ISSN: 1813-4890

Multi-source Learning Data Driven Programming Exercise Recommendation System: Wiscode

Zhao Xinyi, Chen Shuya, Yang Mengyuan, Man Xinyu, Tian Yuke, Cao Yiting,

Bai Kai*

School of Computer Science, Yangtze University, Jingzhou 434023, China

Abstract

With the digital development of programming education, online teaching platforms continuously accumulate multi-source learning behavior data including code submissions, exercise answers, error types, learning processes and human-computer interactions. Aiming at the problem that traditional recommendation methods rely on single scores or fixed rules and fail to dynamically depict students' abilities, this paper proposes a personalized programming exercise recommendation method integrating multi-source learning behavior data combined with the intelligent teaching auxiliary system "Zhima Liangshi". The proposed method constructs learner feature models and exercise resource feature models, and establishes a three-dimensional weighted scoring mechanism that unifies knowledge mastery, error type matching and difficulty adaptation into recommendation decision-making. Large language models are adopted to assist resource labeling and generate interpretable recommendation explanations. Experiments based on 50 students, 75 exercises and 4077 valid submission records show that the proposed method outperforms random recommendation and rule-based recommendation in recommendation accuracy, pertinence to error types and overall learning efficiency, and can provide accurate and interpretable exercise support for programming learning.

Keywords

Intelligent teaching; programming education; personalized recommendation; learning behavior data; multi-source data fusion; ability portrait.

1. Introduction

Programming courses are characterized by strong practicality, closely connected knowledge points and complex error types. Students generate massive recordable behavioral data during the learning process, such as code submission times, test pass status, scores, time consumption, error causes, solution browsing, hint requests and dialogues with AI assistants. These data can not only reflect whether students can solve problems, but also further reveal their weak knowledge points, thinking obstacles and learning habits. Transforming these scattered data into ability portraits for teaching decision-making and recommending appropriate exercises accordingly is a critical issue to be solved by intelligent teaching systems.

Traditional programming exercise recommendation mostly adopts rule matching or collaborative filtering. Rule matching pushes exercises according to curriculum progress, grades or question difficulty. It is easy to implement but cannot reflect individual differences among students. Collaborative filtering generates recommendations based on historical behaviors of similar students, yet it suffers from cold start, data sparsity and insufficient interpretability [1,2]. More importantly, most existing methods mainly take single indicators such as pass rates and learning duration as inputs, ignoring code quality, error types,

ISSN: 1813-4890

knowledge mastery and dynamic learning processes, which makes it difficult to fully reflect the real learning status of students.

In programming education scenarios, recommendation quality directly affects students' learning experience. Overly simple exercises lack challenges, while overly difficult ones may lead to frustration. An ideal recommendation should fall within the zone of proximal development matching students' current abilities, which can compensate for weak knowledge points while maintaining moderate challenges. Therefore, focusing on the "Zhima Liangshi" system, this paper proposes a multi-source learning behavior data fusion method to construct student ability portraits and designs a personalized programming exercise recommendation strategy. This paper mainly addresses three core problems: unified representation of multi-source heterogeneous data, dynamic evaluation of student abilities, and balancing pertinence, adaptability and interpretability of recommendation results.

2. Related Work

Research on programming education recommendation has evolved from rule-driven to data-driven paradigms. Early methods relied on teachers' experience or curriculum paths to recommend exercises in the order of knowledge points, question difficulty and assignment schedules. They are suitable for standardized teaching but respond poorly to individual differences. Collaborative filtering improves personalization by inferring exercise preferences through similar student behaviors, yet it performs unstably for new students, new exercises and sparse interaction scenarios. Knowledge tracing models further connect students' answer sequences with knowledge states to dynamically estimate mastery probabilities, but they mainly take binary correct/incorrect signals as inputs and make insufficient use of code complexity, error types and debugging processes.

In recent years, knowledge graphs, graph neural networks and representation learning have been applied to construct "student-exercise-knowledge point" relational models, which help depict dependencies between knowledge points and correlations among exercises. Code representation models such as CodeBERT also improve the system's semantic understanding of source code. However, these methods usually require large-scale training data, adopt complex model structures, and generate recommendations that are hard to interpret for teachers and students. In educational applications, recommendation systems should not only pursue prediction accuracy, but also reflect teaching objectives, difficulty gradients and learning loads. Thus, recommendation paradigms for e-commerce or content scenarios cannot be directly applied.

Large language models bring new technical possibilities to educational recommendation [3]. Such models can read exercise descriptions, standard answers and student codes to assist in extracting knowledge points, judging difficulty levels, identifying error causes and generating natural language explanations. Nevertheless, directly relying on large language models to determine recommendation lists may lead to unstable outputs, high costs and insufficient controllability. A more feasible solution is to treat large language models as semantic feature extractors and explanation generators, while computable recommendation algorithms complete core ranking decisions, with models supplementing labels, reasoning and feedback. The framework adopted in this paper combines statistical features of behavioral data with semantic capabilities of large language models to form a hybrid recommendation framework suitable for practical teaching systems.

3. System Architecture and Data Sources

The "Zhima Liangshi" system targets the ternary collaborative teaching scenario of "Teachers-AI-Students", designed with separation of front-end and back-end and microservice

ISSN: 1813-4890

architecture. The presentation layer adopts Vue 3, Element Plus and ECharts to display learning progress, ability portraits and recommendation results. The business logic layer is built on Python 3.10 and FastAPI, undertaking user authentication, model service access, ability evaluation, intelligent tutoring and recommendation calculation. The data persistence layer uses MySQL to store structured data of users, courses and exercises, MongoDB to store dialogues and behavior logs, and Redis to provide cache and task queue support. The system is deployed in containers via Docker Compose for convenient expansion and maintenance.

The data used in this paper are mainly derived from interactions between students and exercises, divided into four categories [4]. The first category is programming answer data, including exercise ID, submission time, correctness, scores, attempt times and time consumption, which serve as the basis for calculating knowledge mastery. The second category is error behavior data. The system records eight types of errors including syntax errors, logic errors, runtime errors, type errors, index out-of-bounds, timeouts, output errors and partially correct answers, to identify students' frequent error patterns. The third category is learning process data, such as time spent per exercise, submission intervals, learning frequency and active periods, used to analyze learning engagement and persistence. The fourth category is interactive behavior data, including solution browsing, hint requests and AI dialogues, providing supplementary evidence for analyzing students' confusion.

Experimental data are real learning records generated by the system from January to February 2026, covering 50 students, 75 Python programming exercises, 15 core knowledge points and 4077 valid submission records. Students are divided into four ability levels: excellent, good, medium and weak. Exercise difficulty ranges from level 1 to level 5, and knowledge points cover basic syntax, control structures, functions, data structures, file operations, exception handling, object-oriented programming, recursion, sorting and modules. In the data preprocessing stage, invalid submissions are eliminated first, followed by aggregating correct rates, average scores, frequency of error types and attempt behaviors by student and knowledge point to provide inputs for ability portrait construction and recommendation calculation.

Table 1 Overview of Experimental Dataset

Indicator	Value	Description	Indicator
Number of participating students	50	Covering four ability levels: excellent, good, medium, weak	Number of participating students
Scale of exercise bank	75	43 programming questions, 32 multiple-choice questions	Scale of exercise bank
Number of knowledge points	15	Covering core modules of Python programming	Number of knowledge points
Total submission records	4077	Approximately 81.5 records per student	Total submission records
Overall correct rate	45.1%	Indicating good discrimination of the dataset	Overall correct rate
Number of error types	8	Syntax, logic, runtime, type, index, timeout, output, partially correct	Number of error types

4. Personalized Programming Exercise Recommendation Method

Modeling learner features is the prerequisite for recommendation. This paper describes students from five dimensions: basic information, ability features, behavioral features, knowledge point features and interactive features. Ability features include overall correct rate, average score, total submission times and ability level. Behavioral features cover average time consumption, average attempt times, learning frequency, learning intervals, active periods and

ISSN: 1813-4890

distribution of error types. Knowledge point features contain mastery degree of each knowledge point, weak knowledge points, advantageous knowledge points and untouched knowledge points. Interactive features include completed exercises, attempted but unsolved exercises, correct rates by question type, correct rates by difficulty and feedback on historical recommendations. With these features, the system can judge students' proficient and weak knowledge points as well as their frequent error types.

To unify heterogeneous features for ranking decisions, this paper constructs a multi-factor weighted decision scoring model. Let the full candidate exercise set be P. For any candidate exercise $p \in P$,this paper quantifies scores from three core dimensions: knowledge point matching degree, error type pertinence and difficulty self-adaptation, and calculates the final comprehensive recommendation score $S_{total}(p)$ combined with a rework mechanism for historical wrong questions.

Let M denote students' current mastery degree of the knowledge point covered by exercisep$((0 \le M \le 100))$. The mastery degree calculation formula is: $M = CorrectRate \times 0.6 + (AverageScore/100) \times 0.4$ Following the teaching principles of compensating weaknesses, exploring new knowledge and consolidating strengths, the knowledge point matching score $S_{kp}(p)$ is defined as a piecewise function:

$$S_{kp}(p) = \begin{cases} 50 \times \left(1 - \frac{M}{100}\right), 0 \le M < 55\ (Weakknowledgepointwithatleast2historicalsubmissions) \\ 30, pbelongstountouchedknowledgepoints \\ 15 \times \left(1 - \frac{M}{100}\right), 55 \le M < 100\ (Learnedbutunmasteredknowledgepoints) \\ 2, M \ge 100\ (Fullymasteredknowledgepoints) \end{cases} \quad (1)$$

Let $E_{student}$denote the set of frequent error types of students under the current knowledge point, and $E_{exercise}$denote the set of error types accumulated from all students' answers to exercise p. Define the exercise type feature variable$T_p \in \{choice, program\}$. The error type matching score consists of an overlap score $S_{overlap}(p)$and a question type adaptive correction term $\Delta S_{type}(p)$:

$$S_{error}(p) = S_{overlap}(p) + \Delta S_{type}(p) \quad (2)$$

$S_{overlap}(p)$is linearly weighted by the proportion of specific error types made by students under this knowledge point, with a full score of 30.

Delta $S_{type}(p)$ adjusts the teaching adaptability between question types and error types:

$$\Delta S_{type}(p) = \begin{cases} 15, T_p \equiv programandstudents'frequenterrorsarelogic/runtimeerrors \\ 10, T_p \equiv choiceandstudents'frequenterrorsaresyntax/typeerrors \\ 0, Othersituations \end{cases} \quad (3)$$

Let the absolute difficulty level of exercise p be $D_p \in \{1,2,3,4,5\}$. Students are divided into four ability levels based on their overall correct rate: Excelent $(\ge 70\%)$, Good $(55\% - 70\%)$, Medium $(40\% - 55\%)$, Weak $(< 40\%)$, with corresponding ideal recommended difficulty intervals $[D_{min}, D_{max}]$: Weak → [1,2]; Medium → [1,3]; Good → [2,4]; Excellent → [3,5]. If the exercise difficulty falls within the interval, positive incentive is generated based on difficulty value:

ISSN: 1813-4890

$$S_{diff}(p)=\begin{cases}10+10\times\left(\dfrac{D_p-D_{\min}}{D_{\max}-D_{\min}+\epsilon}\right), D_p\in[D_{\min},D_{\max}]\\ 0, D_p\notin[D_{\min},D_{\max}]\end{cases} \quad (4)$$

where ϵ is an infinitesimal positive number to avoid division by zero, set as $\epsilon = 10^{-6}$.

An explicit correction operator is introduced to strengthen the wrong question elimination mechanism:

$$S_{retry}(p)=\begin{cases}8, Studentsattemptedthisexercisebutfailedinhistory\\ 0, Othersituations\end{cases} \quad (5)$$

The final comprehensive score formula for candidate exercises is:

$$S_{total}(p)=S_{kp}(p)+S_{error}(p)+S_{diff}(p)+S_{retry}(p) \quad (6)$$

The optimization objective of personalized recommendation is formulated as follows: after filtering out exercises that students have fully solved set P_{passed}, select the Top-10 exercises to form recommendation list R with maximized total comprehensive scores:

$$Maximize \sum_{p\in\mathbf{R}} S_{total}(p) \quad s.t. \quad \mathbf{R}\subset(P-P_{passed}),\ |\mathbf{R}|=10 \quad (7)$$

5. Experimental Design and Result Analysis

To verify the effectiveness of the proposed method, three comparison groups are set: random recommendation, rule-based recommendation and the proposed method [5,6]. Random recommendation randomly selects 10 exercises excluding finished items, serving as the lower performance bound. Rule-based recommendation determines the ideal difficulty range according to students' overall correct rates and randomly selects exercises within this range. The proposed method calculates scores of candidate exercises via the three-dimensional weighted scoring model and generates recommendations with the top 10 exercises. Evaluation indicators include Precision@10, weak knowledge point coverage rate, difficulty adaptability, error type pertinence, recommendation diversity and overall learning efficiency[7]. Overall learning efficiency is weighted by the improvement of correct rate and knowledge mastery, measuring the promotion effect of recommendations on learning outcomes.

Table 2 Overall Comparison of Recommendation Performance

Evaluation Indicator	Random Recommendation	Rule-based Recommendation	Proposed Method
Recommendation Precision	0.6000	0.6300	0.9700
Weak Knowledge Point Coverage	0.5825	0.5440	0.6361
Difficulty Adaptability	0.6672	0.9420	0.7224
Error Type Pertinence	0.7200	0.7490	0.9840
Recommendation Diversity	0.7900	0.7160	0.4660
Overall Learning Efficiency	0.6427	0.6818	0.8011

Results in Table 2 show that the Precision@10 of the proposed method reaches 0.9700, significantly higher than 0.6000 of random recommendation and 0.6300 of rule-based recommendation. Its error type pertinence hits 0.9840, rising by 31.4% compared with rule-

ISSN: 1813-4890

based recommendation. The weak knowledge point coverage rate is 0.6361, exceeding both baselines. The overall learning efficiency is 0.8011, 17.5% higher than 0.6818 of rule-based recommendation. This indicates that integrating knowledge mastery and error type information into recommendations can accurately target students' real learning demands and focus training on their weakest ability links.

It is worth noting that the difficulty adaptability of the proposed method is 0.7224, lower than 0.9420 of rule-based recommendation, and its recommendation diversity (0.4660) is also inferior to the other two methods. This does not imply algorithm failure, but reflects the design trade-off of "focusing on making up knowledge weaknesses". Rule-based recommendation only selects exercises within a fixed difficulty scope, so it naturally performs better on the single indicator of difficulty adaptability. Random recommendation covers scattered knowledge points, resulting in higher diversity. The proposed method prioritizes knowledge point matching weight; when exercises corresponding to students' weak knowledge points slightly deviate from the ideal difficulty range, the system still recommends them to fix knowledge loopholes. In experiments, the average correct rate improvement brought by the proposed method is 0.3464, 1.69 times that of rule-based recommendation (0.2046). The average mastery improvement is 7.08, 1.73 times that of rule-based recommendation (4.09), proving that focused training brings higher learning gains.

Table 3 Comparison of Learning Efficiency Improvement

Indicator	Random Recommendation	Rule-based Recommendation	Proposed Method	Improvement vs Rule-based
Average Correct Rate Improvement	0.1825	0.2046	0.3464	+69.3%
Average Knowledge Mastery Improvement	3.65	4.09	7.08	+73.1%

Table 3 demonstrates obvious gaps in learning efficiency under three recommendation strategies. Compared with random and traditional rule-based recommendation, the proposed method achieves prominent advantages. The average correct rate improvements of random and rule-based recommendation are 0.1825 and 0.2046 respectively, while the proposed method reaches 0.3464, an increase of 69.3% against rule-based recommendation. In terms of average knowledge mastery improvement, the three groups are 3.65, 4.09 and 7.08 in sequence, with the proposed method rising by 73.1% relative to rule-based recommendation. It fully verifies that the personalized recommendation strategy can effectively strengthen learning effects and greatly improve students' knowledge mastery and answer accuracy.

6. Conclusion and Future Work

Aiming at the defects that existing programming exercise recommendation relies on single data and fails to dynamically depict students' abilities, this paper proposes a personalized programming exercise recommendation method fused with multi-source learning behavior data. Taking the "Zhima Liangshi" system as the application scenario, the method comprehensively utilizes code submission records, answer results, error types, learning processes and interactive behavior data to construct student ability portraits and exercise feature portraits. A three-dimensional weighted scoring mechanism covering knowledge point matching, error type matching and difficulty adaptation is designed to generate recommendation lists. Experimental results prove that the proposed method can significantly

improve recommendation accuracy, error type pertinence and overall learning efficiency, possessing favorable practical application value.

Future research can be optimized from three aspects. First, introduce time decay, confidence estimation and sequence modeling to make ability portraits better reflect students' recent learning changes. Second, further enhance the analytical capability of large language models for non-standard codes, complex errors and learning dialogues to reduce noise in semantic features. Third, combine more educational psychology theories to optimize the quantitative measurement of the zone of proximal development, striking a stable balance among precise weakness compensation, difficulty adaptation and recommendation diversity. With accumulating data and system iteration, personalized recommendation will further promote the transformation of programming teaching from unified exercises to individualized instruction.

Acknowledgements

This work was supported by the Research Project of Hubei Higher Education Association under Grant No. 2025XDZ022 (Research on the Innovative Application of Large AI Model Technology in Computer Major Education — A Case Study of Artificial Intelligence Series Courses) and the Teaching Reform Research Project of Hubei Provincial Department of Education under Grant No. 2024276 (Research and Practice on the Construction of Artificial Intelligence Series Courses for Postgraduates Majoring in Computer Science Empowered by Large Models).

References

[1] Xi, P. H., Wu, X. Z., Jiang, W. C., et al. (2025). Review on personalized educational resource recommendation. Computer Science.

[2] Zhong, S. C., Yang, L., & Fan, J. R. (2025). Data-driven personalized learning: Practical problems, inherent logic and implementation paths. E-Education Research, 46(1), 13-19+33.

[3] Tang, Q. W. (2025). Research on high school programming teaching model supported by generative artificial intelligence [Master's thesis]. Yangzhou University.

[4] Wang, C. K., Ren, G., & Wei, Y. (2025). Research on multi-source heterogeneous learning behavior data fusion model based on Spark parallel architecture. Journal of Henan Institute of Technology, 33(5), 14-18.

[5] Zhao, D. (2021). Primary and secondary school programming teaching management and resource recommendation system [Master's thesis]. Shanxi University.

[6] Li, Z. (2021). Design and research of programming question recommendation system based on cognitive diagnosis [Master's thesis]. Liaoning University.

[7] Luo, Z. Z. (2023). Research on knowledge tracing and learning resource recommendation model oriented to personalized learning [Master's thesis]. Jiangxi Normal University.