Journal of Contemporary Educational Research, 2026, Volume 10, Issue 6
http://ojs.bbwpublisher.com/index.php/JCER
Online ISSN: 2208-8474
Print ISSN: 2208-8466

Research on the Reconstruction of Data Structure Experimental Teaching Evaluation System and AI Standard Usage Mechanism under the Background of Large Models

Yueyue Liu, Xueqing Lu, Qingqing Liu, Kai Xu

School of Computer and Software Engineering, Anhui Institute of Information Technology, Wuhu 241000, Anhui, China

Abstract: Aiming at the problems in the experimental teaching of data structure such as the replacement of students' learning process, the weakening of algorithm thinking, and the distortion of teaching evaluation results caused by the overuse of large models, this paper analyzes the influence mechanism of large models on students' ability after their intervention in experimental teaching. On the basis of defining the reasonable usage boundary of large models, a new procedural teaching evaluation system is constructed, and an implementation plan for data structure experimental teaching under the constraint of standardized AI usage is proposed. Teaching practice results show that this evaluation model can effectively standardize students' use of AI and has a significant promoting effect on improving students' algorithm understanding ability, program debugging ability, and engineering practical ability.

Keywords: Large model; Data structure; Experimental teaching; Procedural teaching evaluation

Online publication: July 14, 2026

1. Introduction

With the rapid development of generative artificial intelligence, intelligent tools represented by large models such as Tongyi Qianwen and Spark Desk have shown powerful capabilities in code generation, algorithm derivation, program debugging, and other tasks, and have been quickly integrated into college classrooms. Documents including *Notice of the General Office of the Ministry of Education on Organizing and Implementing the Digital Empowerment Teacher Development Action* and *Outline of the Education Power Construction Plan (2024–2035)* have been issued successively [1,2], clearly taking "promoting the innovative integrated application of artificial intelligence technology" as the direction of education reform. Generative artificial intelligence is widely applied in lesson preparation, classroom teaching, curriculum ideological and political education, after-school tutoring, and other links, providing new technical support for teaching innovation. Meanwhile, it also brings new teaching risks.

As a core course for computer majors, Data Structure undertakes the core task of cultivating students' logical thinking, algorithm design, and engineering practical ability [3–6]. Among them, experimental teaching is an important link in cultivating students' core literacy [7]. However, generative artificial intelligence can provide executable code quickly, allowing students to obtain complete programs in a very short time. This phenomenon not only impacts the teaching goal of "ability cultivation as the core," but also leads to practical dilemmas such as homogenized assignments, invisible learning process, and distorted ability evaluation under the traditional result-oriented evaluation method.

In this context, how to give play to the enabling role of large models in teaching while preventing them from replacing students' core abilities has become a key problem to be solved urgently in the teaching reform of Data Structure course. Taking "the degree of AI use in programming classrooms" and "reconstruction of procedural evaluation system" as the main line, this paper explores the innovative path of Data Structure experimental teaching in the era of generative artificial intelligence, and provides a reference for the teaching design and evaluation reform of programming courses in colleges and universities.

2. Current situation and problems of data structure experimental teaching under the background of large models

After freshman year, students have mastered programming syntax such as C language and Java, and have basic code reading and implementation capabilities. However, the teaching objectives of Data Structure require students to have higher-level abilities including abstract modeling, algorithm design, and complexity analysis, as well as systematic programming and debugging capabilities. These abilities are the key for students to transform from "being able to write programs" to "being able to design algorithms and solve complex engineering problems" [8,9].

The project-based teaching content of Data Structure is shown in **Table 1**.

Table 1. Practical project system of data structure

Project No.	Project name	Core investigation content	corresponding course module
1	Simple Sequential List System	Basic operations of sequential list	Sequential List
2	Simple Linked List System	Establishment of linked list by tail insertion and basic operations of linked list	Linked List
3	Base Conversion System	Basic operations of stack and base conversion realized by stack	Stack
4	Josephus Circle System	Basic operations of queue and Josephus Circle realized by queue	Queue
5	Simple Binary Tree System	Establishment and traversal of binary tree	Binary Tree
6	Simple Undirected Graph System	Storage and establishment of graph, calculation of vertex degree	Graph
7	Simple Hash Table System	Storage structure and establishment of hash table	Search
8	Bubble Sort System	Principle and algorithm implementation of bubble sort	Sorting

In traditional project teaching, closed-loop training of "demand analysis—algorithm design—coding implementation—repeated debugging—complexity evaluation" enables students to gradually form cognitive structure and practical skills in problem modeling and engineering implementation. With the large-scale intervention of generative artificial intelligence, students increasingly rely on large models to generate executable code directly in experimental tasks [10]. The original cognitive process centered on generation and error correction is compressed or even partially replaced, and experimental activities tend to "verify AI results" rather than "independently design algorithms."

Reflecting on the above current situation, there are three key problems in the current teaching process.

First, lack of procedural training. With the support of generative artificial intelligence, students only need to input experimental purposes and requirements to obtain complete executable algorithm codes. Key processes such as problem abstraction, algorithm construction, data structure selection, and complexity analysis that should be completed independently are greatly compressed or even replaced, resulting in an obvious weakening of algorithm thinking training.

Second, intensified fear of programming aggravates AI dependence. It is found in actual teaching that some students have varying degrees of fear of programming tasks, lack confidence in the implementation of complex algorithms and patience in debugging. When encountering difficulties, they tend to directly obtain complete codes from AI to avoid learning risks. Although this learning method enhances students' programming self-confidence, it weakens their independent exploration awareness and problem-solving willingness to a certain extent.

Third, failure of evaluation system. Result-oriented evaluation is difficult to distinguish real ability from tool output. The current evaluation method dominated by experimental reports and operation result correctness can hardly distinguish students' real ability level under the background of large models. There is an obvious disconnection between evaluation conclusions and students' algorithm abilities, leading to the failure of teaching incentive mechanism.

Based on the above problems, this paper proposes a teaching evaluation system constrained by AI usage, and makes adjustments from three aspects: defining the boundary of AI usage, reconstructing procedural evaluation system, and adjusting experimental task structure.

3. Practical teaching evaluation system constrained by AI standard usage

3.1. Defining the reasonable usage boundary of large models in data structure experimental teaching

From the essence of teaching, artificial intelligence should serve the development of students' cognitive ability rather than replace their core learning process. The role of large models in Data Structure experimental teaching is positioned as an auxiliary tool, whose reasonable functions are mainly reflected in helping students understand abstract algorithm ideas, prompting possible design ideas, and assisting program error location.

Correspondingly, the following behaviors are defined as typical abuse: directly generating complete experimental codes for assignment submission; directly replacing key algorithm design links; generating complete experimental reports by large models. To enhance operability, this paper constructs a hierarchical specification model for AI usage in data structure experiments: allowed at the concept understanding layer, controlled at the algorithm design layer, strictly restricted at the code implementation layer, and guided at the algorithm optimization layer. The model defines a clear boundary for AI applications.

3.2. Reconstructing data structure experimental teaching evaluation system

Different from traditional evaluation focusing on result correctness, the evaluation system constructed in this paper takes "learning process and ability achievement" as the core and highlights the value of procedural data. Evaluation data mainly come from compilation and debugging times, error type distribution, modification track of key algorithm steps, and complexity change path.

On this basis, a multi-dimensional evaluation indicator system covering algorithm design ability, logical correctness, complexity analysis ability, debugging and optimization ability, and standardized coding ability is constructed, with evaluation weights reconstructed in an ability-oriented manner: procedural data accounts for 40%, algorithm explanation and derivation 30%, on-site operation 20%, and final result correctness 10%. This system effectively weakens the evaluation deviation of only focusing on results.

3.3. Implementation plan of data structure experimental teaching under AI standard usage constraints

Fundamentally, students' fear of programming is closely related to the traditional teaching structure of valuing theoretical deduction over frequent practical training. With limited computer practice training, it is difficult for students to form stable programming proficiency and accumulate debugging experience, making them more likely to feel frustrated when facing comprehensive experimental tasks and thus turn to rely on AI tools. From the two dimensions of strengthening procedural training and optimizing assessment mechanisms, this paper constructs a solution combining behavioral intervention and motivation guidance.

3.3.1. Alleviating programming fear through high-frequency and low-burden practical training

Complete algorithm projects are split into progressive micro-tasks such as single operation, single-step recursion, and local module implementation by increasing short-cycle programming and instant debugging sessions in class. It gradually lowers the psychological threshold for students to complete complex programming tasks. Meanwhile, with the help of online evaluation platforms and visual debugging tools, instant feedback is provided to help students accumulate successful experience in the process of repeated attempts and gradual improvement, so as to enhance programming self-confidence.

3.3.2. Introducing procedural programming ability into final assessment to form institutional guidance

To force students to attach importance to the cultivation of real programming ability from the evaluation level, computer-based examination is formally incorporated into the final assessment system, constituting the final course score together with written examination results, regular performance, and class notes. The computer-based examination focuses on students' ability to complete algorithm implementation and debugging within a limited time, paying attention to problem analysis path, code standardization, and debugging strategies rather than only the correctness of final results. Directly linking real programming ability with course grades prompts students to continuously engage in programming training throughout the semester, fundamentally eliminating the speculative learning path of relying on AI to pass the course.

4. Teaching practice effect and comparative analysis

According to the comparison of data before and after implementation in practical teaching, the homogeneity rate of experimental assignments decreased significantly, students' debugging success rate improved significantly, and the overall scores of algorithm understanding test increased. Students have a more solid grasp of algorithm ideas and complexity analysis. Questionnaire survey results show that most students recognize the AI standard usage mechanism, and believe it helps the real improvement of their abilities rather than merely pursuing correct assignments (**Figure 1**).

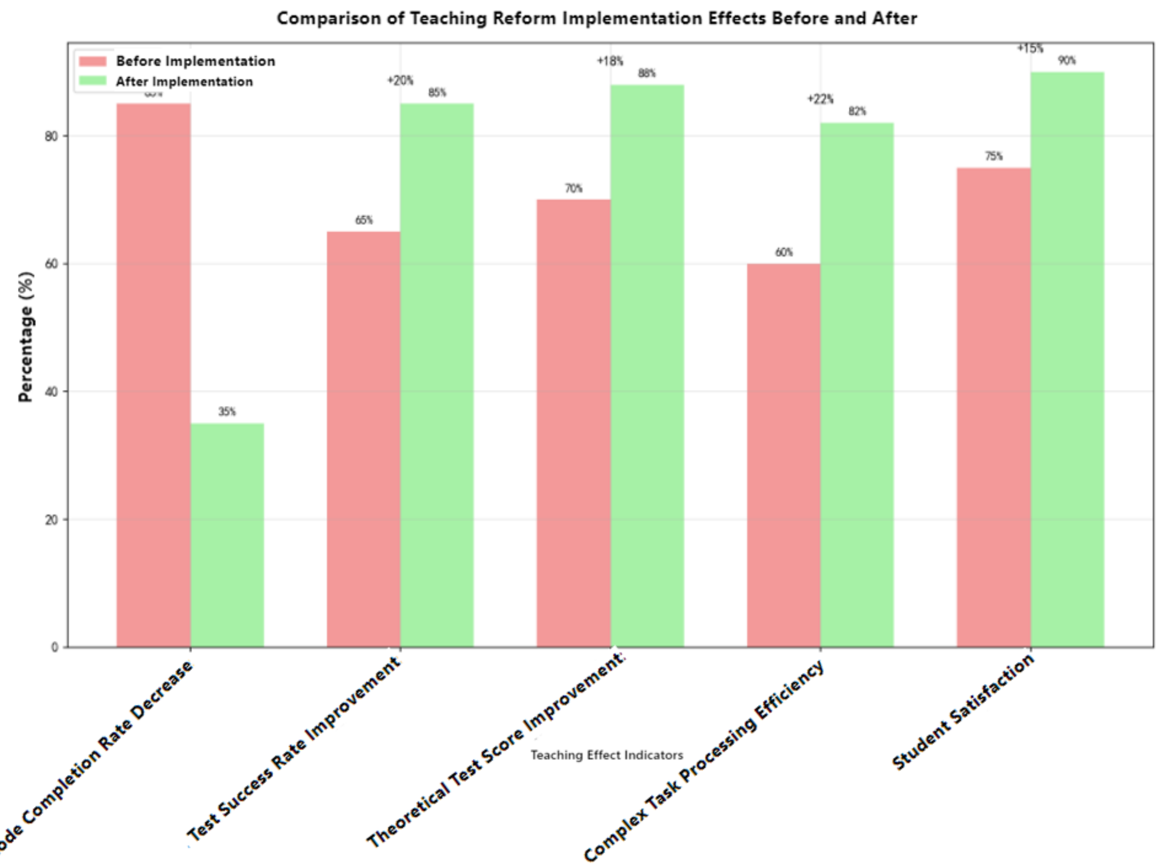

Figure 1. Comparative analysis of teaching reform effects

5. Conclusion and prospects

Focusing on the problem of distorted ability evaluation in Data Structure experimental teaching under the background of large models, this paper systematically explores a new path of experimental teaching reform from the two dimensions of defining a reasonable AI usage boundary and reconstructing the procedural evaluation system. Practical results show that this model can effectively restrain students' AI dependence behavior and promote the coordinated development of students' algorithm ability, debugging ability, and engineering practical ability. In the future, with the further maturity of intelligent teaching data collection and learning analysis technology, the evaluation of Data Structure experimental teaching will become more refined and intelligent. This model can also be extended to core courses such as Operating System, Compilation Principle, and Software Engineering, providing a solid ability guarantee for the cultivation of computer talents in the era of large models.

Disclosure statement

The authors declare no conflict of interest.

References

[1] General Office of the Ministry of Education, 2025, Notice on Organizing and Implementing the Digital Empowerment Teacher Development Action, Official Letter of the Department of Teachers, (13).

[2] The State Council, 2025, Outline of the Education Power Construction Plan (2024–2035).

[3] Ruan R, Wang H, Ye H, 2025, Advanced Teaching Practice of Data Structure under the Background of Emerging Engineering. Computer Education, (04): 70–74.

[4] Li S, Peng F, Liang W, 2025, Hybrid Teaching Mode of Data Structure Based on “One Foundation, Two Basics, Four Beams and Eight Pillars.” Computer Education, (11): 87–92.

[5] Lan H, Xu J, Xue Y, 2025, Exploration on Teaching Reform of Data Structure Course under Emerging Engineering. Computer Education, (02): 180–185.

[6] Pang R, Lu X, 2025, Interactive Visualization Implementation of Data Structure and Algorithm Based on Agent. Journal of East China Normal University (Natural Science Edition), (05): 32–42.

[7] Huang L, Yu H, Li R, 2025, Research on Experimental Teaching Reform of Data Structure Oriented to Core Literacy. Computer Education, (10): 264–268.

[8] Yan R, Li Y, Li D, 2025, Exploration on Hybrid Teaching of Data Structure Course Based on EduCoder Platform. Forestry Education in China, 43(06): 82–90.

[9] Shi H, Zhang M, Zhao W, 2025, Research on the Connection Mode of Basic and Professional Courses Oriented to Innovative Talent Training——Taking Data Structure as an Example. Heilongjiang Education (Theory and Practice), (08): 75–78.

[10] Cui J, Zhang S, Chen J, 2025, Construction and Practice of AI-enabled Programming Course Teaching Mode. Computer Technology and Development, 35(12): 165–173.

Publisher’s note

Bio-Byword Scientific Publishing remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.