International Journal for Housing Science and Its Applications
Publish August 4, 2025. Volume 46, Issue 3 Pages 2889-2905

https://doi.org/10.70517/ijhsa463242

Research on the enhancement mechanism of generative AI-based multilevel computing model on the teaching effect of computer education

Mingxing Zhu[1,*] **and Xin Guo**[1]
[1] Zhixing College, Hubei University, Wuhan, Hubei, 430011, China
Corresponding authors: (e-mail: mxingz@163.com).

Abstract In the Internet era, recommender systems have become very important in daily life, and the combination of Generative Adversarial Networks (GANs) and recommender algorithms provides new opportunities for the development of this field. In order to solve the problem of computer education course resource recommendation, this paper combines the collaborative filtering recommendation algorithm and the sequence generation adversarial network to construct a generative adversarial network (MGFGAN) recommendation model with multi-dimensional gradient feedback, and designs a computer education course resource recommendation system with this as the core algorithm, and explores its role in improving the teaching effect. Compared with ItemPop, MF-BPR, and MGFGAN-I based on user interaction vectors, MGFGAN-A based on user attributes achieves optimal values in all recommendation performance indicators, and improves the performance of Precision@10, Recall@10, NDCG@10, MRR@10, and MRR@10 compared with MGFGAN-I, respectively. 0.0594, 0.0103, 0.0392, and 0.0829, respectively. Using the systematic clustering method, the results of the cluster analysis of the performance of the students in the experimental group using the recommender system of this paper and the control group not using the recommender system show that the experimental group of students achieved better results. This paper provides a methodological path for using generative AI to improve the teaching effectiveness of computer education.

Index Terms recommender system, multidimensional gradient feedback, generative adversarial network (GAN), collaborative filtering, systematic clustering

I. Introduction

The rapid development of computers has made computer technology widely used in social production and life, and the process of cultivating talents in colleges and universities also needs to meet the needs of society, so that the cultivated computer application talents can have strong professional knowledge and practical ability [1]-[3]. Computer education in colleges and universities should give full play to computer science and technology, strengthen the knowledge base of students, and promote the improvement of students' practical ability [4]. How to improve the quality and effect of college students' computer education has become an urgent issue. At the same time, the development of cutting-edge artificial intelligence technology brings computer education into a new realm, which can better meet the needs of college students [5], [6].

In the traditional education model, students are usually forced to learn at the same speed and method, however, each student's learning style and progress is unique, which often leads to a deviation between the teaching objectives and the actual results [7], [8]. Generative AI can provide personalized learning paths by analyzing a student's learning style, needs, and progress, which means that students who master knowledge quickly can move forward more quickly, while those who are confused about certain concepts can receive additional support [9]-[11]. This customized approach helps students to better understand and master computer skills while increasing their self-confidence [12]. Generative AI can also help educators better understand student needs and performance through data analysis [13]. By monitoring students' academic performance and study habits, generative AI can provide educators with valuable insights that can help them adjust their teaching methods to meet students' needs [14]-[16]. Such accurate predictions can help reduce students' academic frustration and improve their academic achievement [17]. Thus, by applying generative AI techniques more broadly, we can provide better teaching and learning in computing education and produce more students with computing skills to build a solid foundation for their careers.

In this paper, by combining collaborative filtering algorithm, sequential generative adversarial network and multidimensional gradient feedback mechanism, we realized the construction of generative adversarial network recommendation model for computer education curriculum resources, and designed a recommendation system with this model as the core. In order to evaluate the effectiveness of the recommendation model, MGFGAN-A based on user attributes is compared with temPop, MF-BPR, and MGFGAN-I based on user interaction vectors, and then combined with the systematic clustering method, the practical effectiveness of the recommender system in the teaching of computer education courses is explored.

II. Generative Adversarial Network Recommendation Model Incorporating Multidimensional Gradient Feedback

Aiming at the computer education network teaching course recommendation problem, this paper fuses collaborative filtering recommendation algorithm and generative adversarial network, and proposes generative adversarial network based on multidimensional gradient feedback mechanism (MGFGAN), in order to realize the intelligent recommendation of the course resources, and then to promote the enhancement of teaching effect.

II. A. Collaborative Filtering Algorithm

The core idea of collaborative filtering recommendation (CF) algorithms [18] is to construct binary relationships between users and items based on their historical behaviors, such as buying or not buying, clicking or not clicking, and then discovering the items that users may be interested in. The method can usually be categorized into three types according to the recommendation process, which are User-based CF, Item-Based CF and Model-Based CF.

II. A. 1) User-based collaborative filtering algorithm

The specific steps of the User-Based CF algorithm are:

(1) Collect data. Collect the user's historical behavioral data, i.e., explicit or implicit input data, from the user behavioral system, and then preprocess the data to obtain the scoring matrix $R_{m\times n}$ as shown in Equation (1):

$$\begin{array}{ccccccc} User\,/\,Item & I_1 & I_2 & \cdots & I_j & \cdots & I_n \\ U_1 & R_{11} & R_{12} & \cdots & R_{1j} & \cdot & R_{1n} \\ U_2 & R_{21} & R_{22} & \cdots & R_{2j} & \cdot & R_{2n} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ U_i & R_{i1} & R_{i2} & \cdots & R_{ij} & \cdots & R_{in} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ U_m & R_{m1} & R_{m2} & \cdots & R_{nj} & \cdots & R_{nm} \end{array} \tag{1}$$

where U_i and I_j represent user i and item j respectively, and R_{ij} represents the rating, with the value range of 1~5.

(2) Obtain the nearest neighbor set of the target user.User-Based CF recommends the target user mainly by using the hobbies of the users in the neighbor set to obtain the recommendation list.The neighbor set is obtained by calculating the similarity between the users, i.e., the first N users with high similarity are taken to construct the user neighbor set.

(3) Generate recommendations. After obtaining the set of similar items through step (2), the rating values of these similar items are taken out, and then the method shown in Equation (2) is calculated to predict the ratings of the unrated items of the users in order to obtain recommendations:

$$p_{u,i} = \overline{r}_u + \frac{\sum_{v\in N_u} sim(u,v)\times|r_{v,i} - \overline{r}_v|}{\sum_{v\in N_u} sim(u,v)} \tag{2}$$

where $p_{u,i}, r_{v,i}$ are the predicted ratings of item i by target user u, the ratings of item i by neighboring user v, respectively, and $\overline{r}_u, N_u$ are the mean value of ratings of all items by target user u, and the neighbor set of target user u, respectively, and $sim(u,v)$ is the similarity between the target user u and the neighbor user v.

II. A. 2) Project-based collaborative filtering algorithms

The Item-Based CF algorithm consists of three main steps:

(1) Data collection. This step is the same as User-BasedCF, which also builds the user-item rating matrix $R_{m\times n}$.

(2) Obtain the set of similar items. Different from User-BasedCF, Item-BasedCF mainly makes recommendations based on the information of similar items, so the core of this algorithm is to obtain the similar item set. The same is that both methods obtain the similar set by calculating the similarity degree.

(3) Generate recommendation. After obtaining the set of similar items through step (2), the rating values of these similar items are taken out, and then the method shown in Eq. (3) is calculated to predict the ratings of the user's unrated items, so as to obtain the recommendation:

$$p_{u,i} = \overline{r}_u + \frac{\sum_{j \in N_i} sim(i,j) \times | r_{u,j} - \overline{r}_j |}{\sum_{j \in N_i} sim(i,j)} \tag{3}$$

where $p_{u,i}, r_{u,j}$ represent the prediction score of the target user u on the project i and the rating of the target user u on the project j, and $\overline{r}_u, \overline{r}_j$ represent the average of the target user's u ratings for all items and the average of all users' ratings for the project j, respectively, and N_i is the set of similar items of the project i, $sim(i,j)$ is the similarity between project i and project j.

II. A. 3) Model-based collaborative filtering algorithms

In contrast to User-Based CF and Item-Based CF, Model-Based CF does not perform similarity calculations, but instead utilizes historical user data for training, and is trained by feeding this historical data into existing statistical modeling algorithms or machine learning algorithms, and then obtains a recommendation model, which can subsequently be used directly to predict specific item ratings. The advantage of this approach is that it does not have to maintain a highly dimensional user-item rating matrix and is more current.

II. A. 4) Similarity calculation method

Similarity calculation is the core step of User-Based CF and Item-Based CF, and the result of the calculation will directly determine the effect of recommendation, and the following types are used more often:

(1) Cosine similarity

Typically, a n-dimensional vector is used to represent a user's rating of an item, and if an item is not rated, it is noted as 0. If the rating vectors of users u and v in the n-dimensional space are R_u and R_v in that order, the similarity of u and u can be obtained by equation (4) as $Sim(u,v)$. i.e:

$$Sim(u,v) = cos(u,v) = \frac{R_u * R_v}{\| R_u \| * \| R_v \|} \tag{4}$$

In general, the size of cosine ranges from [0,1]. When $Sim(u,v) = 0$, it indicates that the similarity between users or projects is 0. When $Sim(u,v) = 1$, it indicates that the similarity between two users or two projects is very high.

In practical applications, cosine similarity is not effective in calculating the difference between comparing the values of the dimensions, thus making the final recommendation not good enough. In order to compensate for the above defects, the researcher has improved the cosine similarity and proposed the modified cosine similarity:

$$Sim(i,j) = \frac{\Sigma_{u \in U} (R_{u,i} - \overline{R}_u)(R_{u,j} - \overline{R}_u)}{\sqrt{\Sigma_{u \in U} (R_{u,i} - \overline{R}_u)^2} \sqrt{\sum_{u \in U} (R_{u,j} - \overline{R}_u)^2}} \tag{5}$$

where U denotes the set of users who have reviewed both item i and item j, $R_{u,i}$ denotes the rating of user u on item i, $r_{u,j}$ denotes the rating of user u on item j, and $\overline{R}_u$ denotes the mean of all ratings of user u on the item.

(2) Pearson correlation coefficient

This coefficient is often used to calculate the linear relationship between variables, and its size ranges from -1 to 1, where 0 represents no correlation, 1 represents complete correlation, and -1 represents complete negative correlation. In User-BasedCF, assuming that $I_{u,v}$ denotes the set of items that user u and user v have reviewed at the same time, the similarity $Sim(u,v)$ between user u and user v is expressed by the Pearson correlation coefficient:

$$Sim(u,v) = \frac{\Sigma_{i\in I_{u,v}}(R_{u,i}-\bar{R}_u)(R_{v,i}-\bar{R}_v)}{\sqrt{\sum_{i\in I_{u,v}}(R_{u,i}-\bar{R}_u)^2}\sqrt{\sum_{i\in I_{u,v}}(R_{v,i}-\bar{R}_v)^2}} \tag{6}$$

where $R_{u,i}$ denotes the rating of user u on item i, $R_{v,i}$ denotes the rating of user v on item i, $\bar{R}_u$ denotes the average rating of user u on all items that have been evaluated, and $\bar{R}_v$ denotes the average rating of user v on average rating on all items that have been evaluated.

(3) Euclidean distance similarity

Euclidean distance calculates the absolute distance between points in space, and the calculation results are closely related to the positional coordinates of each point in space, which can accurately show the absolute difference of numerical characteristics between each individual. It is defined as follows:

$$D(x,y) = \sqrt{\sum_{i=1}^{n}(x_i-y_i)^2} \tag{7}$$

The Euclidean distance calculated by the above formula is a value not less than 0. The similarity between users can then be visualized using the following formula:

$$Sim(u,v) = \frac{1}{1+D(u,v)} \tag{8}$$

From the above equation, it can be seen that when $D(u,v)$ is 0, the inter-user similarity $Sim(u,v)$ is 1, which indicates that the two users' interest preferences are very similar, and as $D(u,v)$ increases, the similarity gradually tends to 0, and the similarity between the two users decreases gradually.

II. B.Generating Adversarial Networks

Generative Adversarial Networks (GANs) [19] are very typical generative models that are able to predict the distribution of sample data after learning the features of those data, and finally generate new data by combining what has been learned.

II. B. 1) Generating Adversarial Network Principles

The GAN contains a generator (G) and a discriminator (D), both G and D consist of neural networks. The optimization function of GAN is shown in equation (9):

$$\begin{aligned}\min_G \max_D V(D,G) &= \mathrm{E}_{x\sim P_{data(x)}}[\log(D(x))] \\ &+\mathrm{E}_{z\sim P_{z(z)}}[\log(1-D(G(z)))]\end{aligned} \tag{9}$$

where $P_{data}(x)$ is the distribution of the real data, $G(z)$ is the process by which the generator G generates the data (converting the input random low-dimensional noise z into a high-dimensional pseudo-sample), and $D(x)$ is the probability that the sample x comes from $P_{data}(x)$.

In making updates to the generator G, the solution is optimal when and only when $P_{data} = P_g$. When updating the discriminator D, its solution is optimal when and only when $D^*(x) = \frac{P_{data}(x)}{P_{data(x)+P_g}}$. The final result of generative adversarial training performed by the GAN is that G generates a distribution close to the false P_{data} of the sample $G(z)$, so that D has no way to determine whether $G(z)$ is real data or fake data, at which point $D(G(z)) = 0.5$.

II. B. 2) Sequence Generation Adversarial Networks

Sequential Generative Adversarial Network (SeqGAN) [20] incorporates reinforcement learning on the basis of traditional GAN with RNN and CNN as generator and discriminator respectively.In policy gradient, the approximate state values are obtained by Monte Carlo search, and the generator G is treated as a stochastic parameter, and the G is trained by the policy gradient.The optimization function of SeqGAN is shown in Equation (10):

$$\min_{G}\max_{D} V(D,G) = -\mathrm{E}_{x\sim P_{data}}[\log(D(x))] - \mathrm{E}_{z\sim P_{z(z)}}[\log(1-D(G(z)))] \tag{10}$$

Since Sequential Generative Adversarial Networks are able to generate discrete data, while SeqGAN uses RNN as a generator to be able to make predictions on time series data, in this study, traditional collaborative filtering algorithms are combined with SeqGAN, and the scoring data in the CF algorithm is generated using SeqGAN predictions to reduce the sparsity of the data and enable the performance of the CF algorithm to be improved.

II. C.MGFGAN Recommendation Modeling

GAN-based recommendation model has the problem of poor stability, for this reason, this paper proposes a multidimensional gradient feedback generative adversarial network (MGFGAN) recommendation model.The structure of MGFGAN is shown in Fig. 1, which mainly includes two parts, the generator and the discriminator. The model requires vectorization of the original dataset. The raw dataset is converted into user rating vectors. And the generator is constructed based on the multilayer perceptual machine model, and the generator learns how to generate data that is consistent with the real vectors. Meanwhile, a negative sampling module is added to the generator to enhance the diversity of the generated model. An auto-encoder is introduced in the discriminator model to judge the gap between the generated data and the real data as a way to provide multi-dimensional gradient feedback for the generator. In the recommendation stage, this paper adopts Top-K recommendation. The training of the model is terminated after the discriminator fails to recognize the generated data and the real data.

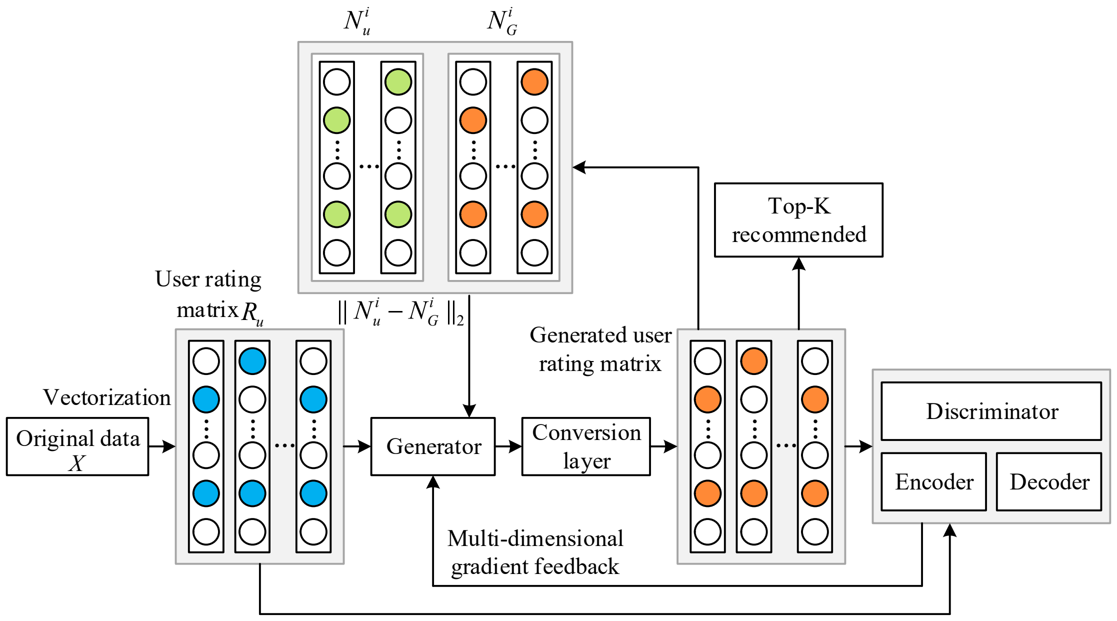


Figure 1: Architecture of MGFGAN

II. C. 1) A discriminator incorporating multidimensional gradient feedback

In order to solve the problem of insufficient gradient feedback obtained by the generator, this paper proposes a multidimensional gradient feedback mechanism by introducing a self-encoder. In the discriminator, the encoder encodes the input user rating vectors and converts them into low-dimensional vector representations to extract their feature information. The decoder, on the other hand, reconstructs the encoded vectors to produce the original data. When feeding back the multidimensional gradient, the Encoder part does not need to be adjusted directly, but is fed back to the generator through the Decoder and the reconstruction error. This feedback process enables the generator to better mimic the real data and improve the quality of the generated model. The discriminator in this paper borrows the idea of reconstruction error, and the expression is shown in equation (11):

$$D(X) = \| Dec(Enc(X)) - X \|_2 \tag{11}$$

where $D(X)$ represents the output of the discriminator, $Dec(\cdot)$ and $Enc(\cdot)$ represent the decoder and encoder of the self-encoder, respectively, $\|\cdot\|_2$ stands for the mean-square error, and X denotes the matrix of the user's real rating vector.

If the input data comes from the user rating vectors generated by the generator, at this time, after the output of the data is encoded and decoded, the discriminator utilizes the reconstruction error value to determine the difference between the user rating data and the real data X, and then obtains the multidimensional gradient feedback value as a guide for the iteration of the generator, and the objective function of the discriminator under this condition is as follows:

$$D(\hat{X}) = \| Dec(Enc(\hat{X})) - X \|_2 = \| Dec(Enc(trans(G(X)))) - X \|_2 \tag{12}$$

To equip the discriminator with the ability to discriminate real data and provide multidimensional gradient feedback, the loss function of the discriminator proposed in this paper contains the reconstruction error values of the above two components. On this basis, the parameter .m. is introduced, which serves to control the multidimensional gradient error. Thus the loss function of the discriminator is as follows:

$$L_D(X) = D(X) + [MGF(X) - m] = \sum_n D(r_n) + (D(\hat{r}_n \odot t_n) - m) \tag{13}$$

r_u represents user u's interactions with all items, while $MGF(\cdot)$ denotes the multidimensional gradient feedback function, m is the expected value of the error, and $trans(\cdot)$ represents the transformation layer in the MGFGAN, which serves to transform the generated vectors into user-rated vectors, where r_u is the generator G generated vectors, t_u is an n-dimensional transformation vector representing user u's interaction with each item, if there is a user u who has rated item i, $t_{ui} = 1$, and vice versa, $t_{ui} = 0$, and after the dot-multiplication operation $\odot$ between the vectors, which in fact converts $\hat{r}_u$ with the value of the unpurchased item corresponding to the true rating vector to 0.

The purpose of the loss function in the discriminator is that when the input data is a real user vector, its reconstruction error tends to 0. And when the input data is a false data generated by the generator, its reconstruction error value is controlled to be within the range of $[0, m]$, which represents the expected error value. When the value of $[m - MGF(\cdot)]$ is greater than 0, it means that the judgment of the discriminator for false data does not reach the expected value of m, and the value of L_D will increase, and vice versa, it decreases. Therefore, the function of this module is to regulate the recognition accuracy of the discriminator for different real data as well as the generated data, and to play the role of multi-dimensional gradient feedback. The role of this module is to regulate the recognition accuracy of the discriminator for different real data and generated data, and to provide multi-dimensional gradient feedback.

II. C. 2) Negative Sampling Based Generator

In the generator model of MGFGAN, the input data does not include the noisy data z, because after the introduction of the multidimensional gradient feedback mechanism in the discriminator, the data dimensions of the model explode, leading to a substantial increase in time complexity. Therefore, dropping the inclusion of noisy data will make the data generation process faster towards the real data, thus reducing the number of training rounds for the model.

In order to prevent the problem that the generator in GAN-based recommendation algorithms is unable to extract the user's features and capture the user's real preferences, the negative sampling module is introduced in this model. The negative sampling module is implemented by randomly selecting uninteracted items and discarding them in each training round. In this way, the model does not suffer a substantial loss in model diversity even if noisy data is discarded. Therefore, the loss function of the generator is shown in equation (14):

$$\begin{aligned} L_G(X) &= \lg(1 - MGF(X)) + \partial \| N_u^i - N_G^i \|_2 \\ &= \sum_u (\lg(1 - D(\hat{r}_u \odot t_u)) + \partial \sum_i (x_{ui} - \hat{x}_{ui})^2) \end{aligned} \tag{14}$$

where the value of $MGF(X)$ represents the gap between the user vectors generated by the generator and the true vectors, and the change in the value of this loss function is inversely proportional to the value of $MGF(X)$, which also means that the larger the gap between the user vectors generated by the generator and the true values, the greater the degree of loss penalty will be obtained, and vice versa.

In addition, Eq. N_u^i represents the set of negatively sampled items not purchased by a randomly selected user in the i th round of training, and x_{ui} denotes the negatively sampled items for each user u, i.e., the items that have not been rated by user u. N_G^i is then the negatively sampled set corresponding to N_u^i selected from the generation of rating vectors in the i th round of training, and thus $\hat{x}_{ui}$ represents the negatively sampled set of the generator-generated user u, and the items of $\hat{x}_{ui}$ correspond to the items in x_{ui}. items correspond sequentially. The δ is a tuning parameter that represents the importance of the negative sampling loss term. After adding the negative sampling mechanism, the generator will make the value of the purchased items close to 1 and the value of the unpurchased items close to 0, so as to prevent the generator from adopting an invalid generation strategy.

II. C. 3) Recommendation Process of MGFGAN Algorithm

The recommendation process of MGFGAN algorithm is as follows:

Inputs: user's original data x, generator learning rate λ_C, discriminator learning rate λ_D, generator negative sampling parameter δ, discriminator error expected value m, generator and discriminator sampling parameters M_c and M_D, generator and discriminator's parameter matrices θ and ϕ.

Output: a parameter matrix θ capable of generating a vector of user ratings.

Step1: Randomly initialize θ and ϕ.

Step2: Vectorize the original data X to get the user rating matrix $R = \{r_1, r_2, \cdots, r_n\}$.

Step3: for total-epochs do.

Step4: for each $u \in U$ do.

Step5: Randomize negative sampling to get negative sampling set N_u^i.

Step6: end for.

Step7: for G-steps do.

Step8: Obtain the user rating vector $\{r_1 r_2, \cdots, r_{M_C}\}$ based on the sampling parameters M_G.

Step9: Input the scoring vector into the generator G.

Step10: Generate and transform the fake user rating vectors $\{\hat{r}_1, \hat{r}_2, \cdots, \hat{r}_{M_C}\}$.

Step11: Update the parameter matrix of the generator $\theta = \theta - \lambda_G \cdot \nabla_\theta L_G$.

Step12: end for.

Step13: for D-steps do.

Step14: Obtain the user rating vector $\{r_1, r_2, \cdots, r_{M_D}\}$. based on the sampling parameters M_0.

Step15: The generator generates and converts the spurious rating vectors $\{\hat{r}_1, \hat{r}_2, \cdots, \hat{r}_{M_D}\}$.

Step16: Input the above vectors into the discriminator D.

Step17: Obtain the multidimensional gradient feedback value of the real purchase vector $D(X) = \| Dec(Enc(X)) - X \|_2$.

Step18: Obtain the multidimensional gradient feedback value $D(\hat{X}) = \| Dec(Enc(\hat{X})) - X \|_2$ for the generated user vector.

Step19: Construct the loss function L_D by combining the above multidimensional gradient feedback values.

Step20: Update the parameter matrix of the discriminator $\phi = \phi - \lambda_D \cdot \nabla_\phi L_D$.

Step21: end for.

Step22: end for.

Step23: return θ.

II. D.Experimental results and analysis

In order to verify that the proposed method can improve the recommendation performance, it is compared with some commonly used top-N recommendation methods on real datasets, and the effects of different negative sampling ratios and regularization factors on the recommendation performance are considered.

II. D. 1) Experimental setup and data set

The environment on which the algorithms in this paper run is Windows 10 operating system, Intel Core i7 CPU, 16GB RAM and Nvidia GeForce GTX 1080 graphics card. Publicly available datasets A and B are selected for experiments, and the datasets are randomly divided into training and testing sets in a 7:3 ratio. The training process uses Adam optimization in stochastic gradient descent (SGD) method. In this paper, implicit feedback is considered, if item i is relevant, i.e., user u rates the record of item i, $r_{ui}=1$, otherwise $r_{ui}=0$. The statistical information of the two datasets is shown in Table 1.

Table 1: The statistical information of the dataset

Data set	Number of users	Number of projects	Score evaluation	Sparsity /%
Dataset A	1132	2018	120000	94.76
Dataset B	7248	4742	1200318	96.24

After several experimental comparisons, the neural network hidden layer is set to three layers, the number of nodes in the hidden layer of the discriminator network is set to $\{1024,128,16\}$, and the number of nodes in the hidden layer of the generator network is set to $\{256,512,1024\}$. And some hyperparameters that perform well empirically are identified, and the sigmoid function is used as the activation function of the neural network, the learning rate is set to 0.0001, and the batch size is 32.

II. D. 2) Evaluation indicators

In this paper, four commonly used accuracy metrics are used for the top-N recommendation task: precision (P@N), recall (R@N), normalized discounted cumulative gain (NDCG@N), and mean reversed rank (MRR@N). The first two metrics focus on how many correct items are included in the recommendation list, and the latter two focus on the rank position of correct items in the recommendation list. For user u recommending N items, denoted as $R(u)$, and the set of items that user u interacts with on the test set is $T(u)$, the accuracy and recall are calculated as:

$$P@N=\frac{\sum_u |R(u)\cap T(u)|}{\sum_u |R(u)|} \tag{15}$$

$$R@N=\frac{\sum_u |R(u)\cap T(u)|}{\sum_u |T(u)|} \tag{16}$$

NDCG@N can be expressed as:

$$NDCG@N=\frac{\sum_u NDCG_u@N}{|U|} \tag{17}$$

MRR@N is calculated as:

$$MRR@N=\frac{1}{|U|}\sum_u \frac{1}{rank_u} \tag{18}$$

where $|U|$ is the number of users in the test set, and $rank_u$ denotes the position where the first recommendation is correctly located in the recommendation list of user u. The larger the above four evaluation indexes are, the better the recommendation performance of the model is. In this paper, the length of recommendation list N is fixed to 10 and 30.

II. D. 3) Comparison experiments

This section compares the recommendation performance of the proposed method in this paper with the commonly used top-N recommendation algorithms to verify whether the proposed method can improve the recommendation accuracy. The specific comparison methods are as follows:

ItemPop: the simplest non-personalized recommendation algorithm that sorts items in descending order of popularity, i.e., the number of purchase records.

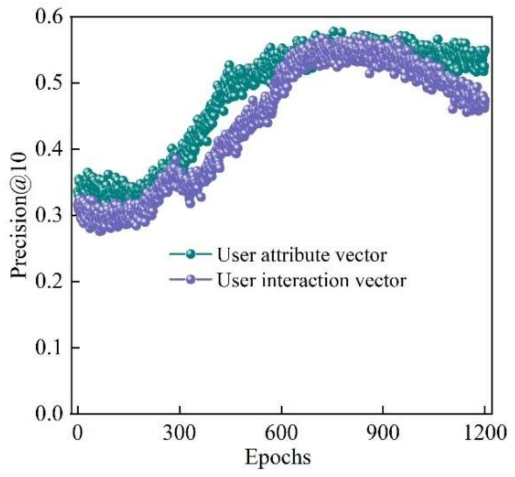
MF-BPR: a sorting algorithm based on matrix decomposition, which does sorting optimization for each user's own item preferences, and requires the user's preference sorting ternary of items for the training set.

MGFGAN-I: Adopting vector-based sequence generation adversarial network to learn real data distribution, the learning condition is the real interaction behavior of users.

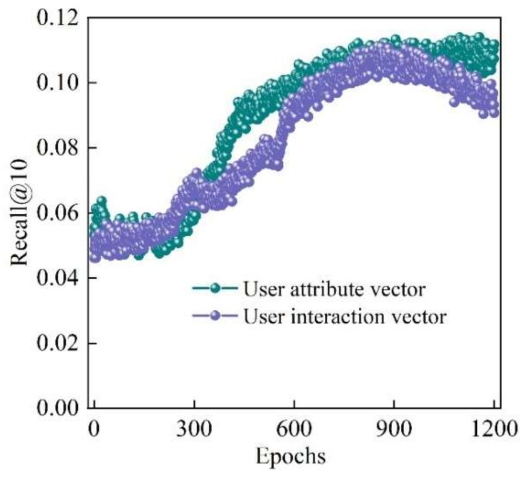
MGFGAN-A: User attributes are encoded and used as bootstrapping conditions for MGFGAN model to simulate user interaction behavior.

The performance comparison based on user attributes and user interaction vectors on dataset A is shown in Fig. 2, where (a)~(d) represent the comparison of four evaluation indexes, Precision, Recall, NDCG, and MRR, respectively, and the number of iterations is set to 1200.

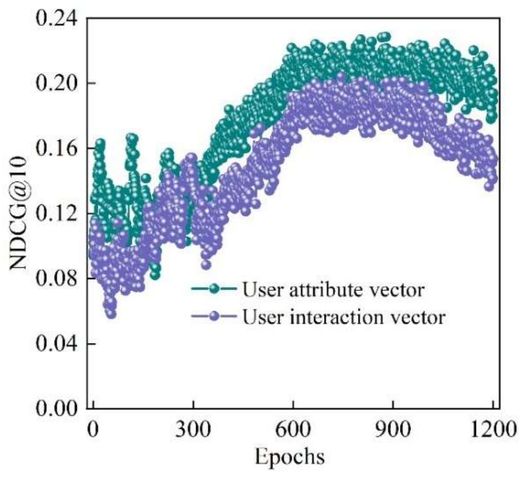
It can be seen that the user attribute-based MGFGAN recommendation method is slightly better than the user interaction vector-based recommendation in each evaluation index in general, which overcomes the error brought by the dichotomous input neural network fully connected layer of the unknown term in the user interaction vector-based MGFGAN model, thus verifying the effectiveness of the proposed method. Secondly, since the zero-sum game based on the GAN model is more difficult to converge and susceptible to hyperparameters. The evaluation indexes of the MGFGAN algorithm based on user interaction show an overall trend of increasing and then decreasing, and it starts to converge when it runs for about 800 rounds, but the evaluation indexes start to decrease with the increase of the number of rounds, which is due to the phenomenon of overfitting of the user interaction vectors, resulting in the decrease of the recommendation accuracy. On the other hand, the evaluation indexes of the MGFGAN model based on user attributes remain stable after convergence. Through the above analysis, it can be seen that the recommendation accuracy of the MGFGAN recommendation model based on user attributes converges more smoothly and stably, which also verifies that the convergence performance of the method is better.



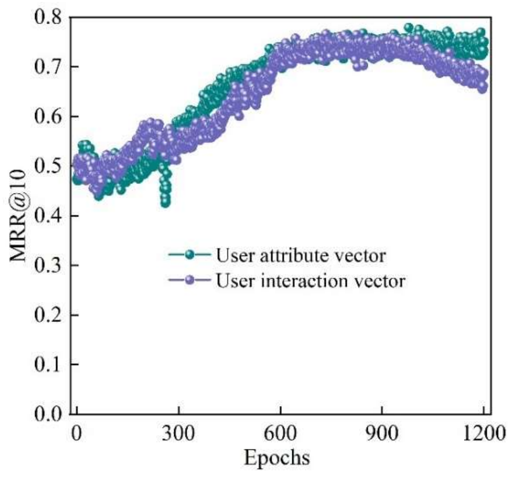
(a) Precision



(b) Recall



(c) NDCG



(d) MRR

Figure 2: Comparison of the recommendation performance of the MGFGAN model

The metrics performance of the two datasets recommending top-10 and top-30 relevant computer education courses under different recommendation methods are shown in Tables 2 and 3, denoted by P, R, G, and M for Precision, Recall, NDCG, and MRR, respectively.

As can be seen from Table 2, MGFGAN-A is slightly better than other recommendation methods in each evaluation index, and the recommendation accuracy P@10 and recommendation position ranking index G@10 are 2.27 times and 1.87 times that of non-personalized recommendation ItemPop, respectively, and compared with MGFGAN-I based on user interaction vector, it is 0.0594, 0.0103, 0.0392 and 0.0829 in P@10, R@10, G@10 and M@10 indicators, respectively. When recommending top-30 courses, MGFGAN-A also has some accuracy improvement compared with other methods, but the improvement effect of MGFGAN-A compared with top-10 is slightly reduced due to the recommendation of courses that are not very relevant. Table 3 also shows that the proposed method is superior to other recommendation methods, which verifies that the MGFGAN recommendation model based on user attributes can improve the recommendation performance.

Table 2: Performance comparison of different recommendation methods in the dataset A

Recommendation algorithm	P@10	R@10	G@10	M@10	P@30	R@30	G@30	M@30
ItemPop	0.2451	0.0824	0.1165	0.4618	0.1857	0.1825	0.1913	0.4866
MF-BPR	0.2302	0.0946	0.1407	0.5344	0.1675	0.2172	0.2135	0.5542
MGFGAN-I	0.4980	0.1029	0.1784	0.6863	0.3826	0.2749	0.3172	0.7095
MGFGAN-A	0.5574	0.1132	0.2176	0.7692	0.3963	0.2756	0.3254	0.7583

Table 3: Performance comparison of different recommendation methods in the dataset B

Recommendation algorithm	P@10	R@10	G@10	M@10	P@30	R@30	G@30	M@30
ItemPop	0.2437	0.0537	0.0915	0.2769	0.1872	0.1263	0.1482	0.2963
MF-BPR	0.2472	0.0558	0.0953	0.3425	0.1845	0.1294	0.1533	0.3725
MGFGAN-I	0.2928	0.0765	0.0887	0.4807	0.2234	0.1792	0.1864	0.5079
MGFGAN-A	0.3656	0.0946	0.1135	0.5754	0.2768	0.2346	0.2375	0.5984

III. Design of MGFGAN-based recommender system for computer education courses

In this chapter, we will design and deploy a computer education course resource recommendation system based on the MGFGAN deep learning algorithm using the dataset Movielens-100k.

III. A. Overall system architecture design

The architecture of the system in this paper is shown in Fig. 3, which is divided into four main layers, namely data layer, algorithm layer, business layer and user layer. The functions of each layer are as follows:

(1) Data Layer: the main task of the data layer is to store the data, using the relational database management system MySQL and distributed file storage system HDFS. where MySQL has the advantages of high performance and efficient service stability, small software memory, easy to install and easy to maintain, low cost, and active community users, etc. HDFS has the following advantages:

a) High fault tolerance, data can be saved in multiple copies, once the copy is lost, it can automatically realize the repair.

b) Suitable for big data processing, data scale can handle GB, TB and PB level data.

c) The use of streaming data access, a write, read multiple times, can ensure data consistency.

d) Can be built on inexpensive machines, highly reliable and highly scalable to support the expansion of more than 10k nodes.

(2) Algorithm layer: The recommendation algorithm in the algorithm layer of the system adopts the MGFGAN algorithm proposed in the previous section.

(3) Business layer: the development of business layer is realized through SpringBoot, Mybatis, Layui, Ajax and other technologies.

(4) User layer: the user layer mainly consists of two kinds of users, ordinary users and administrators.

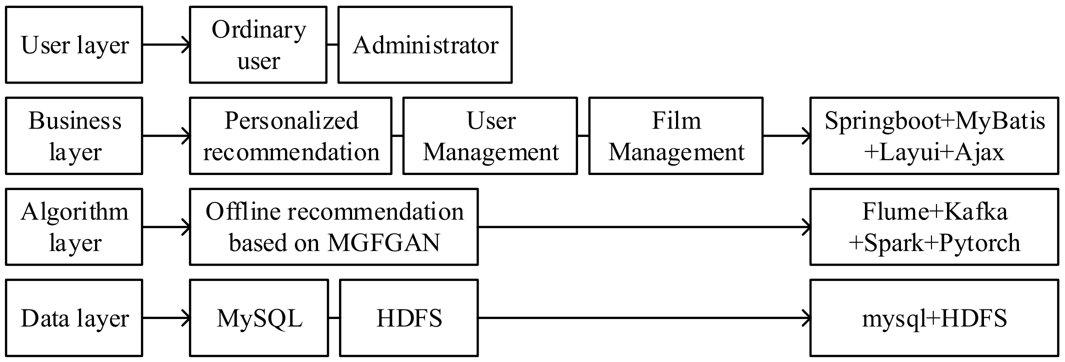


Figure 3: Architecture of the system

III. B. *Design of data processing links*

Following the principle of layered design, the data processing link of the system is shown in Fig. 4, including data source, data acquisition, data storage, data computation, algorithm processing and data display.

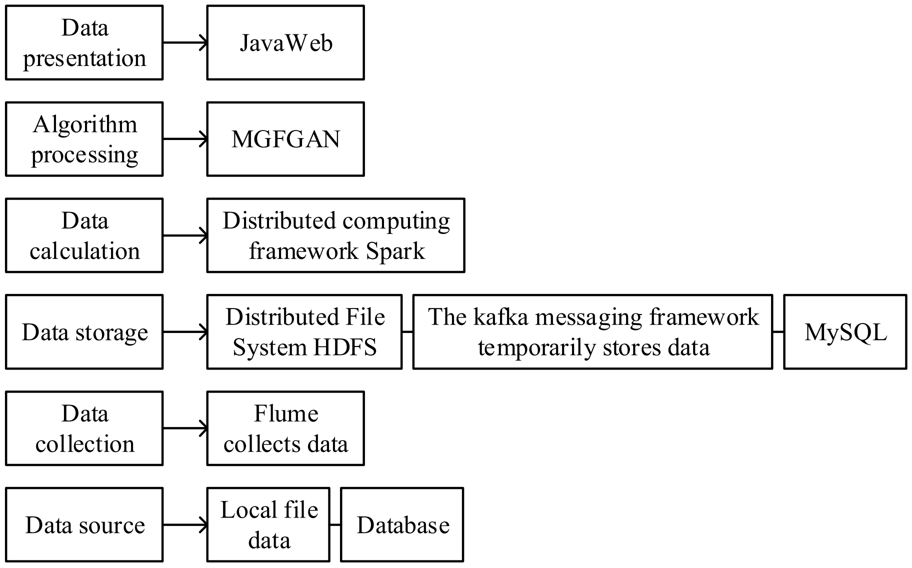


Figure 4: Data Processing Link

The specific functions of each part are as follows:

(1) Data source

The data source consists of two parts, which are offline data and real-time data.

The offline data is mainly Movielens 100k public dataset, which contains 100,000 scoring data. The dataset consists of three main parts of data. The first part is the offline user data, which consists of five fields, namely user id, user age, user gender, user specialty category, and user zip code. The second part is course resource data, which includes five fields, including course id, course name, course date, course link, and type to which the course belongs. The third part is the user's rating data of the course, including a total of 4 fields including user id, course id, rating value and timestamp.

The real-time data comes from the external service of the recommender system. The system generates three kinds of key real-time data during operation, which are new users, new course information and new comments. When users, courses and comments change, the table contents in their corresponding databases change, generating new binlog logs.

(2) Data Collection

Data collection includes offline data collection and real-time data collection. Offline data is collected using Flume, a highly available, highly reliable, distributed framework for massive log collection, which consists of Source data source, Channel buffer and Sink data sink. Real-time data is collected using Maxwell to monitor the real-time binlog logs added to MySQL data tables.Maxwell is a data synchronization software. It can read MySQL binary binlog in real time and generate messages in JSON format. Through the configuration file settings of Maxwell, the generated JSON format data can be sent to the topic of the Kafka messaging framework.

(3) Data storage

Data storage includes temporary data storage and final data storage. Offline data is temporarily stored in HDFS after collection, and then saved in HDFS after batch processing by Spark. While real-time logs are collected by Maxwell and firstly input to Kafka, then sparkstreaming consumes the corresponding topic data of Kafka in quasi-real-time and performs data preprocessing, and finally the preprocessed data is stored on the distributed file system HDFS. Real-time data and offline data are periodically merged and finally stored on HDFS.

(4) Data computation

Data computation refers to the preprocessing of data. The system preprocesses real-time logs in Kafka by batch processing of offline data through SparkCore and quasi-real-time through Spark Streaming, and finally merges the two kinds of data periodically. Because the preprocessed data can be used as input to the algorithm, preprocessing mainly extracts or transforms the features of the raw data. The core work of preprocessing includes preprocessing of user information and scoring data.

(5) Algorithm processing

Algorithm processing, mainly includes model training and model testing.

(6) Data presentation

After training the MGFGAN algorithm using user information data and user's rating matrix data, TopN recommendation results are generated for each user. The research of this paper focuses on the research of recommendation algorithm and the design of recommendation system. After comprehensively considering the rationality of the recommendation system design, it is decided to display the recommendation results in a simple way. The list of courses of TopN output by MGFGAN type is written to a table in MySQL database. For example, if we recommend 5 courses in Top5, we write the user id and the id of the 5 courses in the database. The system queries the course name and the course link through the course id and displays it to the user.

III. C. *Design of Algorithmic Processing Link*

The system's uses an offline training model approach, and the algorithm processing link is shown in Figure 5. The pre-processed user information data and user rating matrix data saved on HDFS are first read. Then the input data is used to train the MGFGAN model. Finally generate TopN recommendation results for each user and save the recommendation results to a table in MySQL for displaying the results at a later stage.

In this paper, the recommendation system consists of offline data and real-time binlog data newly generated in the system. With the implementation of time, the real-time data will be more and more. Therefore periodically the data is merged with the preprocessed offline data and real time data. Then retrain the model and display the latest list of course recommendations to the user.

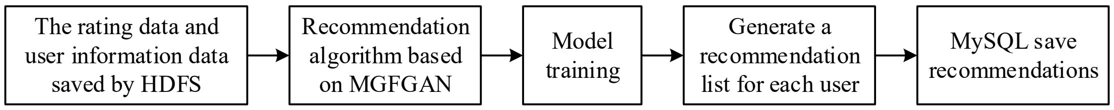

The rating data and user information data saved by HDFS
Recommendation algorithm based on MGFGAN
Model training
Generate a recommendation list for each user
MySQL save recommendations

Figure 5: Algorithm processing link

III. D. *Construction of experimental environment and platform*

In this section, the construction of the big data environment, the construction of the deep learning environment, and the construction of the computer education course recommendation system environment are described.

(1) Construction of big data environment

The data source of the recommendation system designed in this paper includes two parts of data, the first part is the offline dataset and the second part is the real-time log data. The data volume of real-time data will grow rapidly with time, so this system is based on big data Hadoop ecology for data storage and computation. The deployment of the big data platform designed in this paper is shown in Figure 6. The tasks of each component are as follows: the distributed file system HDFS is responsible for storing the files, the memory-based distributed computing framework Spark is responsible for the computation of the data, the message queuing framework Kafka is responsible for receiving the real-time data, the collection framework Flume is responsible for the collection of the offline data, and Maxwell is responsible for parsing the real-time data.

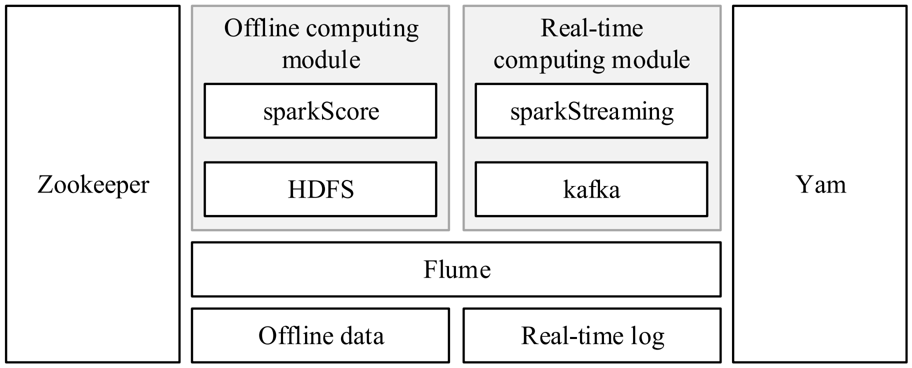


Figure 6: Hadoop big data platform

(2) The construction of deep learning environment

In this paper, we use a laptop whose GPU model is NVIDIA GeForce RTX 2060. The GPU memory is 16G, which can meet the needs of the system. First install Anaconda 4.7.10, then create Python 3.6.13 Pytorch environment, and finally install Pytorch 1.8.1.

(3) Building the course recommendation system environment

The construction of the recommendation system is biased towards background development and front-end development, involving a large amount of code, mainly using Springboot, MyBatis, Layui, Ajax and other frameworks to build.

IV. Analysis and evaluation of the effectiveness of the computer education course recommendation system

In order to assess the effect of the computer education course resource recommendation system designed in this paper on the enhancement of teaching effectiveness, this chapter utilizes the systematic clustering method [21] for experimental investigation.

IV. A. Experimental design

In this experiment, 15 sets of teaching tasks of undergraduate computer education course in J college were selected as the research object, which contains three independent modules, namely, computer basic knowledge and application module, advanced programming language module, and provincial computer level 2 examination module, which are referred to as basic module, language module, and province 2 module, respectively. The weighting ratio of 4:3:3 is used to sum up the Foundation Module, Language Module, and Provincial 2 Module to find the total assessment grade of the students. The computer education course of study requires students to take the Provincial II exam based on the completion of the first two modules. These three modules respectively examined the students' basic computer knowledge, computer programming ability, the comprehensive application of the standardized test environment, which can comprehensively reflect the students' memory, cognitive, and adaptive ability.

In this paper, according to the needs of the experimental group and control group design of the corresponding teaching subgroups respectively according to the three modules to collect the corresponding student performance. The control group (Group1) adopts the traditional teaching method, which is based on theoretical teaching and emphasizes the teacher's control of the classroom and the transmission and expression of information. In the experimental group (Group2), teachers are required to carry out educational and teaching activities based on the online teaching platform and adopt the flipped classroom teaching method to complete the corresponding teaching tasks, so as to strengthen the proportion of students' independent learning time outside the classroom and guide them to acquire the theoretical knowledge and skills of the teaching content through the course recommendation system, thus providing a basis for analyzing and evaluating the performance of the course recommendation system in the teaching and learning of the students in the experimental group according to the results of the different modules. The performance of the course recommendation system in teaching is analyzed and evaluated using the results of different modules of the experimental group. The teaching groups of the instructors and the corresponding module scores of the students they taught are shown in Table 4.

By analyzing the data in Table 4, the performance of students in Group1 is significantly lower than that of the corresponding experimental group Group2 (88.84, 91.06, 72.13, 84.49) in the basic module, language module, province 2 module and the overall assessment (73.07, 75.16, 58.94, 69.46).

Table 4: Teacher groups and corresponding module scores

Teachers	Score of the basic module	Language module score	The score of the second provincial module	Overall evaluation score	Group
Teacher_01	68.74	74.64	46.70	63.90	Group1
Teacher_02	73.75	72.51	67.83	71.60	Group1
Teacher_03	73.79	76.29	59.71	70.32	Group1
Teacher_04	72.68	77.93	61.27	70.83	Group1
Teacher_05	76.41	78.18	62.69	72.83	Group1
Teacher_06	73.02	71.43	55.46	67.28	Group1
Teacher_07	85.25	87.40	74.45	82.66	Group2
Teacher_08	92.03	92.08	63.80	83.58	Group2
Teacher_09	88.68	93.76	76.77	86.63	Group2
Teacher_10	88.68	87.81	74.99	84.31	Group2
Teacher_11	87.66	89.83	69.20	82.77	Group2
Teacher_12	91.66	88.82	63.88	82.47	Group2
Teacher_13	88.46	91.90	71.86	84.51	Group2
Teacher_14	93.03	95.26	70.98	87.08	Group2
Teacher_15	84.07	92.65	83.24	86.40	Group2

For the pre-processing of students' performance data of elective computer education courses, 2076 records of valid student performance data were finally obtained. Through Microsoft EXCEL, the effective students' grades were sorted according to the teachers of the experimental grouping, and then the grades of the three modules were categorized and summarized in order to calculate the average score of each module, and to obtain the average grades of the computer basic module, the average grades of the advanced programming language, and the average grades of the province II computer for each experimental grouping of students. SPSS was used to perform systematic cluster analysis method on the three module grades of 15 groups of students to obtain the clustering information.

IV. B. Experimental results and analysis

IV. B. 1) System Cluster Analysis Option Settings

Systematic Cluster Analysis was used to process the data in the table, in order to quantitatively analyze the differences between the flipped classroom teaching combined with the course recommendation system relative to the traditional education teaching. In this paper, the distance measure used in the systematic cluster analysis is the default Between-group Linkage, while the Measure uses Euclidean Distance, and the data standardization uses Z Scores, which ensures that the data are standardized to have a mean of 0 and a standard deviation of 1. Subsequently, the number of clusters is classified according to the experimental design, and the number of clusters is initially determined to be two. The number was initially determined as 2.

IV. B. 2) Systematic cluster analysis process

The process of analyzing the systematic clustering given by SPSS is shown in Table 5. The size of the distance between the two classes that were merged each time can be seen from the clustering coefficient in the fourth column of the table. During the corresponding cluster analysis process of the system, the change of this coefficient can be used to analyze the optimal number of categories for the 15 experimental subgroups after the systematic cluster analysis.

As can be seen from Table 5, the clustering coefficient of the 13th order is 3.448, which is only 0.467 units larger than the clustering coefficient of the 12th order, which is 2.981. And the clustering coefficient of the 14th order is 7.089 units larger than the clustering coefficient of the 13th order. As a result, it can be considered reasonable that the systematic cluster analysis process ends at the 13th order, when all the data are divided into two categories. And this is exactly in line with the experimental design idea of this paper, i.e., it reflects the difference between the performance of students in the experimental group and the control group.

Table 5: Cluster analysis process

Stage	Cluster combination		Coefficients	The order cluster emerged for the first time		The next stage
	Cluster 1	Cluster 2		Cluster 1	Cluster 2	
1	7	10	0.038	0	0	5
2	9	13	0.052	0	0	7
3	3	4	0.137	0	0	6
4	8	12	0.284	0	0	10
5	7	11	0.331	1	0	8
6	3	5	0.442	3	0	9
7	9	14	0.475	2	0	8
8	7	9	0.649	5	7	10
9	2	3	0.745	0	6	11
10	7	8	1.412	8	4	12
11	2	6	1.461	9	0	13
12	7	15	2.981	10	0	14
13	1	2	3.448	0	11	14
14	1	7	10.537	13	12	0

IV. B. 3) Cluster analysis results

The clustering results after systematic clustering analysis are shown in Table 6, which clearly shows that the students' performance in the experimental group taught by the six teachers in Group1 is in category 1, while the students' performance in the experimental group taught by the nine teachers in Group2 is in category 2. The category 2 scores reflect the impact of the course resource recommendation system on teaching effectiveness during the teachers' teaching process. This indicates that the designed course resource recommendation system can effectively improve the teaching effectiveness of computer education.

Table 6: Cluster analysis results

Case	Cluster	Case	Cluster
Teacher_01	1	Teacher_09	2
Teacher_02	1	Teacher_10	2
Teacher_03	1	Teacher_11	2
Teacher_04	1	Teacher_12	2
Teacher_05	1	Teacher_13	2
Teacher_06	1	Teacher_14	2
Teacher_07	2	Teacher_15	2
Teacher_08	2		

Based on this result, the corresponding clustering tree diagram can be drawn as shown in Figure 7. Further analysis based on Fig. 7 shows that the final clustering produces the results of the experimental grouping taught by Teacher_01 in Class 1 that is significantly different from the results of the rest of Group1. There is also a significant difference between the performance of the experimental grouping taught by Teacher_15 in category 2 and that of the rest of Group2.

After analyzing, the former is due to the fact that the teacher who undertakes the corresponding teaching task is a new teacher, whose teaching management ability and subject expertise are weaker than other teachers, resulting in the students' performance in the corresponding teaching task group being significantly lower than the average performance of Group1. In the latter case, the teacher is a backbone teacher with strong teaching ability, who can consciously connect the curriculum resource recommendation system with the traditional education and teaching process seamlessly, and can make full use of the organization and management function of the curriculum resource recommendation system to strengthen the control and management of the students' learning process, so that the performance of the students in the experimental group taught by the teacher is significantly better than that of the other experimental subgroups.

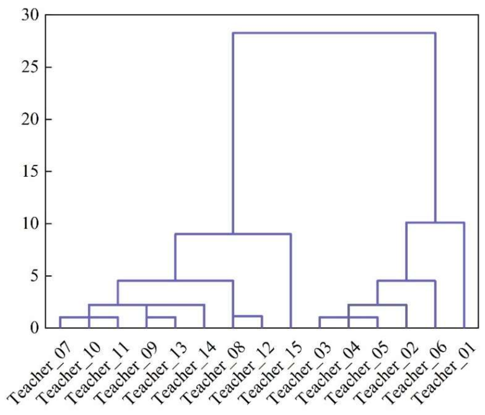


Figure 7: The tree diagram obtained by system clustering

V. Conclusion

Based on the multi-dimensional gradient feedback mechanism, combined with the collaborative filtering algorithm and the sequence generation adversarial network, this paper constructs a computer course recommendation system, and explores its mechanism for improving the teaching effect of computer education.

The MGFGAN-A recommendation method based on user attributes is slightly better than the MGFGAN-I recommendation method based on user interaction vectors in all evaluation indexes, and the recommendation accuracy converges more smoothly and stably, which verifies the effectiveness of the method. MGFGAN-A was slightly better than other recommendation methods in each evaluation index, and the recommendation accuracy P@10 and recommendation position ranking index G@10 were 2.27 times and 1.87 times that of non-personalized recommendation ItemPop, respectively, and were 0.0594, 0.0103, 0.0392 and 0.0829 higher than MGFGAN-I in P@10, R@10, G@10 and M@10 indicators, respectively. At the same time, MGFGAN-A is also better than other comparison methods when recommending top-30 courses, indicating that the MGFGAN recommendation model based on user attributes can improve the recommendation performance.

In the teaching experiment, the scores of the basic module, language module, province 2 module and the overall assessment of the experimental group Group2 using the curriculum resource recommendation system of this paper were 88.84, 91.06, 72.13 and 84.49 respectively, which were significantly higher than those of the control group Group1 using the traditional teaching mode.The results of the systematic clustering showed that the experimental group of 6 teachers teaching the experimental group of Group1 The clustering results of the system show that the students of the experimental group taught by the 6 teachers of Group1 are in category 1, while the students of the experimental group taught by the 9 teachers of Group2 are in category 2, which indicates that the designed curriculum resource recommendation system can have a significant impact on the teaching and learning effect of computer education. In addition, the final clustered results of the experimental group taught by the new teacher Teacher_01 in Category 1 are significantly lower than the results of the rest of Group1, while the results of the experimental group taught by the core teacher Teacher_15 in Category 2 are significantly higher compared to the results of the rest of Group2. This suggests that the flexible use of the recommendation system by core teachers also significantly affects the effectiveness of computer education teaching, further demonstrating the applicability of this paper's system to the task of recommending resources for computer education courses.

Funding

This work was supported by Zhixing College of Hubei University, the 2024 University-level Teaching Reform Research Project – The Transformation Practice of Computer Education Boosted by Generative AI Technology: A Perspective Based on Large Language Models (Project Number: XJY202415).

References

[1] Lei, J., Song, J. Q., & Wang, J. Y. (2024). Research on the Training Mode of Innovative Talents in Information and Computing Science from the Perspective of Scientific and Technological Innovation. International Journal of Education and Humanities, 4(1), 17-26.
[2] Ji, B., Liu, Y., Wang, Y., Wang, Z., & Song, S. (2023). Research on the Multidimensional Integration of Talent Cultivation in the Field of Electronic Information. International Journal of New Developments in Education, 5(21).
[3] Webb, M., Davis, N., Bell, T., Katz, Y. J., Reynolds, N., Chambers, D. P., & Sysło, M. M. (2017). Computer science in K-12 school curricula of the 2lst century: Why, what and when?. Education and Information Technologies, 22, 445-468.
[4] Sentance, S., & Csizmadia, A. (2017). Computing in the curriculum: Challenges and strategies from a teacher's perspective. Education and information technologies, 22, 469-495.

[5] Paiva, J. C., Leal, J. P., & Figueira, Á. (2022). Automated assessment in computer science education: A state-of-the-art review. ACM Transactions on Computing Education (TOCE), 22(3), 1-40.

[6] Mao, J., Chen, B., & Liu, J. C. (2024). Generative artificial intelligence in education and its implications for assessment. TechTrends, 68(1), 58-66.

[7] Dickey, E., Bejarano, A., & Garg, C. (2024). AI-Lab: A Framework for Introducing Generative Artificial Intelligence Tools in Computer Programming Courses. SN Computer Science, 5(6), 720.

[8] Gleeson, M. (2024). Exploring the Use of Generative AI in Computer Science Education in a Technological University in Ireland. European Journal of Open, Distance and E-Learning, 26(S1), 107-118.

[9] Prather, J., Denny, P., Leinonen, J., Becker, B. A., Albluwi, I., Craig, M., ... & Savelka, J. (2023). The robots are here: Navigating the generative ai revolution in computing education. In Proceedings of the 2023 working group reports on innovation and technology in computer science education (pp. 108-159).

[10] Denny, P., Prather, J., Becker, B. A., Finnie-Ansley, J., Hellas, A., Leinonen, J., ... & Sarsa, S. (2024). Computing education in the era of generative AI. Communications of the ACM, 67(2), 56-67.

[11] Ariza, J. Á., Restrepo, M. B., & Hernández, C. H. (2025). Generative AI in Engineering and Computing Education: A Scoping Review of Empirical Studies and Educational Practices. IEEE Access.

[12] Liu, R., Zenke, C., Liu, C., Holmes, A., Thornton, P., & Malan, D. J. (2024, March). Teaching CS50 with AI: leveraging generative artificial intelligence in computer science education. In Proceedings of the 55th ACM technical symposium on computer science education V. 1 (pp. 750-756).

[13] Garcia, M. B. (2025). Teaching and learning computer programming using ChatGPT: A rapid review of literature amid the rise of generative AI technologies. Education and Information Technologies, 1-25.

[14] Wu, Y. (2023). Integrating generative AI in education: how ChatGPT brings challenges for future learning and teaching. Journal of Advanced Research in Education, 2(4), 6-10.

[15] Baidoo-Anu, D., & Ansah, L. O. (2023). Education in the era of generative artificial intelligence (AI): Understanding the potential benefits of ChatGPT in promoting teaching and learning. Journal of AI, 7(1), 52-62.

[16] Ruiz-Rojas, L. I., Salvador-Ullauri, L., & Acosta-Vargas, P. (2024). Collaborative working and critical thinking: Adoption of generative artificial intelligence tools in higher education. Sustainability, 16(13), 5367.

[17] Hashmi, N., & Bal, A. S. (2024). Generative AI in higher education and beyond. Business Horizons, 67(5), 607-614.

[18] Weiwei Wang, Wenping Ma & Kun Yan. (2024). FSPPCFs: a privacy-preserving collaborative filtering recommendation scheme based on fuzzy C-means and Shapley value. Complex & Intelligent Systems,11(1),107-107.

[19] Srikanth Bethu. (2025). Malicious Attack Detection in IoT by Generative Adversarial Networks. SN Computer Science,6(4),372-372.

[20] Wang Yunhu, Zhang Guobao, Sun Tao, Zhang Yongchun, Huang Yongming & Liang Guoan. (2023). Fault Detection of Organic Heat Carrier Boilers Based on SeqGAN-CNN. Journal of Physics: Conference Series,2562(1),

[21] Rizwan Ur Rahman, Pavan Kumar, Aditya Mohan, Rabia Musheer Aziz & Deepak Singh Tomar. (2025). A Novel Technique for Image Captioning Based on Hierarchical Clustering and Deep Learning. SN Computer Science,6(4),360-360.