ISSN 2522-6398 Vol. 7, Issue 10: 223-234, DOI: 10.25236/FER.2024.071036

A Programming Learning Engagement Assessment Model and Teaching Application Based on Online Judgement System

Zuhua Dai[1,a,*], Shiya Lv[1,b], Qicheng Zhang[1,c]

[1]School of Computer Science and Engineering, Northwest Normal University, Lanzhou, China
[a]2812704000@qq.com, [b]874186995@qq.com, [c]415032355@qq.com
[*]Corresponding author

Abstract: *The online judge system is a specialized learning management systems for program assignments commonly used in programming education. The behaviour data recorded by the system can reflect the most authentic thinking and learning situation of students. Mining and analysis the learning behaviour data can help teachers discover students with learning difficulties in a timely manner and design personalized teaching activities. Although the important value of learning behaviour data analysis in promoting the development of online education has attracted widespread attention and reflection, the mining techniques for these data have been underexplored. This study investigated the data on student self-directed learning behaviour in online programming learning and studied the identification method of self-directed learning behaviour patterns. We also explored the applicability of student behaviour differentiation analysis in academic performance warning, personalized teaching activity assistance design, and learning situation analysis. We collected online behaviour data from 152 students in the Algorithm Design and Analysis course at a comprehensive university in China. Students were asked to independently submit programs for 17 programming questions within 18 weeks, and the program used the automatic evaluation results on the online judge system as their regular grades for the course. In the end, 8259 program submission records were collected. We constructed an online programmer learning engagement evaluation model and event analysis, correlation analysis, and K-means algorithm to cluster these behavioural data to determine the differences in student learning engagement patterns. Using the model proposed in this study, 8259 program submission records were processed, and five significantly different programming learning behaviour patterns were identified, namely: progressive high input, high performance input, random input, low performance input, and low input. Thereby demonstrating the usability of the model in evaluating learner differentiated learning engagement patterns. We also propose application suggestions for differentiated student behaviour patterns in teaching scenarios.*

Keywords: *Online Judge System; Programming Learning Behaviour; Assessment of Learning Engagement; Event Analysis; Cluster Algorithm*

1. Introduction

With the continuous advancement and deepening of educational informatization, e-learning, online learning, blended learning has been widely integrated into education and teaching practices. These teaching methods are often implemented with the support of Learning Management Systems (LMS), which collect a vast amount of student behavioral data. The potential correlation structures and underlying patterns within the data can provide insightful reflections of students' genuine thinking and learning conditions, opening up new opportunities for profound research into student psychology and behavior [1]. The utilization of data mining techniques to analyze the data and address various challenges in online education has garnered significant attention and contemplation from researchers.

Aydoğdu [2] developed a neural network model that incorporates the utilization of LMS to forecast student performance, and assessed the impact of LMS usage-related variables on the accuracy of these performance predictions. Qiu and Zhang et al. [3] proposed a learning performance predictor based on online learning behavior classification to achieve real-time supervision and timely feedback during the learning process. Tong et al. [4] identified 12 online learning behavior indicators highly associated with learning performance through data correlation analysis and developed a student MOOC learning performance evaluation model using three algorithms: multiple linear regression, multi-layer perceptron, and classification regression tree. Lin [5] introduced a K-Means algorithm for MOOC user clustering

ISSN 2522-6398 Vol. 7, Issue 10: 223-234, DOI: 10.25236/FER.2024.071036

analysis, featuring three key steps: feature weight calculation, initial cluster center optimization, and the design of a balanced discriminant function to determine the optimal number of clusters. Yan et al. [6] categorized online learning behavior into four classifications: trajectory behavior, social behavior, resource learning behavior, evaluation, and reflective behavior. They employed Pearson correlation analysis to explore the relationship between different learning behaviors and course grades, revealing a strong correlation between metrics such as the number of visits, clicks, online time, and course grades. A grade prediction classification model was then trained using a three-layer feedforward neural network. Li et al. [7] leveraged facial video images and LMS interactive mouse flow data from student learning activities to develop a multimodal data integrated evaluation model for assessing online learning engagement. By correlating the evaluation model results with student survey scale outcomes, they demonstrated the feasibility and effectiveness of the integrated model for evaluating learning engagement. Hu et al. [8] established a multimodal learning analysis model utilizing the HDRBM neural network model. The aforementioned studies [5][6][7][8] on LMS learning behavior models primarily concentrate on the modeling and analysis of MOOC platform data. Within the realm of computer education, there has been a recent surge in learning behavior data analysis for online judge systems (OJ) [9][10]. These research findings not only hold significant value for enhancing the functionality of LMS and improving the overall user experience but also play a crucial role in driving the reform of teaching methodologies in the era of information-based instruction.

The online judge system is a B/S architecture code automatic evaluation system. The system compiles and executes programs submitted by users online, and compares the results with predefined test cases to assess program correctness. The OJ system was originally employed in the International College Student Programming Competition, later evolving into a specialized management system for programming assignments in education. It has been widely utilized in various programming courses, as well as in training and selecting team members for algorithm competitions.

Researches [11][12] indicate that using an OJ system to assist in programming education can help educators improve teaching efficiency and enhance instructional outcomes. Furthermore, through the use of scientifically sound data analysis models to analyze behavioral data such as student response times and the frequency of attempts recorded in the OJ system, educators can gain insights into students' performance and requirements in programming learning [13][14]. This process, in turn, establishes a scientific foundation for personalized learning, enhancement of instructional design, and prediction of student achievement.

Based on the categorization of model tasks, the learning behavior data analysis models of the OJ system include programming ability assessment models, student group identification models, and the expansion of user functionality based on student behavior data, etc. The programming ability assessment model aims to predict students' grades or programming abilities using their online programming behavior data, typically through supervised learning techniques. For example, Chuang and Chang [15] conducted a comparative analysis of the coding behaviors between novices and experts, with a particular emphasis on their problem-solving patterns and the accuracy of their programs. The research findings indicate that experts commit fewer errors, thereby achieving a higher level of program correctness compared to novices. Sunday and Ocheja et al. [16] employed the ID3 and J48 decision tree algorithms to analyze student log data from the course "Introduction to Computer Programming". Xu et al. [10] combined exploratory factor analysis with deep neural networks to evaluate student programming proficiency. The student group identification model recognizes similarities/differences in learning behavior among student groups to aid differentiated instruction, often using association rule mining or data clustering technologies. Mai et al. [17] analyzes the relationship between learning behavior data and learning assessment results, employing Random Matrix Theory (RMT) and community detection to identify groups of learners with similar behaviors. Rico-Juan et al. [18] analyzed student behavior data using explainable AI to identify characteristics and behavior patterns of different student groups for personalized learning needs. Fu et al. [19] developed a visualization learning analytics dashboard for the OJ system, named VisOJ, which includes two types of user interfaces: one for teachers and one for students. The teacher interface displays students' learning status and ranking trends, assisting teachers in monitoring and providing feedback on their learning activities. The student interface offers perspectives such as error type analysis and evaluation, promoting students' self-reflection and self-regulation. Liu et al. [20] propose a novel model called Programming Exercise Recommender with Learning Style (PERS), which simulates learners' intricate programming behaviors.

The engagement of learning behaviors [21] is an important indicator for evaluating learners' active states during the learning process, especially in online learning. By utilizing the log data from LMS to assess and provide feedback on learners' engagement, it can help teachers gain a comprehensive

ISSN 2522-6398 Vol. 7, Issue 10: 223-234, DOI: 10.25236/FER.2024.071036

understanding of learners' online learning situations. It enables timely detection of students who may have issues and allows for appropriate interventions to ensure learning effectiveness. This approach contributes to addressing the prevalent problem of high dropout rates in online education [7] and promotes its progress and innovation.

Inspired by the researches on LMS learning behavior model, this paper focuses on studying the learning behavior data of students in the OJ system. It constructs an analysis model for the sequence of program submission behaviors and explores an evaluation method for measuring learners' engagement in online programming learning within the OJ system. This research provides a modeling approach for analyzing learning behaviors in the OJ system and also offers a research basis for the design of learning activities and optimization of learning evaluation models in the OJ system. Ultimately, this contributes to enhancing the teaching effectiveness and user experience of the OJ system.

2. Model and Methods

This paper explores the relationship between the programming submission behavior recorded by the online judging system and user learning engagement. The research framework is illustrated in Figure 1. Firstly, the learning behavior data from the OJ system is collected and preprocessed. Then, the event attributes of programming learning engagement are defined, and programming learning engagement events are extracted as analysis units from the learning behavior data. User learning engagement event sequences are constructed, and clustering techniques are applied to analyze patterns of learning engagement behavior. Finally, the model is validated using actual data, and further analysis is conducted on the correlation between programming learning engagement factors and learning performance.

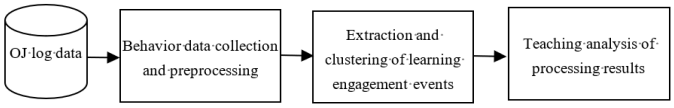


Figure 1: Analysis framework for online programming.

2.1. Description of Programming Learning Behaviour Data

Learning behavior data depends on the functionalities of the Online Judge (OJ) system and the teaching process. The Programming Teaching Assistant (PTA)[1], developed by Zhejiang University, is utilized as programming teaching platform to facilitate the learning process. The teaching procedure is as follows:

1) The teacher creates an empty set of programming problems in the OJ system and specifies the automatic evaluation rules and the learning time window. The set is associated with the student accounts participating in the learning process.

2) According to the teaching progress, the teacher sequentially releases programming problems and promptly notifies the students via email.

3) Students submit their programs to the released programming problems within the learning time window. The OJ system automatically evaluates their submissions, updates the scores, and students can view the evaluation results and scores in the OJ system.

4) Steps 2 and 3 are repeated until the end of the teaching activity.

During the time window, students can submit their program multiple times for system evaluation. The OJ system saves the submission information based on the submission time and calculates the final score for each problem based on the last submission.

There are three data tables in the OJ system related to programming learning behavior.

The problem set $T = \{t_1, t_2, ..., t_i, ..., t_m\}$, where each t_i is defined in the attributes set TA. $TA =$ {problem number, problem title, release time, problem description, programming constraints, test case set, score weight}. The problem numbers are assigned sequentially according to the release time, and m represents the total number of problems. Program submission information table

[1] https://pintia.cn/

ISSN 2522-6398 Vol. 7, Issue 10: 223-234, DOI: 10.25236/FER.2024.071036

$TS=\{ts_1, ts_2, ..., ts_i, ..., ts_N\}$, where each ts_i is defined in the attributes set TSA. TSA = {student ID, problem number, submission date and time, program evaluation status, score}. N is the total number of program submissions during the learning time window. The program evaluation status includes "correct answer", "partially correct", "non-zero return", "wrong answer", "compilation error", "multiple errors", "timeout", "floating point error", "segmentation fault". For the convenience of data classification and statistics, all program evaluation statuses other than {"correct answer", "partially correct"} are categorized as "error states". Student user table $U=\{u_1, u_2,...,u_i,...,u_n\}$, where each u_i is defined in the attributes set UA, $UA=\{student\ ID, student\ name\}$. n represents the total number of students. The relationships between the three tables are illustrated in Figure 2. The programming learning behavior data is mainly extracted from the program submission information table.

Figure 2: The source data table relationship of programming.

2.2. Concepts Related to Programming Learning Engagement Events

Teaching events refer to various educational activities that occur during the teaching process, such as classroom lectures, post class exercises, etc. Teaching events offer learners a pathway to acquire knowledge and skills, guiding and fostering learner engagement behavior. The learning behavior engagement event reflects the degree of behavioral resource engagement exhibited by learners in participating in teaching events. In online programming learning, a programming learning event refers to a programming task that students need to complete. The teacher initiates the programming task based on the problem attributes set TA, and completing the task requires students to possess the corresponding knowledge and skills.

Due to individual differences, students demonstrate different learning styles, strategies, and behavioral engagement in completing programming tasks, which can comprehensively reflect their programming ability level, knowledge mastery level, academic attitude, and so on. To facilitate the representation and computation of data in the model, the following concepts related to programming learning engagement events are defined.

Definition 1 (Program Submission Behavior Sequence, PSBS): The program submission sequence TS_{ij} for student $u_i(0<i\le n)$ regarding programming task $t_j(0<j\le m)$ in the program submission information table is represented as $\{ts_{ij,1}, ts_{ij,2}, ..., ts_{ij,k}, ..., ts_{ij,l}\}$, where $ts_{ij,k}\in TS$, l represents the number of program submissions made by student u_i for task t_j.

Definition 2 (Programming Learning Event, PLE): A programming exercise task $t_j\in T(0<j\le m)$ released by the teacher based on the teaching progress, which includes the release time and grading criteria, constitutes a programming exercise question.

Based on the structure of the program submission information table and the characteristics of programming learning events, this paper uses five dimensions to represent the behavioral characteristics of programming learning investment events: programming task response time (unit: days), program submission count, task completion duration (unit: minutes), program submission status sequence, and task score. Among them, the programming task response time reflects the initiative and academic interest of the programmer in learning activities, demonstrating the student's attitude towards following teaching requirements. A quick task response helps students consolidate knowledge promptly and alleviate learning pressure; the task duration signifies the time students invest in a programming task, reflecting the continuity of programming behavior; the program submission count indicates how students react and perform when facing difficulties and setbacks during learning. When students submit programs, if the evaluation result is incorrect, they do not give up but continuously revise and improve, making multiple submission attempts and investing more time in problem-solving. The program submission status sequence reflects the sequence of program issues feedback by the OJ system and is represented using characters to denote different meanings of program evaluation statuses.

Definition 3 (Programming Learning Engagement Event, PLEE): The programming investment

ISSN 2522-6398 Vol. 7, Issue 10: 223-234, DOI: 10.25236/FER.2024.071036

situation of student u_i on programming task t_j is represented by a ternary relationship: $te_{ij} = \{s, u_i, t_j\}$, where s represents the learning engagement situation. The property set of s includes five attributes: programming task response time, number of program submissions, task completion duration, program submission status sequence, and score, denoted as:

$$SA = \{responsetime, submitcount, duration, statussequence, score\}$$

The values of the five attributes are calculated based on the PSBS of student u_i on programming task t_j as follows:

$s_{responsetime} = ts_{ij,1} - t_j\,(release\ time)$, where $ts_{ij,1}$ is the date and time of the first program submission.

$$s_{summitcount} = |TS_{ij}| = l.$$

The calculation of $s_{duration}$ is slightly complicated. Assuming that student u_i 's program submission behavior for task t_j occurs within c days, let $s_{duration\_d}$ represent the duration of engagement on the d-th day, $s_{duration\_d} = ts_{ij,de}\ (submission\ date\ and\ time) - ts_{ij,ds}(\ submission\ date\ and\ time)$. Here, $ts_{ij,ds}$, $ts_{ij,de}$ indicate the first and last program submission times of student u_i on task t_j in the d-th day, respectively. Therefore, $s_{duration} = \sum_{d=1}^{c} s_{duration\_d}$.

$$s_{statussequence} = \cup_{k=1}^{l} ts_{ij,k}\,(program\ evaluation\ status).$$

$$s_{score} = ts_{ij,l}\,(score).$$

The program submission status sequence is a non-numeric type and requires data conversion processing. The program submission status sequence has two characteristics: (1) the length of the program submission status sequence equals the number of program submissions. (2) the status transition pattern of the program submission status sequence is usually "incorrect, ..., partially correct, ..., correct". By utilizing these two characteristics, the number of incorrect evaluations, the number of partially correct evaluations, and the number of correct evaluations is calculated from the status sequence. These three types of evaluation states are used to replace the program submission status sequence, avoiding the inconvenience of calculating the status sequence while not losing computational information, and reducing duplicate attribute of the program submission count. The adjusted SA becomes $\{responsetime,\ duration, error\text{-}summitcount,\ partlycorrecsummitcount,\ correcsummitcount, score\}$.

Definition 4 (Programming Learning Engagement Event Sequence, PLEES): Within a given learning time window W, all the PLEEs completed by student u_i are arranged according to the release time of the programming tasks, as shown in (1):

$$TE_i = \{te_{i1},\ te_{i2}, \cdots,\ te_{im}\} \quad (1)$$

where m is the number of programming tasks that the student needs to complete. For programming tasks t_j that the student u_i has not submitted any program, a zero-PLEE is inserted at position j in the programming event sequence, denoted as $te_{ij} = \{s_0, u_i, t_j\}$,where $s_0 = \{W_{end} - t_j\ (release\ time), 0, 0, 0, 0, 0\}$ and W_{end} represents the end time of the programming task window. The s_0 is defined to represent zero investment programming status.

2.3. Similarity Calculation Method for the PLEESs

Similarity measurement is the computational foundation for clustering and mining event sequence data [22] The following describes the similarity calculation method for PLEESs.

Let $TE_A = \{te_{A1}, te_{A2}, ..., te_{Am}\}$ represent the PLEES for student u_A, where k represents any attribute of the PLE, i.e., $k \in SA$. Calculate the average value and standard deviation of all PLE attribute k in

ISSN 2522-6398 Vol. 7, Issue 10: 223-234, DOI: 10.25236/FER.2024.071036

TE_A using (2), denoted as $A_k = (AV_k, AS_k)$.

$$AV_k = \frac{\sum_{j=1}^{m}(te_{Aj}(s_k)}{m}, AS_k = \sqrt{\frac{\sum_{j=1}^{m}(te_{Aj}(s_k) - AV_k)^2}{m-1}} \tag{2}$$

Similarly, let vector $B_k = (BV_k, BS_k)$ represent the statistical vector of the PLEES TE_B for student u_B on event attribute k. Use the Cosine Similarity method to calculate the similarity score between A_k and B_k, as shown in (3).

$$sim(A_k, B_k) = \frac{A_k \cdot B_k}{\|A_k\|^2 \times \|B_k\|^2} \tag{3}$$

The similarity between the two PLEESs is calculated using (4):

$$sim(TE_A, TE_B) = \sum_{k=1}^{d} sim(A_k, B_k) \tag{4}$$

Where d is the number of attributes involved in the PLEE, according to definition 3, d=6 in this paper.

2.4. Clustering Algorithm for the PLEES

The K-means clustering algorithm, which selects initial cluster centers based on the principal component method, is a commonly used clustering method suitable for grouping data [23]. In the clustering process of the PLEES, this method can be used to place event sequences with similar features into the same cluster.

The process of K-means clustering algorithm is as follows:

Input: The PLEES dataset $TE_{Data} = \{TE_1, TE_2, ..., TE_n\}$, the number of clusters K, where n is the number of students.

Step 1: Using the principal factor method to select PLEESs from TE_{Data} as the initial clustering centers.

Step 2: For each PLEES $TE_i \in TE_{Data}$, complete the following processing in sequence:

(1) Calculate the similarity between TE_i and the K cluster centers using equation s (3) and (4).

(2) Sort the K similarity values of (1) from highest to lowest.

(3) Assign TE_i to the cluster with the highest similarity.

Step 3: After all data in TE_{Data} is allocated, calculate the cluster center of each cluster using the average method based on all PLEESs in that cluster.

Step 4: Repeat steps 2 and 3 until the algorithm converges.

In step 1 of the above algorithm, the principal factor method to selects the initial cluster center as follows:

(1) Calculate the correlation coefficients between the characteristic attributes of the PLEE and academic performance.

(2) Determine the PLEE attribute with the highest absolute value of the correlation coefficient as the main factor.

(3) Calculate the full range R of the main factor, divide R into K groups, determine the upper and lower limits and the median of each group.

(4) Select the K records in TE_{Data} with principal factor values closest to the K group medians as

ISSN 2522-6398 Vol. 7, Issue 10: 223-234, DOI: 10.25236/FER.2024.071036

the initial cluster centers.

3. Model Experiment and Results Analysis

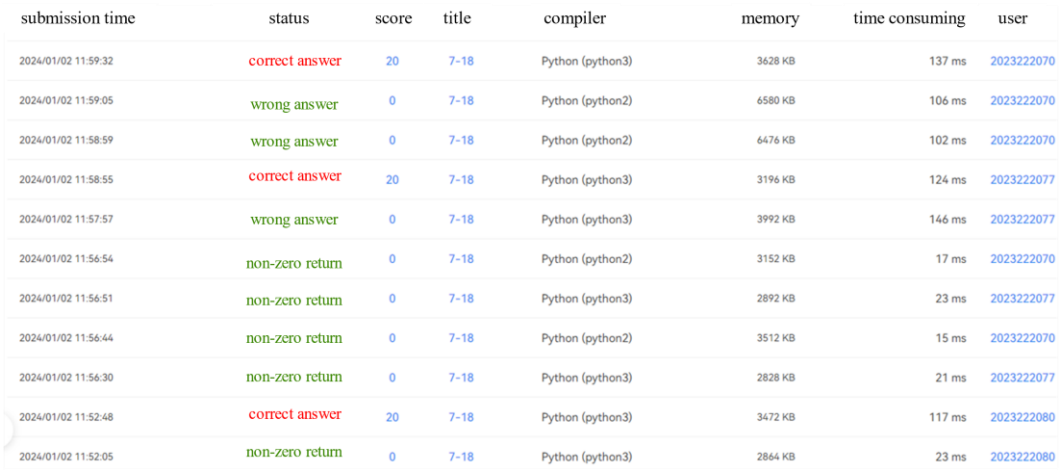
The learning behavior data of 152 students in the "Algorithm Analysis and Design" course offered by a certain university in the 2023 semester were collected. Prior to the course, the user guide of the PTA platform was shared with students to ensure that they could understand the platform's functions.

Students were also encouraged to join the PTA user QQ group. In this group, the technical advisors were available at all times to address any difficult operational issues students encountered on the platform, without interfering with students' independent learning activities. The start and end time window for programming tasks is [2023-9-18 12:00, 2024-1-2 12:00], 17 problems were released according to the teaching progress during this period, and the information examples of the problems are shown in Table 1.

Table 1: Sample information for the programming problems.

Problem number	Release time	Total number of programs passed	Total number of submitted programs	Problem title	Score weight
1	2023/9/18 21:56	189	552	Statistical problem	10
2	2023/9/26 11:37	169	1032	Fibonacci sequence	10
3	2023/10/10 0:00	163	342	Output the full permutation of n numbers	8
4	2023/10/17 0:00	160	376	Hanoi Tower Problem	20
5	2023/10/24 0:00	158	397	A multiplied by B	8
6	2023/10/31 21:22	149	877	Binary search	10
7	2023/11/7 11:37	158	527	Matrix A multiplied by B	20

A total of 8259 program submission records were collected from the PTA, and the data format of the program submission records is shown in Figure 3. The first row in the figure represents the record attributes. The paper selects five attribute data for analysis: submission time, status, score, title, and user. These attributes correspond to TSA's submission date and time, prog ram evaluation status, score, problem number, student ID.



submission time	status	score	title	compiler	memory	time consuming	user
2024/01/02 11:59:32	correct answer	20	7-18	Python (python3)	3628 KB	137 ms	2023222070
2024/01/02 11:59:05	wrong answer	0	7-18	Python (python2)	6580 KB	106 ms	2023222070
2024/01/02 11:58:59	wrong answer	0	7-18	Python (python2)	6476 KB	102 ms	2023222070
2024/01/02 11:58:55	correct answer	20	7-18	Python (python3)	3196 KB	124 ms	2023222077
2024/01/02 11:57:57	wrong answer	0	7-18	Python (python3)	3992 KB	146 ms	2023222077
2024/01/02 11:56:54	non-zero return	0	7-18	Python (python2)	3152 KB	17 ms	2023222070
2024/01/02 11:56:51	non-zero return	0	7-18	Python (python3)	2892 KB	23 ms	2023222077
2024/01/02 11:56:44	non-zero return	0	7-18	Python (python2)	3512 KB	15 ms	2023222070
2024/01/02 11:56:30	non-zero return	0	7-18	Python (python3)	2828 KB	21 ms	2023222077
2024/01/02 11:52:48	correct answer	20	7-18	Python (python3)	3472 KB	117 ms	2023222080
2024/01/02 11:52:05	non-zero return	0	7-18	Python (python3)	2864 KB	23 ms	2023222080

Figure 3: Screenshot of program submission record data style.

Implement the PLEE extraction, event sequence similarity algorithm, and clustering algorithm in Java language. Analyze the correlation between programming learning engagement attributes and course grades using correlation coefficient method. The following is an exposition and analysis of the data processing results.

3.1. Analysis of the Correlation Between the Attributes of PLEEs and Course Grades

The attributes of PLEE related to learning performance are of research value. Calculate the average values of 5 attributes of the PLEE sequence, then conduct a correlation analysis between each attribute and the OJ assessment scores as well as the final exam scores of the course, as shown in Table 2.

The data in Table 2 indicate that $s_{responsetime}$ (the response time to programming tasks) is significantly

ISSN 2522-6398 Vol. 7, Issue 10: 223-234, DOI: 10.25236/FER.2024.071036

negatively correlated with final exam scores, while $s_{correcsummitcount}$, $s_{partlycorrecsummitcount}$, $s_{errorsummitcount}$ (the number of program submissions in three evaluation states) is significantly positively correlated with final exam scores. $s_{duration}$ (the duration of task completion) shows a weak correlation with final exam scores. It is evident that the response time to programming tasks is the major factor influencing learning outcomes. The OJ assessment scores have no significant correlation with $s_{duration}$ (the duration of task completion), $s_{correcsummitcount}$ (the number of program submission errors), $s_{responsetime}$ (the response time to programming tasks), or $s_{partlycorrecsummitcount}$ (the number of partially correct program evaluation states). Only $s_{correcsummitcount}$ (the number of correct program evaluation states) shows a significant correlation with OJ assessment scores. In conclusion, the grading algorithm of the PTA system only considers learning outcomes in the correct program evaluation state, without taking into account other behavioral engagement attributes in the learning process. This lack of consideration is unreasonable and the algorithm should be improved based on learning engagement factors.

Table 2: The correlation between the attributes of PLEE and grades.

Release time	Correlation coefficient	
	Final exam score	OJ evaluation score
$s_{responsetime}$	-0.5**	-0.1
$s_{duration}$	0.1	0
$s_{correcsummitcount}$	0.2*	0.3**
$s_{partlycorrecsummitcount}$	0.3**	0.1
$s_{errorsummitcount}$	0.2*	0
OJ evaluation score	0.1	--

3.2. Cluster Analysis of Programming Learning Behaviour

By using the response time of programming tasks as the principal factor for measuring the PLEE, a cluster analysis was conducted on 8259 program submission records from 152 students, resulting in 5 stable clusters. TABLE 3 presents the statistical values of PLEE attributes for these 5 cluster centers.

It is evident that the data results in Table 3 reflect five distinct patterns of programming learning behaviors within the OJ system. Cluster #1 and Cluster #2 students each account for 12% of the total, while Cluster #5 students have the highest proportion at 36%. By observing the central values of the PLEE attributes for these five clusters, it can be seen that there are significant differences in learning engagement behaviors among the five patterns. Based on these differences, the programming learning behavior engagement patterns are classified as follows: Sequential Progressive High engagement (Cluster #1), High-performance engagement (Cluster #2), Random Participation (Cluster #3), Low-performance engagement (Cluster #4), and Low- engagement (Cluster #5). The following summarizes and describes the characteristics of these five learning engagement patterns based on Table 3.

Table 3: Cluster center aAttribute values for the PLEEs.

Event attribute	Cluster #1 20 people		Cluster #2 20 people		Cluster #3 30 people		Cluster #4 32 people		Cluster #5 50 people	
	AVG	STDEV	AVG	STDEV	AVG	STDEV	AVG	STDEV	AVG	STDEV
$s_{responsetime}$	19.9743	15.4302	20.6037	16.1564	36.0866	16.9592	36.2599	20.271	48.759	22.9149
$s_{duration}$	65.2619	132.4964	12.9233	31.6835	5.0709	12.1854	27.6385	75.6926	4.1605	11.4572
$s_{correcsummitcount}$	1.6852	0.983	1.1614	0.4005	1.1337	0.3639	1.2252	0.5243	1.209	0.3484
$s_{partlycorrecsummitcount}$	2.0265	5.1273	0.6561	1.6913	0.153	0.4893	0.9215	2.7363	0.0944	0.3431
$s_{errorsummitcount}$	5.5238	5.6334	2.0688	3.515	1.4155	2.8738	2.7807	4.7539	0.8774	2.0604

Sequential Progressive High engagement: Students in this cluster have the quickest response time to programming tasks, and their response time is highly correlated with the task release time. The average duration of task completion, the number of program evaluation status times, and their standard deviation are significantly higher than the other clusters. The learning engagement characteristics indicate that this pattern has the highest level of investment. The high level of behavioral engagement in learning reflects the strong emphasis these students place on course learning. Compared to the other four clusters, they are able to generate learning engagement in a timely manner according to the assignment requirements,

ISSN 2522-6398 Vol. 7, Issue 10: 223-234, DOI: 10.25236/FER.2024.071036

following the sequential and progressive cognitive patterns. Analysis of end-of-course grades shows that the average scores of students in this cluster are slightly lower than those in the High-performance engagement cluster, ranking second in overall performance. However, there is a considerable difference in scores, with a greater proportion of high-scoring students compared to the other five clusters.

High-performance engagement: Students in this cluster have a slightly delayed response time to programming tasks compared to Cluster #1, but the average duration of task completion and the number of program evaluation status times are significantly lower. Analysis of grades shows that these students achieve the highest average score in the end-of-course exams, with relatively small differences in performance, indicating a higher level of learning effectiveness.

Random Participation: Students in this cluster have an average response time to programming tasks of about 36 days, which is about two weeks later than Cluster #1 and Cluster #2, indicating a slower response speed to tasks. The average duration of task completion and the number of program evaluation status times are significantly lower than Cluster #1, Cluster #2, and Cluster #3, but slightly higher than Cluster #5. These behavioral engagement characteristics indicate that these students lack a relatively stable learning pattern and demonstrate a higher level of randomness in their participation in online learning. Analysis of grades shows that students in this cluster rank third in average performance, with a considerable difference in grades.

Low-performance engagement: Students in this cluster have an average response time to programming tasks that is similar to Cluster #3, but the average duration of task completion and the number of program evaluation status times are significantly higher than Cluster #3. This indicates that Cluster #4 and Cluster #3 share a similar task response pattern, but Cluster #4 students have a much higher level of programming learning engagement. Analysis of grades shows that these students have lower average scores and pass rates compared to the first three clusters, demonstrating a pattern of high investment but low performance.

Low-engagement: Students in this cluster have the slowest response time to programming tasks among the five clusters. The response time has extremely low correlation with the task release time, with the highest standard deviation. The average duration of task completion and the number of program evaluation status times are significantly lower than the previous four clusters. This indicates that students in this cluster have the lowest overall learning engagement and a very short learning exploration process. Analysis of grades shows that a large proportion (approximately 80%) of students in this cluster have failed grades in the end-of-course exams.

4. Discussion: Teaching Applications of the Assessment Model

Utilizing the data analysis results of the assessment model in online programming learning, effective analysis and prediction of students' learning performance can be conducted, enabling timely identification of students facing learning difficulties or issues. Timely teaching interventions can be implemented for addressing problematic learning behaviors, thereby providing the possibility for designing a hybrid teaching model that aligns with students' learning behavior characteristics [24][25] and supports precise teaching evaluations based on learning process behaviors.

4.1. Prediction or Early Warning of Academic Performance

The analysis results from section 2.1 indicate that factors such as programming task response time and the number of program submissions in three evaluation states can be used to predict academic performance. In teaching, by leveraging the stage-wise learning behavior data recorded by the OJ system, students exhibiting abnormal learning behaviors can be identified, marked, and communicated to teachers, enabling timely communication and intervention. Personalized teaching guidance can be provided to address specific issues faced by students, correcting incorrect or passive learning behaviors, offering additional assistance and support to overcome learning difficulties. By analyzing the average number of program submissions for each task and identifying the types of errors in program evaluation states, the understanding of corresponding knowledge units in teaching can be continuously monitored, allowing for timely adjustment of teaching strategies or supplementation of teaching content.

4.2. Designing Personalized Teaching Activities

Based on the behavioral characteristics of the five types of programming learning engagement

ISSN 2522-6398 Vol. 7, Issue 10: 223-234, DOI: 10.25236/FER.2024.071036

patterns, teachers can conduct group teaching activities and adopt suitable teaching strategies and methods for students with different learning behavior characteristics. For students with random engagement, low-performance engagement, or low- engagement behaviors, teachers need to deeply understand the reasons behind the slow average response time for programming tasks and the challenges faced in completing specific programming tasks. By analyzing the program codes of students in the last submission with errors in evaluation state, teachers can grasp the specific situations encountered by students in task completion, provide effective teaching guidance through teaching Q&A sessions, thematic discussions, etc., and encourage students to actively engage in discussions with teachers and peers, thus enhancing learning enthusiasm. For students who have submitted programs multiple times without passing the evaluation, classmates with progressive high-engagement and high-performance engagement behavioral characteristics can be assigned as course assistants to provide academic support to those students who cannot complete learning tasks within the specified time frame, improving their learning efficiency and facilitating peer discussions. Students with progressive high-engagement and high-performance engagement behavioral characteristics usually complete course learning tasks with high quality; teachers can further understand their advanced learning needs, assign challenging programming tasks, share course reading materials, arrange challenging assignments, and provide them with expanded learning resources and teaching guidance to cultivate their comprehensive thinking and innovative skills, while selecting students as participants for programming design competitions.

4.3. Basis for Diagnostic Teaching Assessment of Student Situations

By utilizing the data analysis results of the programming learning engagement assessment model, teachers can efficiently analyze the overall response status of each programming task, obtain statistical results of learning behavior characteristic types presented in visual formats such as tables, bar graphs, etc., at the class level. These data analysis results serve as the basis for teachers to conduct diagnostic assessment of student situations and design teaching accordingly. Based on statistical data and analysis results, teachers can quickly identify the program codes of students with higher task completion quality and share them as excellent assignment examples with all students. For tasks with abnormal submission frequencies, teachers can review student submissions based on program evaluation state types, assess the difficulty levels of programming tasks and differences in students' programming capabilities, summarize common issues in coding tasks for teaching feedback, and provide learning recommendations.

5. Conclusion and Future Perspectives

In the rapidly developing learning environment of educational informatization, significant changes have taken place in people's learning methods. The use of various LMS for online learning or blended learning has become the norm. By collecting and analyzing student learning activity data from LMS systems, studying students' learning characteristics and cognitive evolution process holds important significance for changing teaching models and enhancing teaching quality. This paper conducted modeling analysis on program submission behavior data in OJ systems, establishing a programming learning engagement assessment model, providing an effective way for teachers to understand the level of student learning engagement. The model utilizes the features of program submission records to extract programming learning engagement event attribute values from OJ systems, transforming programming learning behavior sequences into programming learning engagement event sequences. Subsequently, through the application of correlation analysis, cluster analysis, and other data mining techniques, programming learning behaviors and learning patterns were deeply studied. This model can analyze the long-term trends and patterns in students' programming learning behaviors. In OJ system teaching, it can help teachers effectively evaluate or predict student learning performance, guide teachers in designing personalized learning activities, improve students' learning effectiveness and satisfaction, and promote the development and innovation of online programming education.

This paper introduces the concepts of PLEE and programming learning engagement event sequences, and uses statistical calculation methods to obtain attribute feature values of the PLEE. However, this method overlooks the temporal and unequal length characteristics of program submission behavior sequences, which to some extent impacts the accuracy of measuring the similarity of program submission behavior sequences. Future research could consider introducing dynamic time warping (DTW) algorithms to enhance the model proposed, to better address the issue of measuring the similarity of program submission behavior sequences, thereby improving the accuracy and applicability of the model.

ISSN 2522-6398 Vol. 7, Issue 10: 223-234, DOI: 10.25236/FER.2024.071036

Acknowledgements

This work was supported by University Innovation Fund Project of Gansu (No. 2022B-092), 2022 Northwest Normal University Curriculum Assessment Reform Project.

References

[1] Chai Y M, Lei Ch F. A survey of online learning behaviour research based on data mining technology[J]. Applied Research of Computers, 2018,35(05): 1287-1293.
[2] Aydoğdu Ş. Predicting student final performance using artificial neural networks in online learning environments[J]. Education and Information Technologies, 2020,25(03): 1913-1927.
[3] Qiu F, Zhang G, Sheng X, Jiang L, Zhu L, Xiang Q, Chen P K. Predicting students' performance in e-learning using learning process and behaviour data[J]. Scientific Reports, 2022,12(1): 453.
[4] Tong Y, Zhan Z H. An evaluation model based on procedural behaviours for predicting MOOC learning performance): students' online learning behaviour analytics and algorithms construction[J]. Interactive Technology and Smart Education, 2023,20(03): 291-312.
[5] Lin X. Clustering Research Based on Feature Selection in The Behaviour Analysis of MOOC Users [J]. Journal of Information Hiding and Multimedia Signal Processing, 2019,10: 147-155.
[6] Yan N, Au O T S. Online learning behaviour analysis based on machine learning[J]. Asian association of open universities journal, 2019,14(02): 97-106.
[7] Li Z H, Zhang Z L, Liu H. Research on evaluation method of online learning input based on Model integration [J]. China Distance Education, 2020(10): 9-16+60.
[8] Hu Q T, Wu W Y, Feng G T, Pan F, Qiu K X. Research on interpretability analysis of multimodal learning behaviour supported by deep learning[J]. Journal of Audio-Visual Education Research, 2021.42(11): 77-83.
[9] Wasik S, Antczak M, Badura J, Laskowski A, Sternal T. A Survey on Online Judge Systems and Their Applications [J]. ACM Computing Surveys (CSUR), 2018,51(01): 1-34.
[10] Xu B, Yan Sh, Jiang X, Feng S. SCFHpp: A student analysis model to identify students' programming levels in online judge systems[J]. Symmetry, 2020,12(4): 601.
[11] Bilegjargal D, Hsueh N L. Understanding Students' Acceptance of Online Judge System in Programming Coursespp, A Structural Equation Modelling Approach[J]. IEEE Access, 2021,09: 152606-152615.
[12] Zhang Y, Li Z, Du B, Wu Y, Jiang H. Data Analysis of Online Judge System-Based Teaching Model[C]. In International Conference on Computer Science and Education, edited by W. Hong, and Y. Weng (2022): 531-543, Singapore. Springer Nature Singapore.
[13] Wang J Sh, Lin P Ch, Tang Zh Y, Chen S. How problem difficulty and order influence programming education outcomes in online judge systems[J]. Heliyon, 2023,9(11): e20947.
[14] Yu J H, Chang X Z, Liu W, Huan Z. An online integrated programming platform to acquire students' behaviour data for immediate feedback teaching[J]. Computer Applications in Engineering Education, 2023, 31(3): 520-536.
[15] Chuang Y, Chang T. H. Y. Analyzing Novice and Competent Programmers' Problem-Solving Behaviours Using an Automated Evaluation System[J]. Science of Computer Programming, 2024, 103138.
[16] Sunday K, Ocheja P, Hussain S, Oyelere S, Samson B, Agbo F. Analyzing student performance in programming education using classification techniques[J]. International Journal of Emerging Technologies in Learning (iJET), 2020,15(02): 127-144.
[17] Mai T T, Bezbradica M, Crane M. Learning behaviours data in programming education. Community analysis and outcome prediction with cleaned data[J]. Future Generation Computer Systems, 2022, 127: 42-55.
[18] Rico-Juan J R, Sánchez-Cartagena V M, Valero-Mas J J, Gallego A. Identifying Student Profiles Within Online Judge Systems Using Explainable Artificial Intelligence[J]. IEEE Transactions on Learning Technologies, 2023,16(6): 955-969.
[19] Fu Q, Bai X, Zheng Y, Du R, Wang D, Zhang T. VisOJ: Real-time visual learning analytics dashboard for online programming judge[J]. The Visual Computer, 2023,39(6): 2393-2405.
[20] Liu Y, Zhu R, Gao M, Personalized Programming Guidance Based on Deep Programming Learning Style Capturing[C]. In: Hong, W, Kanaparan, G. (eds) Computer Science and Education. Computer Science and Technology. ICCSE 2023. Communications in Computer and Information Science, 2023, Springer, Singapore.
[21] Wu F T, Zhang Q. Learning behaviour engagement: definition, analytical framework and theoretical model[J]. China Educational Technology, 2018(01): 35-41.

ISSN 2522-6398 Vol. 7, Issue 10: 223-234, DOI: 10.25236/FER.2024.071036

[22] Jiang T C, Li J Y, Lv H Zh. Research of similarity model based on learning behaviour time series[J]. Journal of Qiqihar University (Natural Science Edition), 2019,35(6): 1-3+7.
[23] Lin X Q. Application of K-means clustering algorithm to online learning behaviour research in the era of big data[J]. Electronic Design Engineering, 2021.29(18): 181-184+193.
[24] Bai L, Hu Y L, Zheng L. Design and practice of blended teaching model based on Learning Behaviour Analysis[J]. Computer Engineering and Science, 2018,40: 42-46.
[25] Wang Z R. Design and practice of the blended learning model based on an online judge system[J]. Journal of Computer Education, 2018(8): 126-129.