Journal of Combinatorial Mathematics
and Combinatorial Computing
www.combinatorialpress.com/jcmcc

Combinatorial Press

Research on automatic generation and personalized matching system of ideological and political education content based on intelligent technology

Yu Hang[1], Guanqun Zhang[2,] ✉

[1] *College of Continuing Education, North China Institute of Aerospace Engineering, Langfang, Hebei, 065000, China*

[2] *Hebei University of Engineering Science, Shijiazhuang, Hebei, 050000, China*

ABSTRACT

With the rapid development of artificial intelligence technology, the research on personalized learning in the field of ideological and political intelligence education is increasingly active. In this paper, an improved locust optimization algorithm is proposed, which is applied to the intelligent grouping strategy of ideological and political education. Then a knowledge state-oriented hypergraph self-attention knowledge tracking model is proposed, which consists of a hypergraph module and a self-attention module, and is capable of predicting students' future interaction sequences through their past interaction sequences. In order to realize students' personalized test question matching needs, a Civics test question recommendation algorithm based on the neural graph model is proposed, based on which a personalized Civics test question recommendation exam system is designed and implemented. The intelligent grouping strategy based on the optimized locust algorithm achieves a total score accuracy of 100% in the Civics grouping task. The knowledge tracking model accurately predicts students' knowledge status, and the attention weights of students' learning paths based on this paper's recommendation algorithm are all higher than 0.5. It shows the effectiveness of this paper's strategy of automatic generation of Civics education content based on the locust optimization algorithm and the personalized test question matching model on the students' in-depth understanding of the Civics knowledge and improvement of learning efficiency.

Keywords: Locust Optimization Algorithm, Intelligent Paper Formation Strategy, Knowledge Tracking, Test Question Recommendation, Neural Graph Model

✉ Corresponding author.
E-mail address: zhangguanqun1201@126.com (G. Zhang).
Received 03 March 2024; Revised 20 April 2024; Accepted 10 November 2024; Published Online 15 April 2025.
DOI: 10.61091/jcmcc127a-336

1. Introduction

With the increasing competition for talents in the international arena, the importance of modernization of education has been increasing in various countries. New teaching technologies and teaching concepts, such as intelligent education, data-driven teaching, flipped classroom and other teaching methods are constantly emerging [1]. A development trend of promoting education teaching reform with intelligent technology to advance education modernization has gradually emerged in the international arena. Most of the countries and regions with a relatively high level of educational intelligence have successively set up more specialized intelligent research institutions, providing important support for the modernization of national education [2]. Internationally, both developed and developing countries attach great importance to promoting the modernization of education through intelligent technology, so as to create conditions for the cultivation of talents needed in the era of intelligence. In China, along with the rise of the Internet, cloud computing, big data and intelligent computing technology, the future of teaching will be data-driven teaching, which requires corresponding changes in the teaching mode and organization to achieve personalized teaching to cultivate students' comprehensive abilities [3-4]. This new teaching mode requires changes in teaching content and its generation. Existing fine courses, open courses, online courses and other teaching resources exist in a single form, low sharing, slow updating and generation, which seriously restricts the further development of education [5-7].

Automatic generation tool makes education from the traditional paper text to digital, online learning mode, teachers can use it to quickly and accurately generate a variety of teaching resources, such as courseware, exercises, quizzes, lab reports and so on. This allows teachers to teach more efficiently and save a lot of time and energy [8-11]. In addition, automatic generation helps to realize personalized learning. Teaching materials can be automatically customized according to the different needs and ability levels of students, and by analyzing students' learning and feedback, the automatic generation tool can generate teaching materials, exercises and answers for each student that suit his or her learning style and level [12-14]. In this way, students can learn at their own pace and in their own way, improving their learning effectiveness and interest.

Ideological and political education aims to guide students to form a correct worldview, outlook on life and values, and to cultivate their moral ethics, legal concepts, national concepts and sense of social responsibility. However, as the teaching of various subjects have begun to pay attention to the integration and development of this course and ideological and political education, which has brought new opportunities and challenges to the ideological and political education, the content is diversified and comprehensive at the same time, the content is also more complex, and it is difficult for students to obtain the knowledge they want in the learning process in a targeted way [15], and the automatic generation and personalized recommendation solves this problem perfectly.

In this paper, based on the optimized locust algorithm, an intelligent grouping strategy for Civics education is completed, which improves the grouping efficiency and grouping quality of the traditional grouping strategy. Then a knowledge tracking model is proposed, which combines the advantages of hypergraph module and self-attention mechanism to accurately predict students' Civics learning behaviors in order to reveal students' Civics knowledge mastery level. Personalized matching between students and Civics test questions can further improve students' learning efficiency, therefore, this paper proposes a recommendation algorithm based on the neural graph model and introduces the algorithm into the construction process of personalized Civics test question matching system. Finally, the methods proposed in this paper are verified one by one through simulation experiments using MOOCCube dataset.

2. Establishment and Improvement of Civics Automatic Script-Grouping Model

2.1. Modeling of automatic grouping

Automatic grouping of papers according to the user's multiple constraints on the requirements of the computer system from a large and medium-sized Civics test bank automatically drawn to meet the user's requirements of the test paper, which is a multi-objective combinatorial optimization problem. To solve such problems are currently common in the following methods. (1) Random sampling method. According to the user's requirements, the computer randomly selects a number of different test questions to form a test paper, or until there are no questions in the test bank to be extracted. (2) Retrospective method. Each time the questions are drawn is tentative, if successful, the next step of the trial, if the drawing fails, return to the position of the last extraction, retry down again, if still unsuccessful, and then back one step forward until the test paper generation is completed or the trial is returned to the original point. (3) Genetic Algorithm. The algorithm starts from an initial solution and undergoes genetic evolution to produce generations of populations until the algorithm converges to an optimal solution. (4) Particle swarm algorithm. The algorithm mimics the process of a flock of birds foraging in the sky, with each bird updating its position and speed according to rules until an optimal solution is found.

Let each Civics topic has 6 attributes: topic type, section, score, difficulty factor, time spent, and frequency, and n test question is to be drawn from the test bank to make up a test paper that meets the requirements, which forms a $n\times 6$ one attribute matrix.

Each row of the matrix represents a test question and each column represents an attribute. Matrix S transforms the problem of forming a paper into a matrix solution problem, i.e., selecting a solution of matrix S from the test questions in the test bank according to the user's requirements for forming a paper.

Let the types in the test bank are fill-in-the-blank, multiple choice, judgment, question and answer 4 types, numbered 1~4, according to the user's requirements, set the proportion of each type of questions as $t\in\{t_1,t_2,t_3,t_4\}$, then there are $t_1+t_2+t_3+t_4=1$, the user satisfaction as shown in Equation (1):

$$B_k=1-\left|\left(\sum Aa_{i,1}\right)/T-t_k\right| A=\begin{cases}1 & Which\ a_{i,1}=k\\ 0 & Other\end{cases} \tag{1}$$

Where, $a_{i,1}$ is the question type of the i nd question, T is the total score of the question paper. t_k is the proportion of k types of questions in the question paper, where $k\in\{1,2,3,4\}$. The larger the value of $B_k\left(B_k\le 1\right)$, the higher the degree to which the k types of Civics questions fulfill the user's requirements.

An excellent Civics paper should have questions that cover all chapters as much as possible, and questions from important chapters should score higher than other chapters. Chapter scores (1~9) are given for each test question according to the importance of the chapter. Let there be 5 chapters in the Civics course.

Importance is scored as 1, 4, 7, 6, or 4, with Chapter 3 being the most important with a score of 7. Chapter 1 is the least important with a score of 1. Important chapters are accounted for as shown in equation (2):

$$C=\left(\sum Aa_{i,2}\right)\cdot w_j A=\begin{cases}1 & Which\ a_{i,2}=j\\ 0 & Other\end{cases} \tag{2}$$

Where $a_{i,2}$ is the chapter to which question i belongs. w_j is the importance score j, and the larger the C, the higher the percentage of test question scores for important chapters and the wider the coverage of test questions.

The paper score should be equal to the specified score N, but strictly constraining the paper score to be N is likely to reduce the algorithm's search efficiency. In fact, as long as the generated paper score is within the specified reasonable range, the ideal result can be achieved by appropriate manual fine-tuning. The constraints on the test paper scores are shown in Equation (3):

$$par = \sum a_{i,3} \ And \ \left| \sum a_{i,3} - N \right| \le r \tag{3}$$

where N is the user-specified mark for the paper (typically 100, 120, 150). a_{i3} is the score of the i rd question. r is the error allowed for the paper score.

The difficulty level of the test questions is categorized as 1, 2, and 3, which represent 3 levels of easy, medium, and hard, respectively. The difficulty constraint of the test paper is shown in equation (4):

$$dif = M - \left| \text{int}\left(\frac{\sum a_{i,A}}{n} \right) - M \right| \tag{4}$$

Where $a_{i,4}$ is the difficulty of the i nd question. n is the total number of questions in the paper. M is the difficulty level of the question paper requested by the user. The larger dif is, the closer the difficulty level of the paper is to the user's request.

The test paper time constraint is shown in equation (5):

$$time = \begin{cases} 1 & \left| \sum a_{i,5} - M \right| \le 15 \\ 0 & Other \end{cases} \tag{5}$$

Where, $a_{i,5}$ is the time taken for question i, and M is the user-specified exam time, taken as 90 or 120, with a 15-minute margin of error allowed.

The same question cannot appear more than two times (including two times) in a single exam, and it cannot appear consecutively in the last two consecutive exams. The test question frequency constraint is shown in Equation (6):

$$fre = \begin{cases} 1 & Satisfaction \ of \ conditions \\ 0 & Otherwise \end{cases} \tag{6}$$

$$Fit = time \cdot fre \cdot \left(r_1 \cdot \ dif + r_2 \cdot par + r_3 \cdot \left(\sum B_k \right) + r_4 \cdot C \right) \tag{7}$$

Where r_1, r_2, r_3, and r_4 are the weights of the 4 constraints of difficulty, total score, type of question and the chapter it belongs to, and $r_1 + r_2 + r_3 + r_4 = 1$. Answer time TIME and frequency FRE are practiced with one vote.

There are four types of questions in the Civics question bank: fill-in-the-blank, multiple choice, judgment, and quiz, with 80, 90, 105, and 15 questions, respectively, coded using segmented real numbers. Fill-in-the-blank questions are coded from 1 to 80, and the coding segment length is 2. Judgment questions are coded from 1 to 90 with a code length of 2. Judgment questions are coded from 1 to 105 with a code length of 3. Quiz questions are coded from 1 to 15 with a code length of 2. The codes of the 4 types of questions are combined together to form the code of a particle, or there are 4 zones in the code of each particle, and the code of each zone is also referred to as a gene, and the codes of the different zones may appear to be the same but they do not represent the the same Civics topic.

2.2. Intelligent Grouping Strategy Based on Improved Locust Optimization Algorithm

2.2.1. Locust Optimization Algorithm. Locust optimization algorithm [16] is used to solve the optimization problem. By simulating the foraging behavior of locusts, the small-scale movement of larvae is mapped as the local exploitation of the algorithm, and the large-scale hopping of adults is

mapped as the global exploration of the algorithm. Based on the distance between locusts, the forces can be categorized into attractive and repulsive forces. The position update of the basic GOA algorithm is shown in equation (8):

$$X_i^d = c\left\{\sum_{j=1, j\neq i}^{\kappa} c\frac{ub_d - lb_d}{2} s\left(\left|x_j^d - x_i^d\right|\right)\frac{x_j - x_i}{d_{ij}}\right\} + T_d \tag{8}$$

where ub_d and lb_d are the upper and lower bounds of the d rd dimension, respectively. X_i is the position of locust i. d_{ij} is the distance between locust i and locust j. T_d is the optimal solution for the current iteration round.

c is the linear decreasing coefficient, which is used to balance the process of algorithm development and exploration, as shown in Eq. (9):

$$c = c_{\max} - t \cdot \frac{c_{\max} - c_{\min}}{T_{\max}} \tag{9}$$

where $c_{\max}$ and $c_{\min}$ are the maximum and minimum values of c, respectively. $T_{\max}$ is the maximum number of iterations. t is the current iteration round.

The search process of GOA is the process of updating the position of each locust near the food by relying on the information of the force between the locusts, the optimal position, and the current position during the iteration process. In GOA, the position of each locust is mapped to each solution in the search space, and the advantages and disadvantages of each solution are measured by the fitness function, and the position of the food is mapped to the global optimal solution, and the process of searching for the source of food is the algorithm's optimization process.

2.2.2. Improvement strategies. In the locust optimization algorithm, each locust participates in the search process, and compared with other swarm intelligence algorithms, GOA has higher search efficiency. In order to further improve the search performance of GOA, this paper makes the following improvements to GOA and applies the improved algorithm to the intelligent Civics grouping strategy.

The new position update formula combines the communication information of all locusts, the position of each locust, the optimal position of the iterative process, and the random competitive position, in which the communication information between locusts can enhance the ability of the algorithm to search globally, the random competitive position can expand the algorithm's search space, and the optimal position of the iterative process allows each locust to search in the optimal direction.

As the number of iterations increases, the algorithm is prone to fall into the local optimum, and at this moment either the locusts in the winning group or the losing group will be close to this local optimal solution, leading to premature convergence of the algorithm. Therefore, in the second half of the algorithm, in order to make the algorithm can jump out of the local optimum, this paper introduces the idea of simulated annealing.

Simulated annealing algorithm comes from the annealing principle of solids, the solid is cooled to room temperature at high temperatures, the internal particles will be reduced with the temperature, the reduction of their own internal energy, the state of the particles will be regionally stabilized, and finally reach the base state at room temperature. At a particular temperature, the internal energy inside the solid will change due to the movement of the particles, and the system accepts this change if it proceeds in the direction of decreasing internal energy, and vice versa with a certain probability.

The simulated annealing algorithm follows the Metropolis criterion and accepts the worse solution with a certain probability. The probability calculation rule is shown in equation (10):

$$P_{ij}^T = \begin{cases} 1 & E(j) \le E(i) \\ e^{\left(\frac{E(j)-E(i)}{KT}\right)} & others \end{cases} \tag{10}$$

where $E(i)$ is the internal energy of the solid in the i state. $E(j)$ is the internal energy of the solid in the j state. K is the Boltzmann constant. T is the temperature.

At temperature T, the solid goes from state i to state j and accepts j as the current state if the internal energy of state j is less than that of state i, otherwise the state is accepted with a certain probability according to Eq. (10).

In the simulated annealing idea, the random position of locusts is shown in Eq. (11):

$$X_{newi} = \frac{1}{2}\left(x_i + \frac{(ub - lb) \times 2 \times rand}{l^2} - \frac{ub - lb}{l^2} \right) \tag{11}$$

where $rand$ is a random number in $(0,1)$.

Introducing the simulated annealing idea into the locust optimization algorithm helps to expand the search space of the algorithm, jump out of the local optimum, and enhance the global search capability.

c in GOA is the key parameter that controls the transformation of the algorithm from exploration to exploitation process. In the traditional locust optimization algorithm, c is linearly decreasing, which cannot dynamically adjust the search step of the algorithm, thus affecting the convergence speed of the algorithm. Therefore, this paper proposes a cosine adaptive weight adjustment algorithm for the search step, Eq. (12):

$$C(l) = \cos\left(\frac{\pi}{2} \times \frac{l}{L} \right) \times \left(c_{\max} - \frac{l}{L} \times \left(c_{\max} - c_{\min} \right) \right) \tag{12}$$

Where L is the maximum number of iterations and l is the current iteration round.

The improved formula has a larger step size and slower descent at the beginning of the algorithm, which helps the algorithm to search globally. In the middle of the algorithm, the step size decreases more, which helps to improve the convergence speed of the algorithm. At the end of the algorithm, the step size is smaller and the rate of descent is slower, which helps the algorithm to search locally.

From the position update of the locust optimization algorithm, it can be seen that the position of each locust is determined by the current position, the communication information between the locusts and the optimal position of the current iteration, so the GOA has a strong local search ability, but the global search ability is weak. Therefore, this paper introduces a stochastic competition strategy in the first half of the iteration process. The random competition strategy means that in each round of iteration, two locusts are randomly selected to compare the adaptation values, and the one with higher adaptation is the winning individual, and the one with lower adaptation is the losing individual, which participates in the iterative process of the algorithm in order to expand the search space of the algorithm. The optimized position update is shown below:

$$\begin{aligned} X_{new}(t+1) = {} & \alpha_1 C + \alpha_2 \left(X_{win}(t) - X_i(t) \right) \\ & + \alpha_3 \left(X_{loss}(t) - X_i(t) \right) + \alpha_4 \left(X_{best}(t) - X_i(t) \right) \end{aligned} \tag{13}$$

$$C = \sum_{j=1, j \neq i}^{N} c \frac{ub_d - lb_d}{2} s\left(\left| x_j^d - x_i^d \right| \right) \frac{x_j - x_i}{d_{ij}} \tag{14}$$

α_4 is the optimal solution memory weights with reference to the adaptive weights of parameter c, which helps to control the transformation of the algorithm exploration and development process, Eq. (15):

$$\alpha_4 = \cos\left(\frac{\pi}{2} \times \frac{t}{T_{\max}} \right) \tag{15}$$

2.2.3. Steps of the Improved Locust Optimization Algorithm. The flow of the improved locust optimization algorithm (EGOA) proposed in this paper is shown in Fig. 1.

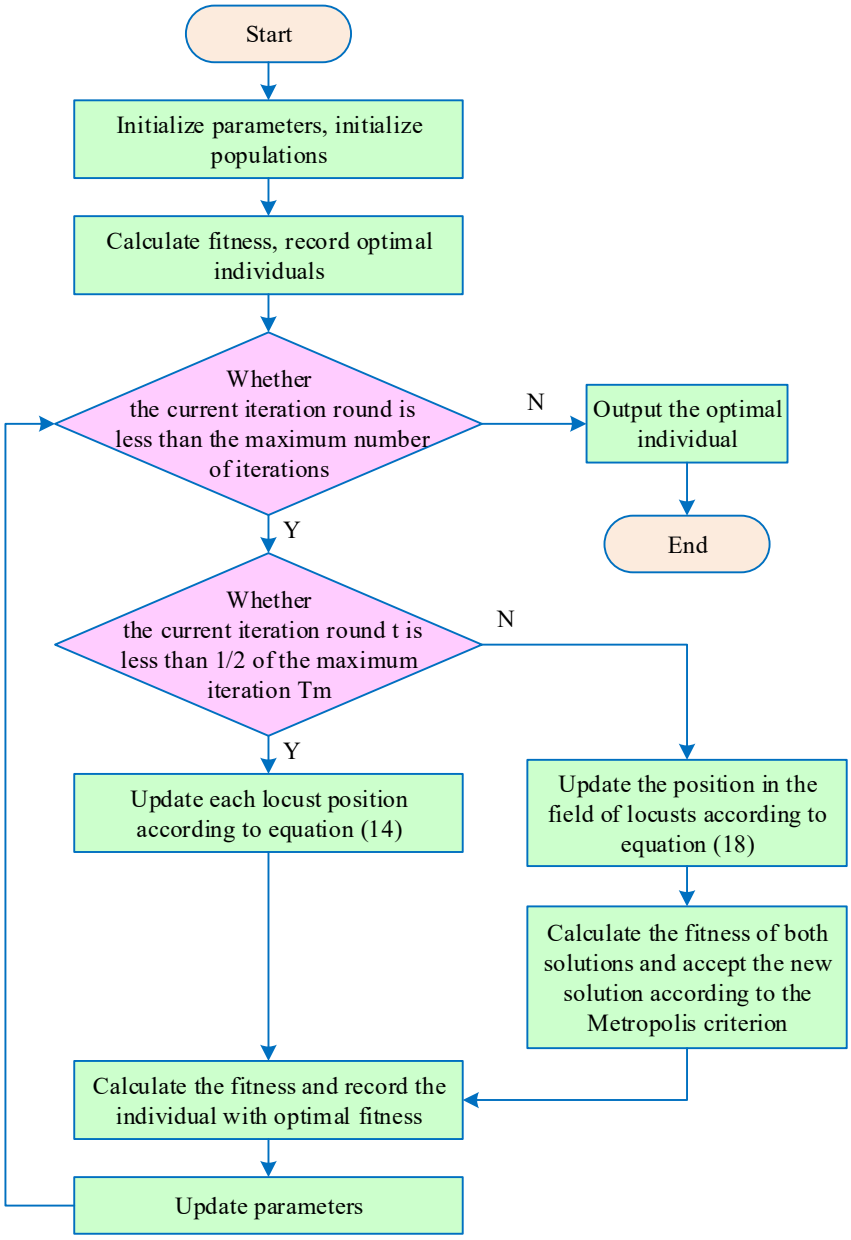

Fig. 1. Flow chart of EGOA

2.2.4. Intelligent Grouping Strategy Based on EGOA. To improve the search efficiency of the algorithm, the encoding method is very important. Currently, the common encoding method is binary encoding, which is relatively simple, but in the intelligent grouping strategy of this paper, the number of questions in the question bank is very large, and the use of binary encoding will lead to too long a code. Since the questions have been categorized according to the question types when the question bank is created, this paper adopts segmented real number encoding, and the encoding between different question types does not interfere with each other.

In this paper, the coding scheme of the question paper is mapped to locusts in EGOA, and the value of the fitness function is calculated to measure the advantages and disadvantages of each combination scheme.

Steps of intelligent paper grouping strategy based on EGOA:

Step 1 Initialize the test paper encoding scheme, each combination scheme is encoded as each locust in the algorithm.

Step 2 Initialize the parameters such as the number of iterations, the total score of the test questions, the question type, the difficulty coefficient, and the differentiation.

Step 3 Find the combination scheme with the optimal fitness value for each coding scheme.

Step 4 If the number of iteration rounds is less than 1/2 of the maximum number of iterations, skip to step 5, otherwise skip to step 6.

Step 5 update the locust location.

Step 6 Updates the position within the locust neighborhood, calculates the fitness of the two solutions, and Meuops subjected to the solution.

Step 7 Calculate the fitness and save the individual with the best fitness.

Step 8 Update the parameters.

Step 9If the number of iterations does not reach the maximum, skip to step 4. If the maximum is reached, output the optimal test combination solution and the algorithm ends.

3. Knowledge state-oriented self-attentive knowledge tracking model for hypergraphs

3.1. Concepts related to knowledge tracking

1) Definition 1: Knowledge concepts refer to the knowledge implicit in the Civics exercises. For example, the Civics exercise of finding the shortest palindrome implicitly includes the knowledge concept of string, and programming learners need to master the knowledge concept of string if they want to correctly answer the question of the shortest palindrome.

2) Definition 2: Knowledge state refers to the learner's mastery of different knowledge concepts, which is quantified by a value between 0 and 1, with 1 representing complete mastery of the relevant knowledge points and 0 representing no knowledge of the relevant knowledge points. It is well known that learners' knowledge state changes continuously during the learning process. According to Ebbinghaus' forgetting curve, students forget something during the learning process, which can cause them to do Civics practice questions wrongly, thus lowering their knowledge state. When learners answer a Civics exercise question correctly, the knowledge state of the knowledge concept related to that Civics exercise question will increase.

3) Definition 3: Civics Exercise Prediction means that the Knowledge Tracking Model takes as input the questions posed at each time step and the student's responses, uses a variety of methods to record how the student's knowledge evolves, and predicts the likelihood that the student will correctly answer new Civics Exercise questions.

$$P\left(R_{k+1}=1 \mid I_1, I_2, \cdots, I_k, E_{k+1}\right) \tag{16}$$

where the probability of correctly completing the $k+1$ st Civics exercise is denoted by $P\left(R_{k+1}\right)$. The knowledge tracking model inputs the first k interaction records $I_1, I_2, \ldots, I_k$ and the $k+1$ th Civics exercise E_{k+1} and outputs the probability of correctly completing the $k+1$ th Civics exercise.

Knowledge tracking [17] is a typical time-series prediction problem where learners answer Civics exercises containing different knowledge concepts at successive time steps. The knowledge concepts contain sequential relationships, e.g., learners first learn sequential tables and chain tables before they are capable of learning stacks and queues. It is this order relationship that makes the knowledge concepts have a natural graph structure, and extracting the graph structure of the knowledge concepts as auxiliary information for modeling learners' knowledge can improve the accuracy of model prediction.

3.2. Structure of the hypergraph self-attentive knowledge tracking model

Hypergraph self-attentive neural network is a knowledge tracking model proposed in this paper, which combines hypergraph and self-attention. The hypergraph module can not only group the Civics exercises based on knowledge concepts, but also directly generate interactive embeddings for learners.

Finally, this paper evaluates students' mastery of a particular Civics exercise in the current time period through a gating mechanism to predict whether learners can correctly answer the provided Civics exercises. This paper improves the self-attentive knowledge tracking model by introducing a deeper attention mechanism module applied to knowledge tracking.

3.2.1. Hypermap module. Hypergraph self-attentive knowledge tracking constructs an interaction hypergraph structure for the students' history doing sequence. Students' correct answer v_{i+} and incorrect answer v_{i-} to the i st Civics exercise question will generate embeddings x_{i+} and x_{i-}, respectively, by hypergraph convolution, and interaction nodes belonging to the same knowledge concept construct dependencies between interactions based on hyperedges of the knowledge concept.

$$h(v,e)=\begin{cases}1 & if\ v\in e\\ 0 & if\ v\notin e\end{cases} \tag{17}$$

Here, $h(v,e)$ denotes the matrix representation of the generated hypergraph. Once the hypergraph structure is constructed, the learner's feedback is immediately embedded in the nodes of the hypergraph, replacing the previous way of independently embedding Civics exercises and answers based on graph neural network knowledge tracking thereby better retaining the interaction information.

The hypergraph module of hypergraph self-attentive knowledge tracking extracts the features of hyperedges using hypergraph convolution based on the features embedded in the learner's historical interactions and the hypergraph adjacency matrix. Then, an aggregation operation is performed on the features of the hyperedge to complete the update of the interaction embedded features.

$$Y=D_v^{-\frac{1}{2}}HWD_e^{-1}H^TD_v^{-\frac{1}{2}}X\theta \tag{18}$$

$$D_v=\sum_{c\in E}\omega(e)h(v,e) \tag{19}$$

$$D_v=\sum_{v\in V}h(v,e) \tag{20}$$

Where, H is the feature matrix of the hyperedge, D_v is the degree of the node and D_e is the degree of the hyperedge. ω is the weight matrix of the hyperedge initialized with all values of 1, and θ is the learnable parameter.

3.2.2. Self-attention module. The self-attention module of the hypergraph self-attention knowledge tracking model is a converter-based encoder-decoder structure. Residual linking and layer normalization are applied at each layer of the self-attention module. First the learner's historical interactions are embedded as the encoder's attentional keys, values and queries. The attentional weights between interactions are computed and output, and then this information is returned to the decoder.

$$\begin{cases}M=LayerNorm(SkipConct(MHA(Q,K,V)))\\ O=LayerNorm(SkipConct(FFN(M)))\end{cases} \tag{21}$$

where M denotes the output of the encoder's self-attention layer, O denotes the output of the encoder, FFN is two superimposed linear layers, and Skip Conct is the residual connection. The first layer of the decoder uses multi-head attention to determine the attentional weights between the

embeddings of the Civic Exercise, while the second layer computes the connection between the interaction activity and the Civic Exercise.

$$\begin{cases} M_1 = LayerNorm(SkipConct(MHA(Q,K,V))) \\ M_2 = LayerNorm\left(SkipConct\left(MHA\left(M_1,O,O\right)\right)\right) \\ Dec = LayerNorm\left(SkipConct\left(FFN\left(M_2\right)\right)\right) \end{cases} \tag{22}$$

where M_1 represents the output of the first self-attention layer of the decoder, M_2 represents the output of the second self-attention layer of the decoder, and Dec represents the final output of the decoder.

In conclusion the self-attention module of hypergraph self-attention knowledge tracking can be described as:

$$\begin{cases} O = Encoder(I) \\ Dec = Decoder(E,O) \end{cases} \tag{23}$$

The formula for the hypergraph self-attentive knowledge tracking model of multiple attention is:

$$\begin{aligned} MHA = Concat\Bigg(& Soft\max\left(Mask\left(\frac{Q_1K_1^T}{2}\right)\right)V_1,\cdots \\ & Soft\max\left(Mask\left(\frac{Q_iK_i^T}{2}\right)\right)V_i\Bigg)W^o \end{aligned} \tag{24}$$

then divided by the square root of d, i.e., the dimensions of q and k, for dimensionality reduction. Finally, a mask is applied and the attentional weights activated by the softmax function are multiplied with v_i to obtain the output of the i th attention head. Finally, the outputs of all the attention heads are concatenated to form the output of the multi-head attention. The output of the multi-head attention is linear, and two layers of linear transformations with ReLU activation need to be added to enhance the expressive power of the hypergraph self-attention knowledge tracking.

$$FFN\left(X\right) = \mathrm{Re}\,LU\left(xW_1 + b_1\right)W_2 + b_2 \tag{25}$$

where x denotes the output of the multi-head attention. W_1, W_2, b_1 and b_2 are the shared offset and weight matrices.

3.3. Cyclic Gating Units and Gating Mechanisms

After initializing the learner's interaction hypergraph, the learner's interaction sequence corresponds to a path in the hypergraph, and in this paper, we further update the learner's knowledge state using a cyclic gating unit (GRU). The current knowledge state is computed by combining the knowledge state h_{1-1} from the previous time step with the storage unit C_1. In this process, the update gate acts as a counterbalance.

$$r_t = \sigma\left(W_r\left[h_{t-1}, x_t\right] + b_r\right) \tag{26}$$

$$u_t = \sigma\left(W_u\left[h_{t-1}, x_t\right] + b_u\right) \tag{27}$$

$$c_t = \tanh\left(W_c\left[r_t \square\ h_{t-1}, x_t\right] + b_c\right) \tag{28}$$

$$h_t = \left(1 - u_t\right) \square\ h_{t-1} + u_t \square\ C_t \tag{29}$$

where the weight matrices of the reset gate, update gate and storage cell are denoted by W_r, W_u and W_c, respectively. b_r, b_u and b_c are the learnable bias terms of the reset gate, update gate and memory cell respectively. $\square$ shows the multiplication of corresponding positional elements between matrices, and $\sigma(\ )$ and $\tanh(\ )$ are nonlinear activation functions.

The hypergraph self-attentive knowledge tracking model uses a gating mechanism to integrate the knowledge states of the two modules in order to combine the respective strengths of the two modules.

$$s_i^e = gate \square\ s_i^h + (1 - gate) \square\ s_i^t \tag{30}$$

The basic idea of the gating mechanism is described as follows:

$$gate = \sigma\left(\left(W_{g1} s_i^h + b_{g1}\right) + \left(W_{g^2} s_i^h + b_{g_2}\right)\right) \tag{31}$$

where the learnable weights of the hypergraph and attention modules are denoted by W_{g1} and W_{g2}, respectively. To balance the importance of the hypergraph and self-attention modules a gating mechanism is added before the prediction layer. The predicted value $\hat{r}_i$ of the learner's answer to Civic Exercise $e_i \in E$ is obtained from the knowledge state s_i^e by a linear layer using a sigmoid activation function.

$$\hat{r}_i = Sigmoid\left(s_i^e w_o + b_o\right) \tag{32}$$

4. Neural Graph Modeling Civics Test Recommendation Algorithm

4.1. Neural Graph Collaborative Filtering Algorithm

The core idea of Neural Graph Collaborative Filtering Algorithm (NGCF) [18] is to apply deep learning techniques in recommender system algorithms and exploits the higher-order connectivity of the user-item interaction graphs to extract deeper features of the users and items, which are used to enhance the Embedding feature representations of the users and items.

The Neural Graph Collaborative Filtering Model designs a neural network approach to propagate for recursion, thus realizing the integration of higher-order connectivity information into Embedding.

The Neural Graph Collaborative Filtering Model contains three main parts: an Embedding layer, multiple Embedding propagation layers and a prediction layer.

1) Embedding layer. It mainly maps the sparse feature vectors of users and items into dense types and generates the initial user and item Embedding. A matrix can be constructed as an Embedding lookup table:

$$E = \left[e_{u_1}, \cdots, e_{u_N} e_{i_1}, \cdots, e_{i_M}\right] \tag{33}$$

Where, $e_u \in R^d \left(e_i \in R^d\right)$, denotes the initial Embedding of users and items and d denotes the dimension. In the above set, the first N items denote users and the last M items denote items.

2) Embedding propagation layer. Based on the message-passing structure of GNN, the collaborative filtering signals are captured along the higher-order connectivity graph and the Embedding is optimized.Each propagation layer contains two stages: message construction and message synthesis.

(1) Message Construction. For a user u, with which an interaction has occurred with an item i, a user-item pair (u,i) is composed, defining a message from i to u:

$$m_{u \leftarrow i} = \frac{1}{\sqrt{|N_u||N_i|}}\left(W_1 e_i + W_2\left(e_i \square\ e_u\right)\right) \tag{34}$$

Where, $\frac{1}{\sqrt{|N_u||N_i|}}$ is the Laplace Paradigm, which denotes the attenuation factor of the propagation process, N_u and N_i denote the number of items that user u has interacted with and the number of users that have interacted with item i, respectively, W_1, W_2 denote two weight matrices, which are the parameters that need to be learned by the model, e_i and e_u denote the initial Embedding, and $\square$ denotes the element-by-element multiplication operator for the vectors.

(2) Message Synthesis. In order to obtain the optimized Embedding of user u after propagation, it is necessary to synthesize the messages from all items that have interacted with user u to u. The synthesis function is defined:

$$e_u^{(1)} = Leaky\,\mathrm{Re}\,lu\left(m_{u\leftarrow u} + \sum_{i\in N_u} m_{u\leftarrow i}\right) \tag{35}$$

where $e_u^{(1)}$ denotes the Embedding of user u after one layer of propagation, LeakyRelu is the activation function, and $m_{u\leftarrow u}$ denotes that the self-connected signal retains the features prior to propagation, $m_{u\leftarrow u} = W_1 e_u$.

Similar to obtaining the Embedding of user u above, it is also possible to obtain the Embedding representation of item i after one layer of propagation $e_i^{(1)}$, again for the above two stages.

Based on one layer of propagation, more layers of propagation can be added to explore higher order connectivity information, and the Embedding representation of user u at layer L is as follows:

$$e_u^{(L)} = Leaky\,\mathrm{Re}\,lu\left(m_{u\leftarrow u}^L + \sum_{i\in N_u} m_{u-i}^L\right) \tag{36}$$

Among them:

$$\begin{cases} m_{u\leftarrow i}^L = \frac{1}{\sqrt{|N_u||N_i|}}\left(W_1^L e_i^{(L-1)} + W_2^L\left(e_i^{L-1}\square\ e_u^{(L-1)}\right)\right) \\ m_{u\leftarrow u}^L = W_1^L e_u^{(L-1)} \end{cases} \tag{37}$$

W_1^L, W_2^L denote the weight matrices of the L rd propagation layer, which are also the parameters to be learned by the model. $e_i^{(L-1)}$ and $e_u^{(L-1)}$ denote the item and user Embedding obtained after the $L-1$th propagation layer. It can be seen that each layer is a recursive implementation based on the output of the previous layer as input. Similarly the representation of item i can be obtained after the L th propagation layer $e_i^{(L)}$.

3) Prediction Layer. After L layers of propagation, the representations $\left\{e_u^1,\ldots,e_u^L\right\}$ of the users u in each layer can be obtained, and the representations obtained in each layer show the preferences of the users u at different levels. Therefore, the representations obtained from each layer need to be connected to form the final Embedding. The same operation is done for item i. The final form is as follows:

$$e_u^* = \left\|e_u^0\right\|\ldots e_u^L, e_i^* = e_i^0\left\|\cdots\right\|e_i^L \tag{38}$$

Finally, the final Embedding of User u and Item i above is inner-producted to compute the ratings.

4.2. Neural Graph Collaborative Filtering Algorithm Improvement

In this paper, we introduce an attention mechanism in the propagation layer of NGCF by adding a variable weight to the construction of messages, each message has a different weight, the size of which

indicates the degree of preference, and the weight itself depends on the Embedding information of the USER and the ITEM.

NGCF is similar to the traditional Matrix Factorization (MF) representation at the prediction layer, which utilizes the inner product of the feature vectors p_u and q_i of the user and item to represent the interaction between user u and item i:

$$\hat{y}_{ui} = f\left(u,i\middle| p_w q_i\right) = p_u{}^T q_i = \sum_{k=1}^{K} p_{uk} q_{ik} \tag{39}$$

So, the inner product interacts by treating user and item features as if the dimensions are independent of each other and have the same weight, which makes it look more like a linear feature model. However, a simple inner product can actually limit the performance of the model.

In this paper, we borrowed the structure and idea of NCF, in the prediction layer of NGCF, firstly, the obtained Embedding of the final user and item are directly connected as the input of the neural network, and then set up a multi-layer neural network, and finally a neuron in the output layer is used as the output to obtain the score.

In this paper, the NGCF is improved on the basis of NGCF propagation layer and prediction layer respectively. First, compared with some traditional collaborative filtering algorithms, the NGCF based on this paper enhances the Embedding representation of user and item by explicitly encoding the interaction information. Second, this paper introduces an attention mechanism in the propagation layer of NGCF, which learns variable weights for user-associated items instead of the original equal weights or heuristic weights, resulting in a model with higher efficiency and better interpretability. Finally, this paper employs neural networks instead of inner product in the NGCF prediction layer to capture the complex nonlinear interactions between users and items, which improves the model accuracy. Finally, this paper proposes an attention model [19] structure NGCF-Att.

1) Embedding layer. The feature representations of users and items are initialized into vector representations, and at the same time, it is equivalent to constructing an Embedding lookup table of users and items.

2) Propagation Layer. For different items, different attention should be given. The attention mechanism can well learn the variable weights and assign them to different items or different components. The level of weights responds to the magnitude of relevance and preference.

(1) In the message construction phase, the message is redefined:

$$m_{u-i} = \frac{1}{\sqrt{\left|N_u\right|\left|N_i\right|}}\left(W_1\alpha_i e_i + W_2\left(\alpha_i e_i \square\ e_u\right)\right) \tag{40}$$

The above message construction process adds a weight α_i to the messages delivered for different interacting items i compared to the NGCF model, and the variable weight α_i indicates the degree to which the user has different preferences for the items. α_i is obtained by user e_u and item e_i computation as follows:

$$\alpha_i = w_1^T \varphi\left(W_3 e_u + W_4 e_i + b_3\right) + b_4 \tag{41}$$

$\varphi(x) = \max\left(0,x\right)$ denotes the ReLU activation function. The computation of the above equation is equivalent to the scoring of item i by user u, obtained by automatic learning of the network:

$$\alpha_i = \frac{\exp\left(\alpha_i\right)}{\sum_{j\in N_i}\exp\left(\alpha_i\right)} \tag{42}$$

(2) Then following NGCF, all the messages are merged in the message synthesis phase to obtain the optimized user Embedding after one propagation layer.Finally, through multi-layer propagation, the

Embedding of the user in each layer is obtained.The same computation is done to obtain the Embedding of the item in each layer.

3) Cascade Layer. The direct connection does not need to train additional parameters to obtain the final representations e_u^* and e_i^* after the connection.

4) Interactive layers. A tower-structured multilayer fully connected network that first connects e_u and e_i^* and then goes through L layers of neural network instead of inner product to capture nonlinear relationships.

$$\varphi = a_L\left(h_L^T\left(a_{L-1}\left(\ldots a_2\left(h_2^T\begin{bmatrix}e_v^*\\e_i^*\end{bmatrix}+b_2\right)\ldots\right)\right)+b_L\right) \tag{43}$$

Where, h_n^T is the weight of each layer of neural network, b_n is the bias of each layer of neural network, a_n is the activation function of each layer of neural network, and φ is the last layer of neural network before the output layer.

5) Output layer. There is only one neuron, and the last layer of the interaction layer is fully connected to that neuron to get a final result, which is the user's rating of the item y_{ui}.

$$\hat{y}_{ui} = a_{out}\left(h^T\varphi\right) \tag{44}$$

where h^T is the output layer weights and a_{out} is the output layer activation function.

5. Design and Implementation of Personalized Test Question Recommendation Examination System

5.1. Overall system design

The advantages of using B/S structure and MVC layered design of personalized test recommendation exam model are that there is a clear division of labor in the development process and the system is highly maintainable. The system is divided into a three-tier structure and the overall architecture diagram is shown in Figure 2.

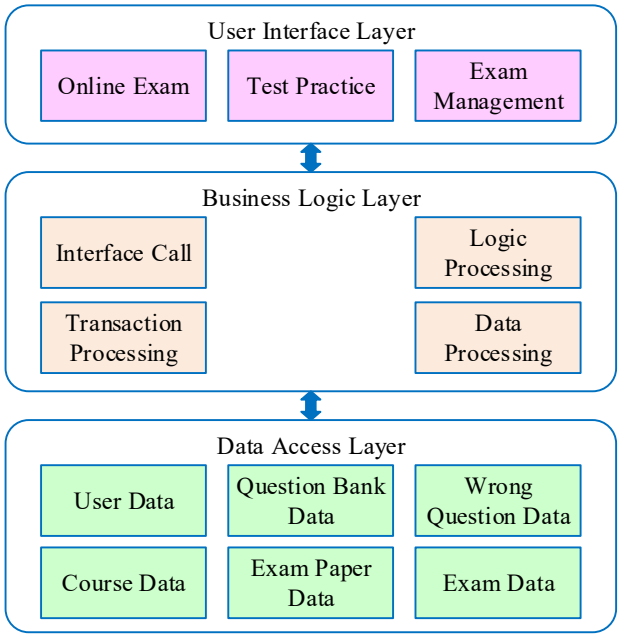

Fig. 2. System layered architecture

5.2. System detailed function realization

Through the demand analysis of the personalized Civics test question recommendation online examination system and the functional design of each user side, the functional implementation of the system as a whole is done, and the overall flow chart of the system is shown in Figure 3.

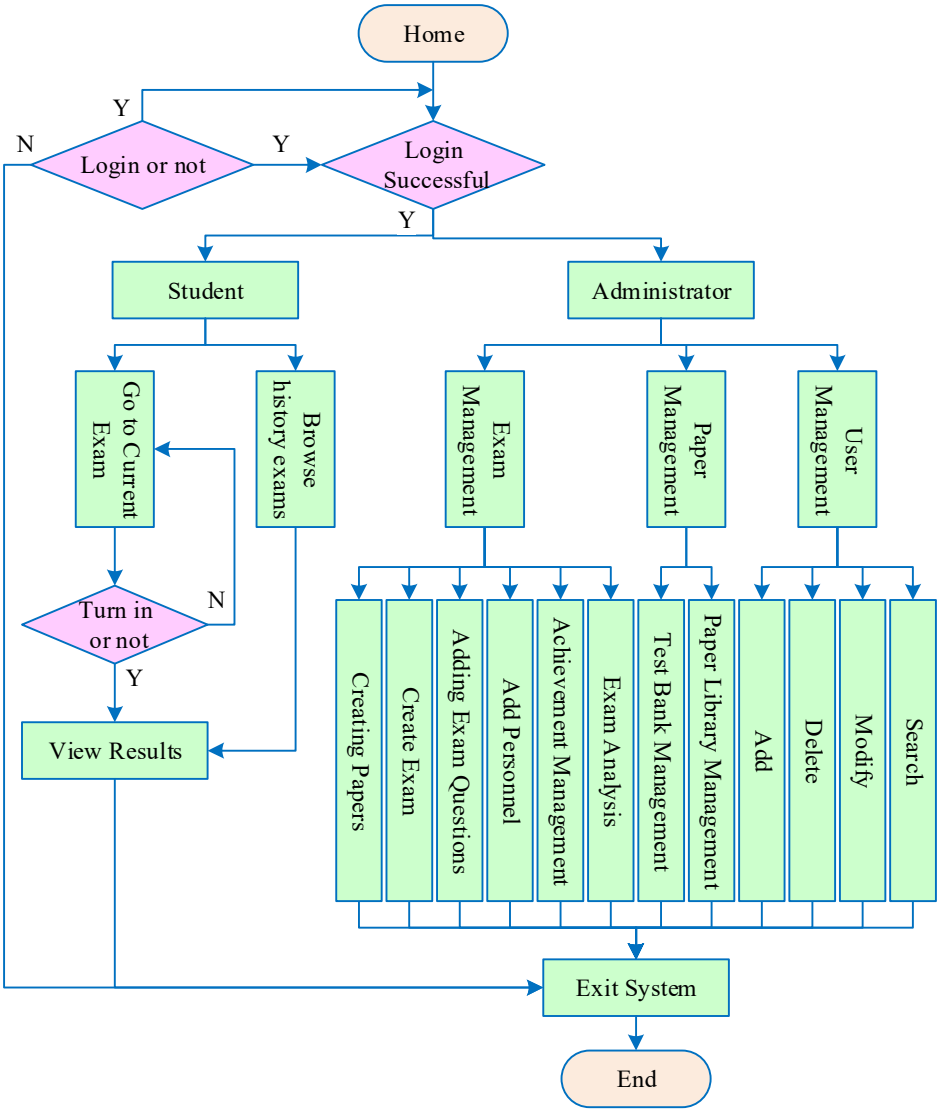

Fig. 3. System chart

Enter the URL to enter the login screen of Ideology and Politics Personalized Test Recommendation Online Exam System. Click the login button to select the management side and student side to login. After successfully logging in, the system will display different functional interfaces according to different roles and permissions.

6. Model simulation verification

6.1. Simulation Validation of Civics Automated Grouping Model

In order to verify that the EGOA-based Civics grouping algorithm proposed in the paper improves the grouping performance, the following experiments are carried out, the setup parameters are shown in Table 1, and the SGA and AGA models are used as a comparison to carry out the programming

simulation, and the maximum adaptability of each algorithm can be obtained under the same conditions as shown in Fig. 4. The EGOA proposed in this paper has a stronger ability to find the optimal and jump out of the local optimal compared with AGA and SGA. The accuracy of EGOA proposed in this paper and the efficiency of paper grouping have achieved a large improvement compared with SGA and AGA, and the total score accuracy of the method in this paper is 100%, which is higher than the remaining two algorithms. At the same time, 20 experiments are conducted to compare the average Civics grouping accuracy, convergence algebra and running time of SGA, AGA and EGOA, respectively, and the comparison results of average Civics grouping accuracy, convergence algebra and running time are shown in Table 2 and Table 3, respectively.

Table 1. Parameter Settings

Parameter name	SGA	AGA	EGOA
Iteration number	1000	1000	1000
Population scale	20	20	20
Pcmax	0.95	0.95	0.95
Pcmin	0.95	0.65	0.65
Pmmax	0.2	0.2	0.2
Pcmin	0.2	0.02	0.02

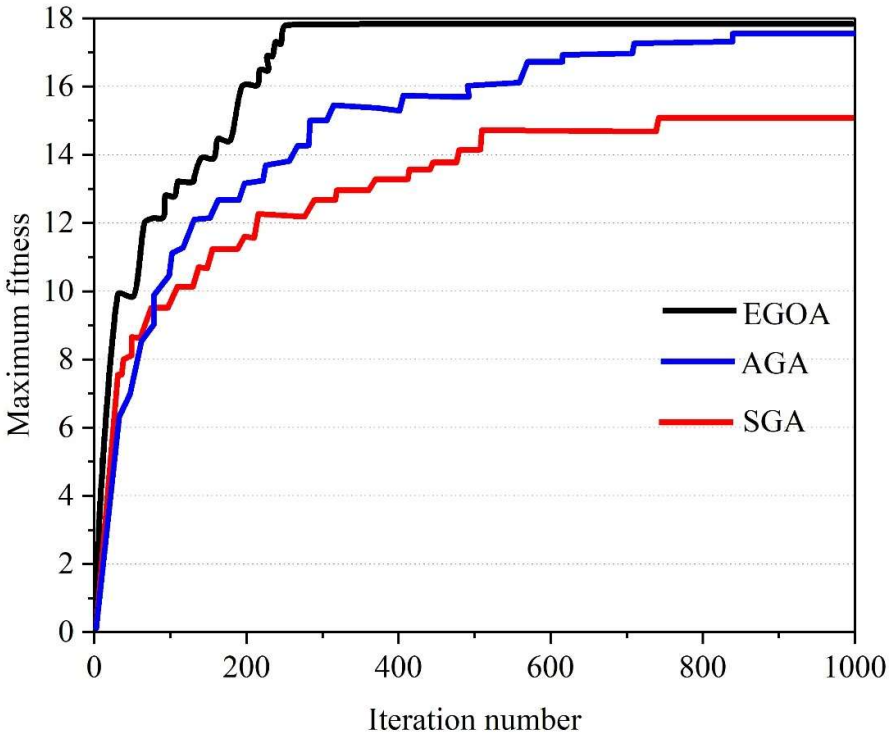

Fig. 4. The maximum fitness ratio of three algorithm species

Table 2. Algorithm group volume accuracy comparison

Parameter name	SGA(%)	AGA(%)	EGOA(%)
Difficulty	90	92	100
Chapters	94	95	100
Exposure time	93	97	100
Type	98	98	100
Total score	93.75	95.5	100

Table 3. The convergence algebra and algorithm time of the three algorithms

Parameter name	SGA	AGA	EGOA
Algorithm convergence algebra	1500	1100	380
Algorithm time/s	1000	1400	350

6.2. Knowledge tracking model simulation validation

The dataset used in this chapter is MOOCCube, which is a large-scale dataset designed for online Civic Education research. In this study, we extracted a subset of the interaction data of the "Data Structure" MOOC course from MOOCCube, which covers the participation records of 891 students and a total of 15,633 interactions.

Before the experiment started, the original Civics course data were rigorously preprocessed. First, some invalid data were eliminated to improve the quality of analysis, specifically including data points with extremely high error rates (more than 90%) and learning sequences that were too short (less than 4 interaction actions in length). Second, in order to maintain the consistency of the dataset and the standardization of model inputs, student learning records containing more than 10 interactions were reasonably cropped, and a special Token-filling strategy was adopted for data in batches of different lengths to ensure that all batches could be inputted into the model in a uniform format for training and analysis.

In order to verify the effectiveness of the knowledge tracking model proposed in this study, the HTNKT model was compared with the following knowledge tracking models:

Deep Knowledge Tracking (DKT) is the first model that applies RNN and LSTM to knowledge tracking. In this paper, LSTM is used to model the changes in the learner's knowledge state.

Self-Attentive Knowledge Tracking (SAKT) introduces a self-attentive mechanism in the knowledge tracking task and uses Transformer to capture the relationship between learners' previous learning interactions over time.

Knowledge tracking using Dynamic Key-Value Memory Networks (DKVMN) borrows the idea of memory networks to capture interpretable learner knowledge states. The forgetting mechanism is considered while updating the learner's knowledge state.

Contextual Attention Based Knowledge Tracking (AKT) employs two self-attention encoders to learn contextualized contextual representations of Civics exercises and answers, and combines self-attention and monotonic attention mechanisms to capture prolonged information. In addition, AKT generates embeddings of Civics exercises based on the Rasch model.

Learning Process Consistent Knowledge Tracking (LPKT) records in detail the changes in learners' knowledge states after each interaction and fully takes into account the dynamic effects of learners' processes of learning new knowledge and forgetting old knowledge.

In this study, we quantitatively analyze the model performance using a dichotomous evaluation framework. Under this framework, the prediction results are categorized into positive and negative categories, and four key indicators-TP, TN, FP, and FN-are used to comprehensively reflect the prediction accuracy and error of the model.

In general, the better the model's performance is evaluated, the higher the values of TP (true case) and TN (true negative case), and the lower the values of FP (false positive case) and FN (false negative case). In this study, for the evaluation of the knowledge tracking model, we analyzed the answer results by transforming them into a binary state: 0 represents the samples that answered incorrectly (negative samples) and 1 represents the samples that answered correctly (positive samples).

In order to fully evaluate the performance of the model, we used two classic classification evaluation metrics-accuracy (ACC) and area under the ROC curve (AUC). Among them, ACC is a measure of the

ratio of the number of correctly categorized samples to the total number of samples, which is calculated by the following formula:

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{45}$$

AUC, the area under the ROC curve, is a comprehensive indicator of the performance of a binary classification model. The size of its value is directly related to the degree of model performance, which is manifested in the fact that the larger the AUC value is, the better the model performance is. When AUC is equal to 0.5, it means that the prediction effect of the model is not better than random guessing. And when the AUC reaches the maximum value of 1, it means that the model shows the best classification ability and can perfectly distinguish between positive and negative samples. On the contrary, if the AUC is 0, it means that the model is completely unable to accurately distinguish between the two types of samples and is in the worst performance state. Therefore, a higher AUC score means that the model has better performance and stability.

The data presented in Table 4 are the results of the comparative analysis of the model's effectiveness, which verifies the superiority of the model proposed in this study in predicting and assessing students' learning status and progress. Through the comparative analysis of the two key performance indicators, AUC and ACC, it can be concluded that compared with the traditional methods that rely only on sequence data or the methods that simulate the learning process alone, this study comprehensively takes into account the time-series characteristics of the students' learning process and their intrinsic learning dynamics, and it makes progress in both evaluation indicators, and it improves by 4.65% in the ACC compared with the DKT. 4.65%.

Table 4. Model effect comparison analysis

Model	AUC/%	ACC/%
DKT	72.59	73.09
DKVMN	72.65	74.05
SAKT	72.59	75.12
AKT	74.58	74.56
LPKT	75.49	73.47
HTNKT	76.88	76.49

A learning sequence of the same learner of length 10 is selected for visualization. The visualization results are shown in Figure 5.

The HTNKT model tracks the changes in the knowledge state of the same learner during the learning process.The HTNKT model predicts the changes in the knowledge state of the learner during the learning process for Knowledge Point 1, Knowledge Point 2, and Knowledge Point 3 and visualizes them. Below each sequence number indicates the knowledge point used in the learner's answer and the correctness or incorrectness of the answer. The darker the color means that the learner has mastered the knowledge point better.

At the beginning, the learner's knowledge status at each knowledge point is set to 0 and is updated through learning. It can be seen that if the student answers correctly, the knowledge status corresponding to the knowledge point increases accordingly. For example, when the learner answers correctly with knowledge point 1, the knowledge status shown in the first row increases accordingly. During the period of exposure to Knowledge Point 3 from the beginning of the 3rd learning sequence until the second learning of the Knowledge Point in the 9th sequence, the HTNKT model shows a decreasing trend in the learner's mastery of Knowledge Point 3. This phenomenon reflects that the model captures the pattern of knowledge forgetting that may occur during the learner's learning process, thus realistically modeling and tracking the dynamic changes in the learner's knowledge mastery. In addition, this learner learns a knowledge point in the current step, but the knowledge

status of this knowledge point changes at other times besides the current step, mainly because of the potential connection between different knowledge points, which may affect the updating of each other's mastery during the learning process. Eventually, we can get the learner's final knowledge state for each knowledge point, and we can see that the student's knowledge state has improved to different degrees compared to the beginning.

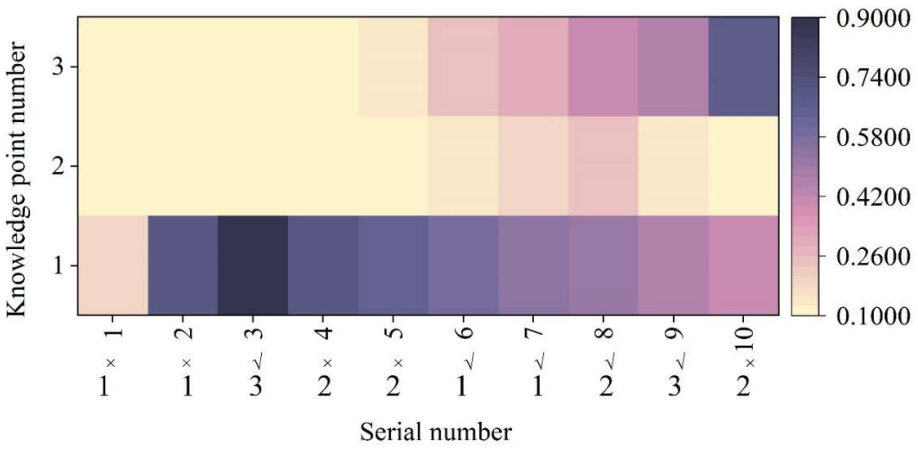

Fig. 5. Learner knowledge state visualization

6.3. Simulation Experiments of Civics Test Recommendation Algorithm

Evaluate the top k ranked Civic Exercise predictions for the user using the following widely used evaluation metrics, where k is set to 5, 10, and 20. Calculate all metrics for each group of 100 concepts in the test set (one of which is interactive and 99 of which are unknown). For each interactive Civics exercise related to user u, a corresponding recommendation list is generated.

In order to demonstrate the effectiveness and robustness of the NGCF method in this paper, it is compared with methods such as KCFRE, KGC and the following baseline methods:

MF: This method uses a matrix decomposition model to consider bias information from students and Civics exercises.

User-CF: It uses user-based collaborative filtering recommendation to calculate the similarity between students, and then recommends undone Civics exercises to students based on the answer records of similar students.

DMF: This model considers explicit interaction and non-preferred implicit feedback as a deep matrix decomposition model, embedding mapping students/Civics exercises.

NCF: Only the Civics exercises that the students did right are used for GMF and MLP interactions.

ACKRec: extracts the user (concept) representation from the same set of meta-paths. However, ACKRec treats this problem as a rating prediction task, where a user's rating of a concept is the number of interactions between the user and the concept. Furthermore, it utilizes user and concept representations as features while extending the matrix decomposition framework.

Table 5 demonstrates the comparison results, for NGCF the HR and NDCG are improved by 41.15% and 65.54% over User-CF, respectively, which indicates that the model in this paper obtains better performance on sparse datasets. Compared to the traditional user-based collaborative filtering recommendation algorithm User-CF, this model uses neural networks to learn user features, in addition the latter four mainstream learning resource recommendation models outperform User-CF overall, also showing the excellence of neural networks. On dense datasets, MF-based methods usually perform better because the dataset contains rich user interaction information, and neighborhood information instead introduces redundant information. However, on sparse datasets, there is a lack of enough interaction information to characterize user preferences, and thus MF-based methods cannot

solve this problem. Instead, the GCN-based approach improves the performance of the model by supplementing the neighborhood information.

Table 5. Experimental comparison

Metric	HR@5	NDCG@5
MF	0.512	0.389
User-CF	0.516	0.399
DMF	0.604	0.485
NCF	0.615	0.495
KCFRE	0.638	0.526
ACKRec	0.645	0.568
KGC	0.709	0.622
NGCF	0.758	0.659

Fig. 6 shows the performance of the proposed method in HR and NDCG at different TOP n. Figure 7 represents the performance of the proposed method with the state-of-the-art method KGC in different HR, NDCG, and MRR, and it can be seen that the present method basically outperforms the KGC method in the metric of MRR, which is improved by 27.05% in this paper's method compared to KGC. These results show that the proposed method can achieve competitive performance in TOP n Civics test recommendation assessment metrics compared to the baseline and state-of-the-art methods.

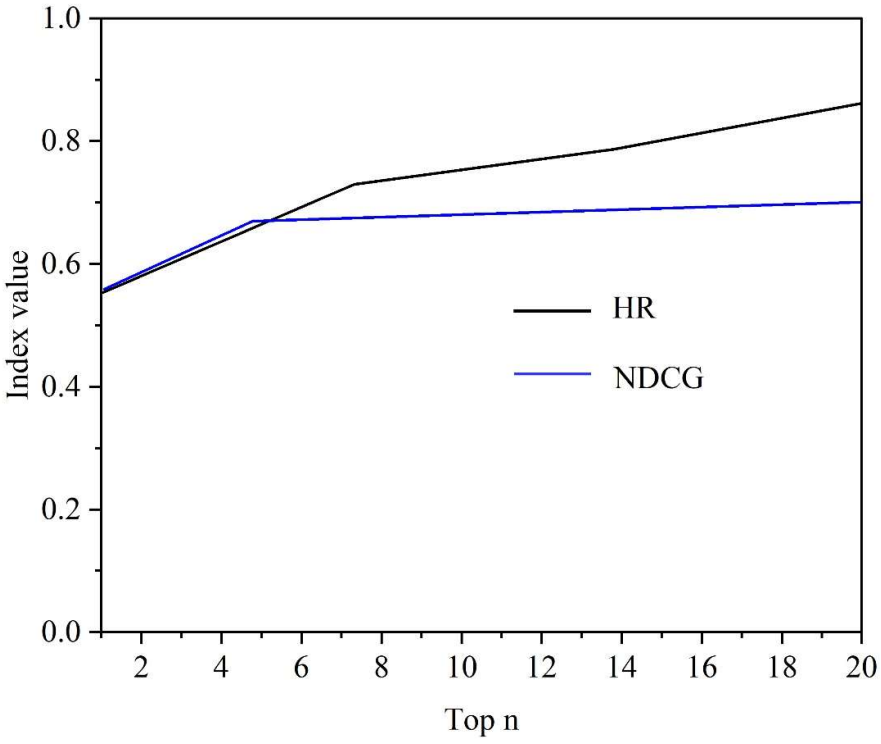

Fig. 6. recommended rendering of different top

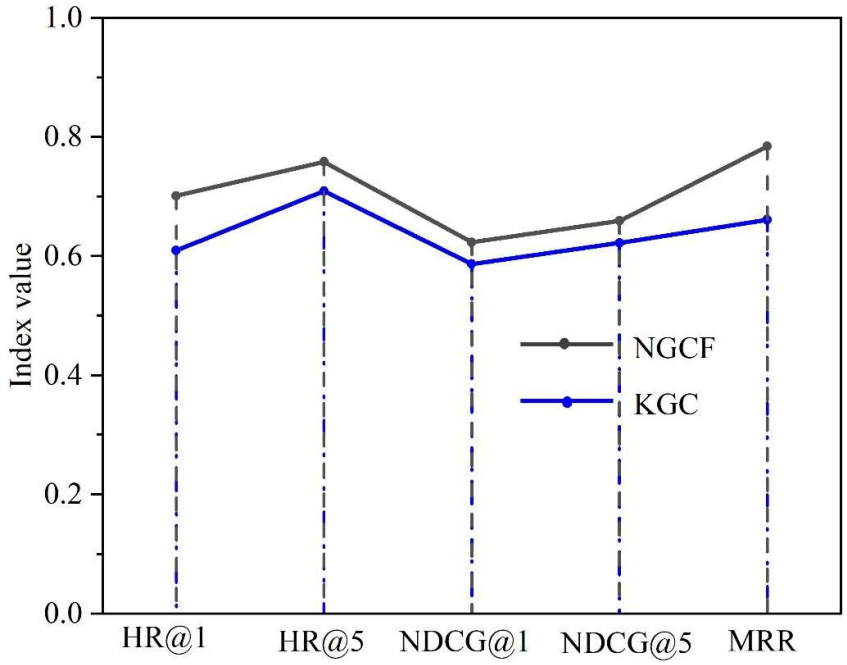

Fig. 7. This method is compared with the students' cognitive module

Figure 8 represents the attention weights of 100 randomly selected learned users with different meta-paths, observing different weights for each user's meta-path. In this heatmap, darker colored cells indicate higher attention weights. Attention can indeed improve performance, and different meta-paths have different importance for deriving user (concept) representations. This can also be further verified by investigating the learned attention weights of different metapaths. For example, Figure 8 shows a heat map of learned attention weights for 100 randomly selected users. In the graph, the X-axis represents the 100 users and the Y-axis represents the attention weights of the 2 different meta-path users. From the graph, it can be noticed that the overall weight of the second one (user one concept-user) is higher than the first one (user one Civics exercise one user), which indicates that in the process of Civics exercise recommendation, the knowledge concepts are the core of Civics exercises, and the students' understanding of the knowledge concepts can only help them to learn better.

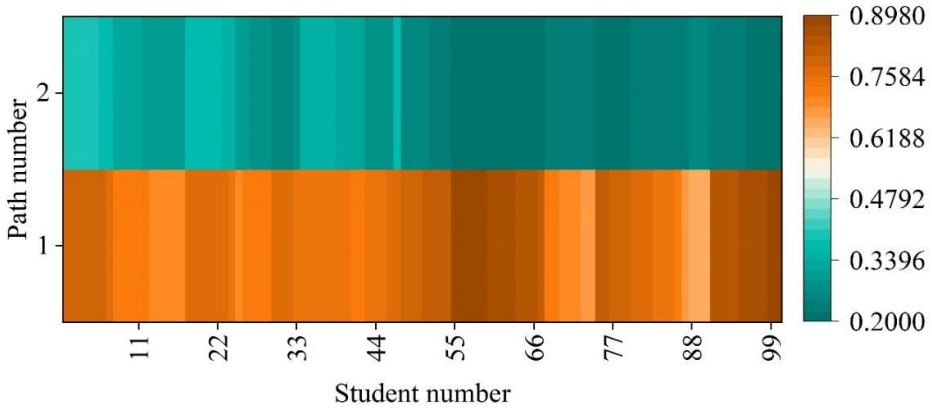

Fig. 8. Two element path heat maps

7. Conclusion

This paper uses the improved locust algorithm to construct an intelligent grouping strategy for Civics, and compares the performance advantages and disadvantages of this paper's intelligent Civics

grouping strategy with SGA and AGA strategies through simulation experiments. The experimental results show that the improved locust algorithm of this paper is better than the comparison algorithm in terms of paper formation accuracy, convergence algebra and running time, and the total score accuracy of this paper's algorithm reaches 100%.

Subsequently combined with the knowledge tracking model to further portray the mastery of students' Civics knowledge, this paper's hypergraphic self-attentive knowledge tracking model has obvious advantages in both evaluation indexes, which improves students' knowledge status. However, the dataset of this experiment only uses the dataset in the field of ideological and political education, and it is hoped that the model can carry out more experiments on datasets in different fields.

Finally, in order to realize the personalized matching between test questions and individuals, the Civics test question recommendation model based on the neural graph model is constructed, and the performance of this paper's recommendation algorithm about HR, NDCG, and MRR indexes is better than the existing excellent recommendation algorithms, and the overall weighting of the attention weights of the users of the learning paths based on this paper's algorithms is higher, which are above 0.5, which shows that this paper's recommendation model in the Civics exercise question recommendation process can help the students to deeply understand the knowledge of Civics and Politics, and help students improve their learning efficiency.

References

[1] Englund, C., Olofsson, A. D., & Price, L. (2017). Teaching with technology in higher education: understanding conceptual change and development in practice. Higher Education Research & Development, 36(1), 73-87.

[2] Lyapina, I., Sotnikova, E., Lebedeva, O., Makarova, T., & Skvortsova, N. (2019). Smart technologies: perspectives of usage in higher education. International journal of educational management, 33(3), 454-461.

[3] Ahmad, K., Iqbal, W., El-Hassan, A., Qadir, J., Benhaddou, D., Ayyash, M., & Al-Fuqaha, A. (2023). Data-driven artificial intelligence in education: A comprehensive review. IEEE Transactions on Learning Technologies.

[4] Park, V., & Datnow, A. (2017). Ability grouping and differentiated instruction in an era of data-driven decision making. American Journal of Education, 123(2), 000-000.

[5] Huang, R., Tlili, A., Chang, T. W., Zhang, X., Nascimbeni, F., & Burgos, D. (2020). Disrupted classes, undisrupted learning during COVID-19 outbreak in China: application of open educational practices and resources. Smart Learning Environments, 7, 1-15.

[6] Sorokova, M. G. (2020). E-course as blended learning digital educational resource in university. Psikhologicheskaya nauka i obrazovanie, 25(1), 36-50.

[7] Grecu, Y. V. (2022). Overcoming Obstacles to Differentiate Instruction When Implementing Prepared Curricular Resources in a Diverse Classroom. Anatolian Journal of Education, 7(1), 167-180.

[8] Wang, Q., Yu, S., & Wang, X. (2024). Research on the Role of an Automatic Resource Generation System to Promote Chinese as a Second Language Learners' Learning in Colleges. Journal of Educational Computing Research, 62(2), 501-531.

[9] Schroeder, K. T., Hubertz, M., Van Campenhout, R., & Johnson, B. G. (2022). Teaching and Learning with AI-Generated Courseware: Lessons from the Classroom. Online Learning, 26(3), 73-87.

[10] Kurdi, G., Leo, J., Parsia, B., Sattler, U., & Al-Emari, S. (2020). A systematic review of automatic question generation for educational purposes. International Journal of Artificial Intelligence in Education, 30, 121-204.

[11] Ch, D. R., & Saha, S. K. (2018). Automatic multiple choice question generation from text: A survey. IEEE

Transactions on Learning Technologies, 13(1), 14-25.

[12] Duan, X. (2019). Automatic generation and evolution of personalized curriculum based on genetic algorithm. International Journal of Emerging Technologies in Learning (Online), 14(12), 15.

[13] Afzaal, M., Nouri, J., & Aayesha, A. (2023, August). A Transformer-Based Approach for the Automatic Generation of Concept-Wise Exercises to Provide Personalized Learning Support to Students. In European Conference on Technology Enhanced Learning (pp. 16-31). Cham: Springer Nature Switzerland.

[14] Li, Y., Shao, Z., Wang, X., Zhao, X., & Guo, Y. (2018). A concept map-based learning paths automatic generation algorithm for adaptive learning systems. IEEE Access, 7, 245-255.

[15] Zhang, W. (2023). Opportunities and Challenges of Ideological and Political Education for College Students in the Internet Era. Applied Mathematics and Nonlinear Sciences, 9(1).

[16] Baofeng Sun,Gendao Li,Shuai Wang & Cheng Xie. (2025). Two-dimensional bin-packing problem with conflicts and load balancing: A hybrid chaotic and evolutionary particle swarm optimization algorithm. Computers & Industrial Engineering110851-110851.

[17] Zhang Pei,Hu Ping,Li Gang,Lv Yingying & Wan Meng. (2024). DHKFN: Knowledge Tracking Model Based on Deep Hierarchical Knowledge Fusion Networks. International Journal of Knowledge Management (IJKM)(1),1-17.

[18] Sheng Jinfang,Liu Qingqing,Hou Zhengang & Wang Bin. (2023). A Collaborative Filtering Recommendation Algorithm Based on Community Detection and Graph Neural Network. Neural Processing Letters(6),7095-7112.

[19] Anushka Joshi,Pradeep Singh & Balasubramanian Raman. (2025). IsoMapGen: Framework for early prediction of peak ground acceleration using tripartite feature extraction and gated attention model. Computers and Geosciences105849-105849.