Utilizing Large Language Models to Assist in the Teaching of Fundamentals of Computer Courses

Yi Qiao, JunSong Ren
Sichuan Polytechnic University, Deyang, Sichuan, China

Abstract: The Fundamentals of Computer Science course faces the teaching contradiction between "broad coverage" and "in-depth delivery". Against the backdrop of educational digital transformation, Large Language Models (LLMs) provide a new possibility to resolve this contradiction. This paper systematically analyzes the role positioning, practical functions, and implementation paths of LLMs in the teaching of this course: from the teacher's perspective, LLMs can assist in constructing knowledge graphs, generating question banks and automatic grading, and enriching teaching activities; from the student's perspective, LLMs can help solidify foundational theories, assist in knowledge deduction, cultivate programming competencies, and broaden academic horizons. On this basis, the paper proposes a "Teacher-Medium-Student-LLM" four-in-one collaborative teaching model, and conducts a teaching demonstration with the "linked list module" in data structures as an example. Meanwhile, the paper also deeply discusses potential issues in the application of LLMs, such as data privacy risks, students' over-reliance, knowledge "hallucinations", and the inability to replace teachers' role in moral and value cultivation. The research aims to provide ideas for teaching innovation in computer-related fields, and also offer references for educators to adapt to the era of educational large models and update their teaching philosophies and methods.

Keywords: Computer Fundamentals; Undergraduate Education; Large Language Models; Artificial Intelligence

1. Introduction

As a core introductory course in the field of computer science, Fundamentals of Computer Science covers a broad range of topics while maintaining foundational importance. However, constrained by limited teaching hours, it is challenging to delve deeply into each subfield. This not only restricts the in-depth delivery of teaching content but also hinders students who wish to pursue further studies in computer science from acquiring the in-depth knowledge required for subsequent exploration. How to resolve this contradiction between "broad coverage" and "in-depth delivery" within fixed teaching hours, and maximize the quality of teaching and students' learning outcomes, has become a common challenge for frontline teachers of the Fundamentals of Computer Science course.

Against the backdrop of the digital transformation in education, university teachers have an urgent demand for new types of teaching auxiliary tools. In recent years, Large Language Models (LLMs), which have emerged prominently in the field of artificial intelligence, have become highly promising teaching aids due to their user-friendly nature and powerful functions. After 2020, artificial intelligence technology entered a stage of explosive development, with breakthroughs in LLMs attracting particular attention. In 2023, OpenAI released ChatGPT, whose exceptional natural language understanding and high-quality text generation capabilities triggered a global technological boom, injecting new vitality into the innovation of teaching models in the education sector [1-3].

As LLMs have gradually become an important research paradigm in multiple disciplinary fields [4-6], their application value in educational scenarios has also been gradually highlighted. Existing studies have shown that LLMs can achieve a level of performance comparable to that of average students in standardized tests covering various question types in disciplines such as mathematics and computer science [7]. In educational practice, LLMs can serve as intelligent assistants to support students in writing training and reading analysis [8,9]. Latest research further confirms that ChatGPT

possesses the ability to generate logically consistent answers across disciplines, ensuring both the breadth of knowledge and the depth of content [10]. Studies have also indicated that teachers who utilize LLMs demonstrate excellent performance in evaluating and optimizing the instructional design of courses and learning activities [11]. Additionally, a number of academic opinion papers have discussed the application scenarios of LLMs in classrooms, proposing that LLMs can play a crucial role in links such as teacher-student collaboration, personalized learning, and automated assignment evaluation [12,13].

At the same time, the application of LLMs in education also faces a series of practical issues: the risk of plagiarism when students use LLMs to complete assignments, potential cognitive biases in AI-generated content, the tendency of students to over-rely on LLMs, and the inequality faced by non-English speakers in accessing LLM resources. All these issues need to be carefully addressed in teaching practice [14].

Against this backdrop, the teaching model of the *Fundamentals of Computer Science* course has also ushered in an opportunity for in-depth transformation. This paper conducts a systematic analysis of the role positioning, practical functions, and implementation paths of LLMs in the teaching of this course. The specific chapter arrangement is as follows: Chapter 1, from the perspective of teachers, analyzes how LLMs empower the optimization of instructional design; Chapter 2 focuses on the student perspective, exploring the role of LLMs in promoting students' independent learning abilities and the development of core disciplinary competencies; Chapter 3, integrating both teacher and student perspectives, proposes a "four-in-one collaborative teaching model (Teacher-Medium-Student-LLM)"; Chapter 4 takes the "linked list module in data structures" as specific teaching content to conduct a demonstration case analysis of LLM-assisted teaching; Chapter 5, addressing the current lack of a sound institutional system for LLM applications, conducts an in-depth analysis of potential issues in aspects such as adherence to academic standards, knowledge accuracy (to avoid misleading), and the cultivation of students' in-depth thinking; Chapter 6 summarizes the research conclusions, practical implications, and future development directions of the entire paper, as shown in Figure 1.

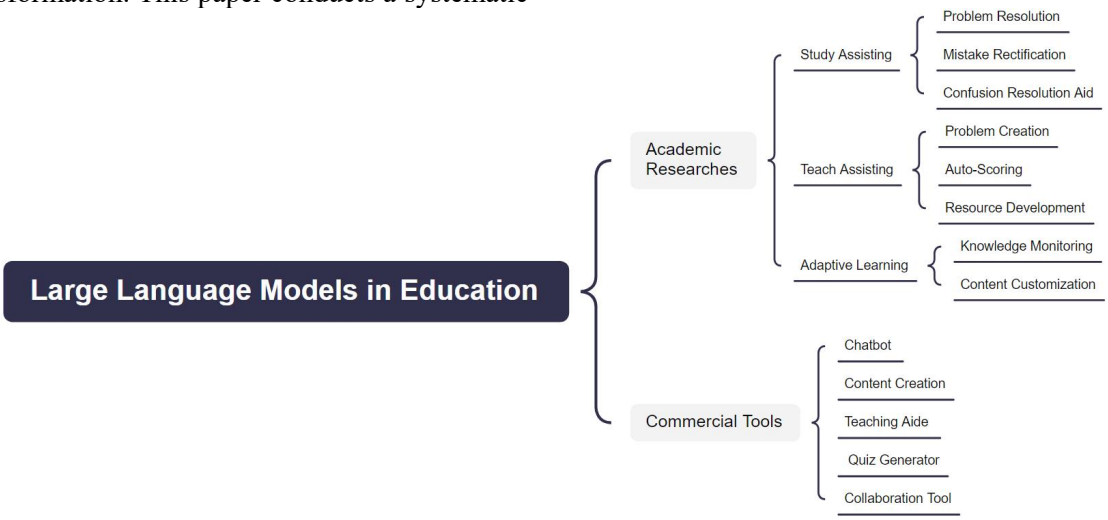

Figure 1. Large Model Classification for Educational Applications

2. The Teaching Significance of Large Language Models for Teachers of Fundamentals of Computer Courses

In the teaching of the Fundamentals of Computer Science course, LLMs provide support for the optimization of teachers' teaching from three key dimensions.

First, they facilitate the construction of knowledge graphs and the systematic design of teaching content. This course covers various types of knowledge, such as computer hardware, software, and programming languages, with closely interconnected knowledge points. LLMs can deeply mine massive resources including textbooks, syllabi, and classic cases. Through semantic recognition and logical correlation algorithms, they sort out the subordinate and intersecting relationships among knowledge points, and construct a complete visual

knowledge graph. With the help of this graph, teachers can clearly grasp the overall structure of the knowledge system, identify key points (e.g., "operating system process management") and difficult points (e.g., "pointer and memory address mapping"), and design teaching modules and schedules more scientifically. This avoids redundant explanations of knowledge points (e.g., repeated teaching of "file operations" in different chapters) or omissions (e.g., neglecting details of the data link layer in the "OSI seven-layer model of computer networks"), thereby ensuring the systematicness and coherence of the teaching process. For instance, when designing the teaching of the "Computer Organization Principles" chapter, the knowledge graph can intuitively present the data flow relationships among hardware components such as CPU and memory, as well as the interaction logic between hardware and operating systems/application software. This helps teachers reasonably allocate teaching hours between "hardware structure explanation" and "case analysis of software-hardware collaboration" and optimize the order of explanation.

Second, they empower question generation and automatic grading, alleviating the burden of teaching assessment. On one hand, based on teaching objectives, LLMs can automatically generate hierarchical and multi-type test questions to meet the dual needs of "theoretical understanding" and "practical application" in the Fundamentals of Computer Science course. For theoretical knowledge points (e.g., "IP address classification" and "characteristics of stacks in data structures"), they can generate objective questions such as multiple-choice questions, true/false questions, and short-answer questions, covering basic concepts and error-prone points. For practical skills (e.g., "application of Python basic syntax" and "Excel function operations"), they can generate programming practice questions, case analysis questions, and even simulate debugging tasks in programming environments. Teachers can adjust the difficulty of questions according to students' learning progress (e.g., preview before class, consolidation after class, and unit tests). For example, regarding the knowledge point of "loop structures", basic programming questions (e.g., "summing numbers from 1 to 100") can be generated for students with weak foundations, while comprehensive questions (e.g., "realizing matrix transposition using nested loops") can be designed for advanced students. This eliminates the need for teachers to compile test papers manually, significantly saving time spent on question preparation. On the other hand, LLMs possess efficient automatic grading capabilities: for objective questions, they can quickly match and grade answers through predefined answer banks and generate accuracy statistics; for subjective questions, especially programming-related ones, they can achieve accurate grading through code verification, logic correctness detection, and efficiency analysis (e.g., time complexity evaluation)-not only identifying syntax errors (e.g., missing semicolons, undefined variables) but also providing optimization suggestions (e.g., replacing ordinary loops with list comprehensions); for short-answer questions, they can judge whether students' answers cover core points (e.g., whether conditions such as "resource mutual exclusion" and "circular waiting" are mentioned when explaining "deadlock") based on keyword matching and logical integrity algorithms, and generate personalized grading feedback. For example, after the unit test of "Python function programming", LLMs can automatically calculate the error rate of questions related to "function parameter passing" in the class, mark frequent errors (e.g., confusing positional parameters with keyword parameters), provide data support for teachers' subsequent targeted explanations, and reduce the repetitive work of manual grading.

Third, they enrich the forms of teaching activities and enhance teaching effectiveness. On one hand, LLMs provide strong support for the implementation of flipped classrooms. Teachers can use LLMs to create pre-class materials such as animation scripts for knowledge points (which can be converted into videos) and hierarchical preview questions (integrated with the aforementioned automatic question generation function), allowing students to preview core course content in advance (e.g., "basics of database SQL statements"). In class, teachers can focus on organizing interactive sessions such as group discussions (e.g., "analyzing the application scenarios of different sorting algorithms") and case-based practical operations (e.g., "querying student information tables using SQL"), so as to deepen students' understanding and application of knowledge. On

the other hand, for answering questions about technical details such as programming technologies and computer hardware composition, LLMs can respond to students' personalized needs more quickly and accurately. When students encounter problems with programming syntax (e.g., "differences between Python lists and tuples") or questions about hardware working principles (e.g., "causes of differences in read-write speeds between solid-state drives (SSDs) and mechanical hard drives (HDDs)") after class, LLMs can immediately provide detailed graphic-text explanations and even simulate hardware working process animations. This not only reduces teachers' burden of after-class question answering but also solves students' problems in a timely manner, ensuring the continuity of learning.

3. The Significance of Large Language Models for Students of Fundamentals of Computer Courses

LLMs exert positive influences on students' learning of fundamentals of computer courses in multiple aspects, facilitating the improvement of learning effects and the development of comprehensive competencies.

(1) Solidifying Foundational Theoretical Knowledge

Based on Bloom's Taxonomy of Educational Objectives, the personalized and progressive explanation path of LLMs aligns with the human spiral cognitive law. For abstract knowledge domains such as computer architecture and operating system principles, students can continuously ask follow-up questions according to their individual learning situations. The LLM will gradually deepen its explanations from conceptual elaboration to principle analysis, guiding students to move from superficial cognition (memorizing definitions) to deep-level thinking (applying theories to solve problems), and ultimately helping them form a solid knowledge structure.

For instance, when learning about process management in operating systems, students can start by inquiring about the concept of "processes," and then gradually delve into more in-depth content such as process scheduling algorithms. Throughout this learning process, the LLM will provide corresponding explanations and illustrative cases in line with the students' learning progress.

(2) Assisting in the Deduction of Computer Knowledge

In the learning of binary numbers and logic algebra, LLMs explain the principle behind "a binary number ending in 0 being an even number" (i.e., the characteristic of bit weight) and demonstrate the steps of the "division-by-2 and remainder-taking method" using the example of converting the decimal number 15 to a binary number. In the learning of programming languages and algorithms, LLMs illustrate the process and principle of bubble sort-"comparing and swapping adjacent elements to make the largest value 'sink to the bottom'"-through demonstrations or textual descriptions, and explain the cyclic recursive logic of the "accumulative summation" algorithm by analyzing changes in variable values. In the learning of data structures, taking an int-type array as an example, LLMs explain the reason why "array indices start from 0" from the perspective of calculating "starting address + element size × index," helping students understand the deduction process of knowledge,as shown in Table 1.

Table 1. Questioning Levels and Examples Based on Bloom's Cognitive Taxonomy Theory

Cognitive Level	Key Definition	Key Question Verbs	Example Question
1. Remember	Recall factual information, basic concepts, or previously learned details. It is the lowest level of cognitive ability, focusing on "repetition".	Remember, list, identify, name, define, label	"List the main components of a basic computer hardware system."
2. Understand	Interpret, explain, summarize, or paraphrase information to show comprehension of meaning. It focuses on "translation and interpretation".	Explain, summarize, paraphrase, describe, interpret, clarify	"Explain the main functions of an operating system in a computer."
3. Apply	Use learned knowledge, concepts, or skills to solve problems, complete	Apply, solve, use, implement,	"Use the operating system's file management function to create a new

	tasks, or apply to new situations. It focuses on "practical application".	calculate, demonstrate	folder named 'Computer Basics Notes' on the desktop."
4. Analyze	Break down information into parts, identify relationships between components, or examine structure and logic. It focuses on "disassembly and connection".	Analyze, compare, contrast, distinguish, examine, identify relationships	"Compare and contrast the differences between a hard disk drive (HDD) and a solid-state drive (SSD) in terms of speed, capacity, and price."
5. Evaluate	Make judgments, assess value, justify opinions, or critique based on criteria and evidence. It focuses on "judgment and argumentation".	Evaluate, assess, judge, justify, critique, defend	"Evaluate which type of laptop (a lightweight ultrabook or a high-performance gaming laptop) is more suitable for a college student majoring in computer science, and justify your opinion with specific needs."
6. Create	Generate new ideas, design solutions, construct original works, or combine elements to form something new. It is the highest level of cognitive ability, focusing on "innovation and construction".	Create, design, develop, invent, construct, formulate	"Design a simple home network setup plan that allows 3 computers,2 smartphones, and 1 smart TV to connect to the internet simultaneously, and list the required network devices."

(3) Cultivating Students' Programming Competencies

When students encounter obstacles in programming thinking, LLMs can provide guidance. For example, when writing a program to calculate the average score, LLMs guide students to think about data acquisition, accumulative calculation, exception handling, and other key links, helping them sort out programming logic. When there are errors in the code, LLMs can quickly locate the errors (such as syntax errors and logical errors), analyze the causes, and provide modification suggestions. Additionally, based on students' programming proficiency, LLMs can recommend practice tasks ranging from syntax exercises to project development, enabling students to gradually enhance their programming competencies.

(4) Broadening Students' Horizons

LLMs can integrate the latest developments in the computer field, cutting-edge technologies (e.g., artificial intelligence, big data, and cloud computing), and industry application cases in real time, and present them in a accessible manner. Through interaction with LLMs, students can learn about the applications of these technologies in both the computer field and other industries such as healthcare, transportation, and finance-for example, the achievements of artificial intelligence in image recognition and the cases of big data in user behavior analysis. This not only helps students understand industry trends and broaden their knowledge scope but also stimulates their interest in exploring the computer field, laying a solid foundation for their future learning and career development.

4. Changes in Course Teaching Methods Brought About by The Large Language Model

The traditional basic computer courses adopt a "teacher-teaching tool-student" tripartite teaching model: teachers dominate knowledge impartation and control the teaching progress; teaching tools (such as teaching materials, courseware, etc.) bear the function of knowledge transmission; students passively receive knowledge. Due to the fixed nature of teaching content and methods, this model struggles to meet personalized learning needs and restricts students' initiative in learning.

After the introduction of LLMs, the teaching system has evolved into a "teacher-student-teaching tool-LLM" four-in-one structure.

The role of teachers has shifted from knowledge imparters to teaching organizers and guides. By virtue of LLMs, teachers can integrate teaching resources and develop customized learning plans to meet the personalized needs of students.

The dominant position of students has become prominent. Students can independently choose learning content and methods, seek answers to questions through LLMs, and collaborate to

complete tasks-thereby enhancing their participation and collaborative capabilities.

Teaching tools have achieved functional extension. Combined with LLMs, they have become a link connecting the three parties (teachers, students, and tools): they not only display teaching resources but also record learning data to assist teachers in adjusting their teaching strategies.

LLMs play a core auxiliary role: they support teachers in optimizing teaching, provide personalized guidance for students, and promote teacher-student as well as student-student interactions. By breaking the constraints of time and space, LLMs facilitate resource sharing and contribute to the improvement of teaching effects, as shown in Figure 2.

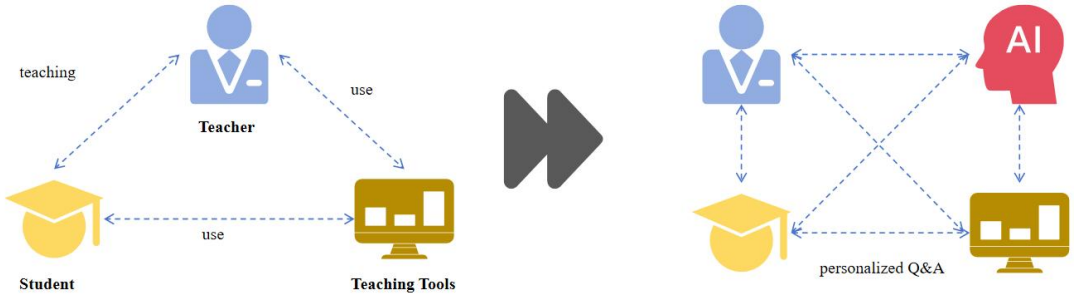

Figure 2. Schematic Diagram of the Transformation of Teaching Mode

5. A Case Study of Using Large Language Models to Assist Fundamentals of Computer Courses Teaching-Taking The Data Structure Linked List Module As An Example

Linked lists, a challenging topic in fundamental computer science courses, involve definitions, storage structures, basic operations, and applications, often posing comprehension difficulties for students. The following concisely analyzes the auxiliary effects of LLMs from the dual perspectives of teachers and students.

5.1 From the Perspective of Teachers' Teaching

Firstly, the linked list knowledge graph constructed by LLMs sorts out classifications, storage structures, operation methods, differences from arrays, and application cases. This helps teachers identify that the storage and operations of singly linked lists are key points and pointer handling is a difficulty, thereby optimizing teaching content.

Secondly, LLMs enrich teaching activities: in flipped classrooms, they generate preview materials (concept videos, schematic diagrams, and thinking questions); during class, teachers organize groups to discuss operation cases proposed by LLMs, with LLMs providing thought prompts; after class, students can seek immediate answers from LLMs.

5.2 From the Perspective of Students' Learning

Firstly, LLMs help solidify foundational knowledge. When students have doubts about concepts such as node composition and linked storage structures, LLMs guide them to deepen understanding through interactive Q&A and schematic diagrams.

Secondly, LLMs assist in derivation. Taking insertion/deletion operations in singly linked lists as examples, LLMs demonstrate steps (e.g., locating predecessor nodes and handling pointers during insertion) through specific cases and explain the underlying principles (e.g., avoiding chain breakage).

Thirdly, LLMs foster programming skills. When students encounter programming difficulties, LLMs guide their thinking (e.g., traversal functions requiring structure definitions and cyclic access), identify syntax/logic errors with suggestions for correction, and recommend exercises of appropriate difficulty.

Fourthly, LLMs broaden students' horizons by introducing the applications of linked lists in memory management, databases, and the implementation of stacks/queues, as well as optimization trends in big data scenarios, helping students understand their practical value and development prospects.

6. Rational Thinking About LLM

Although LLMs can efficiently output knowledge content, they remain difficult to replace the systematic teaching methods that have been verified through long-term practice. There are still many critical issues in their educational applications:

Educational data contains a wealth of minors'

privacy, and there are hidden risks of data leakage in model training and academic performance analysis, along with a lack of unified governance standards. Of greater concern is the issue of ideological security-biases in training data may lead LLMs to generate content with stereotypes, requiring the use of value alignment technologies to ensure consistency with the goal of fostering virtue through education.

Meanwhile, there exists a trap of over-reliance. Students' over-reliance on LLMs to complete assignments has already shown signs of declining critical thinking abilities, which necessitates guiding independent thinking through tools such as "inquiry-based chain thinking tools." Additionally, the "hallucination" problem of LLMs may spread incorrect knowledge, and there is currently no clear set of norms for human-AI collaboration or accountability mechanisms.

From the perspective of teaching process regulation, systematic teaching methods emphasize that teachers dynamically adjust teaching strategies based on real-time classroom feedback (e.g., students' in-class interaction performance, homework completion quality) and promptly identify common knowledge gaps. In contrast, LLMs can only passively respond to questions based on pre-set algorithms and cannot proactively detect deviations in the teaching process.

More crucially, systematic teaching methods bear the dual functions of "imparting knowledge" and "cultivating people": while delivering knowledge, teachers cultivate students' teamwork abilities through in-class interactions, guide students to develop independent thinking habits through critical discussions, and even convey values through their own words and deeds. However, LLMs lack the ability for real emotional perception and value judgment-they can neither simulate the emotional resonance between teachers and students nor help students develop higher-order thinking skills and sound personalities. This is precisely the core value of systematic teaching methods that cannot be replaced by technology.

7. Conclusion

The rapid advancement of LLMs is profoundly reshaping the overall ecosystem of the education sector. In the face of this transformation, educators must proactively embrace the trend: in the context of the era of educational large models, they should not only conduct in-depth reflections on the essence and goals of education to be fully prepared for industry changes, but also continuously update their teaching philosophies and practical methods, and take the initiative to learn and apply technologies related to educational large models, so as to better adapt to the development needs of the times and the growth demands of students.

This paper explores the current challenges faced in the application of LLMs in the education sector, as well as future development directions, aiming to provide ideas and inspiration for research on teaching innovation in computer-related fields.

References

[1] Chen L, Chen P, Lin Z. Artificial intelligence in education: A review[J]. IEEE access, 2020, 8: 75264-75278.

[2] Chiu T K F, Xia Q, Zhou X, et al. Systematic literature review on opportunities, challenges, and future research recommendations of artificial intelligence in education[J]. Computers and Education: Artificial Intelligence, 2023, 4: 100118.

[3] Denny P, Prather J, Becker B A, et al. Computing education in the era of generative AI[J]. Communications of the ACM, 2024, 67(2): 56-67.

[4] Chen Z, Mao H, Li H, et al. Exploring the potential of large language models (llms) in learning on graphs[J]. ACM SIGKDD Explorations Newsletter, 2024, 25(2): 42-61.

[5] Zhao Z, Fan W, Li J, et al. Recommender systems in the era of large language models (llms)[J]. IEEE Transactions on Knowledge and Data Engineering, 2024, 36(11): 6889-6907.

[6] Jin M, Zhang Y, Chen W, et al. Position paper: What can large language models tell us about time series analysis[J]. CoRR, 2024.

[7] Achiam J, Adler S, Agarwal S, et al. Gpt-4 technical report[J]. arXiv preprint arXiv:2303.08774, 2023.

[8] Malinka K, Peresíni M, Firc A, et al. On the educational impact of chatgpt: Is artificial intelligence ready to obtain a university degree?[C]//Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 1. 2023: 47-53.

[9] Rahman M M, Watanobe Y. ChatGPT for education and research: Opportunities, threats, and strategies[J]. Applied sciences, 2023, 13(9): 5783.

[10] Wen Q, Liang J, Sierra C, et al. AI for education (AI4EDU): Advancing personalized education with LLM and adaptive learning[C]//Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2024: 6743-6744.

[11] Zhang X, Zhang C, Sun J, et al. Eduplanner: Llm-based multi-agent systems for customized and intelligent instructional design[J]. IEEE Transactions on Learning Technologies, 2025.

[12] Kamalov F, Santandreu Calonge D, Gurrib I. New era of artificial intelligence in education: Towards a sustainable multifaceted revolution[J]. Sustainability, 2023, 15(16): 12451.

[13] Tan K, Pang T, Fan C, et al. Towards applying powerful large ai models in classroom teaching: Opportunities, challenges and prospects[J]. arXiv preprint arXiv:2305.03433, 2023.

[14] Kasneci E, Seßler K, Küchemann S, et al. ChatGPT for good? On opportunities and challenges of large language models for education[J]. Learning and individual differences, 2023, 103: 102274.

