Educational Process Mining for Discovering Students' Problem-Solving Ability in Computer Programming Education

Fang Liu, Liang Zhao, Jiayi Zhao, Qin Dai, Chunlong Fan, and Jun Shen, *Senior Member, IEEE*

***Abstract*—Educational process mining is now a promising method to provide decision-support information for the teaching–learning process via finding useful educational guidance from the event logs recorded in the learning management system. Existing studies mainly focus on mining students' problem-solving skills or behavior patterns and intervening in students' learning processes according to this information in the late course. However, educators often expect to improve the learning outcome in a proactive manner through dynamically designing instructional strategies prior to a course that are more appropriate to students' average ability. Therefore, in this article, we propose a two-stage problem-solving ability modeling approach to obtain students' ability in different learning stages, including the pre-problem-solving ability model and the post-problem-solving ability model. The models are trained with Gradient Boosting Decision Tree (GBDT) on the historical event logs of the prerequisite course and the target course, respectively. With the premodel, we establish the students' pre-problem-solving ability profiles that reflect their average knowledge level before starting a course. Then, the instructional design is dynamically chosen according to the profiles. After a course completes, the post-problem-solving ability profiles are generated by the postmodel to analyze the learning outcome and prompt the learning feedback, in order to complete the closed-loop teaching process. We study the modeling of coding ability in computer programming education to show our teaching strategy. The experimental results show that the generalizable problem-solving ability models yield high classification precision, while most students' abilities have been significantly improved by the proposed approach at the end of the course.**

***Index Terms*—Educational process mining (EPM), Gradient Boosting Decision Tree (GBDT), problem-solving ability model, student ability profile, two-stage modeling.**

Manuscript received 18 July 2021; revised 16 September 2022, 12 October 2022, and 17 October 2022; accepted 17 October 2022. Date of publication 25 October 2022; date of current version 14 December 2022. This work was supported in part by the Research Project on Undergraduate Teaching Reform of Liaoning Province under Grant 2022-160 and in part by the Undergraduate Education Reform Funding of Shenyang Aerospace University under Grant JG202202C. *(Corresponding author: Liang Zhao.)*

Fang Liu, Liang Zhao, and Chunlong Fan are with the School of Computer Science, Shenyang Aerospace University, Shenyang 110136, China (e-mail: liufang@sau.edu.cn; lzhao@sau.edu.cn; fanchl@sau.edu.cn).

Jiayi Zhao was with the School of Computer Science, Shenyang Aerospace University, Shenyang 110136, China. He is now with Tencent Technology Co., Ltd., Shenzhen 518052, China (e-mail: mikusakula@gmail.com).

Qin Dai is with the College of Information, Shenyang Institute of Engineering, Shenyang 110136, China (e-mail: daiqin@sie.edu.cn).

Jun Shen is with the School of Computing and Information Technology, Faculty of Engineering and Information Sciences, University of Wollongong, Wollongong, NSW 2522, Australia (e-mail: jshen@uow.edu.au).

Digital Object Identifier 10.1109/TLT.2022.3216276

I. Introduction

THE use of a learning management system (LMS) has grown exponentially in recent years for both online learning courses and blended-learning courses [1], [2]. The LMS has many advantages including freedom of learning and practicing, and collecting data on all student activities at different levels of granularity [2], [3]. Besides their traditional classrooms, most universities nowadays use the LMS, and the exemplar platforms are Canvas [4] and EDx [5]. With the LMS, teachers can flexibly assign tasks and give feedback to students for their whole learning processes performed online. On the other hand, students can have individual practices and complete assignments and experiments anywhere and anytime.

The aforementioned features provide a convenient facility for the teaching–learning process, whereas the LMS still suffers significant drawbacks and limitations. At present, the LMS mainly focuses on student learning and task arrangement, which means that it is mostly used as a teaching assistant tool [6]. Besides a learning platform, it is also a database storing the students' learning behavior data online that can be transformed into decision-support information for the teaching–learning process. This information can help educators to formulate a course's instructional design for each semester that is adaptive to specific students. The instructional design, taken consideration as in a generic term, is the description of the educational process, e.g., the teaching method of the learning contents and how to organize the learning activities [7].

In the LMS, the event logs of the learning process can reveal students' problem-solving processes and abilities, which is suitable for most computer programming courses. The problem-solving process involves the analytical and interactive aspects to find an appropriate solution or approach to reaching the desired goals [8], [9]. The problem-solving ability refers to learners' learning outcome and performance, expressed as the quality and proficiency of completion, e.g., how long it takes and how many times it has been submitted. This revelation can be achieved with the process mining (PM) technique [10], [11], adopting these logs to discover, monitor, and improve the educational processes. More specifically, educational process mining (EPM) is the application of PM to the raw educational data to discover the information helpful for exploring the principles of teaching and learning technologies that indicate the relationship between various factors in education and the trend of pedagogical development [1], [12], [13]. This article follows the EPM to model students' problem-solving ability and, then, generates their cohorts and individual ability profiles, expressed as high or low. Moreover, we also study how to make the dynamic learning design and intelligent learning output analysis according to the ability profiles.

Several studies have proposed to extract knowledge from the event logs recorded by the LMS with the EPM technology [2], [12], [14], [15], [16], [17], [18], [19]. They aim to understand the factors that influence skills acquisition and create models for the students' learning patterns or problem-solving skills to evaluate the learning outcomes [2], [8], [10], [17], [18]. The existing LMS usually provides logs recording the student attributes, course attributes, browsing behaviors, and some statistical information [2], [8]. With the third-party plugins, some studies can analyze the low-level interactions (e.g., keyboard key presses, mouse movements, etc.) [12], [15], [16], [17], [18] and the high-level interactions of students' learning behavior (e.g., all the commands issued at Integrated Development Environment (IDE) level [15], [16], [17]. Tóth et al. [8] applied the PM method to students' behavior data to analyze their problem-solving patterns and skills. Vatutin et al. [20] checked and analyzed the solutions of students' home, control, and midterm work and used machine learning methods to access students' mathematical knowledge and problem-solving abilities. Some studies [15], [16], [17] mined learners' coding behavior to evaluate and improve their coding abilities. The intelligent tutoring system (ITS) is also a computer-aided system to model learners' psychological states to improve their learning outcomes [21], [22], [23]. It helps learners to acquire domain-specific knowledge and provide feedback about the correctness of their responses. ITS researchers have adopted a variety of student modeling approaches to detect and correct the individual misconception in exercise. With the above mining results, educators can check students' learning status and carry out interventions during a course, but these can only be conducted in the latter phase or at the end of the course after the learning behaviors are complete. Indeed, educators prefer to obtain learners' actual ability in advance to build a more suitable instructional design prior to a course, improving the learning outputs actively. The existing methods are difficult to meet such requirements.

Owing to the above complexity, a one-shot design is infeasible. One key challenge is how to acquire students' problem-solving ability prior to a course. The curriculum system of education is a rigid whole, and the students' ability cultivation is a continuous process. The event logs of the prerequisite courses can reflect students' problem-solving ability before a follow-up course [24], [25]. With the course relevance, this article proposes a two-stage modeling strategy to mine students' problem-solving ability, implemented before and after a course. The two models are used for guiding the instructional design and assessing the learning outcomes, respectively. In the ability modeling stage of a course, students' preability is modeled according to its prerequisite course's activity process data [26], while the postability is modeled according to the present logs. The models are built with the machine learning algorithm Gradient Boosting Decision Tree (GBDT) [27], [28]. All the event logs that can obtain from the online learning platform are helpful for modeling the models. In our opinion, the logs about students' quality of task completion can better reflect students' ability, e.g., grades of quizzes and assignments, time to complete the question, quality of question completion, completion proportion of assignments, etc. In the model application stage, the pre-problem-solving ability profiles generated by the premodel are used to guide the instructional design prior to a course. The postability profiles are used to access the learning outcomes at the end of the course.

The main contributions of this article are as follows.

1) We employ the activity process data on the LMS and the EPM to mine students' problem-solving ability for a course. The analysis results can reflect students' problem-solving ability and provide effective decision-support information for the course's learning.
2) We propose a two-stage problem-solving ability modeling strategy to finely design and evaluate the learning process in the full use of the course relevance in the curriculum system.
3) The students' preability profiles are used to plan an appropriate instructional design prior to a course. Moreover, the students' postability profiles are employed for learning outcome analysis and learning feedback to form a closed-loop learning process.

The rest of this article is organized as follows. In Section II, we review the related work. The educational process based on the LMS is introduced in Section III. Dynamic instructional design based on students' problem-solving ability profiles is described in detail in Section IV. In Section V, experimental results are given and analyzed. Finally, Section VI concludes this article.

II. Related Work

A. Educational Process Mining

EPM is to extract knowledge from event logs recorded by the LMS, massive open online courses (MOOCs), or the ITS. It focuses on the development of a set of intelligent tools and techniques aimed at extracting process-related knowledge from event logs [15], [18], [29]. The application of EPM can be divided into five dimensions, which are about discovering learning behavior patterns, predicting the trend of learning outcome, improving teaching evaluation and feedback, providing teaching decision support, and improving education management service. Trcka et al. [11] converted students' examination records of multiple courses into event logs to analyze elective courses' learning paths. Mukala et al. [30] used a fuzzy mining algorithm to extract students' learning patterns in MOOCs. It has been found that the learning behavior pattern of unsuccessful students is poor and unpredictable, while the behavior pattern of successful students is generally similar. Pechenizkiy et al. [31] applied process discovery and other technologies to analyze online multiple-choice question data and evaluate feedback on the trend of students' answering behavior. Cairns et al. [32] used conformance checking to analyze the fitness degree between the employees' training paths and the established curriculum constraints. Anuwatvisit et al. [33] used conformance checking to detect discrepancies between the flows prescribed in a students' registration model and the actual process instances.

B. EPM and Problem-Solving Ability

Tóth et al. [8] described how to extract useful information from event logs and applied the most appropriate PM method to discover the learning process model. It used visualization, clustering, and classification method to understand students' problem-solving skills. Vatutin et al. [20] used machine learning and data mining methods to assess learners' mathematical problem-solving skills. They aimed to use computational methods in educational assessment. Mayilvaganan and Kalpanadevi [34] evaluated student skills on problem-solving resources that were classified using the naive Bayes technique based on predefined rules. It modeled students' cognitive skills based on student attributes, domain value of specialties, and parents' qualifications. Ardimento et al. [17] adopted the Fuzzy-based PM techniques to model and study the developers' coding process. Real et al. [2] aimed to adopt the EPM techniques to mine the students learning paths in an *Introductory Programming* course. The analysis of these results provided general and specific information on students' learning paths and can help teachers observe students' behavior patterns. Ardimento et al. [16] used conformance checking to test coding behaviors from event logs generated from IDE usage and studied how developers carry out coding activities and what hurdles they usually face. Etinger et al. [35] attempted to uncover and analyze the patterns of behavior performed by students with higher scores in contrast to those with lower scores to understand the online course usage patterns and their relationship with learning outcomes.

C. Gradient Boosting Decision Tree

The Decision Tree is a supervised machine learning technique aiming at the classification of instances. Some studies used Decision Tree for classification or attribute selection in PM. Horita et al. [36] proposed an approach for business goal achievement prediction using Decision Tree based on the event logs in an information system. Rozinat et al. [37] adopted Decision Tree to analyze how data attributes influence business process' choices based on past process executions in an information system.

GBDT is an iterative Decision Tree algorithm, which is composed of multiple Decision Trees, and the results of all trees are agglomerated to make the final answer [27], [28]. GBDT produces competitive, highly robust, and interpretable procedures for both the regression and classification problems. It is a highly effective and widely used machine learning method, considered a robust generalization algorithm together with support vector machine [38], [39] when was first proposed. GBDT has been demonstrating state-of-the-art results on many standard classification benchmarks [28], [40]. The performance suggests that we employ GBDT to model students' problem-solving abilities in this article.

D. Summary of Existing Literature

The above approaches focus on analyzing the interaction data to mine users' problem-solving ability or learning patterns in a course or the learning process. Differently, our approach models the students' problem-solving ability before and after a course that can assist to build a more appropriate instructional design and cast insight how to improve their learning outcome. Furthermore, our method is suitable for most Science, Technology, Engineering, and Mathematics courses carried out on the LMS, if there are logs that can reflect students' special learning actions.

III. Educational Process Based on the LMS

The LMS provides a convenient and new way for teaching and learning. In this article, we study how to carry out learning activities based on the system to take full advantage of it.

A. Support of LMS on Learning

Higher education's core teaching objective is to acquire the thinking skills and ability to solve problems. The key to such an ability acquisition is the sufficient quantity and quality of practices. The LMS provides solid support for practical ability training in learning that includes online assignments, online experiments, online examinations, score summaries, and other course management functions. Online experiments are similar to online assignments, i.e., the publishing and submission of experimental topics are completed on the LMS platform and the operation process can conduct offline. Furthermore, the systems also have an online judgment model to automatically judge and also score the answers to the objective questions and other types of questions in the assignments, experiments, and examinations. This can eliminate the bottleneck of teachers' limited time to assess students' work. The support of LMS for learning is illustrated in Fig. 1.

B. Statistics Indexes of the Programming Submission Process in Course Grading (CG)

Computer programming education's core teaching objective is to acquire the problem-solving ability of coding. In the following, we take the *CG* [41] platform as the LMS and the coding ability to explain our EPM and corresponding learning strategy. The platform has a complete curriculum management system, which provides an excellent management platform for specialized courses in computer programming education. Moreover, it covers the training of problem-solving ability and system ability of programming with less burden on teachers.

The *CG* system has an online programming judgment module that can generate detailed statistics on the tasks' completion process, including the work activity data and the answer detail data. The logs provided by different platforms may vary, in which they reflect the students' coding process and ability. Work activity data figure out statistics on all tasks completed by each student, such as *Number of submitted questions* and *Right questions*. The detailed index definition and value range of work activity data are shown in Table I.

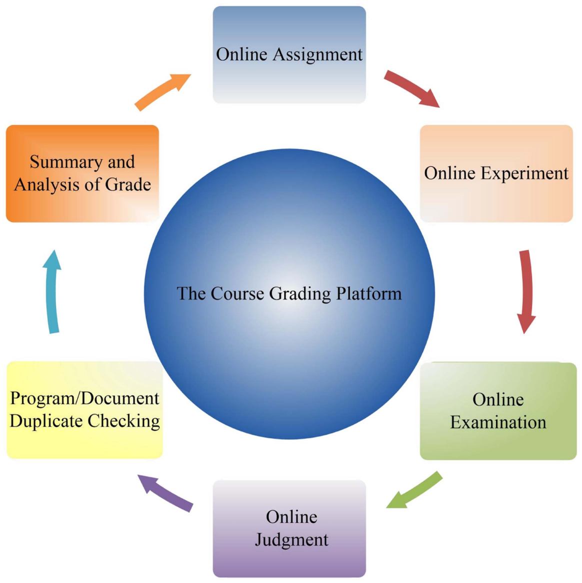

Online Assignment
Online Experiment
Online Examination
Online Judgment
Program/Document Duplicate Checking
Summary and Analysis of Grade
The Course Grading Platform

Fig. 1. Learning process with the LMS for a course.

The answer detail data records the completion process of each question by each student, such as *Pass or not*, *Completion time*, and *Number of submissions for AC* (accept, meaning that the program passes the test cases). The detailed index definition and value range of the answer detail data are shown in Table II.

The indexes in Tables I and II record the overall completion and specific process of students' programming behaviors in detail, giving comprehensive feedback on students' coding ability. Herein, we take the following two cases as examples: students can skillfully and quickly complete the program and pass the test, and the other students need a long time to code and submit many times to pass. For the scores, there is no difference between the two types of students. Nevertheless, in the submission process, there is a big difference. The indexes in Tables I and II can reflect this coding ability difference [42]. Therefore, mining and modeling the process event logs would have a good presentation on students' problem-solving ability.

IV. Dynamic Instructional Design Based on Students' Problem-Solving Ability Profiles

To improve a course's learning outcomes, we carry out dynamic instructional design and intelligent learning outcome analysis based on the students' problem-solving ability profiles. For this purpose, we propose a two-stage problem-solving ability modeling strategy to generate students' ability profiles at different learning stages.

A. *Framework of the Learning Process*

The LMS has accumulated many learners' behavior data. EPM technology believes that effective processing, mining, and analysis of these data can transform them from complex data into clear decision-support information that can reveal students' actual problem-solving abilities. Therefore, we preprocess these event logs as training data and employ GBDT technology to model the students' problem-solving ability (take the coding ability as an example).

The training of coding ability is a long and continuous process in all the programming courses. Thus, we propose a two-stage modeling strategy to mine students' coding ability at different learning stages. The two models can be used to generate students' ability profiles before and after a course, involving the course relevance. Meanwhile, the dynamic instructional planning and the intelligent learning output are, respectively, carried out according to the preability profiles and postability profiles. The learning process based on students' ability profiles is illustrated in Fig. 2.

As in Fig. 2, a course's learning process based on students' ability profiles is divided into the modeling stage and the model application stage. In the modeling stage, the modes are built on the historical event logs. The prerequisite course's learning behavior logs can objectively reflect the students' problem-solving ability before the target course. Therefore, we first use GBDT to build the students' preability model on its event logs. After the target course, when students' problem-solving ability has been enhanced, we build the postability model based on this course's event logs again.

The problem-solving ability models can support the target course's instructional design. In the model application stage of a course, students' preability profiles are first generated according to the event logs of the current grade students' prerequisite course and the preability model. Furthermore, the instructional design is dynamically planned in light of these profiles. After the course, students' postability profiles are generated according to their study process logs and postability model. We combine the postability profiles with the final scores to analyze the students' learning outcome and grading results. Simultaneously, their postability profiles are compared with the preability profiles to analyze the changes of average ability among different cohorts and individual ability.

According to the statistical analysis results, we summarize the learning outcome and adjust the instructional design of the next round of teaching. Moreover, the post-problem-solving ability model is updated according to the new event logs. Thus, the whole closed-loop teaching process is complete.

B. *Modeling of Problem-Solving Ability With GBDT*

GBDT is an additive machine learning model with strong learning ability. In the traditional machine learning algorithm, it is among the top three mostly used algorithms.

1) Feature Preprocessing: We define the input dataset as $D = \{(x_i, y_i)\}$, $i \in 1, 2, \ldots, n$, where $x_i \in \mathbb{R}^z$ is the log feature of student i with z dimensions, $y_i \in [0, 1]$ is the label of the feature, and n is the number of samples. The label y represents a student's problem-solving ability, and we define it as high and low.

For the ith student sample, the feature x_i is the concatenation of task detail feature td_i and answer details feature ad_i, and the index items of these two vectors are in Tables I and II. Before concatenation, each student's answer detail data are

TABLE I
WORK ACTIVITY DATA

ID	Index	Index Definition	Value Range
1	Number of submitted questions	The number of questions that submitted answers.	[0, $+\infty$]
2	Right questions	The number of scored questions.	[0, $+\infty$]
3	Number of code lines	The number of effective code lines in the program.	[0, $+\infty$]
4	Number of total submissions	Total number of submissions of all questions.	[0, $+\infty$]
5	Answer time	The time that students stay on the question pages and have submitted activities.	[0, $+\infty$], unit, minute.

TABLE II
ANSWER DETAIL DATA

ID	Index	Index Definition	Value Range
1	ID of question	The ID of a question in the question database.	Long integer
2	Pass or not	Whether the final answer is correct or not. If the pass rate of a question is greater than or equal to 0.6, the answer is correct. Pass rate = the number of passed test cases / total number of test cases.	[0, 1]
3	Completion time	The time spent on the question page.	[0, $+\infty$], unit, second.
4	Number of submissions	Total number of submissions of the question.	[0, $+\infty$]
5	First AC	Does the code pass for the first time of submission.	[0, 1]
6	First AC time	The difference between the first pass time and the first submission time for a question.	[0, $+\infty$], unit, second.
7	Number of submissions for AC	The number of submission times before a student has passed for the first time, including this time.	[0, $+\infty$]
8	Optimization times	The number of submissions after a student has passed for the first time.	[0, $+\infty$]
9	Effective optimization times	The number of AC times after a student has passed for the first time.	[0, $+\infty$]
10	Number of code lines	The number of lines in a program.	[0, $+\infty$]
11	Average cyclomatic complexity	The average cyclomatic complexity of all functions in a program.	[0, $+\infty$]
12	Number of functions	The number of functions in a program.	[0, $+\infty$]

accumulated according to the questions and averaged by the number of completed questions, as follows:

$$\begin{aligned} xa_i &= \left\{ \frac{1}{|Q_i|} \sum_{\forall j \in Q_i} ad_{ij} \right\} \\ xt_i &= td_i \\ x_i &= con(xt_i, xa_i), x_i = x_i / \sum_{j=1}^{z} x_{ij} \end{aligned} \tag{1}$$

where Q_i is the submitted questions set for student i, ad_{ij} is his log event of the jth submitted assignment, and con is the vector concatenation function. Moreover, the connected features are processed by L_1 norm.

2) Problem-Solving Ability Model: According to the learning behavior logs of the prerequisite course and the target course, we build the students' pre-problem-solving ability model pF_M and the post-problem-solving ability model aF_M, respectively. Both the models are built with GBDT [27], [43] in the same way. The specific modeling approach is as follows, and we represent the two models unified as F_M.

For the given student coding process features D, a tree ensemble model F_M with M additive functions to predict the output is

$$\begin{aligned} F_M(x_i) &= \sum_{m=1}^{M} t_m(x_i) \\ \hat{y}_i = \phi(x_i) &= \frac{1}{1 + e^{-F_M(x_i)}} \end{aligned} \tag{2}$$

where $F = \{t(x) = w_{q(x)}\}$ $(q : \mathbb{R}^m \rightarrow T, w \in {}^T)$ is a set of trees. q represents the structure of each tree, and T is the number of leaves in the tree. Each t_m corresponds to an independent tree structure q and leaf weights w.

To learn the functions used in the model, we minimize the following regularized objective function:

$$\begin{aligned} L &= \sum_{i=1}^{n} l(\hat{y}_i, y_i) + \sum_{m=1}^{M} \Omega(t_m) \\ \Omega(t) &= \gamma T + \frac{1}{2} \lambda \|w\|^2 \end{aligned} \tag{3}$$

where l is the cross-entropy loss function that measures the difference between the prediction label $\hat{y}_i$ and the ground-truth label y_i. The second term Ω penalizes the complexity of the model.

The model is trained in an additive manner. Formally, let $\hat{y}_i$ be the prediction of the ith instance at the kth iteration; we will need to add t_m to minimize the following objective function:

$$L^k = \sum_{i=1}^{n} l\big(\hat{y}_i^{k-1}, y_i + t_k(x_i)\big) + \Omega(t_k). \tag{4}$$

These trees are all optimized by this loss function, and we can finally get the classifier pF_M and aF_M. The original algorithms are presented in detail in [27] and [43].

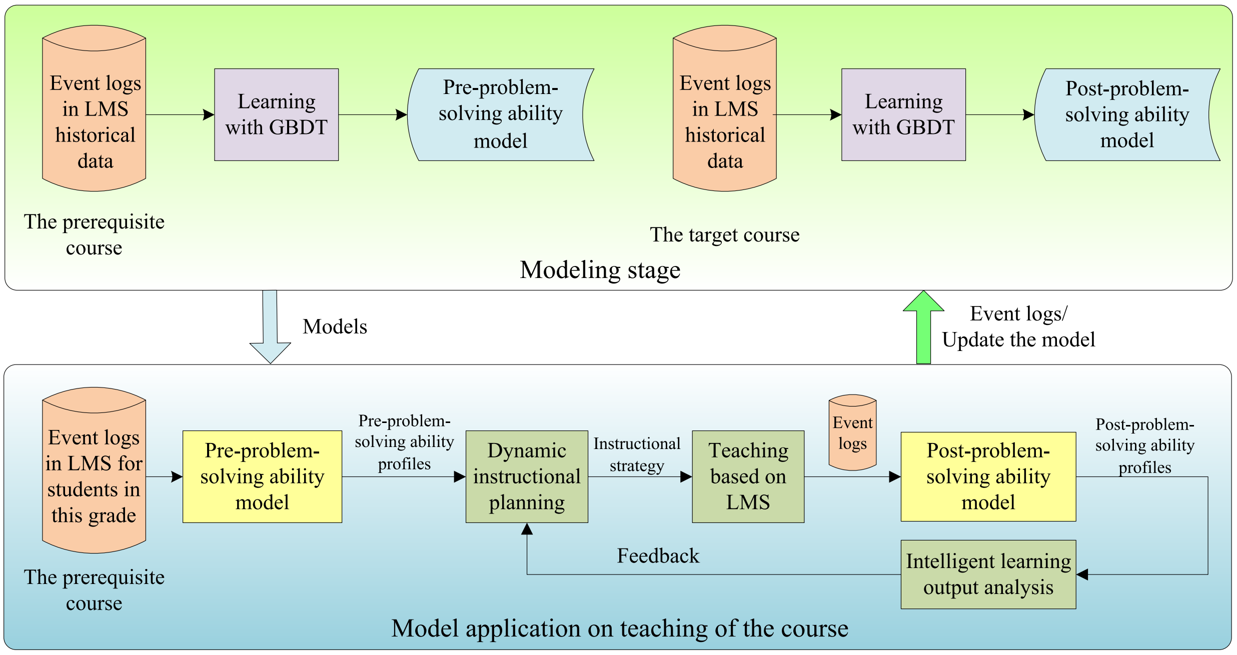

Event logs in LMS historical data
Learning with GBDT
Pre-problem-solving ability model
The prerequisite course
Event logs in LMS historical data
Learning with GBDT
Post-problem-solving ability model
The target course
Modeling stage
Models
Event logs/ Update the model
Event logs in LMS for students in this grade
Pre-problem-solving ability model
Pre-problem-solving ability profiles
Dynamic instructional planning
Instructional strategy
Teaching based on LMS
Event logs
Post-problem-solving ability model
Post-problem-solving ability profiles
Intelligent learning output analysis
Feedback
The prerequisite course
Model application on teaching of the course

Fig. 2. Learning process based on students' problem-solving ability profiles.

C. *Dynamic Instructional Planning Based on Learner's Ability Profiles*

Learners' ability is different, and so is their average ability. Therefore, the instructional design should be in line with their ability to obtain a better learning outcome. However, the existing learning process mostly uses the same instructional strategy every year, which is not conforming to the actual learning states of students. Therefore, we dynamically plan instructional design based on learners' problem-solving ability profiles.

The instructional planning problem TP is defined as a triple $TP = (D, C, P)$, where D, C, and P are instructional design, instructional content, and students' problem-solving ability profile, respectively.

This article divides the instructional planning of a course into two levels, which are global instructional planning and individual instructional planning. Global instructional planning designs a course's learning strategy, while individual instructional planning carries out personalized cultivation for students with excellent problem-solving ability [44]. Dynamic instructional planning based on students' ability profiles can combine instructional contents, instructional designs, and students' abilities to formulate instructional design suitable for their cognitive ability.

1) Global Instructional Planning: Before a course starts, we first choose the specific instructional design according to students' average ability. If their average ability is acceptable, the teaching keynotes should focus on new knowledge and difficult points. If their average ability is general, the instructional design should first ensure that students have a good mastery of basic knowledge and then are difficult points. This dynamic planning strategy can design a more appropriate learning process according to students' ability, which is in line with the teaching concept of different teaching methods for different students.

In the global instructional planning, P represents the overall distribution of students' ability profiles calculated by model pF_M. Its value is the ratio between the number of students with the higher problem-solving ability and the total number of students. We define a vector $Pv = (pv_0, pv_1, \ldots, pv_r)$ to represent r different distribution intervals of students' ability profiles, where the value of r is determined according to a specific course, $pv_i < pv_{i+1}, pv_0 = 0$, and $pv_r = 100$. The ith distribution interval is $I_i = \{P | pv_{i-1} < P \leq pv_i\}, 0 < i \leq r$.

The instructional design for a course $D = (d_1, d_2, \ldots d_r)$ is a set of predefined learning designs, where each d_i is a specific instructional design, $d_i = (d_{i1}, d_{i2}, \ldots d_{icn})$, and cn is the number of instructional contents. d_{ij} is a weight vector that represents the teaching weight of each knowledge point in chapter j. In instructional planning, the specific instructional design d is selected according to the distribution of students' ability profiles. If $P \in I_i, 0 < i \leq r, d = d_i$.

Instructional contents $C = (c_1, c_2, \ldots c_{cn})$ are determined by the course syllabus. If content c_i contains cp knowledge points, $1 \times cp$ row vector $d_{ij} = (d_{ij}^1, d_{ij}^2, \ldots d_{ij}^{cp})$ represents each knowledge point's weight in chapter j under the corresponding instructional design d_i. Where $\sum d_{ij}^k = 1$, the larger the weight of a knowledge point, the more class hours it takes. Weight is 0 means that this round of teaching does not contain the knowledge point. Teachers and educators predefine these weights based on the syllabus and students' problem-solving abilities.

Taking the *Data Structure and Algorithms* course as an example, the learning process planning is shown in Fig. 3. In the course, r is set to 3 according to teaching experiences, and the instructional contents $C = (c_1, c_2, c_3, c_4, c_5)$ are *linear list*, *tree*, *graph*, *search*, and *sort*. It can be seen from the figure that the global instructional design d_i is determined according to the

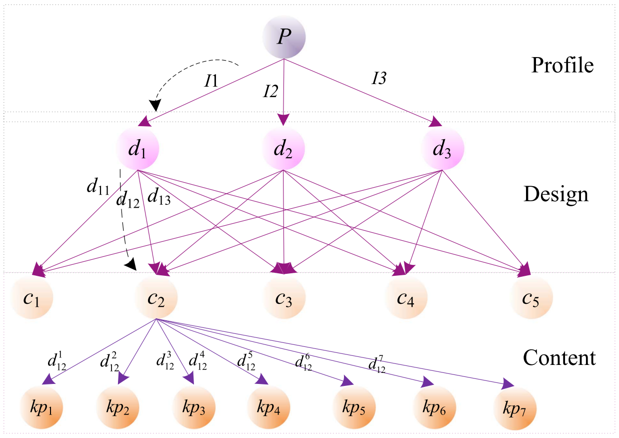


Fig. 3. Learning process based on students' ability profiles.

students' ability profile P. The selection path of the instructional strategies is $P \in I_i \rightarrow d_i$.

Taking the chapter tree as an example, the instructional contents are *the introduction of a tree*, *the definition of a binary tree*, *the computation of a binary tree*, *the recursive traverse of a binary tree*, *the non-recursive traverse of a binary tree*, *thread binary tree*, *the transformation between the binary tree and the forest*, and *the Huffman tree*, where $c_2 = (kp_1, kp_2, \ldots kp_7)$. Supposing that $P \in I_1$, each knowledge point kp_k is taught in line with d_{12}^k to conduct the appropriate teaching in class.

2) *Individual Instructional Planning:* The individual instructional planning aims at students with excellent problem-solving abilities. We first get information about the awards of students participating in competitions and their pre-problem-solving ability profiles. If the specific students participate in the competitions and win the prize and their ability profiles are deemed as high, they have excellent problem-solving abilities. These students can selectively learn the classroom knowledge and practice more difficult programming problems to carry out individualized training. On the other hand, we certainly should also pay attention to students with poor problem-solving abilities and urge them to practice more.

V. Experiments

This section is the exploratory study we conducted to evaluate our framework with empirical experiments.

A. Experiment Setup

We take the *Data Structure and Algorithms* course as the target course and the *Advanced Programming* course as the prerequisite course to validate our instructional strategy. The study involved three grades in the School of Computer Science at Shenyang Aerospace University. All of them had previously studied the *Advanced Programming* course. The problem-solving ability is the coding ability. The students' programming behaviors of the two courses are recorded as event logs in the *CG* platform with indexes in Tables I and II. The event logs of the previous two grades of 620 students are used as training data for the coding ability models. Meanwhile, the teaching based on students' coding ability profiles is conducted on 45 students from the last grade.

The training data include all the event logs of the programming questions in the assignments, the experiments, the stage exams, and the final exams. In training, we adopt the tenfold cross-validation. Table III reports the event logs of students after preprocessing according to (1) without L_1 norm. The columns from f0 to f9 are *Completion time*, *Number of submissions*, *First AC*, *First AC time*, *Number of submissions for AC*, *Optimization times*, *Effective optimization times*, *Number of code lines*, *Average cyclomatic complexity*, and *Number of functions*. The columns from f10 to f14 are *Number of submitted questions*, *Right questions*, *Number of code lines*, *Number of total submissions*, and *Answer time*. L is the label of the samples. Some of the indexes in Tables I and II are removed, for example, *ID of question*.

B. Evaluation of the Ability Models

We employ the XGBoost [43] toolkit to model the students' coding ability. XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible, and portable. It implements a machine learning algorithm under the GBDT that can solve many data science problems quickly and accurately.

The coding ability models' classification performances are evaluated by the following measurement metrics, precision, recall, and F-score:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$
$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \quad (5)$$

While F-score is the harmonic mean of the precision and the recall,

$$\text{F-score} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}. \quad (6)$$

In our experiments, the ground truth of the students' coding ability is manually labeled. Moreover, we annotate the students' coding ability as high and low according to their event logs and final scores. In the experiments, the high-ability students are positive samples, and low-ability students are negative samples. TP is true positives meaning that we label the student's ability as high, and the model also classifies it as high. FP is false positives meaning that we annotate the student's ability as low, but the model classifies it as high. FN is false negatives meaning that we annotate the student's ability as high, but the model classifies it as low.

C. Training of the Coding Ability Models

1) *Feature Importance Analysis:* To obtain the underlying rules that determine students' coding ability, we first employ the Decision Tree to analyze the importance of students' programming process behavior features. The feature importance

TABLE III
CODING ABILITY FEATURES OF STUDENTS AFTER EVENT LOGS PREPROCESSING

ID	f0	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	f12	f13	f14	L
1	71.59	0.16	0.64	17.97	0.23	0.65	0.66	20.26	0.21	0.62	20	19	1029	99	215	1
2	246.55	0.19	0.51	36.19	0.04	0.46	0.53	27.64	0.49	1.33	15	6	661	66	107	1
3	159.06	0.72	0.84	7.19	0.76	0.85	0.85	4.37	0.4	0.51	24	12	733	22	364	1
4	677.57	0.59	0.45	216.96	0.41	0.42	0.47	20.7	1.92	0.42	23	21	724	75	439	1
5	110.19	0.22	0.33	0.69	0.19	0.2	0.38	22.32	0.99	1.09	12	10	732	38	197	1
6	696.94	5.1	0.39	17293.2	3.59	1.02	0.02	20.57	1.14	1.36	40	20	1124	142	282	0
7	686.71	2.27	0.09	6221.86	1.15	0.69	0.33	31.49	2.54	1.06	46	27	1016	147	370	0
8	929.23	7.79	0.11	17449.71	6.68	0.79	0.43	36.28	2.77	1.45	29	16	992	39	336	0
9	1148.71	5.62	0.23	19255.52	3.54	1.39	0.57	33.11	1.67	1.79	12	7	204	137	227	0
10	263.84	1.26	0.15	24571.71	1.23	0.33	0.33	37.31	2.35	1.82	11	8	421	45	205	0

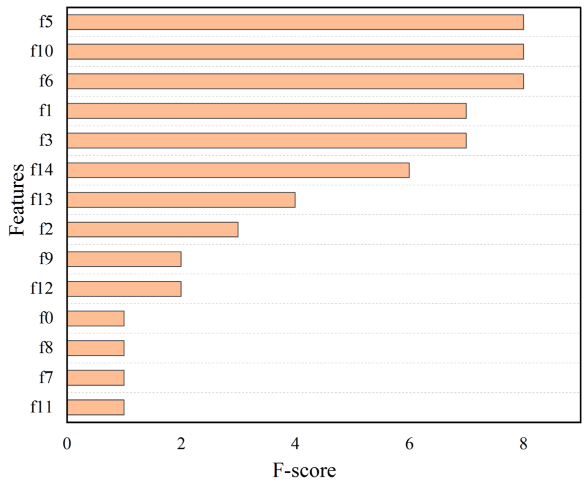


Fig. 4. Feature importance histogram of the premodel.

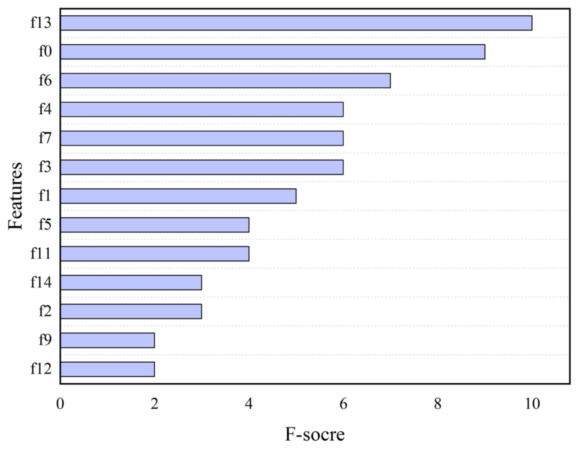


Fig. 5. Feature importance histogram of the postmodel.

histograms of the two models are shown in Figs. 4 and 5. The features show that, for the premodel, *Optimization times*, *Number of submitted questions*, and *Effective optimization times* are the most critical factors, while for the postmodel, *Number of total submissions*, *Completion time*, and *Effective optimization times* are the most critical factors. These factors are all related to the completion quality and quantity of the problems.

The factor *Number of submissions for AC* is removed by the premodel. Because it is necessary for beginners to submit multiple times to pass the test, this factor has less discriminative power. On the other hand, the factors *Average cyclomatic complexity* and *Number of submitted questions* are removed by the postmodel. This means that at this stage, there is no obvious difference in the overall logic of the codes and the number of submitted questions between the high ability students and the low ability students.

The factors *First AC*, *Average cyclomatic complexity*, *Number of functions*, *Right questions*, and *Number of code lines* have less impact on the two ability models, indicating that the distinguishing ability of these features is average. The discrimination degrees of the factors *Completion time* and *Number of code lines* have big differences between the two models. The reason is that students' coding ability is the same when learning the *Advanced Programming* course, but has significant differences in the *Data Structure and Algorithms* course.

The experimental result indicates that the factors and the target data jointly determine the effectiveness of all the features. Although the importance of different factors is inconsistent for the two models, the overall trend is basically the same.

2) Validation of the Two Coding Ability Models: In this section, we train the coding ability models pF_m and aF_m with the XGBoost toolkit and validate their performance. The parameter settings in XGBoost and the modeling process of the two ability models are the same. The experimental results of the two models are as follows.

We adjust the classification threshold of the binary classification scores, and the receiver operating characteristic (ROC) curves of the two models are shown in Figs. 6 and 7. The ROC curve is a graphical plot that illustrates the performance of a binary classifier system as its discrimination threshold is varied. The horizontal axis and vertical axis in the curve represent the FP ($1 -$ Specificity) and TP (Sensitivity) of the models under different thresholds. From the two figures, we can find that when the classification threshold is close to 0.5, the two models obtain ideal classification results. Therefore, we set the threshold as 0.5 for the coding ability classification.

In the experiments, the precision, the recall, and the F-score of the two models are listed in Table IV.

In Table IV, for model pF_m, the precision, recall, and F-score are 93.2%, 88.5%, and 90.8%, respectively, while for model aF_m, they are 90.9%, 97.8%, and 94.2%, respectively. The experimental results show that the models can judge students' coding ability and pick up most students with high

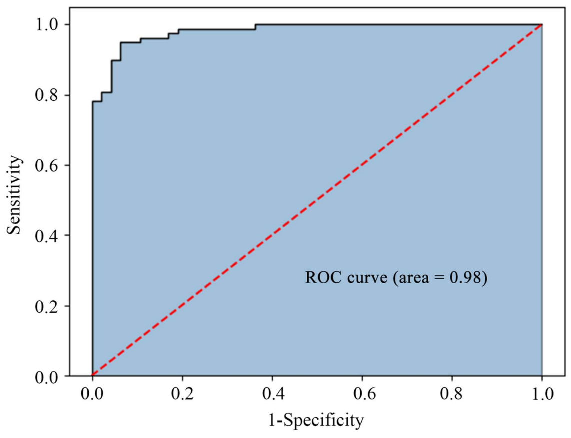


Fig. 6. ROC curve of the premodel.

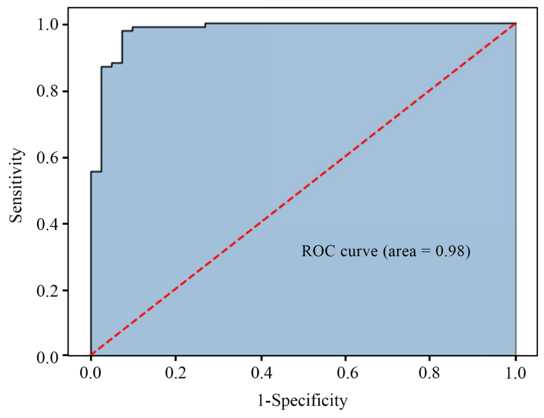


Fig. 7. ROC curve of the postmodel.

TABLE IV
VALIDATION RESULTS OF THE TWO MODELS (%)

model	precision	recall	F-score
pF_m	93.2	88.5	90.8
aF_m	90.9	97.8	94.2

coding ability. Furthermore, the values of the F-score indicate that there is a good balance between precision and recall.

D. Instructional Planning and Learning Outcome Analysis

Moreover, we take 45 students from the last grade to test the dynamic instructional planning and intelligent learning outcome analysis according to their coding ability profiles. First, their preability profiles are generated with the model pF_m. The students' preability profile is on the left of Fig. 8. We set $P = (p_0, p_1, p_2) = (0, 30, 60, 1)$ according to the teaching experience. The proportion of the high-preability students of the previous year's 40 students was 46%. That of this year is 42%, when the specific instructional strategy is chosen for this round of teaching.

After the *Data Structure and Algorithms* course, we generate the postability profiles of the students. The postability profile is on the right of Fig. 8. In this figure, this year's proportion of the high-coding-ability students increases to 65%. Although the

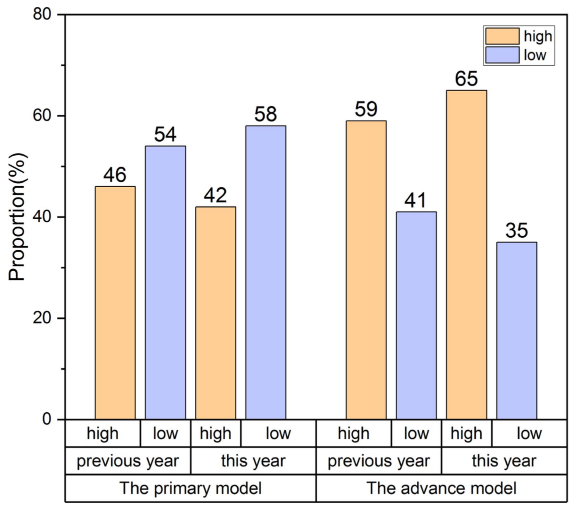


Fig. 8. Students' coding ability profiles.

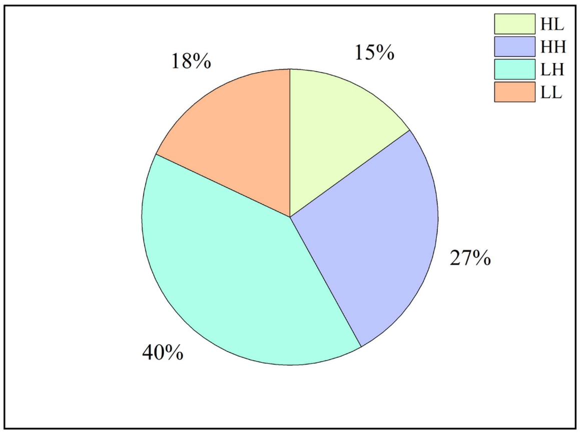


Fig. 9. Changes of students' coding ability profiles.

cohorts' high ability of the previous year is increased by 13%, it is still a bit lower than that of this year's increase which is 23%, showing the effectiveness of the new instructional strategy.

We further compare the ability change of the students between their preprofiles and postprofiles. We statistically calculate the number of students whose ability improved (LH), decreased (HL), and whose ability has no change (LL for low ability and HH for high ability). The results are shown in Fig. 9. From the experimental results, we can find that the students' coding ability is enhanced for 40% students and kept high for 27%. This means that the instructional design is effective for these students. Moreover, 15% students' ability has decreased and 18% students' ability is still low. For these students, there might be the following reasons: the *Data Structure and Algorithms* course is much more complicated than the *Advanced Programming* course for them, and their abilities are below the average level, or they do not study very diligently for some unseen reasons.

VI. CONCLUSION

To improve the learning effects and students' generic problem-solving ability for the computer programming education, this article employs EPM technology to mine students' learning process data on the LMS. We extract the features from these logs and use GBDT to model students' problem-solving ability. To involve the course relevance and provide scientific data support for the follow-up courses, we propose a two-stage modeling strategy. We take the *CG* platform and coding ability as examples to validate the dynamic instructional strategy based on problem-solving ability profiles.

The experimental results show that the preability model and the postability model have ideal classification precision, and they can be used to classify students' problem-solving ability. Students' pre-problem-solving ability profiles can accurately reflect their coding ability before the course starts. Moreover, the instructional strategy based on the ability profiles is more in line with the students' cognitive level and can effectively improve the students' learning outcome. At the end of the course, the postability profiles objectively describe students' learning outcomes and show the effectiveness of our framework.

In the follow-up learning process, we should further analyze the influence of event logs on students' problem-solving ability and investigate more effective instructional strategies.

References

[1] C. Romero, R. Cerezo, A. Bogarín, and M. Sánchez-Santillán, "Educational process mining: A tutorial and case study using Moodle data sets," in *Data Mining Learning Analytics: Applications in Educational Research*. Hoboken, NJ, USA:Wiley Press2016, pp. 1–28.

[2] E. M. Real, E. P. Pimentel, L. V. de Oliveira, J. C. Braga, and I. Stiubiener, "Educational process mining for verifying student learning paths in an introductory programming course," in *Proc. IEEE Front. Educ. Conf.*, 2020, pp. 1–9.

[3] R. Dolak, "Using process mining techniques to discover student's activities, navigation paths, and behavior in LMS moodle," in *Proc. Int. Conf. Innov. Technol. Learn.*, 2019, pp. 129–138.

[4] The Canvas Website, Oct. 2022. [Online]. Available: https://www.instructure.com/en-au/canvas

[5] The EDx Website, Jun. 2021. [Online]. Available: https://www.edx.org/

[6] C. Romero, S. Ventura, and E. García, "Data mining in course management systems: Moodle case study and tutorial," *Comput. Educ.*, vol. 51, no. 1, pp. 368–384, 2008.

[7] J. C. Vidal, B. Vázquez-Barreiros, M. Lama, and M. Mucientes, "Recompiling learning processes from event logs," *Knowl.-Based Syst.*, vol. 100, pp. 160–174, 2016.

[8] K. Tóth, H. Rölke, F. Goldhammer, and I. Barkow, "Educational process mining: New possibilities for understanding students' problem-solving skills," in *Educational Research and Innovation*. Paris, France: Org. Economic Cooperation Develop., 2017, pp. 193–209.

[9] M. Rahman, ""21st century skill "problem solving": Defining the concept," *Asian J. Interdiscipl. Res.*, vol. 2, no. 1, pp. 64–74, 2019.

[10] J. E. Cook and A. L. Wolf, "Automating process discovery through event-data analysis," in *Proc. IEEE 17th Int. Conf. Softw. Eng.*, 1995, pp. 73–73.

[11] N. Trcka, M. Pechenizkiy, and W. van der Aalst, "Process mining from educational data," in *Handbook of Educational Data Mining*. Boca Raton, FL, USA: CRC, 2010, pp. 123–142.

[12] A. Bogarín, R. Cerezo, and C. Romero, "A survey on educational process mining," *Wiley Interdiscipl. Rev.: Data Mining Knowl. Discov.*, vol. 8, no. 1, 2018, Art. no. e1230.

[13] A. H. Cairns, B. Gueni, M. Fhima, A. Cairns, S. David, and N. Khelifa, "Process mining in the education domain," *Int. J. Adv. Intell. Syst.*, vol. 8, no. 1, pp. 219–232, 2015.

[14] A. van denJ. BeemtBuijs, and W. van der Aalst, "Analysing structured learning behaviour in massive open online courses (MOOCs): An approach based on process mining and clustering," *Proc. Int. Rev. Res. Open Distrib. Learn.*, vol. 19, no. 5, pp. 37–60, 2018.

[15] R. Cerezo, A. Bogarí, M. Esteban, and C. Romero, "Process mining for self-regulated learning assessment in e-learning," *J. Comput. Higher Educ.*, vol. 32, no. 1, pp. 74–88, 2020.

[16] P. Ardimento, M. L. Bernardi, M. Cimitile, and F. M. Maggi, "Evaluating coding behavior in software development processes: A process mining approach," in *Proc. IEEE/ACM Int. Conf. Softw. Syst. Process.*, 2019, pp. 84–93.

[17] P. Ardimento, M. L. Bernardi, M. Cimitile, and G. De Ruvo, "Learning analytics to improve coding abilities: A fuzzy-based process mining approach," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, 2019, pp. 1–7.

[18] H. Ariouat, A. H. Cairns, K. Barkaoui, J. Akoka, and N. Khelifa, "A two-step clustering approach for improving educational process model discovery," in *Proc. IEEE 25th Int. Conf. Enabling Technol.: Infrastruct. Collaborative Enterprises*, 2016, pp. 38–43.

[19] Y. Wang, T. Li, C. Geng, and Y. Wang, "Recognizing patterns of student's modeling behaviour patterns via process mining," *Smart Learn. Environ.*, vol. 6, no. 1, pp. 1–16, 2019.

[20] A. Vatutin, M. Moskalenko, M. Skryabin, M. Svintsov, and A. Trifanov, "Computational psychometric approach for assessing mathematical problem-solving skills," *Procedia Comput. Sci.*, vol. 193, pp. 250–255, 2021.

[21] W. Ma, O. O. Adesope, J. C. Nesbit, and Q. Liu, "Intelligent tutoring systems and learning outcomes: A meta-analysis," *J. Educ. Psychol.*, vol. 106, no. 4,, 2014, Art. no. 901.

[22] L. Alfaro, C. Rivera, E. Castaneda, J. Zuniga-Cueva, M. Rivera-Chavez, and F. Fialho, "A review of intelligent tutorial systems in computer and web based education," *J. Adv. Comput. Sci. Appl.*, vol. 11, no. 2, pp. 755–763, 2020.

[23] E. Aimeur and C. Frasson, "Analizing a new learning strategy according to different knowledge levels," *Comput. Educ.*, vol. 27, no 2, pp. 115–127, 1996.

[24] D. M. Olivé, D. Q. Huynh, M. Reynolds, M. Dougiamas,, and D. Wiese, "A quest for a one-size-fits-all neural network: Early prediction of students at risk in online courses," *IEEE Trans. Learn. Technol.*, vol. 12, no. 2, pp. 171–183, Apr./Jun. 2019.

[25] D. Baneres, M. E. Rodríguez-Gonzalez,, and M. Serra, "An early feedback prediction system for learners at-risk within a first-year higher education course," *IEEE Trans. Learn. Technol.*, vol. 12, no. 2, pp. 249–263, Apr./Jun. 2019.

[26] V. T. N. Chau and N. H. Phung, "ST-OS: An effective semisupervised learning method for course-level early predictions," *IEEE Trans. Learn. Technol.*, vol. 14, no. 2, pp. 238–256, Apr. 2021.

[27] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Ann. Statist.*, vol. 29, no. 5, pp. 1189–1232, 2001.

[28] J. H. Friedman, "Stochastic gradient boosting," *Comput. Statist. Data Anal.*, vol. 38, no. 4, pp. 367–378, 2002.

[29] B. Bakhshinategh, O. R. Zaiane, S. ElAtia, and D. Ipperciel, "Educational data mining applications and tasks: A survey of the last 10 years," *Educ. Inf. Technol.*, vol. 23, no. 1, pp. 537–553, 2018.

[30] P. Mukala, J. C. Buijs, M. Leemans, and W. M. van der Aalst, "Learning analytics on coursera event data: A process mining approach," in *Proc. 5th Int. Symp. Data-Driven Process Discov. Anal.*, 2015, pp. 18–32.

[31] M. Pechenizkiy, N. Trcka, E. Vasilyeva, W. Van der Aalst, and P. De Bra, "Process mining online assessment data," in *Proc. 2nd Int. Conf. Educ. Data Mining*, 2009, pp. 279–288.

[32] A. H. Cairns, B. Gueni, J. Assu, C. Joubert, and N. Khelifa, "Analyzing and improving educational process models using process mining techniques," in *Proc. IMMM 5th Int. Conf. Adv. Inf. Mining Manage.*, 2015, pp. 17–22.

[33] S. Anuwatvisit, A. Tungkasthan, and W. Premchaiswadi, "Bottleneck mining and Petri net simulation in education situations," in *Proc. IEEE 10th Int. Conf. ICT Knowl. Eng.*, 2012, pp. 244–251.

[34] M. Mayilvaganan and D. Kalpanadevi, "Cognitive skill analysis for students through problem solving based on data mining techniques," *Procedia Comput. Sci.*, vol. 47, pp. 62–75, 2015.

[35] D. Etinger, "Discovering and mapping LMS course usage patterns to learning outcomes," in *Proc. Int. Conf. Intell. Hum. Syst. Integr.*, 2020, pp. 486–491.

[36] H. Horita, H. Hirayama, T. Hayase, Y. Tahara, and A. Ohsuga, "Process mining approach based on partial structures of event logs and decision tree learning," in *Proc. 5th IIAI Int. Congr. Adv. Appl. Informat.*, 2016, pp. 113–118.

[37] A. Rozinat and W. M. van der Aalst, "Decision mining in ProM," in *Proc. Int. Conf. Bus. Process Manage.*, 2006, pp. 420–425.

[38] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf, "Support vector machines," *IEEE Intell. Syst. Appl.*, vol. 13, no. 4, pp. 18–28, Jul./Aug. 1998.

[39] C. C. Chang and C. J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 1–27, 2011.

[40] P. Li, "Robust logitboost and adaptive base class (ABC) logitboost," in *Proc. 26th Conf. Annu. Conf. Uncertainty Artif. Intell.*, 2010, pp. 302–311.
[41] The CG Website, Jun. 2021. [Online]. Available: http://cg.sau.edu.cn
[42] B. Jiang, S. Wu, C. Yin, and H. Zhang, "Knowledge tracing within single programming practice using problem-solving process data," *IEEE Trans. Learn. Technol.*, vol. 13, no. 4, pp. 822–832, Oct./Dec. 2020.
[43] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2016, pp. 785–794.
[44] L. Meng, W. Zhang, Y. Chu, and M. Zhang, "LD-LP generation of personalized learning path based on learning diagnosis," *IEEE Trans. Learn. Technol.*, vol. 14, no. 1, pp. 122–128, Feb. 2021.

Fang Liu received the M.S. degree in computer science from the School of Information Science, Liaoning University, Shenyang, China, in 2007, and the Ph.D. degree in computer architecture from Northeastern University, Shenyang, in 2021.

Since 2009, she has been a Lecturer with the College of Computer Science, Shenyang Aerospace University, Shenyang. Her research interests include machine learning, computer vision, and video understanding.

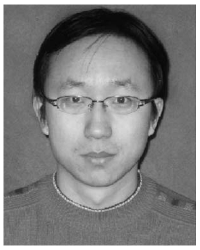
Liang Zhao received the Ph.D. degree in computing from the School of Computing at Edinburgh, Napier University, Edinburgh, U.K., in 2011.

From 2012 to 2014 before joining Shenyang Aerospace University, he was an Associate Senior Researcher with Hitachi (China) R&D. He is currently a Professor with Shenyang Aerospace University, Shenyang, China. He has authored or coauthored more than 100 peer-reviewed papers. His research interests include intelligent transportation systems, vehicular ad-hoc networks, wireless mesh networks, and software-defined networking.

Dr. Zhao is a Guest Editor for journals such as IEEE TRANSACTIONS ON NETWORK SCIENCE AND ENGINEERING and *Springer Journal of Computing*. He was the recipient of the best/outstanding paper awards at 2015 IEEE International Conferences on Ubiquitous Computing and Communications (IEEE IUCC), 2013 ACM International Conference on Advances in Mobile Computing and Multimedia, and 2020 IEEE International Symposium on Parallel and Distributed Processing with Applications. He was the Chair and Co-Chair of more than 20 international conferences and workshops such as 2021 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (Program Co-Chair) and 2020 IEEE IUCC (General Co-Chair).

Jiayi Zhao received the bachelor's degree in network engineering from Shenyang Aerospace University, Shenyang, China, in 2021.

He is currently with Tencent Technology Co., Ltd., Shenzhen, China. His research interests include machine learning and algorithm optimization.



Qin Dai received the B.S. and M.S. degrees in computer science and technology from Liaoning University, Shenyang, China, in 2003 and 2007, respectively, and the Ph.D. degree in computer architecture from Northeastern University, Shenyang, in 2018.

He is currently an Associate Professor with the Shenyang Institute of Engineering, Shenyang. His research interests include computer vision and image processing.

Chunlong Fan received the M.S. degree in software engineering and the Ph.D. degree in computing application technology from Northeastern University, Shenyang, China, in 2005 and 2021, respectively.

He is currently a Professor with Shenyang Aerospace University, Shenyang, where he has been a Lecturer with the College of Computer Science since 1997. His research interests include complex network analysis, agent testing and evaluation, and automated testing technology.

Jun Shen (Senior Member, IEEE) received the Ph.D. degree in computer science from Southeast University, Nanjing, China, in 2001.

He was with the Swinburne University of Technology, Melbourne, VIC, Australia, and the University of South Australia, Adelaide, SA, Australia, before 2006. He is currently an Associate Professor with the School of Computing and Information Technology, University of Wollongong, Wollongong, NSW, Australia, where he has been the Head of Postgraduate Studies and a Chair of the School Research Committee since 2014. He has authored or coauthored more than 250 papers in journals and conferences in computer science/information technology areas. His research interests include computational intelligence, bioinformatics, cloud computing, and learning technologies, including Massive Open Online Courses.

Dr. Shen is a Senior Member of the Association for Computing Machinery (ACM) and Australian Computer Society. He is an Editor, Program Committee Chair, Guest Editor, and Program Committee Member for numerous journals and conferences published by IEEE, ACM, Elsevier, and Springer. He was also a Member of ACM/AIS Task Force on Curriculum MSIS 2016. His publications appeared at IEEE TRANSACTIONS ON LEARNING TECHNOLOGIES, *Briefs in Bioinformatics*, *International Journal of Information Management*, and many others.