DOI Link: https://doi.org/10.59671/9QQzY
Vol.408, Issue.6, Part.1, June 2025, PP. 150-170

Research on the Application of Data-Driven Smart Teaching in Programming Courses

Yao Wang, Xiao-Han Long, Liang Gao, Xing-Zhi Chang*
School of Software and Big Data, Changzhou College of Information Technology, Changzhou, China
Corresponding author: changxz@163.com
Received April 2025 Accepted May 2025 Published June 2025

Abstract: The insufficient data collection and low teacher-student interaction efficiency in programming courses are common problems encountered. In order to solve the problems, this study presents a data-driven smart teaching model (DD_STM) with a four-layers architecture, which are teaching interaction layer, data collection and storage layer, intelligent computing layer and intelligent decision application layer. Moreover, a smart teaching platform has been developed to support the implementation of DD_STM. Empirical study from "Foundations of Python Programming instruction" indicates that the smart teaching model significantly improves teacher-student interaction efficiency and promotes higher level of students learning participation. The positive outcomes may be attributed to the timely response from the smart platform and the effective feedback on students' difficulties, which enhance students' initiative in learning and thereby promote greater engagement in the learning process.

Keywords: Data-driven smart teaching model, AI assistant, Learning participation, Interaction efficiency.

1 Introduction

With the rapid development and extensive application of new digital technologies, the field of education is undergoing a profound digital transformation. This transformation, driven by technological innovation, is not only reshaping teaching philosophies, but also revolutionizing teaching formats and models, which thereby facilitates the shift from traditional to smart education model [1-3]. The process of educational reform is initially reflected in the construction of digital resources and the exploration of innovative educational models. According to the Blue Book of China's Smart Education (2022), China has initially established a digital education environment. In this context, it is particularly necessary to conduct in-depth exploration and research on the practical application of education digitization.

Modern teaching is based on learning data and learning analysis. However, traditional teaching platforms usually only collect simple structured data, such as access frequency statistics, knowledge test score records, attendance rates and so on. This kind of data can only meet the basic requirements of teaching support and knowledge evaluation of some subjects, but can not satisfy the needs of complex teaching process of programming courses. The lack of process data in

ISSN: 0034-8082

programming practice courses makes it difficult for teachers to obtain data of students' learning behaviors in time, thus hindering the immediate feedback and guidance to students. As a result, the organization and promotion of teachers' teaching process often rely on their subjective judgment, lacking objectivity and rigor. In programming courses, how to acquire learning data and utilize technologies, such as big data and artificial intelligence, for learning analysis, optimizing the teaching process, and enhancing the efficiency of teacher-student interaction has sparked extensive discussions [4-5].

Nowadays, students can use diverse tools or platforms to perform the assigned task, which leads to the generation of large amounts of data. Cirigliano et al. [6] provided a framework for collecting student data, which is helpful in showing student engagement in an online environment. However, educators can rarely understand the impact of this data on curriculum progress and what relevant decisions can be made [7]. Methods and tools of learning analytics can analyze data generated during learning process and provide informative insights [8], thereby providing instructional decision support for teachers, and promoting the improvement of teaching quality.

Teacher-student interaction is a pivotal element in the learning process, which provides essential support and assistance for students both academically and emotionally. It also creates a high-quality classroom atmosphere at the psychological level, which promotes student engagement in the online classroom and delivers better learning experiences and outcomes [9]. Teachers have addressed such requirements to some extent, but expanding their supports to match the growing class size and diversity remains a significant challenge [10]. They fail to timely and comprehensively grasp and address the statuses and issues of all students throughout the entire teaching process. The human resource capacities of teachers, as well as their availabilities and accessibilities, limit the interaction efficiency between teachers and students [4, 11]. Figure 1 illustrates the issue of single-threaded interaction between teachers and students in the traditional teaching mode. This single-threaded interaction mode makes teachers a bottleneck in the overall teaching process, leaving students who encounter problems to often be confused and wait, thus resulting in low teaching and communication efficiency.

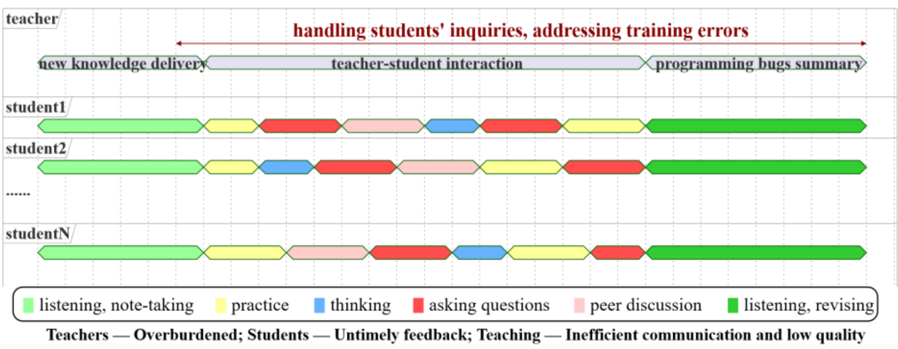

Figure 1 Single-threaded communication between teachers and students

ISSN: 0034-8082

Enhancing teachers’ abilities to support and respond to students’ needs is crucial. An effective approach is to utilize learning analytics to extract useful information from learning process data, thereby helping teachers make decisions. Rienties et al. [12] used learning analytics to investigate students’ behaviors, satisfaction, and learning performances. Foster and Siddle [13] investigated the effectiveness of learning analytics for identifying low engagement and at-risk students. In addition, AI-enabled education is playing a more important role as learning requirements promote [14]. Intelligent education systems provide timely and personalized instructions and feedbacks for both teachers and learners. They are designed to improve learning value and efficiency by using multiple new generation information technologies, especially machine learning related technologies [15-16]. Some studies also focused on the capabilities of LLMs (Large Language Models) in solving programming exercises, code explanation, error detection, question answering and interaction. Artificial intelligence has been demonstrated to provide personalized assistance to students, freeing educators from the overwhelming task of constantly answering questions [17-18].

The self-determination theory identifies autonomy, competence, and relatedness as the three key factors that stimulate intrinsic motivation. In teacher-student interaction, teachers can enhance students' sense of autonomy and competence by providing constructive and meaningful feedback and support, thereby boosting their learning motivation and academic performance [19]. Achuthan et al. [20] emphasized that student engagement and academic performance can be improved by enhancing dialogue, optimizing course structure, and promoting learner autonomy to reduce transaction distance. Xiao et al. [21] utilized multimodal data analysis, incorporating platform behavior and emotional data to explore the impact of teacher-student interactions and emotions on students' academic performance within online classrooms, which indicated that interactive behaviors have a positive effect on real-time quiz performance for adult learners. Wang et al. [22] conducted a randomized experiment to investigate the impacts of adding interactive elements to pure online courses on students' academic performance and personality traits of K-12 grade students, which revealed that adding interactive components could improve students' test scores.

As it can be concluded, the difficulties and insufficiencies in collecting multisource process data, as well as the teacher-student interaction efficiency, remain to be the key problems affecting modern teaching model. In this study, an intriguing emerging method to solve such challenges is to construct smart teaching platforms to analyze data and provide teaching assistant to students [23-24].

The remaining sections of this paper are organized as follows: In section 2, a data-driven smart teaching model (DD_STM) including four different layers has been constructed. The implementation and application of DD_STM exhibit its effectiveness and interaction efficiency in Section 3. Conclusions and summaries are in section 4.

ISSN: 0034-8082

2 Construction of a DD_STM

2.1 System Architecture Design

Based on the advancements of new-generation information technology and the practical needs for smart teaching reform, a DD_STM architecture (illustrated in Figure 2) was designed. It consists of four layers: teaching interaction layer, data acquisition and storage layer, intelligent computing layer, and smart decision application layer. This architecture supports real-time multimodal data acquisition, system learning analysis, and dynamic adjustment of teaching strategies based on learning analysis. The introductions of different layers are listed below.

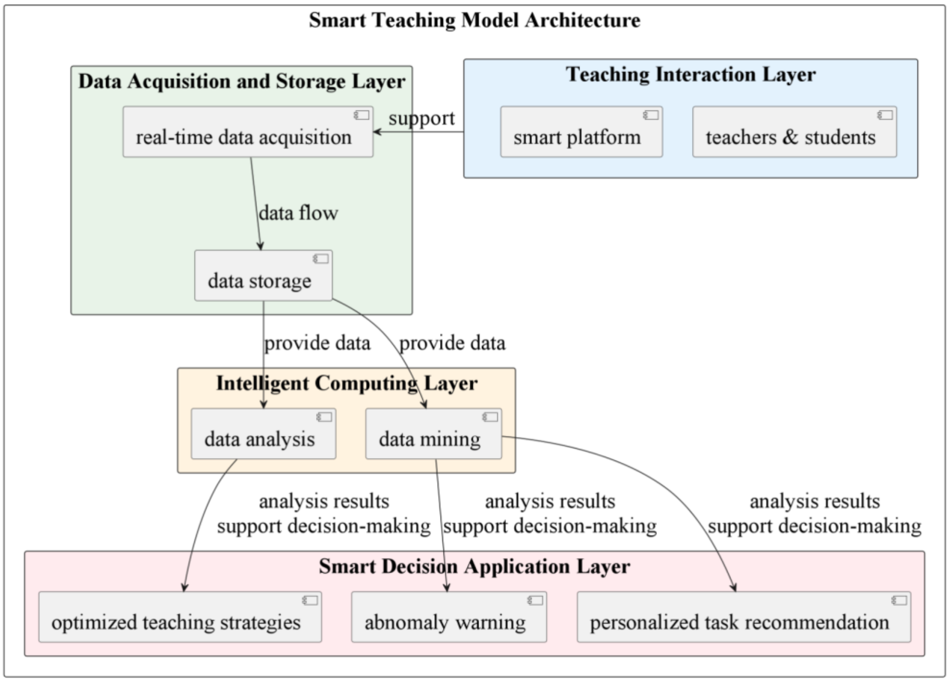

Figure 2 Four-layers DD_STM architecture

2.1.1 Teaching Interaction Layer

The teaching interaction layer provides teachers and students the entrance of conducting teaching and learning activities based on smart platform. Both teachers and students can log into the smart platform. Through the teaching interface layer, teachers can assign programming tasks, view the analysis results and conduct teaching management, while students can receive programming practice tasks, study the learning resources, complete the programming task assigned by teachers or selected by themselves and view the feedback of data analysis.

2.1.2 Data Acquisition and Storage Layer

The data acquisition and storage layer is responsible for real-time collection and storage of multi-source heterogeneous data generated in the teaching environment, and is a crucial link between the teaching interaction layer and the intelligent computing layer.

This layer not only processes structured data (such as academic achievements and examination scores) but also collects and analyzes unstructured data (such as programming practical training tasks, error debugging, access to digital textbooks, interactions with AI teaching assistant, comments and feedback, etc.).

2.1.3 Intelligent Computing Layer

The intelligent computing layer is responsible for the in-depth processing and intelligent analysis of the data throughout the teaching process from the data collection and storage layer. It serves as a crucial link connecting the data acquisition and storage layer and the smart decision application layer.

This layer employs big data analysis techniques and machine learning algorithms to extract valuable information and patterns from multi-source heterogeneous data. The analysis methods include batch processing and real-time analysis. Batch processing is suitable for processing large-scale historical datasets, which can reveal long-term trends and patterns, and contribute to the iterative improvement of teaching plans. Real-time analysis, through the analysis of immediate data streams, provides rapid feedback and suggestions for dynamic adjustment, ensuring the efficiency and adaptability of classroom teaching activities.

The intelligent computing layer provides a reliable learning analysis foundation for the data visualization presentation, information insight within the data, abnormal situation warning, and teaching strategy optimization in the Smart Decision Application Layer.

2.1.4 Smart Decision Application Layer

The Smart Decision Application Layer is the core part of the DD_STM for realizing precise teaching and intelligent management. Based on the data processing and pattern discovery completed by the Intelligent Computing Layer, the Smart Decision Application Layer further utilizes the results of analysis and mining to provide teachers with scientific and precise teaching decision support, and to offer path guidance for students to meet their personalized learning needs.

The Smart Decision Application Layer, together with other layers, jointly establishes a closed-loop smart teaching system. Based on the learning analysis results provided by the intelligent computing layer, the Smart Decision Application Layer generates intuitive graphs, tables, reports and warnings for abnormal situations. This helps teachers quickly understand the main information in the data. It efficiently integrates data reports from learning analysis and transforms it into practical teaching action suggestions. This promotes the intelligence and humanization of education, and contributes to the construction of an orderly and efficient teaching environment.

2.2 System Functionality Design

The DD_STM architecture (as shown in Figure 2) supports real-time acquisition of multi-modal data, learning analysis and visualization, and teaching decision based on analysis results during programming course. Specifically, we implement the Four-layers DD_STM architecture described in the previous text by constructing a smart platform that consists of four subsystems, The smart platform is shown in Figure 3, and it contains four subsystems including AI assistant system, digital textbook system, programming practice management system and programming task resource system. The following is a detailed introduction to the specific functions and usage of the four subsystems.

AI Assistant System: It provides 24/7 theoretical learning and practical operation guidance. Students can ask questions at any time and receive immediate answers. Teachers utilize the smart platform to obtain learning analysis and feedback from the AI assistant, so as to grasp the learning dynamics of students. AI assistant responds to students' questions in real time and recommends programming tasks and network links to digital textbook systems related to the questions, which helps students consolidate their knowledge base and strengthen skills.

Digital Textbook System: It provides learning resources related to programming courses. In addition to text resources, it also provides a variety of media resources to help students understand abstract concepts in a more intuitive and accessible way, such as rich videos, animations and hands-on demonstrations.

Programming Practice Management System: It offers necessary training management and practice environment, supporting functions such as task management, task completion, error messages, and task completion data collection and status analysis. These functions enable teachers to assign practice tasks, receive data feedback, and adjust teaching strategies based on system feedback, and provide students with targeted guidance that meeting their needs.

Programming Task Resource System: It contains tasks of different difficulty levels and types, serving as a resource for teacher-assigned tasks, AI-assisted personalized recommendations, and students' self-directed practice. The task types include foundational consolidation tasks, advanced extension tasks, and targeted reinforcement tasks, which are aimed at strengthening basic knowledge, enhancing knowledge application abilities, and addressing skill gaps.

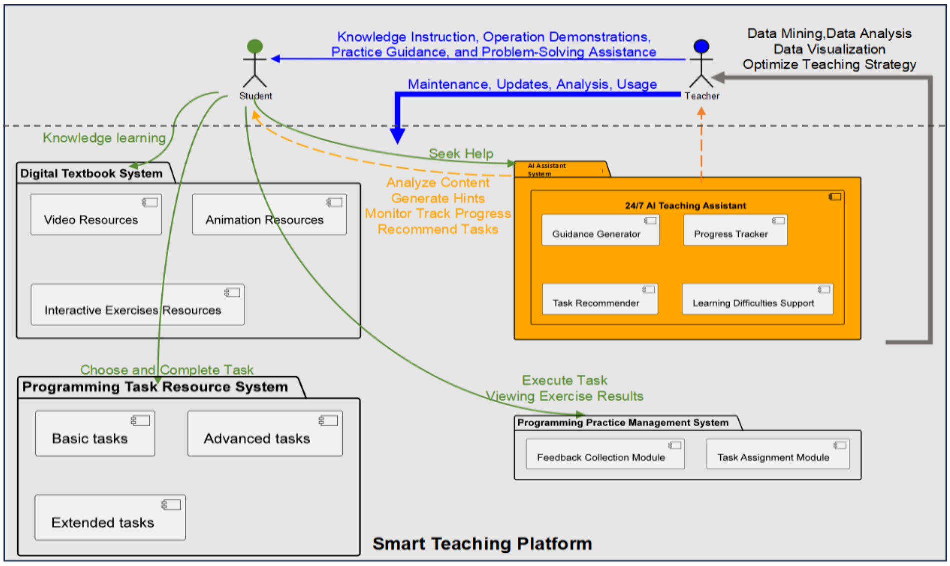

Figure 3 Composition of the smart teaching platform

Students and teachers are users of the smart teaching platform, teachers can use videos, animations and operational demonstrations from the digital textbook system to deliver theoretical knowledge, and utilize programming practice management system for training management. The resource development within the smart teaching platform is open to teachers, who can integrate actual teaching needs, course characteristics, and students' learning preferences into the platform. It enriches platform resources, and promotes the sharing and optimal allocation of high-quality educational resources. This approach ensures that the materials are both in line with the actual teaching situation, and meet the specific teaching needs of different teachers and courses.

Students can access the digital textbook system anytime and anywhere to study course content, retrieve tasks from programming practice management system, and complete programming tasks, including code writing, testing, and result submission within the system. When encountering programming difficulties, they can ask teachers or the AI assistant for immediate feedback. Additionally, students can choose tasks of different types and difficulty levels based on their learning situations to meet personalized learning needs, thereby increasing their sense of achievement and learning motivation.

The support of the smart platform restructures the traditional "single-thread" mode of teacher-student interaction in conventional teaching, and forms a "multi-thread" mode involving the triadic relationship of "teacher-smart platform-students". With the help of such new teaching mode, the DD_STM provides round-the-clock personalized real-time guidance and feedback through the smart teaching platform, thereby promoting efficient interaction between teachers and students, and making classroom teaching more flexible and efficient.

3 DD_STM Implementation and Application

In the smart teaching model, the smart platform forms the technical foundation for implementing intelligent teaching. The smart teaching platform adopts a Browser/Server (B/S) architecture, supporting login and usage by teacher and student users.

3.1 Usage of the Smart Teaching Platform

Students log into the system using their usernames and passwords. They program as required to complete the tasks assigned by teachers, and debug the code based on error reports and guidance from the AI assistant system. Students can view the feedback after submitting the task on the smart teaching platform. Data such as submission records, error logs, time spent on tasks, post-submission task analysis, evaluations, and scores are all collected and stored within the smart teaching platform. Figure 4 shows the programming practice interface for the student side on the smart teaching platform. On the left side of the figure is the list of programming tasks, and on the right side of the figure is the area for viewing and answering programming tasks.

ISSN: 0034-8082

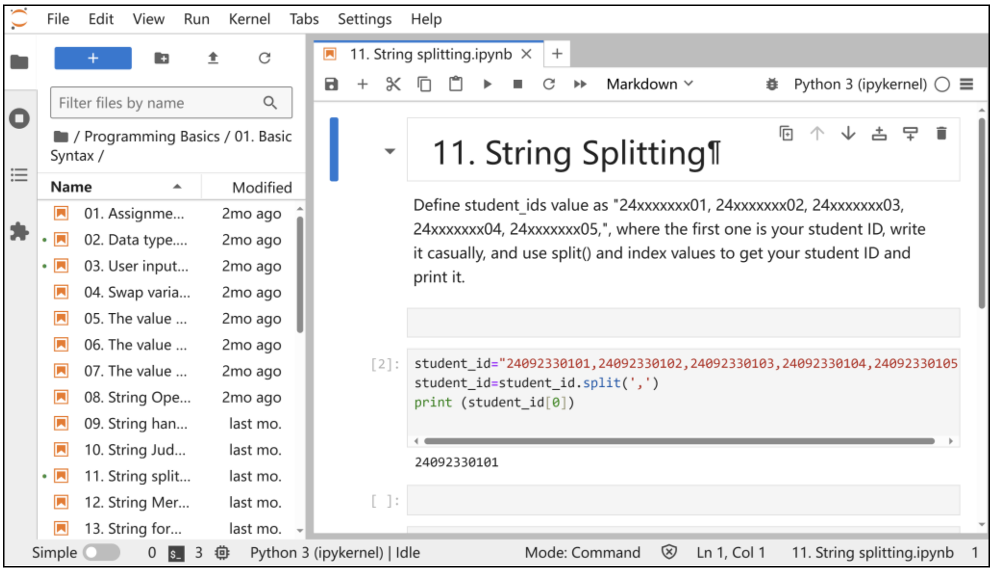

Figure 4 The programming interface for the student side

Teachers can also log into the platform with their usernames and passwords to build the functional modules of the system, configure and improve the resources, assign programming tasks, and view the results of the tasks. Figure 5 shows the teachers' interface for programming tasks management on the smart teaching platform. Teachers are able to obtain students' detailed code snapshots, and derive inspection and analysis results for individual students and the entire class, based on the analysis results. They can promptly adjust the teaching strategies according to the analysis reports and suggestions from the intelligent decision-making layer.

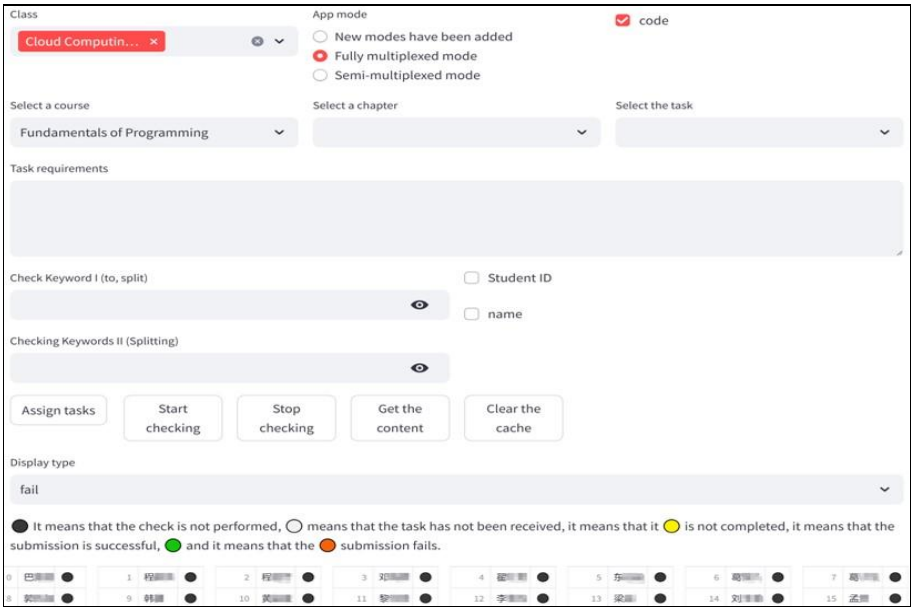

Figure 5 Teachers' interface for programming tasks management

ISSN: 0034-8082

The AI assistant system provides multi-threaded support for answering questions and guiding information, effectively empowering teaching and stimulating students' autonomous learning motivation. Figure 6 shows the AI assistant interaction interface on the smart teaching platform. The system user edits a question in the question input box and press Enter to submit the question, and then can view the answer provided by the AI assistant.

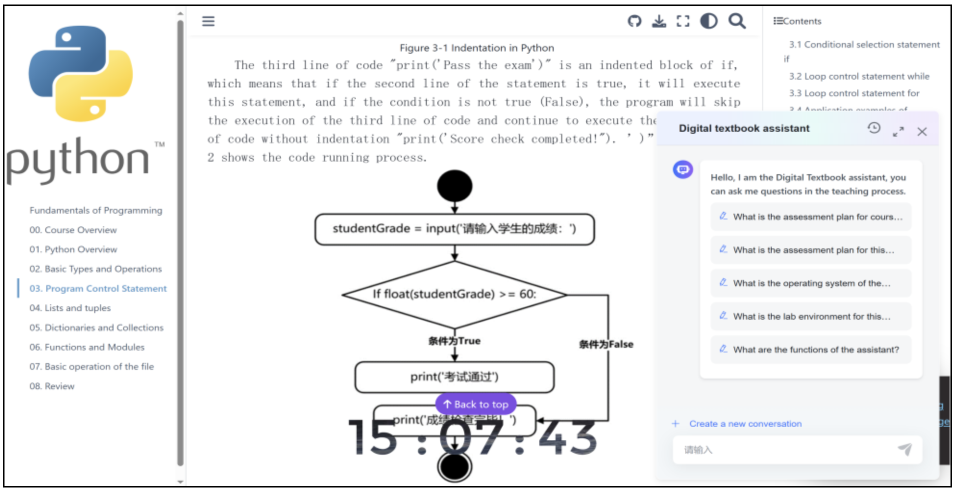

Figure 6 AI assistant interaction interface in digital textbook system

3.2 Collection and Storage of Learning Process Data

In the smart teaching model, the students' learning behavior data is stored in a persistent data storage format. Table 1 shows the main tables that store the students' learning behavior data in the database. The data collected throughout the students' programming process is stored, including objective data from the platform, process data of task completion, and the interaction data between students and AI assistant. The objective data from the platform includes the situation of task receipt, task completion, task submission, task scores, task duration and the number of times that tasks are resubmitted, while the process data of task completion contains task snapshots, task error reports and debugging logs. The collected data is continually stored into the database, which supports the subsequent analysis of both real-time and non-real-time students' learning behavior data.

Table 1 Information of the table for storing students' learning behavior data

ID	Table Name	Description
1	user_info	User information (e.g. name, ID num, identity)
2	tasks	Programming Task Information
3	task_check	Task inspection results, timestamps, etc.
4	task_snap	Snapshots of all students' practical programming tasks at a given moment
5	AI_interaction	Interaction information between students and AI

3.3 Intelligent Computing and Smart Decision Application

The smart teaching platform collects the real time behavior data of students throughout the entire process of programming. Through in-depth analysis of the students' behavior data, it provides objective data support for teachers to grasp students' current learning status, promote the teaching process, and optimize teaching implementation. It also provides timely feedback on students' practical training and autonomous learning, forming positive incentive measures, fully unleashing the potential of "teachers-platform-students" in the teaching field, and improving the teaching effectiveness. The following are the learning analysis and visualization strategies included in the smart teaching platform to support teachers' smart decision-making and teaching strategy adjustment.

3.3.1 Analysis of Students' Programming Task Completion

By analyzing and summarizing the data throughout the entire process of practical training tasks, class snapshots, duration statistics, debugging frequency analysis, code change rate analysis, current node average score analysis, corresponding to skill point programming exercises are obtained, and presented in visual form to assist teachers in understanding the completion progress and scores of students' current practical training tasks. Figure 7 shows the analysis and statistics interface of programming task completion from the teacher's perspective. We can quickly get information from the figure that two students failed to submit their tasks, as the submission status flags are marked in orange.

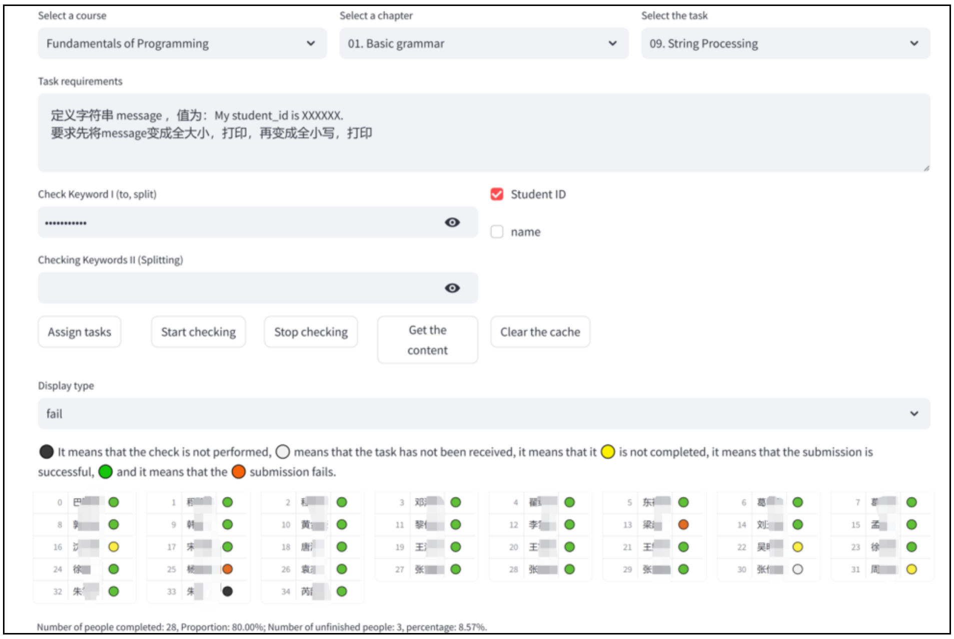

Figure 7 Analysis and statistics of programming task completion

3.3.2 Analysis of Programming Error Types

According to the statistical results, teachers can quickly identify high and low frequency programming errors among students. In such a way, teachers can quickly understand the main difficulties students encounter in the process of transforming their knowledge into programming code. Subsequently, teachers can conduct targeted key explanations, strengthen practice, and adjust strategies for resolving students' difficulties. Figure 8 shows the analysis of error types during the programming practice process. We can quickly obtain the information from the figure: the main error types are SyntaxError (45.7%), TypeError (21%) and NameError (17.3%).

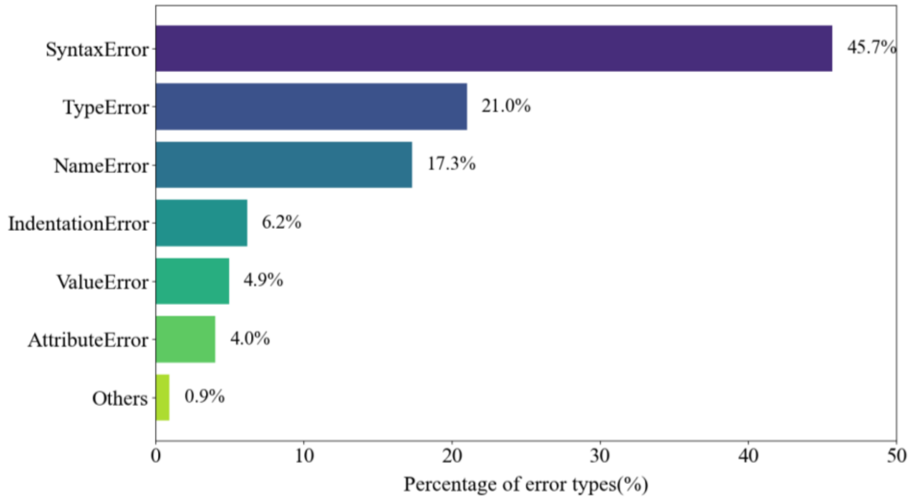

Figure 8 Analysis of error types during the programming practice process

3.3.3 Analysis of Students and AI assistant System Interaction Frequency and Interactive Content

The AI assisted system provides students with timely interaction and difficulty support for difficulties through the multi-threaded mode. Students can seek for help from the AI assisted system at any time during knowledge learning, programming practice, after-class consolidation exercises, and self-review. The AI assisted system also offers round-the-clock intelligent companionship for learning. To a certain extent, the frequency and type of interactions between students and the AI assistant reflect their understanding and application of knowledge, as well as the type of difficulties they encounter. By analyzing the interactive information, teachers can receive timely feedback on students' learning situations. Figure 9 shows the average frequency of student-AI interactions per time period within a class session, and Figure 10 presents a word cloud of the knowledge points involved in student-AI interactions. From figure 9, we can easily get that the highest interaction frequency occurs at 15 minutes, with a total of 43 interactions, and we can obtain information from figure 10 that during Python programming, the main problems that students encountered include the use of functions, for loops, dictionaries, and debugging syntax errors.

ISSN: 0034-8082

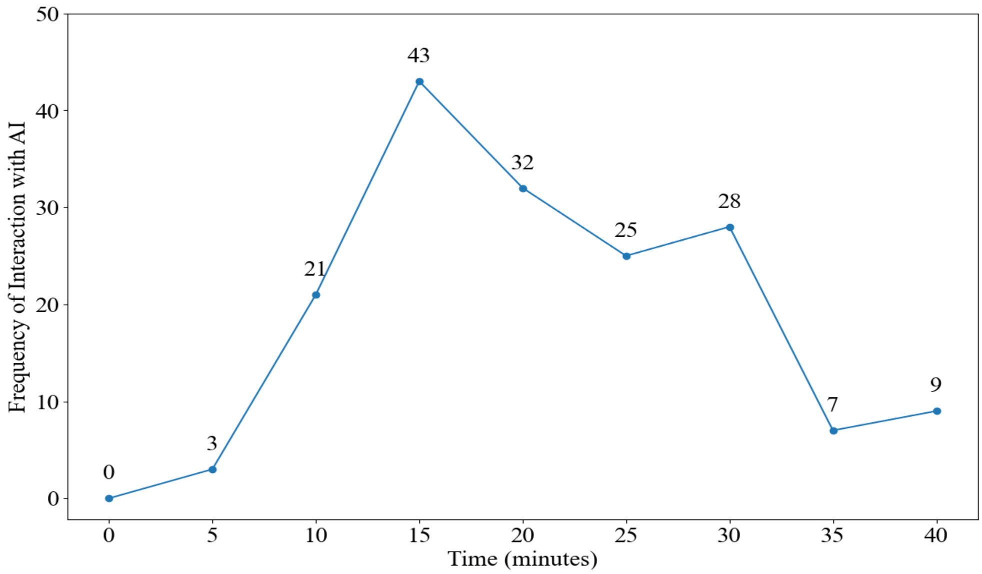

Figure 9 Average frequency of student-AI interactions per time period within a class session

Figure 10 Word cloud of the knowledge points involved in student-AI interactions

3.4 Empirical Analysis of the Programming Course

3.4.1 Samples

In this study, a comparative experiment was carried out in the teaching of the Basic Programming course for two first-year classes majoring in Cloud Computing Technology Application at a vocational university in Jiangsu Province, China. 46 students from Class A were selected as the experimental sample, and 48 students from Class B were selected as the control sample. The teaching duration was 12 weeks (one academic semester). The DD_STM was applied in Class A, while traditional teaching methods were adopted in Class B. That is, in Class B, the smart teaching platform was neither used to support data collection and analysis throughout the teaching process, nor was it used for data-driven decision-making. Additionally, the 24-hour intelligent learning companionship provided by the AI assisted system was not utilized.

In Class A, students learn programming theory through digital textbooks, complete programming exercises via the programming practice management system, and can interact with the AI assistant during their learning process. Teachers explain key concepts through the digital textbook system, manage practical training tasks via the programming practice management system, and use the smart platform to view students' comprehensive learning data and generate decision-making plans. Learning process data can be extracted from students' activities on the smart platform. Teachers can implement precise teaching interventions based on learning analysis provided by the platform, and learning outcomes are closely related to student engagement and effective teacher interventions.

The aim of this study is to enhance effective teacher-student interaction and data-supported decision-making of teachers. It provides supports and references for teachers to promptly understand students' learning status and perform effective teaching interventions, thereby improving teaching effectiveness.

3.4.2 Evaluation Metrics

This study employs two metrics, i.e. Learning Participation (LP) and INteractive Efficiency (INE), to evaluate the impact of a DD_STM on interactive efficiency and teaching quality. Students' learning participation has a positive effect on academic performance. The higher the interactive efficiency and learning participation, the better the academic performance.

The specific formula for calculating interactive efficiency (*INE*) is as follows:

$$INE = \frac{I_{effect}}{I_{total}} \exp\left(\frac{n}{t}\right) F\left(IN_{context}, IN_{feedback}, IN_{noise}\right) \tag{1}$$

where I_{effect} and I_{total} represent the effectively communicated information and total expected conveyed information, respectively, t stands for the time taken to resolve the questions, $F(IN_{\text{context}}, IN_{feedback}, IN_{noise})$ denotes the impact of other factors on interactive efficiency, which is a function of the communication environment, feedback and noise interference, and n is a positive integer representing the number of concurrently asked questions from students.

In this study, the value of I_{effect} for each question posed by one student can be 0, 0.5, or 1, where 0 indicates that the question was neither understood nor resolved, 0.5 indicates that the question was understood but not resolved, and 1 indicates that the question was both understood and resolved. The value of I_{total} for each question posed by a student is set as 1. To maintain consistency in interaction before and after the introduction of the smart teaching model, $F(IN_{\text{context}}, IN_{feedback}, IN_{noise})$ is set to be 1. Therefore, the key factors affecting interactive efficiency are the time invested in understanding and solving problems. The more time invested in understanding and resolving an issue during teaching interactions, the lower the

ISSN: 0034-8082

interactive efficiency, and vice versa.

Under ideal circumstances, the time taken by the AI assisted system and the teacher to solve the same problem is approximated to be the same, setting as *T*. For solving multiple problems at the same moment, the AI teaching assistant can process them concurrently, so the consumed time is always *T*, and the number of solved problems depends on the actual number of concurrent questions. When communicating with students, regardless of the number of questions they ask problems, the teacher can only solve one problem at a time. That is, when students ask the same type of problems concurrently, the interactive efficiency between the teacher and the student is always a constant. Table 2 provides a detailed comparison of the interactive efficiency between the smart and traditional teaching model under ideal conditions. Figure 11 illustrates a qualitative analysis of the idealized theoretical curve for comparing the interactive efficiency between the smart and traditional teaching model.

Table 2 Comparison of interactive efficiency between the smart teaching model and traditional teaching model in solving a specific problem under ideal conditions

Concurrent Questions Asked	Smart Teaching Model	Traditional Teaching Model
1	e(1/T)	e(1/T)
2	e(2/T)	e(1/T)
3	e(3/T)	e(1/T)
…	…	…
N	e(N/T)	e(1/T)

By calculating the *INE* values in the teacher-student single-thread interactive model and the teacher-platform-student multi-threaded interactive model, we can achieve a comparison of *INE* between smart and traditional teaching models. According to the analysis of Table 10 and Figure 11, it can be concluded that for a certain type of question, the *INE* in the smart teaching model significantly improves as the number of concurrent questions increases. In contrast, the *INE* in the traditional teaching model is limited by the human resources of the teacher, resulting in a noticeably lower processing efficiency compared with that in the smart teaching model.

ISSN: 0034-8082

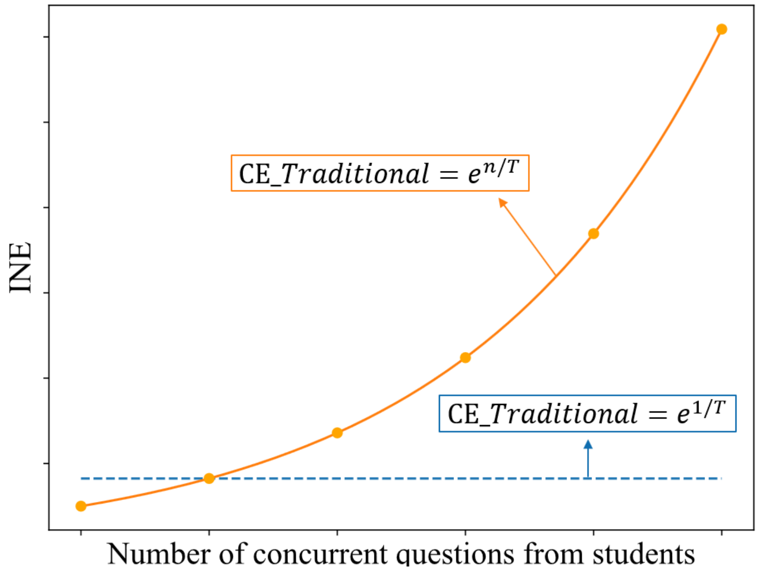

Figure 11 *INE* in solving a specific problem under ideal conditions

The specific calculation formula for learning engagement is as follows:

$$LP = \frac{\sum A_i W_i}{N} \quad (2)$$

where A_i represents the participation mode in learning, which can be any indicator reflecting the students' behavioral activities, such as answering questions, taking quizzes, and completing practical training exercises, W_i is the corresponding weight, and N denotes the total number of all indicators for calculating the learning participation degree. In this study, in order to maintain the consistency of the evaluation before and after the introduction of the smart teaching platform, the behavior patterns involved in learning in both traditional and smart teaching models are knowledge quizzes and practical training exercises, with a weight of 50% for each mode.

3.4.3 Application Results and Analysis

The course "Fundamentals of Python Programming" spans 12 weeks, covering topics such as basic Python syntax, string types, numeric types, program control structures, and composite data types. Throughout the course, 30 in-class programming assignments and 8 knowledge quizzes were assigned. Students completed programming exercises and knowledge tests through a smart teaching platform during class hours. After class, students accessed and utilized educational resources on the platform based on their personalized learning needs.

In the previous text, we defined and theoretically analyzed the teacher-student interactive efficiency for both smart and traditional teaching models under ideal conditions. In this subsection, we simplify the analysis of interactive efficiency to interaction frequency, given that the smart teaching platform can concurrently handle various issues, whereas classroom interactions primarily aim to resolve student queries. This simplification is reasonable and sufficiently reflects differences in interactive efficiency.

Figure 12 presents the interactive number between students and teachers in traditional teaching model, as well as that between students, teachers, and the AI teaching assisted system in smart teaching model. By comparing the average interactive number per session over 12 weeks (one class per week, each consisting of four 40-minute sessions) for Class A and B, it was found that in the traditional model the bottleneck in teacher human resources limits the frequency of interaction, with the number of teacher-student interactions stabilizing at an average of 54 times. In the smart teaching model, AI teaching assistant significantly enhances the efficiency of teaching interaction, increasing the average number of interactions to 386 times, which is more than seven times that of the traditional teacher-student interaction model. As observed from Figure 12, it is particularly noteworthy that as the course progresses, the complexity of learning content and the comprehensiveness of programming tasks keep increasing, and correspondingly, the number of personalized problems is also increasing. The support provided by the AI teaching assistant became increasingly apparent, resulting in a remarkable improvement in interactive efficiency within the teaching process.

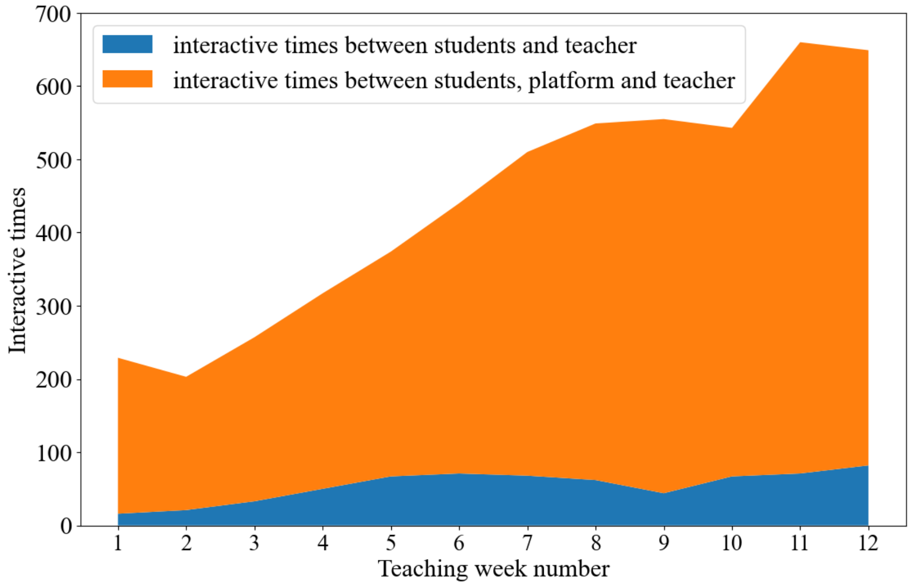

Figure 12 Interactive times in smart teaching model and traditional teaching model

Figures 13 and 14 compare students' participation in knowledge tests and practical training tasks between traditional and smart teaching models. Students' engagement in course learning to some extent reflects their enthusiasm for learning. By comparing the specific performance of this metric, it is evident that students show higher participation rates in both knowledge learning and programming practical training in the smart teaching model. In the traditional teaching model, the average participation rates in knowledge tests and programming practice tests were 91% and 83%, respectively. In the smart teaching model, these rates were 98% and 96%, respectively. It shows an increase of 7% in participation for knowledge tests and 15% for programming practice.

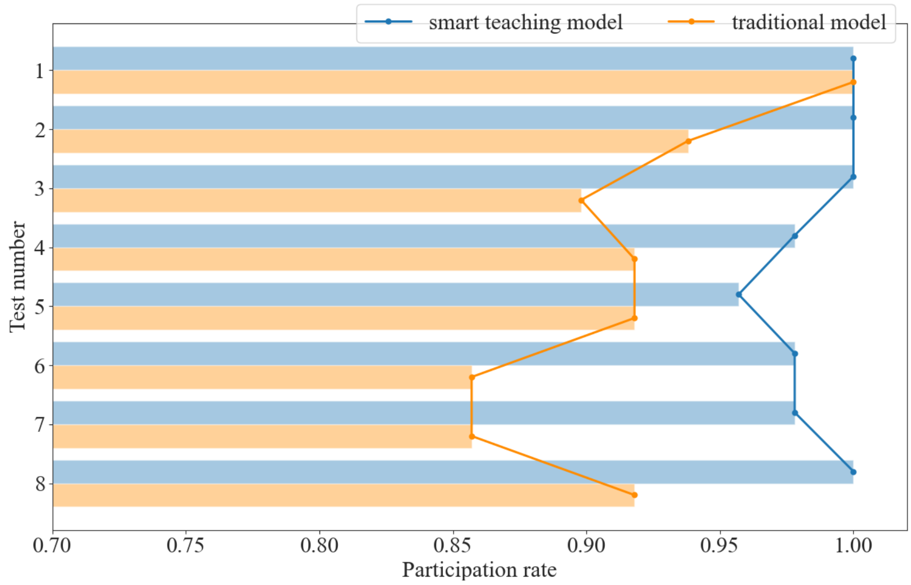

Figure13 Participation rate of knowledge test

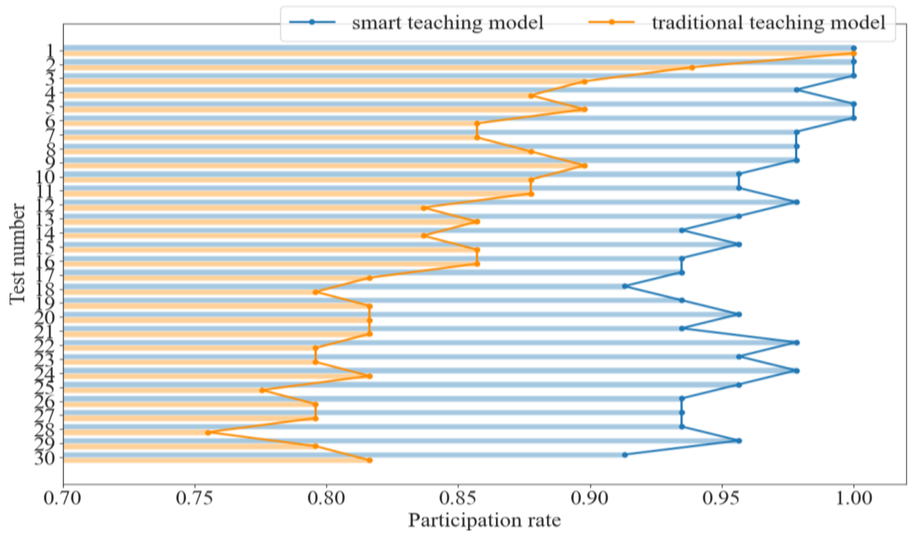

Figure 14 Participation rate of practical test

Additionally, Figure 15 analyzes data on student self-directed learning rates after class from the smart teaching platform. According to the data, 85% of students engage in self-directed learning after class. It indicates that many students continue to use the smart teaching platform for knowledge learning and practical exercises after class, demonstrating that the smart teaching platform effectively supports students' after-class self-directed learning. In contrast, traditional teaching models offer insufficient support for such learning due to a lack of relevant resources and platforms.

ISSN: 0034-8082

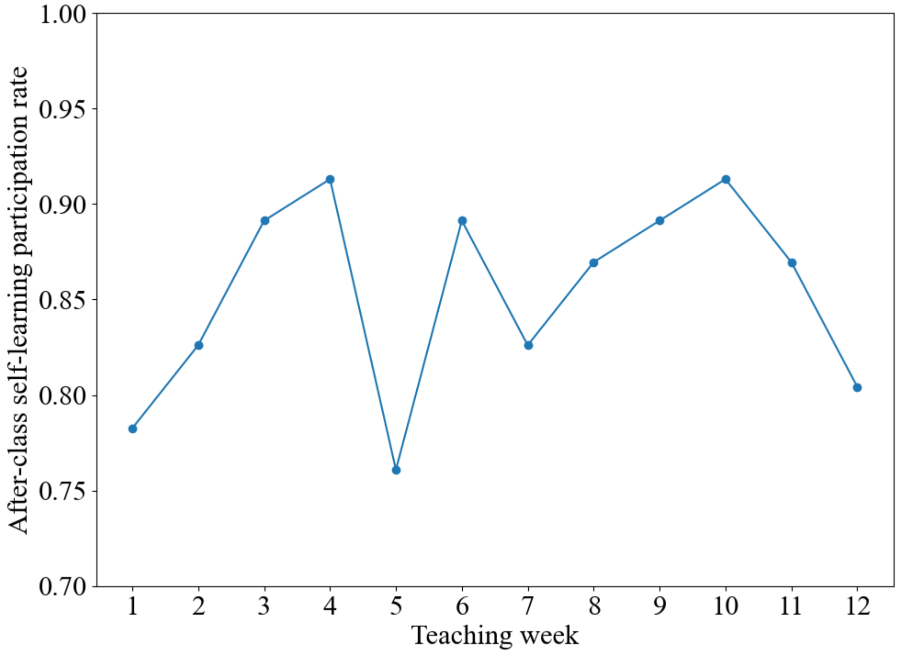

Figure 15 Student engagement in self- learning after class

4 Conclusion

With the launch and implementation of China's educational digitalization strategy, research on data-driven smart teaching has gradually deepened. This study presents a meaningful study on the implementation of digital transformation in education. We construct a four-layers DD_STM comprised of teaching interaction layer, data acquisition and storage layer, intelligent computing layer, and smart decision application layer. An intelligent education platform is introduced as the third element in the teaching environment, forming a collaborative paradigm among teachers, smart teaching platform, and students. An empirical study in Foundations of Python Programming validates this model. The results show that the proposed DD_STM helps to improve the participation rate of students and the interaction efficiency between teachers and students, thereby indirectly enhancing teaching quality. In the empirical study of Foundations of Python Programming, the application of this model increased students' engagement, specifically a 7% increase in knowledge learning engagement and a 15% increase in programming practice engagement. In addition, the interactive efficiency was improved to seven times that of traditional models (simplified as interaction frequency).

The main contributions of this study include: (1) Proposing a new architecture for a smart teaching model, including a teaching interaction layer, a collection and storage layer, an analysis and mining layer, and a smart decision-making layer, based on an analysis of existing problems in programming course instruction. (2) Developing a smart teaching platform containing four core subsystems—a digital textbook system, a task repository system, a practical training system, and an AI teaching assistant system—based on the four-layers architectural model. These subsystems complement each other, achieving a closed-loop from multi-source heterogeneous learning analysis to teaching strategy generation. (3) Presenting a formula for calculating interactive efficiency, which can be used to quantitative comparison of teacher-student interaction efficiency between smart teaching model and traditional teaching model.

This study offers valuable references for achieving higher levels of digitization and intelligence in the field of education, especially in the field of STEM (Science, Technology, Engineering, and Mathematics) education. However, to make the smart teaching model more universal, the smart teaching platform's data collection and parsing could be more sufficient. Moreover, dimensions for educational data analytics could be further enriched.

Acknowledgements

This work is supported by the following projects: The 2023 Provincial Education Science Planning Project titled “Research on the Construction of the Learning Support System for Self-study Examinations” (K-A/2023/09). Research Project of Jiangsu Province Higher Education Teaching Reform in 2023 titled “Research and Practice on the Smart Teaching Mode of Practical Training Courses in Higher Vocational Colleges” (2023JSJG230). Research Topic on China Association of Higher Education’s 2024 Annual Higher Education Scientific Research Planning Project titled “Research on Application Innovation and Efficiency Optimization Path of Artificial Intelligence Technology Empowering Higher Education Self-examination” (24ZK0304), and the Qinglan Project of Jiangsu Province of China.

References

[1] Ben Williamson, Rebecca Eynon. : Historical threads, missing links, and future directions in AI in education, Learning, Media and Technology, **45**(2020), 223-235.

[2] Neil Selwyn. Ed-Tech Within Limits. : Anticipating educational technology in times of environmental crisis, E-Learning and Digital Media, **18**(2021), 496-510.

[3] Hwang G J , Xie H , Wah B W , et al. : Vision, challenges, roles and research issues of artificial intelligence in education, Computers and Education: Artificial Intelligence, **1**(2025), 3-24.

[4] Solis Tiana, Secules Stephen, Impact of Undergraduate Teaching Assistants (UTAs) on gender-inclusive student engagement in an introductory computer programming course, (2024), 47566.

[5] Lee Changyoon, Myung Junho, Han Jieun, et al. : Learning from teaching assistants to formulate subgoals for programming tasks: exploring the potential for AI teaching assistants, CEUR Workshop Proceedings, **3840**(2024), 1-9.

[6] Cirigliano Matthew M., Guthrie Charles D., Pusic Martin, V. : Click-level learning analytics in an online medical education learning platform, Teaching and Learning in Medicine, **32**(2020), 410–421.

[7] Maria Zotou, Efthimios Tambouris, Konstantinos Tarabanis. : Data-driven problem based learning: enhancing problem based learning with learning analytics, Educational Technology Research and Development, **68**(2020), 3393-3424.

[8] El Alfy, S., Gómez, J. M., et al. : Exploring the benefits and challenges of learning analytics in higher education institutions: a systematic literature review , Information

Discovery and Delivery, **47**(2019), 25–34.
[9] Ong Sharmaine Gek Teng ,Quek Gwendoline Choon Lang.: Enhancing teacher–student interactions and student online engagement in an online learning environment, Learning Environments Research, **26**(2023), 681-707.
[10] Ilić, M., Mikić, V., Kopanja, L. et al.: Intelligent techniques in e-learning: a literature review, Artificial Intelligence Review, 56(2023), 14907–14953.
[11] Xundiao Ma, Yueguang Xie, Xin Yang, ey al. : Structural model construction and analysis for teacher-student interaction in smart classroom based on the development of higher-order thinking, Education and Information Technologies, **29**(2024), 21691-21717.
[12] Rienties, B., Nguyen, Q., Holmes, W., et al. : K. A review of ten years of implementation and research in aligning learning design with learning analytics at the Open University UK, Interaction Design and Architecture(s), **33**(2017), 134–154.
[13] Foster Ed, Siddle Rebecca. : The effectiveness of learning analytics for identifying at-risk students in higher education, Assessment & Evaluation in Higher Education, **45**(2020), 842–854.
[14] Zhang Y, Zhang H. : The development of intelligent education and teachers' professional informatization led by an intelligent ubiquitous network, Comput Intell Neurosci. **2022**(2022), 36248945.
[15] Alkhatlan A, Kalita J.: Intelligent tutoring systems: a comprehensive historical survey with recent developments. International Journal of Computer Applications, **181**(2019), 1–20.
[16] Mousavinasab E, Zarifsanaiey N, R Niakan Kalhori S, rt al.: Intelligent tutoring systems: a systematic review of characteristics, applications, and evaluation methods. Interactive Learning Environments, **29**(2021), 142–163.
[17] Pokrivcakova S.: Preparing teachers for the application of AI-powered technologies in foreign language education, Journal of Language and Cultural Education, **7**(2019), 135-153.
[18] Xu, W., Ouyang, F.: The application of AI technologies in STEM education: a systematic review from 2011 to 2021, International Journal of STEM Education, **9**(2022), 59.
[19] Richard M. Ryan ,Edward L. Deci.: Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being, American Psychologist, **55**(2000), 68-78.
[20]Achuthan Krishnashree, Kolil Vysakh Kani, Muthupalani Sharanya, et al.: Transactional distance theory in distance learning: past, current, and future research trends, Contemporary Educational Technology, **16**(2024), ep493.
[21] Xiao Jun, Chen Mo, Yang Yule, et al.: An exploratory multimodal study of the roles of teacher-student interaction and emotion in academic performance in online classrooms, Education and Information Technologies, 2025. https://doi.org/10.1007/s10639-025-13426-7, preprint.
[22] Wang Chunchao, Wang Yao, Xiao Aiping, et al.: Student development in teacher–student interaction: evidence from a randomized experiment in online education, Journal of Public Economics, **242**(2025), 105285.
[23] Du X, Yang J, Hung J-L, Shelton B.: Educational data mining: a systematic review of research and emerging trends, Information Discovery and Delivery, **48**(2020), 225–236.
[24] Shan Wang, Fang Wan, Zhen Zhu, et al.: Artificial intelligence in education: A systematic literature review, Expert Systems with Applications, **252**(2024), 124167.

ISSN: 0034-8082

Authors and Affiliations
Yao Wang, Xiao-Han Long, Liang Gao, Xing-Zhi Chang*.
Xing-Zhi Chang*
changxz@163.com
Yao Wang
wangyao@czcit.edu.cn
Xiao-Han Long
longxiaohan@czcit.edu.cn
Liang Gao
gaol4265@163.com
School of Software and Big Data, Changzhou College of Information Technology, Changzhou, China

Yao Wang is currently a lecturer at School of Software and Big Data, Changzhou College of Information Technology. Her research interests include the application of learning analysis and artificial intelligence in vocational education, and cloud computing technology in education.

Xiao-Han Long is currently a lecturer at School of Software and Big Data, Changzhou College of Information Technology. His research interests include the design and implementation of digital textbook, and automatic code judge system in programming learning.

Liang Gao is a currently lecturer at School of Software and Big Data, Changzhou College of Information Technology. He once worked at Huawei Technologies Co., Ltd. His research interests include the application of automatic code judge system, and cloud computing technology in education.

Xing-Zhi Chang is a professor at School of Software and Big Data, Changzhou College of Information Technology. He earned his Ph.D. from Northeastern University. His research focuses on innovative applications of new generation information technology in education, especially the design and implementation of artificial intelligence in education.