International Journal for Housing Science and Its Applications
Publish August 5, 2025. Volume 46, Issue 3 Pages 3854-3865

https://doi.org/10.70517/ijhsa463327

A Comprehensive Study of Artificial Intelligence to Promote Reasoning Thinking Skills in Programming Instruction

Yuanyuan Xiao[1,*], Yingxin Zhang[1] and Jianzhi Sun[1]
[1] School of Computer and Artificial Intelligence, Beijing Technology and Business University, Beijing, 100048, China
Corresponding authors: (e-mail: yyxiao24@126.com).

Abstract With the surge of the wave of artificial intelligence, the innovation and practice of the teaching mode of programming courses in colleges and universities can improve the quality of teaching. Using artificial intelligence technology, this paper proposes four paths, including optimization of teaching resources, strengthening practical innovation ability, enhancing practical teaching, and constructing a diversified reasoning thinking evaluation system. Through the learning resource network, it presents the trajectory of students' use of programming course resources, accurately obtains the three key attributes of students' use of resources: content, type, and frequency, accurately portrays students' learning behaviors, and reflects their reasoning thinking ability in the learning process. Based on students' learning behavior data, the cognitive layer is modeled to assess students' reasoning thinking ability. The controlled experiment shows that the mean value of the students' reasoning thinking scores in the experimental class increased from 65.616 to 73.379, an increase of 7.763 points, and the overall progress of the reasoning thinking ability in the experimental class is relatively large. Meanwhile, the t-test results of the two classes show that the P-values of cooperation ability, problem solving ability and critical thinking ability are 0.001, 0.002 and 0.043 respectively, which are less than 0.05, and there is a significant difference between the two classes.

Index Terms artificial intelligence technology, learning resource network, learning behavior portrayal, reasoning thinking ability

I. Introduction

Programming courses are core courses in computer science and its related disciplines, aiming to cultivate students' computational thinking, enhance their employability, and support interdisciplinary applications [1]. However, the traditional teacher-centered teaching methods, although having certain advantages in knowledge transfer and course management, tend to lead to students being in a passive learning state and lacking opportunities for active exploration and practice [2]-[4]. Especially in programming courses involving complex logic and abstract concepts, it is difficult to effectively improve students' logical reasoning ability if it is simply explained, and the lack of a bridge for transition can easily lead to poor learning results [5]-[7]. If we can let students improve their logical reasoning ability in the process of learning programming knowledge, it will be helpful for students to improve their problem solving ability in other subjects and even in daily life [8], [9].

Artificial Intelligence, as an important driving force leading the new round of technological revolution and industrial change, has become a powerful boosting technology for economic development and social progress, bringing more possibilities for the modernization of education. The original intention of integrating AI technology into programming teaching is not only to cultivate students' programming ability, but also to recognize important mathematical and computational aspects in students' creation and learning process [10], [11]. At the same time, it enhances students' access to creative thinking, logical reasoning, and collaborative work with the assistance of AI technology. This is very similar to the pedagogical objectives of the Algorithms and Programming module in Information Technology classes [12]-[15]. Therefore, AI can be used as an auxiliary tool for students' programming learning so as to motivate them to learn and develop their ability to utilize logical reasoning way of thinking to solve real-world problems [16], [17].

In this paper, from the four aspects of optimization of teaching resources, strengthening the practical innovation ability, enhancing practical teaching, and establishing a diversified evaluation system, artificial intelligence technology is reasonably used to reform the teaching mode of program design. The three parts, namely, Lian Li learning network, recommendation algorithm and personalized learning resources, constitute the personalized recommendation model, which generates the final recommended learning resources for the learning objectives and students' personalized needs. Through the learning resource usage trajectory, reflecting the three key

attributes of students' content, type, and frequency of resource usage, we determine the learning resources to be recommended to students. Diagnose the reasoning thinking ability of the student population based on student learning behavior data. Embed the recommendation algorithm into the open-source implementation of program design teaching, and construct controlled experiments and pre- and post-tests to provide feedback on the effect of students' reasoning thinking ability improvement in AI-based program design teaching.

II. Programming teaching mode reform strategy under artificial intelligence

II. A.Resource Optimization and Personalized Learning Based on Intelligent Teaching and Learning

The construction of an intelligent teaching mode is crucial to improving the teaching quality and effect of programming courses. With the help of artificial intelligence, we can efficiently integrate rich and diverse high-quality teaching resources to create a learning environment that is both personalized and dynamically adaptable to meet the unique learning needs and progress of different students.

II. A. 1) Analyze student learning data

In-depth analysis of students' learning data. The platform is able to create personalized learning paths for students, intelligently recommending the most suitable learning content and resources, such as professional tutorials, programming exercises, and online proficiency assessment, based on students' learning foundation, current progress, and personal interests [18]. As a result, students can flexibly choose learning materials according to their own needs, and are no longer restricted by the traditional “one-size-fits-all” teaching materials.

II. A. 2) Monitoring student progress and achievement

With the platform's powerful data analysis function, real-time monitoring of teaching progress, student performance, classroom interaction and homework submission and other key indicators. It helps students clearly grasp their own learning status and provides teachers with accurate feedback. Teachers should have real-time insight into students' knowledge mastery, problems and learning bottlenecks, so as to flexibly adjust the teaching content and methodology, and provide students with more targeted counseling and support.

II. A. 3) Efficient integration of teaching resources

Efficiently integrates numerous high-quality teaching resources, such as open-source programming tools, online programming environments, exciting teaching videos and classic cases. Students can access these resources anytime and anywhere, which in turn dramatically improves learning efficiency, optimizes the allocation of teaching resources, and ensures the maximum use of educational resources.

II. B.Implement project-driven teaching to strengthen practical and innovative abilities

By designing project tasks that are closely related to reality, students are prompted to apply theoretical knowledge to practice, which in turn improves their problem-solving ability and deeply explores their innovative potential. With the help of artificial intelligence, the relevance and effect of project-driven teaching has been significantly improved. Practical activities not only let students deeply realize the wide range of applications of programming skills, but also inadvertently cultivate their critical thinking ability when dealing with complex problems.

To further enhance the effectiveness of the project-driven teaching method, multi-level project tasks are designed for different students' abilities and interests. For students with a weak foundation, they can start with simple projects and gradually increase the difficulty, while for students with a better foundation, they can challenge more complex projects. In teamwork, the roles and responsibilities of each member are clarified to ensure that students can utilize their expertise in the project. Enhancing teamwork not only improves students' communication skills, but also develops their teamwork spirit.

II. C.Integrate Artificial Intelligence and Enhance Practical Teaching Links

Artificial intelligence can not only optimize the teaching process, but also significantly improve the effectiveness of practical teaching. In programming courses, practical teaching is especially critical, allowing students to refine their practical skills through hands-on participation in programming tasks, and then cultivate excellent problem-solving skills.

The combination of artificial intelligence and big data analysis technology explores students' learning paths and programming habits to provide more personalized feedback and improvement measures. In the process of practical teaching, AI monitors the students' practical process in real time, which enables teachers to grasp the students' practical status and learning progress, and flexibly adjust the teaching strategy and tutoring focus according to the data generated by the students in the process of practical teaching.

II. D.Establish a diversified evaluation system to reflect students' reasoning thinking ability

With the help of artificial intelligence, it is possible to create a new evaluation framework that integrates process evaluation and outcome evaluation [19]. This framework will collect and analyze data from students throughout their learning process to deeply assess their performance in multiple dimensions such as programming practice, problem solving, and creative thinking. Using AI, programming assignments, project progress reports, and code quality submitted by students can be exhaustively analyzed, thus providing real-time data support to instructors so that they can accurately grasp the learning dynamics and technical mastery of students. Artificial intelligence can also help realize formative assessment, which provides timely feedback and points out the direction of improvement based on students' real-time performance in course learning. This type of evaluation focuses on the entire growth trajectory of students, thus mapping out their competency level more realistically. In the perspective of formative assessment, it not only focuses on the students' academic performance, but also their comprehensive quality such as practical ability, innovative thinking and teamwork ability.

With the help of artificial intelligence, teachers will be able to formulate personalized guidance plans for students based on rich learning data and accurate evaluation results. This highly personalized evaluation and feedback mechanism will help students steadily improve, and ultimately achieve a fundamental shift in the learning model, from passive knowledge receivers to active learners and explorers.

III. Artificial intelligence-based personalized learning and diagnosis of reasoning thinking skills

III. A. Personalized Recommendation Mode

The personalized recommendation model consists of three parts: learning network, recommendation algorithm, and personalized learning resources, as shown in Figure 1 [20]. Among them, the learning network is the basis of the recommendation model, consisting of two network types: learning resources and learning behavior. The learning resource network uses data mining technology to deeply analyze the trajectory of resource usage as the basis for personalized recommendation. Learning behavior network can accurately portray student characteristics, including cognitive ability, learning preferences, programming knowledge mastery, etc., so as to recommend resources that meet personalized learning needs [21]. The resource recommendation algorithm is the core of the recommendation model, which takes the recommendation completeness rate and recommendation accuracy rate as the evaluation criteria, and generates the final recommended learning resources according to the learning objectives and students' personalized needs.

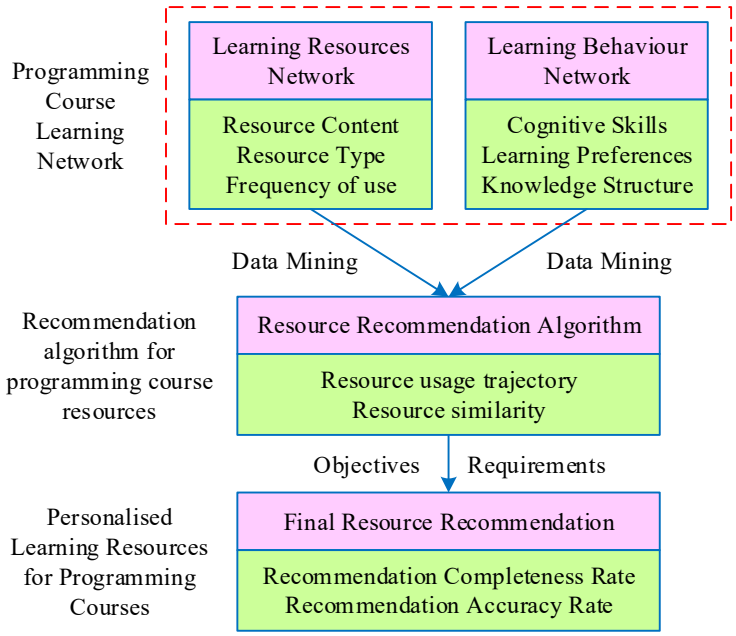

Figure 1: The program design course learning resource personalized recommendation

III. B. Building learning networks

III. B. 1) Learning resource networks

The main role of the learning resource network is to accurately present the learning resource usage trajectory, accurately reflecting the 3 key attributes of students' usage of the resources: content, type and frequency. In the learning resource network, the resource usage trajectory is denoted by R_{ij}^{n}, the resource nodes are denoted by V_c^t, and the edges between the nodes are denoted by E^f, see Equation (1). Where the parameter ij denotes the position of each resource usage trajectory in the matrix, and the parameter n denotes the number of trajectories, and when n is zero it means that no resource has been used:

$$R_{ij}^{n} = \left(V_c^t, E^f\right) \tag{1}$$

III. B. 2) Learning behavior networks

The main role of the learning behavior network is to accurately portray students' learning behaviors, accurately reflecting their cognitive ability, learning preferences and knowledge structure, 3 key features. In the learning behavior network, learning behavior is the process of students using learning resources to accomplish learning goals, denoted by L_{ij}^{n}, and students are nodes in the network, denoted by V_k^s. The edges between the nodes are the degree of preference of the students in using the resources, denoted by E^m, see Equation (2). Where the parameter ij denotes the position of a particular learning behavior in the matrix and the parameter n denotes the number of trajectories, when n is zero it means that there is no any learning behavior:

$$L_{ij}^{n} = \left(V_k^s, E^m\right) \tag{2}$$

III. C. Recommendation algorithms

On the basis of learning resource network and learning behavior network, learning resource usage statistics formula and learning behavior similarity calculation formula are used to design learning resource personalized recommendation algorithm, and the flow chart is shown in Figure 2.

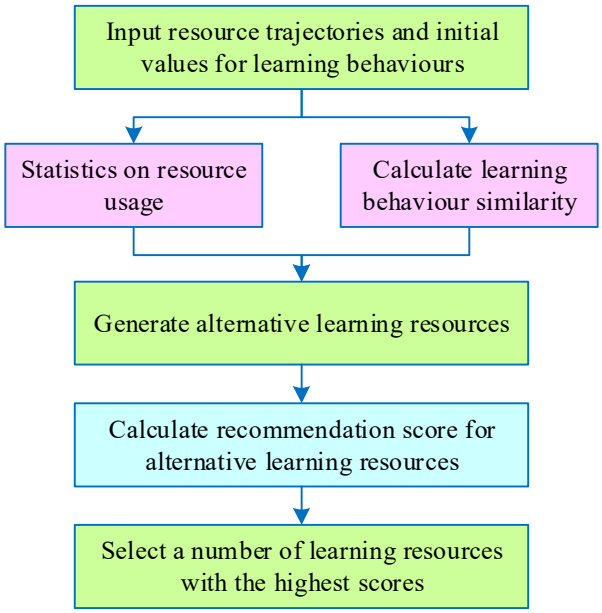

Figure 2: Flowchart of learning resource recommendation algorithm

III. C. 1) Algorithm Initial Values

Using the calculation formula (1) in the learning resource network to count the number of learning resource usage trajectories, and using the calculation formula (2) in the learning behavior network to count the number of times the same learning resource has been used by different students, the value of the results of these two calculations is used as the initial value of the resource recommendation algorithm.

III. C. 2) Resource use and similarity of learning behaviors

Resource usage is described in terms of the degree of similarity of resource usage trajectories in the learning resource network, as defined in Equation (3). Where $S_R\left(p_i, p_j, N\right)$ is used to characterize the learning resource usage trajectory, p_i and p_j are two of the resource usage sequences, and N is the number of sequences:

$$S_R\left(p_i, p_j, N\right)=\begin{cases}\dfrac{2N}{N_i+N_j} & N_a \geq N \\ 0 & N_a < N\end{cases} \tag{3}$$

The degree of learning behavior similarity is described in terms of students' preference for the use of resources, which is defined in Equation (4). Where $S_P(L_i, L_j)$ denotes the degree of similarity in preferences for learning resources, and the parameters L_i and L_j represent the i th and j th preferences for using learning resources in the learning behavior network, respectively. The parameters $S_1(L_i, L_j)$ and $S_1(L_i, L_j)$ denote the degree of similarity of resource use preferences based on students' cognitive ability and knowledge structure, respectively, and ω is a computational coefficient with a range of 0~1, which serves to regulate the weight of the two degrees of similarity:

$$S_P(L_i, L_j)=\omega \times S_1(L_i, L_j)+(1-\omega) \times S_1(L_i, L_j) \tag{4}$$

III. C. 3) Generating alternative learning resources

Based on the statistical results of the degree of similarity between resource usage and learning behaviors, a number of resource usage trajectories and a number of resource usage preferences with the highest degree of similarity are filtered out, which are computed in Equation (5). Where $E_i(N_R, N_P)$ is the ensemble of recommended alternative resources, which is presented in the form of a two-dimensional matrix C_i^R. N_R and N_P are the similarity values of resource usage trajectories and resource usage preferences, and δ is a computational coefficient used to adjust the weights of the two degrees of similarity. μ_R denotes the number of errors that there may be learning resources that do not meet the recommendation criteria but are recommended, and μ_P denotes the number of errors that there may be learning resources that meet the criteria but are not recommended:

$$E_i(N_R, N_P)=C_i^R=\delta \times(N_R-\mu_R) \times(N_P-\mu_P) \tag{5}$$

III. C. 4) Calculating alternative learning resource scores

After determining the recommended alternative learning resources, each resource must be scored, and the scoring index includes two aspects of resource importance and learning needs, resource importance is described by the resource's utilization ranking, and learning needs are described by the degree of preference for the use of the resource, and the calculation method is shown in Equation (6). Where $S_c(R)$ represents the final score of an alternative learning resource, $D_i(R)$ represents the value of the utilization ranking of the i th alternative resource in the learning resource network, and S_R represents the value of the similarity degree of resource preference, and the result of the calculation of the multiplication of the two is the final score of the alternative learning resource:

$$S_c(R)=D_i(R) \times S_R \tag{6}$$

III. C. 5) Identification of final recommended learning resources

After the final score of each alternative resource is calculated, it is sorted in descending order to determine the final recommended learning resources, see Equation (7). Where $F_k(R)$ denotes the set of final recommended learning resources, $Q_c(R)$ denotes the ensemble of sorted alternative learning resources, and μ is a calculation coefficient with a value range of 0~1, which serves to determine the number of final recommended learning resources:

$$F_k(R)=\mu \times Q_c(R) \tag{7}$$

III. D. Diagnosis of group reasoning thinking skills based on multi-task learning

This chapter is the core research of this paper, which investigates the group-level assessment method of reasoning thinking ability, i.e., diagnosing the knowledge status and ability level of the group, based on the learning behavior data of the student group.

III. D. 1) Multi-task learning

In order to effectively utilize the information in the student-practice response data to mitigate the sparsity of the group-practice response data, this study designed a multi-task learning framework to jointly model these two types of interaction data, as shown in Figure 3.

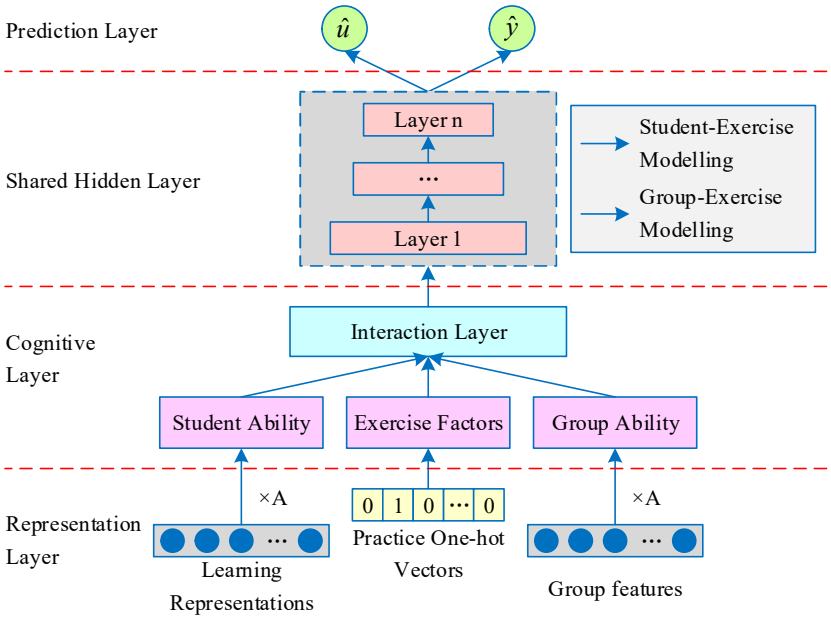

Figure 3: Joint modeling students - Practice response and groups - Practice response

III. D. 2) Group Representation Learning

Since a group is composed of its students, there is a strong correlation between group representations and student representations, and this study naturally obtains group representations from the representations of the students it contains, an advantage of this approach is that it facilitates the transfer of information between the two performance prediction tasks, leading to better diagnostic results. In this study, each student and group is identified with a unique id, and all are assigned a one-hot vector to go as model input in order to facilitate training in the neural network.

For student representations, the one-hot vector x_{s_j} for each student is directly projected into the hidden space using the matrix R to obtain the embedding vector r_{s_j}, which is the representation vector associated with that student:

$$r_{s_j} = x_{s_j} \times R \tag{8}$$

where R is available a trainable matrix. Next, this study aggregates the student embedding in a population to obtain the embedding of the population, i.e., the representation of the population. Before describing the specific aggregation methods, the following will first outline some common aggregation strategies.

There are several predefined strategies to aggregate embedding in neural networks, such as maximum pooling, average pooling and minimum pooling. Typically, these aggregation strategies, also known as heuristics, first predict students' proficiency scores on specific knowledge concepts, and then aggregate the predicted scores of each member of the population through these strategies to obtain the proficiency level of the population. These aggregation strategies can be interpreted in terms of the cognitive proficiency level of the group. For example, the maximum pooling strategy attempts to maximize the cognitive ability of the team by selecting the highest proficiency scores on each knowledge concept among its members.

However, these heuristic-based aggregation strategies lack flexibility in adjusting the weights of the members in the group, and thus are insufficient to model the relationship between the students' knowledge state and the group's knowledge state. It is worth noting that student roles in groups are highly correlated with student characteristics (e.g., knowledge level), and the importance may vary depending on the characteristics of the group context (e.g., collaboration). For example, due to the diversity of learning contexts, two students with similar abilities in different groups may have different influence in their respective groups. To this end, an adaptive weighted sum operation, inspired by attentional mechanisms in neural networks, is designed. The use of r_{g_i} to characterize the group g_i is obtained through Equation (9):

$$r_{g_i} = \sum_{j \in G_i} \lambda_j r_{s_j} \tag{9}$$

where λ_j denotes the influence weight of student s_j. In order to dynamically model students' influence weights in different contexts, this study designs a novel context-aware attention network that learns weight values from historical data of group-practice responses:

$$\begin{aligned} o_j &= h^T \tanh\left(W_K r_{s_j} + W_Q c_i\right) \\ \lambda_j &= soft\max\left(o_j\right) = \frac{\exp(o_j)}{\sum_{j' \in G_i} \exp\left(o_{j'}\right)} \end{aligned} \tag{10}$$

where c_i is the population level context vector for population g_i, which can be obtained by multiplying the one-hot vector of g_i by the trainable matrix W_C, i.e., $c_i = x_{g_i} \times W_C$ times. W_K and W_Q are the key-value matrix and query matrix of the attention network, respectively, which are linearly transformed and projected to the same eigenspace to do the alignment operation for the students' embedding and group context vectors, respectively. In this study, tanh is used as the activation function, and then the weight vector h is used to map the activation value to get the attention score o_j. Finally, the score is normalized using the softmax function, which makes the attention network probabilistically interpretable.

III. D. 3) Cognitive Layer Modeling

The goal of cognitive level modeling is to obtain interpretable student competencies and group competencies and to model complex interactions between students, groups, and exercises. The details are described below.

Group competence. After obtaining a representation of a group, the next goal is to model the group's cognitive ability, which characterizes the group and influences the results of the group's response to practice questions. Specifically, this study uses the cognitive ability vector h_g to characterize a group ability level:

$$h_g = sigmoid\left(r_g \times A\right) \tag{11}$$

where $h_g \in (0,1)^{1 \times K}$ and A is a trainable matrix. Note that the sigmoid activation function is used here to constrain the group ability to be a real value between 0 and 1.

Student ability. The framework proposed in this paper requires co-training of a student performance prediction task and a group performance prediction task, and thus requires modeling the cognitive abilities of the students to predict the outcome of their responses to practice questions. Similarly, a vector h_s is used to represent students' cognitive abilities:

$$h_s = sigmoid\left(r_s \times A\right) \tag{12}$$

where the matrix A is shared in the MGCD framework, which facilitates information transfer in multi-task learning.

Practice factor. For group-level cognitive diagnosis, the practice factor is another important factor to be considered, which represents the characteristics of the practice problems. In this research work, the first practice factor considered is the knowledge concept Q_e related to the practice questions to ensure the interpretability of the designed model at the cognitive level. In the cognitive diagnostic task, the Q matrix serves as the a priori knowledge labeled from pedagogical experts to indicate which knowledge concepts are needed for each exercise, and thus Q_e can be obtained in the following way:

$$Q_e = x_e \times Q \tag{13}$$

where $Q_e \in \{0,1\}^{1\times K}$, and x_e are the one-hot vectors of exercise e. In addition, two other important practice factors are considered in this study: knowledge point difficulty h_{diff} and practice question differentiation h_{disc}, which are widely used in cognitive diagnostic models to help obtain more effective diagnostic results. Where h_{diff} denotes the difficulty of each knowledge concept associated with a given exercise, given by the following equation:

$$h_{diff} = sigmoid\left(x_e \times B\right) \tag{14}$$

where B is a trainable matrix. The h_{disc} is the ability of the practice questions to distinguish groups or students with different levels of proficiency in knowledge, which can be obtained by the following formula:

$$h_{disc} = sigmoid\left(x_e \times D\right) \tag{15}$$

where D is a trainable matrix.

IV. Analysis of the application of artificial intelligence in the teaching of programming

IV. A. Application of Personalized Recommendations in Teaching Programming

In this paper, the recommended algorithms are embedded in an open source programming teaching implementation. The loop control in the course “basic computer programming” as a teaching case, the knowledge points of loop control include: the concept of loop control, while loop, do while loop, for loop, loop nesting, comparison of loops, goto loop, break statement and continue statement. .

Thirty students were divided into a control group and an experimental group of 15 students each, using voluntary enrollment and automatic division into groups when the quota was full. All students participated in 3 hours of study (including learning the first 5 points of loop control, i.e., the first 5 lessons, a total of 60 learning objects). Two groups of students were studied, where the first 2 hours were devoted to the course and the second 1 hour to a mini-examination (the same for both groups, both designed with 5 test questions) to assess what the students had learned during the course. The experimental group was required to answer 3 short questions to provide feedback on the experimental group of students' perceptions of personalized recommendations.

IV. A. 1) Scoring

The two groups of students were taught using different teaching methods. The experimental group used an instructional model that used personalized recommendations to guide students, while the control group used traditional programming instructional methods. They were evaluated in 3 dimensions: effectiveness, efficiency, and attractiveness, respectively. Effectiveness refers to the number of learning objects that students complete correctly in a lesson, efficiency refers to the time spent by students to reach their goals, and attractiveness refers to students' satisfaction with the model.

First, the completion of learning objects and knowledge points of each student was compared. The control group completed more learning objects in the first 2 sessions, while the experimental group completed more knowledge points, as shown in Figure 4. The experimental group completed the recommended learning objects for the knowledge points of program design, and at the same time, the experimental group recommended more knowledge points than the control group, and recommended the knowledge points of program design learning to 15 students, with a mean score of 3.467 for all learning objects.

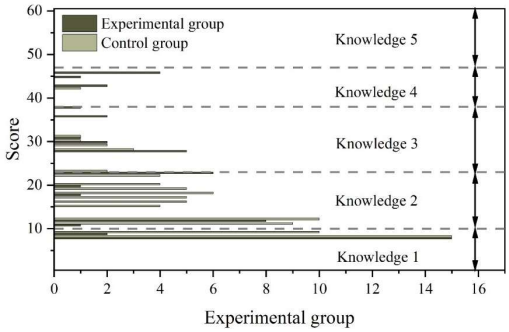

Figure 4: A comparison of the scores of each learning object and knowledge point

IV. A. 2) Learning efficiency

Secondly, in terms of efficiency, the author counted the time spent by students in each group in learning each knowledge point. The time spent by the control group was mainly concentrated in the first 3 lessons, while the experimental group learned 4 lessons in the same time, as shown in Figure 5. As a result, compared with the control group, the experimental group's learning efficiency and effectiveness improved considerably, and the average time spent on the 5 knowledge points was 16,224.2s, which saved 1905.6s compared with the control group.

Finally, a questionnaire star was used to collect the opinions of the students in the experimental group on the recommendations. The questions included 1) whether the recommendations were generated quickly, 2) whether the recommendations were helpful in completing the course, and 3) whether they were satisfied with their final grades considering the course time. 88.456% of the students thought that the personalized recommendations could be generated quickly, indicating that the students in the experimental group were satisfied with the personalized recommendations. The above results indicate that students who used personalized recommendations for learning were much more efficient and effective in their adaptive learning. In the same time, students can learn more knowledge points by using the learning path of personalized recommendation, and students are very satisfied with the recommendation. The recommendation algorithm collects user interaction data in real time, such as the student's learning score and time spent, and avoids wasting students' time while learning by adjusting the recommended path in real time.

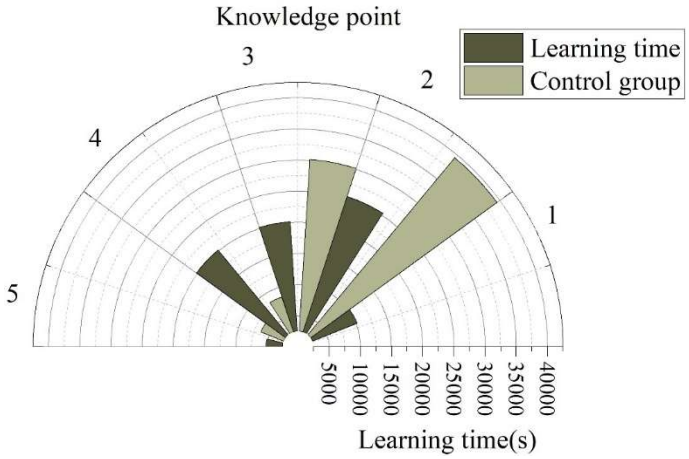

Figure 5: Comparison of the time consumption for learning each knowledge point

IV. B. Evaluation of Reasoned Thinking Skills

IV. B. 1) Analysis of data on the overall level of reasoned thinking skills

In order to investigate whether the teaching model constructed in this study can effectively improve students' computational thinking ability, the computational thinking evaluation scale was used to conduct computational thinking ability pre-tests and post-tests for the experimental class and the control class before the implementation of the teaching experiment and after the end of the teaching experiment, respectively, to measure students' mastery of the basics of algorithms and program design from the five competency elements of computational thinking, and to obtain the learners' real ability level. The collected data are statistically analyzed below.

Firstly, IBM SPSS 24.0 was utilized to test whether the pre-test and post-test data of the experimental and control classes conformed to normal distribution. The results of the analysis are shown in Table 1, which shows that the significance of the pre-test and post-test data of the experimental and control classes are 0.658, 0.248, 0.234 and 0.788, respectively, which are greater than 0.05, indicating that the pre-test and post-test data of the experimental and control classes conformed to normal distribution. The fact that both the pre-test and post-test data conformed to normal distribution indicates that an independent sample test can be conducted on the collected data to test whether there is a significant difference between the experimental and control classes before and after the experiment.

Table 1: Shapiro-Wilk test data for pre- and post-test tests of experimental and control classes

Shapiro-Wilk testing		Pretest		Posttest	
Class		Control group	Experimental group	Control group	Experimental group
Sample size		15	15	15	15
Normal parameter	Mean	64.894	65.518	68.648	74.715
	Standard deviation	14.135	13.325	13.312	11.432
Statistic		0.942	0.964	0.964	0.982
Significance		0.658	0.248	0.234	0.788

Before conducting the teaching, a pre-test was conducted for both classes using the same set of Computational Thinking Evaluation Scale, and the results of the Computational Thinking Evaluation Scale of the two groups of students were analyzed by T-test through SPSS software, and the results of the analysis are shown in Table 2, and the results show that the P-value is 0.845, which is greater than 0.05, indicating that there is no significant difference between the pre-test scores of the two classes before starting the teaching experiment. After the end of the teaching experiment, a posttest was conducted for the students of the two classes, and the results showed that the P-value was 0.036, which is less than 0.05, indicating that there was a significant difference between the two groups of students in terms of their posttest scores by conducting an independent samples t-test.

It can be seen that the design-based learning model constructed in this study produces certain pedagogical effects and has a positive effect on developing students' computational thinking skills.

Table 2: Independent sample t test

/		Levene test of the variance equation		T test of the mean equation		
		F	P	T	Df	P
Pretest	Assumed equal variance	0.074	0.765	-0.245	35	0.845
	Unassuming equal variance			-0.245	31.636	0.845
Posttest	Assumed equal variance	0.642	0.466	-2.348	35	0.036
	Unassuming equal variance			-2.348	30.468	0.036

IV. B. 2) Analysis of pre-test and post-test scores of experimental and control classes

Table 3 shows the analysis of the pre-test and post-test scores of inferential thinking of the experimental and control classes. The mean score of the experimental class improved from 65.616 in the pre-test to 73.379 in the post-test, which is an improvement of 7.763 points, indicating that the overall improvement of the inferential thinking ability of the experimental class is relatively large. The mean score of the control class increased from 65.008 in the pre-test to 68.722 in the post-test, an increase of 3.714 points, indicating a smaller overall improvement in the reasoning thinking ability of the control class compared to the experimental class.

Table 3: Prethinking and post-test results analysis

Class	Type	Number	Overall average	The mean of creativity	Algorithm thought total mean
Experimental group	Pretest	15	65.616	10.278	11.975
	Posttest	15	73.379	11.469	14.345
Control group	Pretest	15	65.008	10.164	11.948
	Posttest	15	68.722	11.315	13.045
Class	Type	Number	The average of the total	The mean of critical thinking	Problem resolution
Experimental group	Pretest	15	13.648	12.326	17.389
	Posttest	15	15.249	13.248	19.068
Control group	Pretest	15	12.786	12.598	17.512
	Posttest	15	13.498	13.268	17.596

IV. B. 3) Posttest Data Analysis of Five Dimensions of Reasoned Thinking

The posttest data of the five dimensions of computational thinking ability of the experimental and control classes were analyzed, and the results of the data analysis are shown in Table 4. After the completion of the teaching experiment, the probability of companionship P of each dimension of computational thinking is 0.436, 0.945, 0.636, 0.798, 0.976 for creativity dimension, algorithmic thinking dimension, cooperation ability dimension, problem solving ability dimension, and critical thinking dimension, respectively, which are all greater than 0.05, and the

assumption of equal variance indicates that the variance of the five dimensions of computational thinking in the two classes has a chi-square. After the t-test of the equations of means, only the p-value of the cooperative ability dimension (0.001), the problem solving ability dimension (0.002), and the critical thinking dimension (0.043) is less than 0.05, while the p-value of the creativity dimension (0.248), and the algorithmic thinking dimension (0.053) is greater than 0.05, which indicates that among the five dimensions, the experimental class and the control class only have the cooperative ability dimension, problem solving ability dimension and critical thinking dimension have significant differences, and the other two dimensions (creativity dimension, algorithmic thinking dimension) do not have significant differences.

Table 4: The analysis of the data of the reasoning thinking ability is analyzed

/		Levene test of the variance equation		T test of the mean equation						
		F	P	T	Df	P	Mean difference	Standard error value	The difference is 95% confidence interval	
									Lower limit	Upper limit
Creativity	Assumed equal variance	0.536	0.436	-0.316	32	0.248	-0.053	0.154	-0.348	0.756
	Unassuming equal variance			-0.316	31.936	0.248	-0.053	0.154	-0.348	0.756
Algorithm thinking	Assumed equal variance	0.005	0.945	-1.755	32	0.053	-0.348	0.163	-0.715	0.085
	Unassuming equal variance			-1.755	31.933	0.053	-0.348	0.163	-0.715	0.085
Cooperative ability	Assumed equal variance	0.189	0.636	-2.069	32	0.001	-0.485	0.224	-0.836	0.044
	Unassuming equal variance			-2.062	31.248	0.001	-0.485	0.224	-0.836	0.044
Critical thinking	Assumed equal variance	0.134	0.798	-1.736	32	0.043	-0.315	0.186	-0.654	0.085
	Unassuming equal variance			-1.736	31.875	0.043	-0.315	0.186	-0.654	0.085
Problem solving	Assumed equal variance	0.002	0.976	-2.069	32	0.002	-0.295	0.136	-0.593	0.043
	Unassuming equal variance			-2.069	31.354	0.002	-0.295	0.136	-0.593	0.043

V. Conclusion

Based on the program design teaching reform strategy under artificial intelligence, this paper proposes to realize personalized recommendation of program design course and diagnosis of students' reasoning thinking ability with the help of artificial intelligence technology. A control experimental group is designed to study the practical application of personalized recommendation model in program design teaching and evaluate students' reasoning thinking ability.

Comparing the knowledge point completion of each student, the personalized recommendation algorithm used in the experimental group recommended the learning knowledge points of programming to 15 students, and the mean score of all learning objects was 3.467.In the analysis of the learning efficiency, the experimental group showed a greater improvement in the learning efficiency and effectiveness, and the average time consumed for the five knowledge points was 16,224.2s, which is a saving of 1,905.6s compared with that of the control group.

Using BM SPSS 24.0 to test whether the pre-test and post-test data of the experimental and control classes conformed to normal distribution, the mean score of the experimental class increased from 65.616 points in the pre-test to 73.379 points in the post-test, which is an increase of 7.763 points, indicating that the experimental class had a relatively large improvement in the overall ability of reasoning thinking.

References

[1] Kong, S. C., Lai, M., & Sun, D. (2020). Teacher development in computational thinking: Design and learning outcomes of programming concepts, practices and pedagogy. Computers & Education, 151, 103872.

[2] Medeiros, R. P., Ramalho, G. L., & Falcão, T. P. (2018). A systematic literature review on teaching and learning introductory programming in higher education. IEEE Transactions on Education, 62(2), 77-90.

[3] Bowman, N. A., Jarratt, L., Culver, K. C., & Segre, A. M. (2021). The impact of pair programming on college students' interest, perceptions, and achievement in computer science. ACM Transactions on Computing Education, 21(3), 1-19.
[4] Umapathy, K., & Ritzhaupt, A. D. (2017). A meta-analysis of pair-programming in computer programming courses: Implications for educational practice. ACM Transactions on Computing Education (TOCE), 17(4), 1-13.
[5] Alammary, A. (2019). Blended learning models for introductory programming courses: A systematic review. PloS one, 14(9), e0221765.
[6] AbdelRahman, S. M., AL-Syabi, B., Al Sharji, E., & Al Kaabi, S. S. (2016). The reasons behind the weakness of some students in programming courses in the college of applied science, ibri. International Journal of Modern Education and Computer Science (IJMECS), 8(1), 48-54.
[7] Cárdenas-Cobo, J., Puris, A., Novoa-Hernández, P., Parra-Jiménez, Á., Moreno-León, J., & Benavides, D. (2021). Using scratch to improve learning programming in college students: A positive experience from a non-weird country. Electronics, 10(10), 1180.
[8] Yildiz Durak, H. (2020). The effects of using different tools in programming teaching of secondary school students on engagement, computational thinking and reflective thinking skills for problem solving. Technology, Knowledge and Learning, 25(1), 179-195.
[9] Mathew, R., Malik, S. I., & Tawafak, R. M. (2019). Teaching Problem Solving Skills using an Educational Game in a Computer Programming Course. Informatics in education, 18(2), 359-373.
[10] Hidalgo, C. G., Bucheli-Guerrero, V. A., & Ordóñez-Eraso, H. A. (2023). Artificial intelligence and computer-supported collaborative learning in programming: A systematic mapping study. Tecnura, 27(75), 175-206.
[11] Yilmaz, R., & Yilmaz, F. G. K. (2023). The effect of generative artificial intelligence (AI)-based tool use on students' computational thinking skills, programming self-efficacy and motivation. Computers and Education: Artificial Intelligence, 4, 100147.
[12] Abichandani, P., Iaboni, C., Lobo, D., & Kelly, T. (2023). Artificial intelligence and computer vision education: Codifying student learning gains and attitudes. Computers and Education: Artificial Intelligence, 5, 100159.
[13] Kose, U., & Arslan, A. (2017). Optimization of self-learning in Computer Engineering courses: An intelligent software system supported by Artificial Neural Network and Vortex Optimization Algorithm. Computer Applications in Engineering Education, 25(1), 142-156.
[14] Timcenko, O. (2024, November). Case Study: Using Artificial Intelligence as a Tutor for a Programming Course. In 2024 21st International Conference on Information Technology Based Higher Education and Training (ITHET) (pp. 1-4). IEEE.
[15] Wu, Y. (2019, September). Object-oriented programming course reform using python language in the background of artificial intelligence. In 2019 3rd International Conference on Education, Management Science and Economics (ICEMSE 2019) (pp. 93-96). Atlantis Press.
[16] Tsai, C. C., Chung, C. C., Cheng, Y. M., & Lou, S. J. (2022). Deep learning course development and evaluation of artificial intelligence in vocational senior high schools. Frontiers in Psychology, 13, 965926.
[17] Thomas, G. (2021). Redesign of an elective introductory artificial intelligence course in a credit-limited computer science curriculum. Journal of Computing Sciences in Colleges, 37(2), 98-104.
[18] Changzhi Sun,Shijie Huang,Banghui Sun & Shiwei Chu. (2025). Personalized learning path planning for higher education based on deep generative models and quantum machine learning: a multimodal learning analysis method integrating transformer, adversarial training and quantum state classification. Discover Artificial Intelligence,5(1),29-29.
[19] Mehdi Darban. (2025). The future of virtual team learning: navigating the intersection of AI and education. Journal of Research on Technology in Education,57(3),659-675.
[20] Ping Chen. (2025). Construction and Research of Personalized University English Learning Platform Based on Recommendation Algorithm. Applied Mathematics and Nonlinear Sciences,10(1),
[21] Sohaib Ahmed,Syed Ahmed Hassan Bukhari,Adnan Ahmad,Osama Rehman,Faizan Ahmad,Kamran Ahsan & Tze Wei Liew. (2025). Inquiry based learning in software engineering education: exploring students' multiple inquiry levels in a programming course. Frontiers in Education,10, 1503996-1503996.