$ , its value is 1, otherwise its value is 0.

In GCN, node feature representation learning is carried out by aggregating features of neighbor nodes, which is called graph convolution information propagation mode. On bipartite graphs, the information propagation mode of graph convolution is defined as equation 2,3:

$$e_u^{(l+1)} = AGG(e_u^{(l)}, \{e_i^{(l)} : i \in N_u\}), \quad (2)$$

$$e_i^{(l+1)} = AGG(e_i^{(l)}, \{e_u^{(l)} : u \in N_i\}), \quad (3)$$

where  $AGG(\cdot)$  represents the graph information propagation model,  $e_u^{(l)}$  and  $e_i^{(l)}$  respectively represent the feature embeddings of user  $u$  and course  $i$  after  $l$  layers graph convolution.  $N_u$  represents the set of courses selected by user  $u$ , and  $N_i$  represents the set of users selected by course  $i$ . In LightGCN, the information propagation takes into the number of node connection edges (degrees), the calculation method is as show in equation 4, 5:

$$e_u^{(l+1)} = \sum_{i \in N_u} \frac{1}{\sqrt{|N_u|} \sqrt{|N_i|}} e_i^{(l)}, \quad (4)$$

$$e_i^{(l+1)} = \sum_{u \in N_i} \frac{1}{\sqrt{|N_u|} \sqrt{|N_i|}} e_u^{(l)}, \quad (5)$$

where  $|N_u|$  and  $|N_i|$  represent the degree of node  $u$  and node  $i$  respectively. The calculation matrix form is as show in equation 6, 7, 8:

$$E^{(l+1)} = \tilde{A}_g E^{(l)}, \quad (6)$$

$$\tilde{A}_g = D_g^{-\frac{1}{2}} A_g D_g^{-\frac{1}{2}}, \quad (7)$$

$$E^{(l)} = E_u^{(l)} || E_i^{(l)}, \quad (8)$$

where  $\tilde{A}_g$  is the symmetrically normalized adjacency matrix of  $A_g$ ,  $D_g$  is the degree matrix of  $A_g$ , and the diagonal value



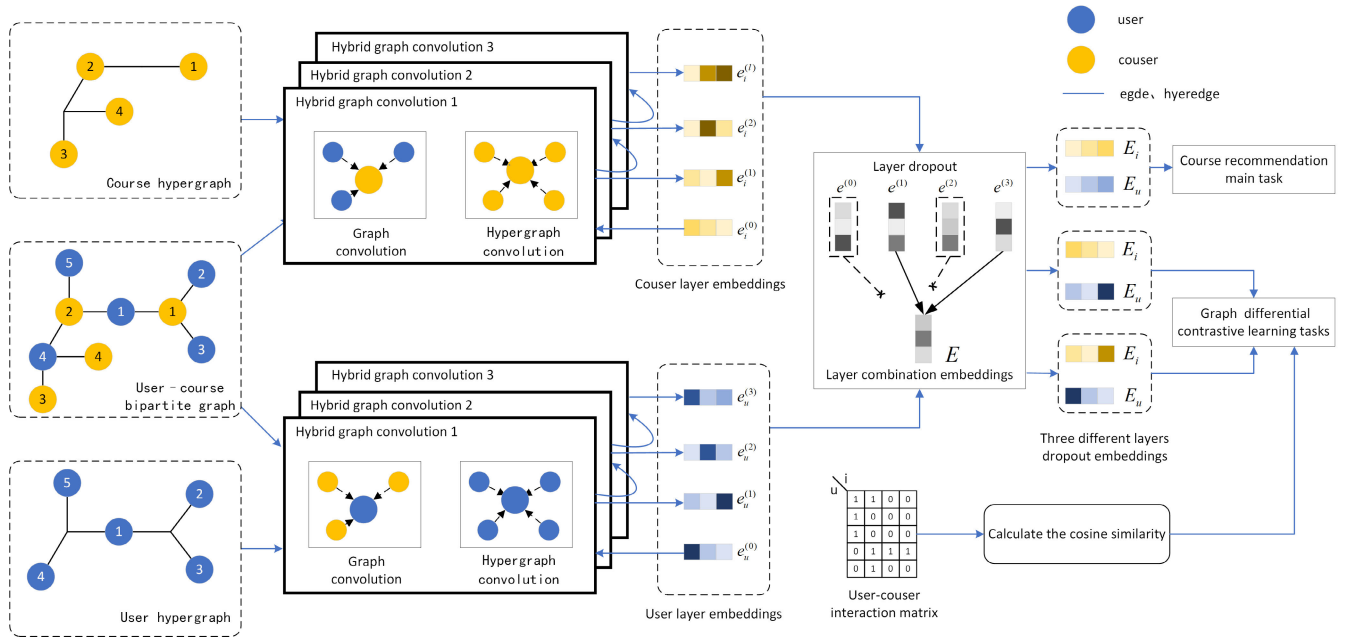


FIGURE 1. The model diagram of course recommendation model based on layer dropout graph differential contrastive learning.

is the number of non-zero elements per row of the matrix  $A_g$ .  $E^{(l)}$  is a concatenation matrix of layer  $l$  user feature embeddings matrix  $E_u^{(l)}$  and course feature embeddings matrix  $E_i^{(l)}$ .

In the information propagation of each layer of GCN, all the information of user nodes comes from adjacent course nodes, and all the information of course nodes comes from adjacent user nodes. Therefore, graph convolution on bipartite graph can effectively capture the user-course cooperative relationship information, but the user-user and course-course relationship information is not captured. New methods are needed to capture these relational information.

## 2) USER HYPERGRAPH AND COURSE HYPERGRAPH

In the previous related research work [5], the generation of hypergraph on the user-course bipartite graph can effectively obtain the higher-order relationship information such as user-user and course-course. The complex higher-order relationship is expressed through the hyperedge, and the adjacency matrix of the graph corresponds to the incidence matrix of the hypergraph. The following describes the construction of user hypergraph and course hypergraph and the process of node feature embeddings representation learning using hypergraph convolution on hypergraph.

The user hypergraph is constructed according to the first-order reachable neighborhood of the course and connects the user nodes in the neighborhood. In Figure 1, the first-order reachable neighborhood nodes of the course 1 node of the user-course bipartite graph have user 1, user 2, and user 3 nodes, so a hyperedge is used to connect the three user nodes 1, 2, and 3. The user hypergraph incidence matrix is defined as  $H_u \in R^{M \times N}$ , where  $M$  and  $N$  represent the number

of users and hyperedges, respectively. Since the hypergraph is constructed with order 1 reachability, the incidence matrix of the user hypergraph is the same as the user-course interaction matrix, that is,  $H_u = H$ .

The course hypergraph is constructed according to the user's first-order reachable neighborhood to construct the hyperedge and connect the course nodes in the neighborhood. In Figure 1, the course 2, Course 3, and course 4 nodes are the first-order reachable neighborhood nodes of the user 4 node, also connected by a hyperedge. The course incidence matrix in the course hypergraph is defined as  $H_i \in R^{N \times M}$ , and the course incidence matrix of the course hypergraph is the transpose of the user-course interaction matrix, that is,  $H_i = H^T$ .

Similar to GCN, hypergraph convolution also aggregates neighborhood node information for node feature embeddings representation learning. The information propagation mode of hypergraph convolution is defined as equation 9,10:

$$e_u^{(l+1)} = HAGG(e_u^{(l)}, \{e_u^{(l)} : u \in N_u^1\}), \quad (9)$$

$$e_i^{(l+1)} = HAGG(e_i^{(l)}, \{e_i^{(l)} : i \in N_i^1\}), \quad (10)$$

where  $HAGG(\cdot)$  represents the hypergraph information propagation model,  $N_u^1$  is the first-order neighborhood node set of node  $u$ , and  $N_i^1$  is the first-order neighborhood node set of node  $i$ .

The symmetric normalized Laplacian matrix of hypergraphs is used to carry out hypergraph convolution operations. The matrix form of user hypergraph convolution operation is as show in equation 11, 12:

$$E_u^{(l+1)} = \tilde{H}_u E_u^{(l)} \quad (11)$$

$$\tilde{H}_u = D_{uv}^{-\frac{1}{2}} H_u D_{ue}^{-1} H_u^T D_{uv}^{-\frac{1}{2}} \quad (12)$$

where  $D_{uv} \in R^{M \times M}$  is the nodal degree matrix of the matrix  $H_u$ , and the diagonal value is the number of non-zero elements per row of the matrix  $H_u$ .  $D_{ue} \in R^{N \times N}$  is the hyperboundary matrix of the matrix  $H_u$ , and the diagonal value is the number of non-zero elements in each column of the matrix  $H_u$ . After the above calculation, the size of the normalized Laplacian matrix  $\tilde{H}_u$  of the user hypergraph is  $R^{M \times M}$ .

The matrix form of course hypergraph convolution operations is as show in equation 13, 14:

$$E_i^{(l+1)} = \tilde{H}_i E_i^{(l)} \tag{13}$$

$$\tilde{H}_i = D_{iv}^{-\frac{1}{2}} H_i D_{ie}^{-1} H_i^T D_{iv}^{-\frac{1}{2}} \tag{14}$$

where  $D_{iv} \in R^{M \times M}$  is the nodal degree matrix of the matrix  $H_i$ , and the diagonal value is the number of non-zero elements per row of the matrix  $H_i$ .  $D_{ie} \in R^{N \times N}$  is the hyperboundary matrix of the matrix  $H_i$ , and the diagonal value is the number of non-zero elements in each column of the matrix  $H_i$ . After the above calculation, the size of the normalized Laplacian matrix  $\tilde{H}_i$  of the course hypergraph is  $R^{N \times N}$ .

### B. HYBRID GRAPH CONVOLUTION

Inspired by EHGCN, a hybrid graph convolution is used to capture the relationship information between user-course, user-user and course-course at the same time, and to mine more information from the recommendation data. The hybrid graphs convolution is defined as equation 15, 16:

$$e_u^{(l+1)} = \sum_{i \in N_u} \sum_{u \in N_u^1} (\tilde{A}_g e_i^{(l)} + \tilde{H}_u e_u^{(l)}), \tag{15}$$

$$e_i^{(l+1)} = \sum_{u \in N_i} \sum_{i \in N_i^1} (\tilde{A}_g e_u^{(l)} + \tilde{H}_i e_i^{(l)}). \tag{16}$$

In the hybrid graph, bipartite graph is used to model the user-course low-order correlation, and hypergraph is used to model the user-user and course-course high-order correlation. In the process of information propagation, each layer of convolution adopts the optimized hybrid graph information propagation to carry out multi-layer learning. Each node feature embeddings update can obtain information from the graph neighborhood node and hypergraph neighborhood node, the interactive information is fully mined from two different levels of low order correlation and high order correlation by graph and hypergraph respectively. Therefore, compared with bipartite graph and hypergraph, the hybrid graph can obtain richer user and course feature embeddings information. Finally, same as LightGCN, the feature embeddings of users and courses at each layer are aggregated to obtain the final feature embeddings of users and courses representing  $E$ . The calculation formula is as equation 17:

$$e = \frac{1}{L} \sum_{l=1}^L e^{(l)}, \tag{17}$$

where  $e^{(l)}$  is the feature embeddings of user and course node after  $l$  layers hybrid graph convolution.

### C. LAYER DROPOUT GRAPH CONVOLUTION NETWORK

In neural networks, when the linear model is too complex, the model tends to overfitting. To solve this problem, researchers proposed the dropout [21]. By adding Gaussian noise to the input of the model. In each training iteration, sampling noise from a distribution  $\varepsilon \sim N(0, \sigma^2)$  with a mean of zero is added to the input  $x' = x + \varepsilon$ , resulting in a disturbance point  $x$ , which is expected to be  $E[x] = x$ . In standard dropout regularization, the deviation of each layer is eliminated by normalizing the characteristics of the nodes that are retained (not dropout). In other words, each interneuron activity value  $h$  is randomly zeroed with a probability of  $p$ , as equation 18:

$$h' = \begin{cases} 0 & \text{if } r \leq p \\ h & \text{otherwise} \\ \frac{h}{1-p} & \end{cases} \tag{18}$$

where  $r \in [0, 1]$  is the random number for each dropout operation. According to the design of this model, its expected value remains the same, that is,  $E[h'] = h$ .

In traditional GCN, the dropout is often used after linear transformation and activation of node features, but in LightGCN, these two parts are removed and the dropout is also changed to randomly dropout the edges of the graph to prevent overfitting. However, when the randomly dropout edge is important to the node, this operation will hinder the information propagation between the nodes, make the features representation learned by the model have noise, and reduce the recommendation effect.

In order to solve the above problems, a new layer dropout is proposed in this paper, which is applied to the graph recommendation model to prevent the overfitting problem. The layer dropout is used in the process of multi-layer feature aggregation of node features, and a specified number of layer features are random dropout. This calculation as shown in equation 19,20:

$$e'^{(l)} = \begin{cases} 0, & \text{if } r_l \geq \text{top}(p, R) \\ \frac{e^{(l)}}{1-p/L}, & \text{otherwise} \end{cases}, \tag{19}$$

$$e' = \frac{1}{L} \sum_l e'^{(l)}, \tag{20}$$

where  $e^{(l)}$  is the graph node feature embeddings of the  $l$  layer,  $e'^{(l)}$  is the feature dropout through the layer, and  $e'$  is the multi-layer aggregate feature embeddings after layer dropout.  $p$  represents the set number of layer dropout,  $L$  represents the number of layers in the graph convolution,  $R = [r^{(1)}, r^{(2)}, \dots, r^{(L)}]$  is a group of random numbers used for layer dropout to randomly select the number of layers.  $\text{top}(p, R)$  represents the  $p$ th largest number in  $R$ . If the random number  $R_l$  of the corresponding layer is greater than or equal to  $\text{top}(p, R)$ , then the layer is dropout, otherwise the data elements of the layer are enlarged to ensure that the expected value of  $E$  is equal to that without layer dropout.

The layer dropout makes the following improvements to the previously used dropout:

1.The layer dropout applies the dropout to the layer aggregation stage, rather than the graph convolution stage. Using dropout in the convolution phase of graph is easy to introduce extra noise, which leads to the failure of normal training of the model. However, using dropout method in the layer aggregation stage can alleviate this problem, mainly because the graph convolution of each layer captures different feature information, even if the layer dropout number is set to a large value, the graph recommendation model will only degenerate into a matrix factorization model.

2.Normalizes the number of random dropout. In the previous dropout, the random number is compared with the specified dropout probability to determine whether to dropout. This method is feasible when there is a large number of drops each time, but the number of layers in the graph convolutional network is small, usually no more than 5 layers. If it is completely random and does not control the number of layer dropout, all or none of the layers will be dropout. Therefore, the method of strictly controlling the quantity of dropout is adopted, each layer is dropout by selecting the graph convolution layer corresponding to the largest value of  $p$  before the random number  $R$ , so as to ensure that the number of dropout layer of each layer is consistent with the set.

**D. LAYER DROPOUT GRAPH DIFFERENTIAL CONTRASTIVE LEARNING**

In the previous graph contrast learning, the Simgcl model with better results generates a contrastive learning enhanced view by adding Gaussian noise to the graph node features of each layer. Different views of the same node are regarded as positive examples, and different views of different nodes are regarded as negative examples. The task of contrastive learning is to narrow the distance between positive examples and widen the distance between negative examples. Graph contrastive learning can make the features learned by the model more uniform and obtain higher quality features, thus improving the recommendation effect. However, the way of using Gaussian noise to generate contrastive view has some defects. On the one hand, additional graph convolution operation is required. On the other hand, although most of the original features are retained by adding noise, some node features will inevitably be distorted.

The layer dropout strategy, described in previous section, is a way to reduce the number of feature layers of graph nodes. The method of layer dropout does not require additional graph convolution operation, and can avoid the problem of node feature distortion to the greatest extent. Therefore, graph contrastive learning based on layer dropout is proposed. Generate two contrastive views using two different layer dropout. InfoNCE was used to calculate the contrastive learning loss:

$$\mathcal{L}_{cl} = \sum_{i \in B} -\log \frac{\exp(\text{sim}(s'_i, s''_i)/\tau)}{\sum_{j \in B} \exp(\text{sim}(s'_i, s''_j)/\tau)}, \quad (21)$$

where  $i$  and  $j$  are users (courses) in the same batch of  $B$  during batch calculation,  $s'$  and  $s''$  are users (courses) embeddings

matrix after layer dropout,  $\text{sim}(x, y)$  is cosine similarity, and  $\tau$  is temperature hyperparameter. The goal of contrastive learning loss is to increase the similarity distance between positive pairs  $s'_i$  and  $s''_i$ , and decrease the similarity distance between negative pairs  $s'_i$  and  $s''_j$ .

In the traditional graph contrastive learning loss calculation, the similarity distance between positive pairs of nodes is close to each other, while the similarity distance between negative pairs of nodes is far away from each other without distinction. As shown in Figure 2(a),The target of the contrastive learning between anchor  $x$  and its negative samples  $x_1^-, x_2^-, x_3^-$  is all distance  $d$ . This is because contrastive learning was originally used for image classification, where image types are completely different types and there is no connection between the different types. Contrastive learning increases the similarity between the same species and decreases the similarity between different species. However, in the recommendation, different users and courses are not really different types, and some users who have chosen the same course and users who have not chosen the same course should have a higher similarity in feature. Therefore, contrastive learning in recommendation should be different for different users and courses to optimize the target distance  $d$ , but the existing graph contrastive learning does not take this into account.

In order to solve the above problem, a differentiated contrastive learning method is proposed. For different pairs of positive and negative nodes, differentiated contrastive learning is set to optimize the target distance. As shown in Figure 2(b), when  $x_1^-$  node is highly similar to  $x$ , the optimized distance is the set minimum optimization distance  $d_{\min}$ ; when  $x_2^-$  negative samples are somewhat related to  $x$ , their similarity should be extended to  $d_2$ . While the negative samples are not associated with  $x_3^-$ , pull their similarity distance further away to  $d_3$ . The improved contrastive loss is:

$$\mathcal{L}_{cl} = \sum_{i \in B} -\log \frac{\exp(\text{sim}(s'_i, s''_i)/\tau)}{\sum_{j \in B} \exp(\text{sim}(s'_i, s''_j)/\tau + \text{mask}_{ij})}, \quad (22)$$

where  $\text{mask}_{ij}$  is responsible for controlling whether the similarity distance between the negative sample and the anchor point continues to pull away. The  $\text{mask}_{ij}$  value is calculated by calculating the observed user course interaction.

A priori similarity is used to determine the differential distance of contrastive learning between nodes. Specifically, the observed user course interaction matrix  $H$  is used for calculation, and the interaction matrix is transformed into user one-hot feature matrix and course one-hot feature matrix. The user one-hot feature matrix is  $E^{\text{hotu}} = H$ . The user prior similarity matrix  $A^{\text{cosu}}$  is obtained by calculating the cosine similarity between the user's one-hot feature. The  $A^{\text{cosu}}$  is calculated as shown in equation 23:

$$A^{\text{cosu}} = \frac{E^{\text{hotu}} \cdot E^{\text{hotu}T}}{\|E^{\text{hotu}}\|^2}, \quad (23)$$

where  $\|E^{hotu}\|$  represents the norm of  $E^{hotu}$ . Similarly, the one-hot feature matrix of the course is  $E^{hoti} = H^T$ . The course prior similarity matrix  $A^{cosi}$  is calculated as shown in equation 24:

$$A^{cosi} = \frac{E^{hoti} \cdot E^{hotiT}}{\|E^{hoti}\|^2}. \quad (24)$$

When the model calculates the contrastive learning loss, the prior similarity  $A^{cosi}$  is compared with the  $sim(s'_i, s''_j)$ . If  $sim(s'_i, s''_j)$  has a smaller value than  $A^{cosi}$ , then  $mask_{ij}$  will be set to  $-inf$ , preventing the similarity between  $i$  and  $j$  from getting larger. The calculation method of mask is as equation 25:

$$mask_{ij} = \begin{cases} -inf & \text{if } sim(s'_i, s''_j) < (A_{cosij} - d_{min}) \\ 0 & \text{otherwise} \end{cases}, \quad (25)$$

where  $mask_{ij}$  and  $A_{cosij}$  represent the contrast loss mask and prior similarity of node  $i$  and node  $j$  respectively, and  $d_{min}$  is the number of interval  $[0, 1]$ , which is used to control the minimum similarity distance in contrastive learning.  $d_{min}$  is added because the prior similarity is calculated through the observed user-course interaction, and the remaining unobserved interaction makes the calculated similarity fluctuate up and down compared with the actual similarity value. Here, the lower limit of the floating is taken, and the prior similarity is subtracted by a smaller value  $d_{min}$ .

### E. MODEL TRAINING AND PREDICTION

The parameters of DGDCL model training are user feature embeddings and course feature embeddings. The optimization task of the model consists of the following two parts.

The first is the main task of recommendation. This part is to represent and learn the feature embeddings of users and courses through the hybrid graph convolutional network, and calculate the loss through the bayesian personalized ranking (BPR) [22]. The main goal of BPR loss is to learn the personalized preferences of the user by placing the courses that the user likes more before the courses that the user likes less. The calculation formula is as equation 26,27:

$$\mathcal{L}_{BPR} = - \sum_{u=1}^M \sum_{i \in N_u} \sum_{j \notin N_u} \ln(\sigma(e'_u{}^T e'_i - e'_u{}^T e'_j)) \quad (26)$$

$$\sigma(x) = \frac{1}{1 + e^{-x}} \quad (27)$$

where  $e'_u, e'_i$  and  $e'_j$  are finally feature embeddings of user  $u$ , course  $i$  and course  $j$ , respectively. Course  $i$  is randomly sampled from the set of courses  $N_u$  that user  $u$  has interacted with, course  $j$  is randomly sampled from the set of courses that user  $N_u$  has not interacted.  $\sigma(x)$  is the activation function *sigmoid*, which maps values to the interval  $(0, 1)$ .

The second is the comparative learning assistance task, which is designed to introduce additional self-supervised information so that the model can learn higher quality

features. The total training loss of the model is the weighted sum of the two. The formula is as 28:

$$\mathcal{L}_{all} = \mathcal{L}_{BPR} + \lambda \mathcal{L}_{cl} \quad (28)$$

where  $\lambda$  is a hyperparameter that controls the ratio of comparative learning loss. In the prediction process, layer dropout is not used, and the multi-layer aggregation is directly used as the final user and course features. The user selects the predicted score of each course as the dot product of their features embeddings. The calculation formula is as 29:

$$P(u, c) = e_u^T e_c \quad (29)$$

where  $e_u$  represents the feature embeddings of user  $u$ , and  $e_c$  represents the feature embeddings of course  $c$ . Finally, the top K courses with the highest predicted score are selected and recommended to the user.

TABLE 1. Student information.

| Dataset  | user | course | interactions | data sparsity |
|----------|------|--------|--------------|---------------|
| XuetangX | 8216 | 667    | 129643       | 0.0236        |
| MOOCCube | 6540 | 620    | 97569        | 0.0240        |

## IV. EXPERIMENT

We conducted experiments on two real user-selected course dataset and evaluated the model, answering the following research questions:

Q1: How does DGDCL compare with current collaborative filtering recommendation methods?

Q2: How does different hyperparameters settings affect DGDCL?

Q3: How does the layer dropout and graph differential contrast learning affect DGDCL?

### A. EXPERIMENTAL SETTINGS

#### 1) DATASET

Two public datasets: XuetangX and MOOCCube are selected in the experiments.

1) XuetangX [2]: The dataset comes from XuetangX, one of the largest online learning platforms in China. The data set contains the learning records of users who selected courses on the learning platform between October 1, 2016 and March 31, 2018. The dataset consists of information such as user number, course number, course name, course type, and course selection time.

2) MOOCCube [23]: The dataset is an open data warehouse for researchers of natural language processing, knowledge graph and data mining related to large-scale online education, including teaching videos, knowledge concepts, course selection records and video viewing records. In this paper, the course selection record of the user is selected as the data set.

The course selection record contains the user number, course number and course selection time.



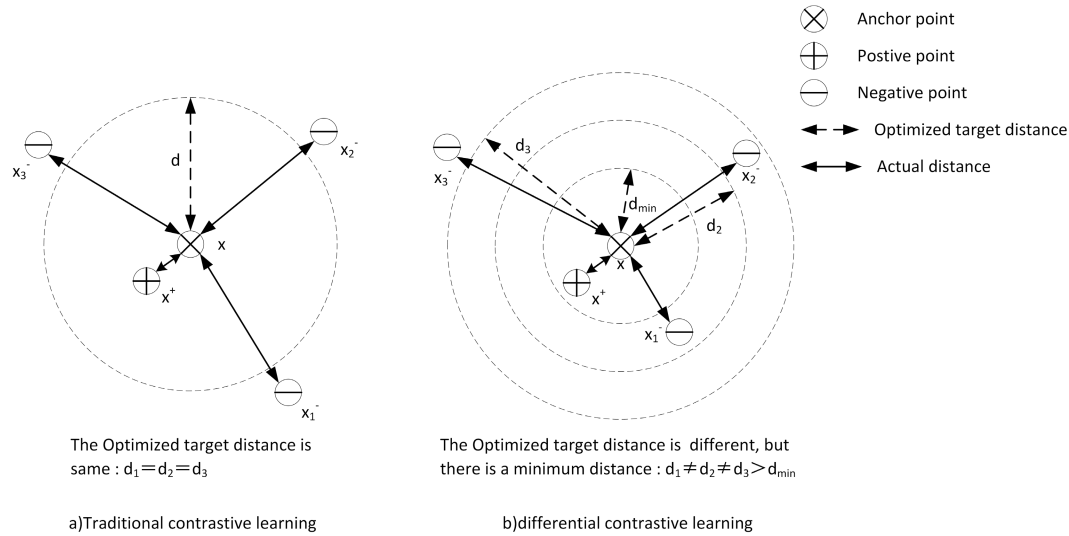


FIGURE 2. Two different contrastive learning.

According to the data processing method in [4] and [15], two datasets are preprocessed. Retaining users who have selected courses more than 10 times, and courses that have been selected more than 5 times. Two datasets information after processing is shown in Table 1. Each dataset is randomly divided into 7:1:2. During model training, 70% of the data is used as the training set and 10% is used as the validation set to adjust the hyper-parameters. After the model training is completed, the training set and the test set are combined, and the model recommendation effect is evaluated on the remaining 20% data as the test set.

## 2) EVALUATION METRICS

We use the following metrics to evaluate the model.

1) Recall: It represents the proportion of the number of courses hit in the predicted result to the set of courses selected by real users. The formula is as follows:

$$\text{Recall@K} = \frac{1}{|U|} \sum_{u \in U} \frac{|\tilde{C}_{u,K} \cap C_u|}{|C_u|}, \quad (30)$$

where @K represents the former top-K recommendation result,  $U$  represents the set of all users,  $\tilde{C}_{u,K}$  represents the predicted user selection set, and  $C_u$  represents the real user selection set.

2) Precision: It represents the proportion of the number of courses hit in the predicted result. The formula is as follows:

$$\text{Precision@K} = \frac{1}{|U|} \sum_{u \in U} \frac{|\tilde{C}_{u,K} \cap C_u|}{K}, \quad (31)$$

3) Normalized Discounted Cumulative Gain (NDCG): It evaluates the ranking of hit courses in the predicted results. The higher the ranking, the larger the index value. The

calculation formula is as follows:

$$\text{NGCG@K} = \frac{\text{DCG@K}}{\text{IDCG@K}} = \frac{\sum_{i=1}^K \frac{2^{rel_i} - 1}{\log_2(i+1)}}{\sum_{i=1}^K \frac{|REL_k| \cdot 2^{rel_i} - 1}{\log_2(i+1)}}, \quad (32)$$

where DCG represents the sum of ranking scores,  $rel_i$  represents the ranking position  $i$  in the prediction sequence, and the ranking scores of each position decrease by ranking position. IDCG indicates the perfect ranking score, i.e. the courses predicted to be hit are at the top of the ranking.

4) F1-measure (F1): It is for the overall evaluation of Precision and Recall, the calculation formula is as follows:

$$\text{F1@K} = \frac{2 \times \text{Precision@K} \times \text{Recall@K}}{\text{Precision@K} + \text{Recall@K}} \quad (33)$$

For all evaluate metrics, We set  $K = 5$ .

## 3) BASELINES

To demonstrate the effectiveness of the DGDCL model, it was compared to the following representative baselines:

1) MF [3]: This is a matrix factorization model using BPR loss optimization, which only uses user-item interaction with the course as the target value for feature embeddings representation learning.

2) LightGCN [4]: This model constructs a bipartite graph of user and course interaction, uses a light GCN to learn the feature representation of nodes, and the embeddings of nodes in each layer graph convolved are added as the final embeddings of nodes

3) EHGCN [12]: This is an effective hybrid graph and hypergraph convolution network for collaborative filtering, which uses graph and hypergraph to model the correlation among nodes in the interaction graph for multilevel learning. Then the improved DenseGCN model framework is used to implement multi-layer graph convolution.

4) SGL [14]: This model is a basic graph contrastive learning model. This method generates two graph contrastive views on the original bipartite graph by randomly adding and subtraction edges, uses LightGCN to learn node embeddings on the original bipartite graph and the two enhanced graphs, and constructs the recommended supervision task and the auxiliary task of contrastive learning for joint optimization.

5) SimGCL [15]: This model proposes a simple and effective recommendation model for graph graph-augmentation-free CL method for recommendation without graph augmentation, which can realize more effective feature level data enhancement by adding uniform random uniform noises to the user and course original representations.

6) XSimGCL[16]: Based on SimGCL, this method is improved by using noise enhancement and interlayer contrastive learning, and achieves the same effect as SimGCL by unifying task recommendation and task comparison, while reducing the time complexity of the model.

#### 4) HYPERPARAMETER SETTINGS

In the experiments, the embeddings size of users and courses is set to 64 and initialized by Xavier method. Adam was used to optimize the model parameters and the learning rate set to 0.001,  $L_2$  regularization set to  $10^{-4}$ , training batch size set to 1024, graph convolutional layer number set to 3, layer dropout number set to 1, and contrastive learning minimum distance is searched in the range [0.2, 0.1, 0.05]. For the contrastive model SGL, SimGCL, XSimGCL and DGDCL, the contrastive learning ratio is searched in the range of [0.1, 0.05, 0.01, 0.005], and the temperature is searched in the range of [1.0, 0.7, 0.5, 0.2]. Other parameters refer to the original paper, the SGL edge dropout rate is set to 0.1, and the SimGCL and XSimGCL random noises is set to 0.1.

## B. RESULTS AND ANALYSIS

### 1) BASELINES COMPARISON(Q1)

To verify the validity of the proposed model in course recommendation, DGDCL was compared with the baselines on XuetangX and MOOCCube datasets. The baselines are divided into graph models and graph contrastive learning models. The experimental results are shown in Table 2.

The experimental results show that DGDCL achieves the best recommendation effect. The best performance in the graph model is EHGCN, because EHGCN uses hybrid graph convolution and improved information aggregation mode, but EHGCN recommendation effect is not as good as EHGCN. On the XuetangX dataset, DGDCL increased by 3.9% on Recall, and 4.3% on Precision, and 4.3% on NDCG, and 4.1% on F1-measure compared with EHGCN. On the MOOCCube dataset, DGDCL by 5.0% on Recall, and 5.6% on Precision, and 5.9% on NDCG, and 5.4% on F1-measure compared with EHGCN. This shows that graph contrastive learning can effectively improve the recommendation effect of the model, because contrastive learning alleviates the problem of data sparsity in the recommendation data.

By introducing additional self-supervised signals, contrastive learning enables the model to obtain additional information, thus improving the recommendation effect.

In the graph contrastive learning model, DGDCL has the best recommendation effect, and SimGCL is the next best. SimGCL generates two contrastive views by adding evenly distributed noise to node embeddings of each layer, while SGL generates two contrastive views by randomly adding and deleting edges of the graph. SimGCL can avoid the absence of node information caused by the deletion of important edges. SimGCL has a better recommendation effect than SGL, but SimGCL has a worse recommendation effect than DGDCL. On the XuetangX dataset, DGDCL increased by 1.6% on Recall, and 1.9% on Precision, and 1.8% on NDCG, and 1.8% on F1-measure compared with SimGCL. On the MOOCCube dataset, DGDCL increased by 3.7% on Recall, and 3.9% on Precision, and 4.1% on NDCG, and 3.8% on F1-measure compared with XSimGCL. This shows that the proposed layer dropout graph differential contrastive learning is superior to the existing graph contrast learning model in improving the model recommendation effect.

There are two reasons for this: First, DGDCL uses layer dropout to generate contrastive views, while in other graph contrast learning models, noise data is added to the model to generate contrastive views, and DGDCL avoids the influence of noise on model recommendation. Second, DGDCL adopts graph differential contrastive learning to make the embeddings similarity distance of different nodes far apart. At the same time, the prior similarity is used to set their maximum embeddings similarity distance differently, so as to avoid those nodes whose embeddings similarity distance should be small being pulled apart indiscriminantly, resulting in the model learning sub-optimal node embeddings.

### 2) PARAMETER ANALYSIS(Q2)

Layer dropout and graph contrastive learning play an important role in DGDCL, so the parameters used in these two parts are analyzed.

The parameters  $p$  in layer dropout control the number of embeddings randomly dropout by the model, which can prevent the overfitting of the graph convolution. In the parameter analysis experiment,  $p$  is selected from [0, 1, 2], and the experimental results are shown in Table 3. On the whole, the model recommendation effect showed a trend of first increasing and then decreasing.

When  $p$  is 0, no layer dropout, no contrastive views are generated, and only the original multilayer graph embeddings are used for recommendation tasks and contrastive learning tasks. At this time, the model is similar to SGL-WA in SimGCL. Compared with the non-graph contrast learning model, the recommendation effect of this method is improved, because the contrastive learning task makes the embeddings distribution learned by the model more uniform, and a more uniform embeddings distribution can retain the intrinsic features of nodes and improve the generalization ability. When  $p$  is 1, the model recommendation effect is

TABLE 2. Baselines comparison.

| Dataset      | XuetangX      |               |               |               | MOOCCube      |               |               |               |
|--------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|
|              | Method        | Recall        | Precision     | NDCG          | F1            | Recall        | Precision     | NDCG          |
| MF           | 0.3424        | 0.2168        | 0.3432        | 0.2655        | 0.2476        | 0.1555        | 0.2388        | 0.1910        |
| LightGCN     | 0.3615        | 0.2311        | 0.3659        | 0.2820        | 0.2765        | 0.1741        | 0.2680        | 0.2136        |
| EHGCN        | 0.3669        | 0.2347        | 0.3692        | 0.2863        | 0.2833        | 0.1784        | 0.2754        | 0.2189        |
| SGL          | 0.3674        | 0.2357        | 0.3717        | 0.2872        | 0.2778        | 0.1755        | 0.2698        | 0.2151        |
| SimGCL       | <u>0.3752</u> | <u>0.2401</u> | <u>0.3780</u> | <u>0.2928</u> | 0.2843        | 0.1798        | 0.2776        | 0.2203        |
| XSimGCL      | 0.3740        | 0.2394        | 0.3772        | 0.2919        | <u>0.2869</u> | <u>0.1813</u> | <u>0.2802</u> | <u>0.2222</u> |
| <b>DGDCL</b> | <b>0.3812</b> | <b>0.2447</b> | <b>0.3850</b> | <b>0.2980</b> | <b>0.2974</b> | <b>0.1884</b> | <b>0.2916</b> | <b>0.2307</b> |

TABLE 3. Influence of the number  $p$  of layer dropout.

| Dataset | XuetangX |        | MOOCCube |        |
|---------|----------|--------|----------|--------|
|         | $p$      | Recall | NDCG     | Recall |
| 0       | 0.3763   | 0.3798 | 0.2859   | 0.2796 |
| 1       | 0.3812   | 0.3850 | 0.2974   | 0.2916 |
| 2       | 0.3770   | 0.3800 | 0.2879   | 0.2803 |

the best, because the layer dropout strategy adopted can alleviate the overfitting problem of the graph on the one hand, and generate two related and different contrastive views on the other hand. Different views can help the model learn different features, observe and compare the data from multiple perspectives, and help improve the robustness and generalization ability of the model. When  $p$  is 2, the model recommendation effect decreases. At this time, the number of layer dropout is too large, resulting in the loss of information obtained by the model in the multi-layer aggregation stage, and it is difficult to obtain the user and course feature information.

The contrastive learning parameter  $\lambda$  controls the proportion of contrastive learning in model training. The comparative learning task is usually positioned as an auxiliary task. In the parameter analysis experiment,  $\lambda$  is selected from [0.1, 0.05, 0.01, 0.005], and the experimental results are shown in Figure 3. The model recommendation effect is the best when  $\lambda$  is set to 0.01 on XuetangX dataset. The model recommendation is best when  $\lambda$  is set to 0.01 on the MOOCCube dataset. When  $\lambda$  is too small, it is difficult for contrastive learning to affect the model, and when  $\lambda$  is too large, it will inhibit model training. A moderate value can balance the relationship between recommendation tasks and comparative learning tasks to achieve the best performance.

The contrastive learning parameter  $\tau$ , also known as temperature parameter, is the penalty degree of control contrastive learning on difficult negative samples. In the parameter analysis experiment,  $\tau$  is selected from [1.0, 0.7, 0.5, 0.2], and the experimental results are shown in Figure 4. When  $\tau$  is set to 0.5 on the two datasets, the model recommendation

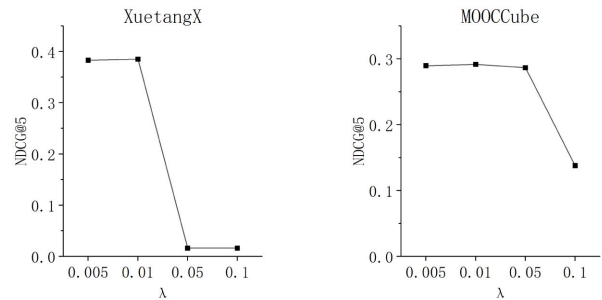


FIGURE 3. The influence of  $\lambda$  of contrastive learning.

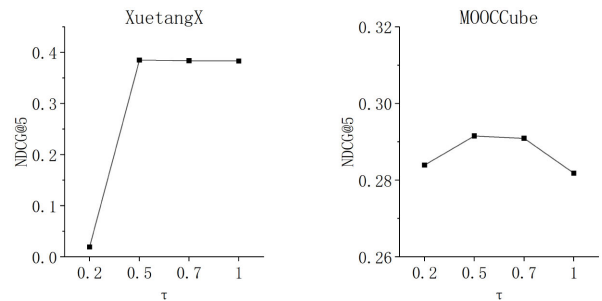


FIGURE 4. The influence of  $\tau$  of contrastive learning.

effect is the best. When  $\tau$  is 1, that is, no parameter is introduced, and when there is a difficult negative sample, the contrastive loss is relatively small, and the penalty for the parameter is also small. Since contrastive learning wants all feature vectors to be as far away as possible, it is necessary to increase the penalty for all misclassified samples, so the value of  $\tau$  must be less than 1 to amplify the penalty for difficult negative samples. However, when the value of  $\tau$  is too small, contrastive learning will over-focus on difficult tasks, which will make normal recommendation tasks cannot be completed, and the model recommendation effect will decline.

### 3) ABLATION EXPERIMENT(Q3)

In order to further verify the effectiveness of the proposed method, ablation experiments were conducted on

**TABLE 4. Composition of ablation model.**

| Method     | LD <sup>a</sup> | GCV <sup>b</sup> | CL <sup>c</sup> | DCL <sup>d</sup> |
|------------|-----------------|------------------|-----------------|------------------|
| DGDCL-WD   |                 | ✓                |                 | ✓                |
| DGDCL-WCL  | ✓               |                  |                 |                  |
| DGDCL-WA   | ✓               |                  |                 | ✓                |
| DGDCL-WDCL | ✓               | ✓                | ✓               |                  |
| DGDCL      | ✓               | ✓                |                 | ✓                |

<sup>a</sup>Layer Dropout(LD)<sup>b</sup>Generat Contrastive Views(GCV)<sup>c</sup>Contrastive Learning(CL)<sup>d</sup>Differential Contrastive Learning(DCL)**TABLE 5. Ablation experiments results for DGDCL.**

| Dataset      | XuetangX      |               | MOOCCube      |               |
|--------------|---------------|---------------|---------------|---------------|
|              | Recall        | NDCG          | Recall        | NDCG          |
| LightGCN     | 0.3615        | 0.3659        | 0.2765        | 0.2680        |
| DGDCL-WD     | 0.3767        | 0.3803        | 0.2883        | 0.2811        |
| DGDCL-WCL    | 0.3745        | 0.3769        | 0.2855        | 0.2792        |
| DGDCL-WA     | 0.3791        | 0.3813        | 0.2903        | 0.2851        |
| DGDCL-WDCL   | 0.3794        | 0.3826        | 0.2949        | 0.2875        |
| <b>DGDCL</b> | <b>0.3812</b> | <b>0.3850</b> | <b>0.2974</b> | <b>0.2916</b> |

two datasets, mainly comparing the layer dropout method proposed in this paper, layer dropout contrastive learning and graph differential contrastive learning. The experimental results are shown in Table 4. The full DGDCL model achieved the best recommendation results, and while the remaining three variants had poor course recommendations, they also outperformed the baseline model LightGCN.

The DGDCL-WD model does not use the layer dropout training strategy in the recommended main task. DGDCL-WD model course recommendation effect decreased, indicating the necessity of layer dropout. Layer dropout can alleviate the overfitting problem of graph convolution in training and make the model have good generalization ability.

The DGDCL-WCL model removes the contrastive learning task and uses only layer dropout. The declining course recommendation effect of DGDCL-WCL model indicates that graph contrastive learning plays an important role in alleviating data sparsity and enabling models to learn even and high-quality embeddings.

DGDCL-WA model does not generate contrastive view, but directly uses the embeddings in the main task of recommendation for contrastive learning. The decline in the DGDCL-WA model course recommendation indicates the effectiveness of using layer dropout to generate contrastive views. Although DGDCL-WA further simplifies contrastive learning, it is better to use layer dropout contrastive learning. The view generated by layer dropout contrastive learning comes from the original multi-layer graph convolution features. Multiple different contrastive views can help the

model learn different features and observe and compare the graph convolution features from multiple perspectives, which helps to improve the robustness and generalization ability of the model.

The DGDCL-WDCL model removes the differential calculation in the contrastive learning loss function. The main reason for the decline in the course recommendation effect of DGDCL-WDCL model is that the original contrastive learning simply divides graph nodes into positive and negative categories, so for some graph nodes that should have high similarity, the similarity distance is far away from each other, which hinders the completion of the recommendation task and fails to bring the best effect of contrastive learning.

## V. CONCLUSION

Personalized recommendation model has been widely used in various fields of the Internet, but there is relatively little research on course recommendation. Therefore, this paper proposes a course recommendation model based on layer dropout graph differential contrastive learning. In order to improve the shortcomings of the existing recommendation model, the high level information of user-user and course-course is obtained through hypergraph. Using the idea of dropout method in neural networks, a graph contrastive learning model based on layer dropout is proposed. On the one hand, layer dropout can alleviate the overfitting problem caused by multiple graph stacking in graph neural networks, and on the other hand, data sparsity can be alleviated by graph contrastive learning. In addition, to improve the contrastive loss, we put forward differential contrastive learning, which effectively improves the effect of contrastive learning in recommendation. Experiments on two course selection datasets show that the proposed method can effectively improve course recommendation quality, and the effectiveness of each component of the model is fully verified by ablation experiments.

In this model, only the user-course information is used for course recommendation, which has certain universality of recommendation model. However, there is still a wealth of associated information in the characteristic information of the course, such as the knowledge points contained in the course, the teachers of the university teaching the course and other information. How to integrate other types of auxiliary information into the recommendation model to further improve the effect of course recommendation will be the next research direction.

## REFERENCES

- [1] X. Wang, W. Ma, L. Guo, H. Jiang, F. Liu, and C. Xu, "HGNN: Hyperedge-based graph neural network for MOOC course recommendation," *Inf. Process. Manage.*, vol. 59, no. 3, May 2022, Art. no. 102938.
- [2] J. Zhang, B. Hao, B. Chen, C. Li, H. Chen, and J. Sun, "Hierarchical reinforcement learning for course recommendation in MOOCs," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 435–442.
- [3] X. Li, X. Li, J. Tang, T. Wang, Y. Zhang, and H. Chen, *Improving Deep Item-Based Collaborative Filtering With Bayesian Personalized Ranking for MOOC Course Recommendation*. Cham, Switzerland: Springer, 2020, pp. 247–258.



- [4] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "LightGCN: Simplifying and powering graph convolution network for recommendation," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2020, pp. 639–648.
- [5] S. Ji, Y. Feng, R. Ji, X. Zhao, W. Tang, and Y. Gao, "Dual channel hypergraph collaborative filtering," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2020, pp. 2020–2029.
- [6] G. Li, M. Müller, A. Thabet, and B. Ghanem, "DeepGCNs: Can GCNs go as deep as CNNs?" in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9266–9275.
- [7] G. Zhu, W. Lu, C. Yuan, and Y. Huang, "AdaMCL: Adaptive fusion multi-view contrastive learning for collaborative filtering," in *Proc. 46th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2023, pp. 1076–1085.
- [8] F. Wu, X.-Y. Jing, P. Wei, C. Lan, Y. Ji, G.-P. Jiang, and Q. Huang, "Semi-supervised multi-view graph convolutional networks with application to webpage classification," *Inf. Sci.*, vol. 591, pp. 142–154, Apr. 2022.
- [9] X. Wang, X. He, M. Wang, F. Feng, and T. Chua, "Neural graph collaborative filtering," in *Proc. SIGIR*. New York, NY, USA, 2019, pp. 165–174.
- [10] U. Jeong, K. Ding, L. Cheng, R. Guo, K. Shu, and H. Liu, "Nothing stands alone: Relational fake news detection with hypergraph neural networks," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2022, pp. 596–605.
- [11] H. Yuan, J. Yang, and J. Huang, "Improving hypergraph convolution network collaborative filtering with feature crossing and contrastive learning," *Appl. Intell.*, vol. 52, no. 9, pp. 10220–10233, Jul. 2022.
- [12] X. Li, R. Guo, J. Chen, Y. Hu, M. Qu, and B. Jiang, "Effective hybrid graph and hypergraph convolution network for collaborative filtering," *Neural Comput. Appl.*, vol. 35, no. 3, pp. 2633–2646, Jan. 2023.
- [13] L. Huang, S. Dai, and Z. He, "Few-shot object detection with dense-global feature interaction and dual-contrastive learning," *Appl. Intell.*, vol. 53, no. 11, pp. 14547–14564, Jun. 2023.
- [14] J. Wu, X. Wang, F. Feng, X. He, L. Chen, J. Lian, and X. Xie, "Self-supervised graph learning for recommendation," in *Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2021, pp. 726–735.
- [15] J. Yu, H. Yin, X. Xia, T. Chen, L. Cui, and Q. V. H. Nguyen, "Are graph augmentations necessary: Simple graph contrastive learning for recommendation," in *Proc. 45th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Jul. 2022, pp. 1294–1303.
- [16] J. Yu, X. Xia, T. Chen, L. Cui, N. Q. V. Hung, and H. Yin, "XSimGCL: Towards extremely simple graph contrastive learning for recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 2, pp. 913–926, 2024.
- [17] X. Jing and J. Tang, "Guess you like: Course recommendation in MOOCs," in *Proc. Int. Conf. Web Intell.*, New York, NY, USA, Aug. 2017, pp. 783–789.
- [18] D. Sheng, J. Yuan, Q. Xie, and L. Li, "ACMF: An attention collaborative extended matrix factorization based model for MOOC course service via a heterogeneous view," *Future Gener. Comput. Syst.*, vol. 126, pp. 211–224, Jan. 2022.
- [19] D. Sheng, J. Yuan, Q. Xie, and P. Luo, *MOOCRec: An Attention Meta-Path Based Model for Top-K Recommendation in MOOC*. Cham, Switzerland: Springer, 2020, pp. 280–288.
- [20] Y. Lin, S. Feng, F. Lin, W. Zeng, Y. Liu, and P. Wu, "Adaptive course recommendation in MOOCs," *Knowl.-Based Syst.*, vol. 224, Jul. 2021, Art. no. 107085.
- [21] J. Shu, B. Xi, Y. Li, F. Wu, C. Kamhoua, and J. Ma, "Understanding dropout for graph neural networks," in *Proc. Companion Proc. Web Conf.*, New York, NY, USA, Apr. 2022, pp. 1128–1138.
- [22] R. Qiu, Z. Huang, H. Yin, and Z. Wang, "Contrastive learning for representation degeneration problem in sequential recommendation," in *Proc. 15th ACM Int. Conf. Web Search Data Mining*, New York, NY, USA, Feb. 2022, pp. 813–823.
- [23] J. Yu, G. Luo, T. Xiao, Q. Zhong, Y. Wang, W. Feng, J. Luo, C. Wang, L. Hou, J. Li, Z. Liu, and J. Tang, "MOOCcube: A large-scale data repository for NLP applications in MOOCs," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, 2020, pp. 3135–3142.



**YONG OUYANG** received the M.S. degree from the Hubei University of Technology, Wuhan, China, in 2007. He is currently an Associate Professor with the School of Computer Science, Hubei University of Technology, where he is also the Director of the Department of Computer Science and Technology, School of Computer Science. His research interests include data mining and intelligent education.



**HAO LONG** received the M.S. degree from the Hubei University of Technology, Wuhan, China, in 2024. His main research interests include deep learning and data mining.



**RONG GAO** received the Ph.D. degree from Wuhan University, Wuhan, China, in 2018. He is currently an Assistant Professor with the School of Computer Science, Hubei University of Technology, Wuhan. His research interests include data mining and intelligent recommendation.



**JINGHANG LIU** received the Ph.D. degree from the Wuhan University of Technology, Wuhan, China, in 2018. He is currently an Assistant Professor with the School of Computer Science, Hubei University of Technology, Wuhan. His research interests include machine learning and artificial intelligence.

...