Impact of Programming Language Skills in Programming Learning

Md. Faizul Ibne Amin
Dept. of Computer Science and Information Systems
The University of Aizu
Aizu-Wakamatsu, Japan
aminfaizul007@gmail.com

Md. Mostafizer Rahman
Dept. of Computer Science and Information Systems
The University of Aizu
Aizu-Wakamatsu, Japan
mostafiz26@gmail.com

Yutaka Watanobe
Dept. of Computer Science and Information Systems
Senior Associate Professor
The University of Aizu
Aizu-Wakamatsu, Japan
yutaka@u-aizu.ac.jp

Muepu Mukendi Daniel
Dept. of Computer Science and Information Systems
The University of Aizu
Aizu-Wakamatsu, Japan
mdaniel108@gmail.com

***Abstract*—In this modern era of the internet and information technology, a mentionable amount of data is generated from different sources consistently which refers to big data. This huge amount of data not only draws great attention for further research but also helps to extract different knowledge and information in various areas. The Information and Communication Technology (ICT) area is not apart from that, as the huge amount of data in this area is enhancing opportunities for further research and development. In the ICT, most of the courses especially programming-related courses are designed to improve practical skills. With the increasing demand for software engineering and other related fields, programming education or learning plays a vital role. However, in programming learning, the impact of programming language is also important to enrich the programming and technical skill. This paper aims to analyze the impact of programming language skills in programming learning by collecting real-world data from a programming course. In this paper, we used a dataset from submission logs of a programming course in an Online Judge (OJ) system. We selected the users randomly and considered single and multiple languages used for acceptance. Finally, we have presented the analysis of the overall acceptance rate, single and multiple languages used acceptance rate, and compared them. Moreover, the analytical result of this paper can help students and programmers as well as the improvement programming learning.**

***Index Terms*—Programming learning, Programming language, Online judge, Acceptance rate, Big data, Data analysis.**

I. Introduction

Over the last few years, an extensive amount of digital data are being generated by the internet of things and information technology. This big amount of data is generated from various sources including numerous sensors and cameras, mobile devices, GPS devices, and social networks. With this huge amount of data, not only a new scope of research and development is being opened but also growing the opportunity to extract new and hidden features, for further development and useful information. The information and communication technology (ICT) sector is also producing a certain amount of data every day from various sources such as learning platforms (e.g., online judges (OJ)). With this big data, research and development are being done and consistently extracting valuable features, knowledge, and information. For instance, programming models, systems analysis, languages, performance development, and algorithms are being developed by this big data.

In ICT and other engineering-related disciplines, usually, the academic courses are designed on a practical basis [1]. To improve logical and innovative thinking, problem-solving skills, implementation skills, and practical-based learning play a vital role. Computer programming is one of the examples of practical-based learning in computer science. The importance of programming learning does not only draw attention to the growing demand for ICT skills and programming education but also plays an important role in almost all other practical-based disciplines. Because, programming education is the basement of modern applications, artificial intelligence (AI), data analysis, and numerical analysis. Thus, computer programming become a very essential part of the ICT discipline specifically in computer science.

To inspire and encourage students, and novice programmers for programming learning, many e-learning systems, and online programming platform are being used. Where a large number of learning materials (e.g., problems, solutions) related to computer science are being offered. Students can use to make their learning more meaningful and study effectively to reach their desired goal by using those platforms. OJ systems are supplemental platforms to traditional programming education which contains a large number of interesting programming problems [2] that inspire students and novice programmers to continue their programming learning. The idea of OJ systems has introduced in the 1977 international collegiate programming contest (ICPC) [3].

The traditional programming learning environment in educational institutions is not enough to prepare skilled programmers due to the limited number of exercise classes, limited scope for practices, and lack of individual tutoring [4]. As one of the fundamental course is computer programming in the ICT discipline, students need to learn programming through lots of practicing and individual tutoring to enrich their programming skills and knowledge. There is no parallel of more programming practice to gain problem-solving skills and logical thinking. The OJ systems have been introduced to cope up with this issue to enhance the scope and opportunities for practice and individual learning in addition to traditional classroom-based learning.

Nowadays, many educational institutes are being used OJ systems for conducting courses related to programming, software engineering, and ICT-related disciplines [5]. Many institutes have also created automated program assessment (APA) systems to do programming learning more smoothly [6], [7]. Consequently, a mentionable amount of programming-related data (e.g., submission logs, scores, source codes) are being generated consistently which can be worthy raw materials for further programming education research and development. Therefore, the main research objective of this paper is to use programming-related data for the analysis.

In this research, our goal is to explore the impact of programming language skills in programming learning by comprehensive analysis through real-world e-learning platform data. We consider a practice-based exercise class of a basic computer programming course that consists of programming exercises and coding tests. After processing the data from OJ systems, tried to explore the impact and importance of programming language skills which are mainly derived from submission logs and verdicts (Acceptance rate). The key points of this work are as follows:

- We have presented the statistical analysis of the used language for solving specific problems.
- We have considered the single and multiple languages used for problem-solving and the acceptance rate for both cases.
- We have presented the average acceptance rate for all users based on OJ systems, single, and multiple language acceptance rates and compared them, and finally analyzed the result.
- We expect that the analysis is helpful to students and novice programmers to improve their programming learning.

The remainder of this paper is structured as follows. In section II related works are presented. Section III provides a brief description of the data source. Section IV focuses on the proposed approach. The experimental results and analysis are presented in section V. In section VI conclusion and future work are presented.

II. Related Works

In this section, some recent research work related to Educational big data analysis and the development of programming learning has been taken into account.

In research [8], the authors addressed an interesting topic for the students and beginners in programming learning, what they want to learn, what sequence they should follow for the learning, and what should be avoided initially. To do so, they have proposed a learning path and approach for the learning sequence based on the analysis of large-scale data from an OJ system. To make learning more meaningful and efficient for students and novice programmers, the authors have recommended a learning path (problem sequence for the learning process) based on the data analysis and machine learning process.

In research [1], the authors have presented the impact of practical skills on academic performance based on the analysis of large-scale data from an OJ system. They have addressed two important issues: practical skills (programming skills) and academic performance. They have considered submission logs and scores for the analysis and extracted the hidden features responsible for the practical skill. Based on that analysis, they have discussed how the analyzed result impacts academic performance. They have followed the educational data mining (EDM) and machine learning approach.

In another research [4], the authors have focused to support programming learning followed by the EDM approach. They have exploited the machine learning approach for extracting useful information from real-world problem-solving data collected from an OJ system. They have proposed a framework that contains various hidden features, patterns, rules, and knowledge, and based on the analysis and findings they have provided some useful recommendations for possible improvements to programming learning.

In the research [9], the authors have analyzed large-scale programming data and discussed various principles, and also provided some solutions. They have provided a structured overview of programming models and systems for big data analysis. In [10], the authors have approached the fuzzy recommender system for OJ including suggestions to students and learners for upcoming problems based on their past activities data, and also provided useful information to students and programmers.

In [11], the authors have focused on the learning topics and problem difficulty level in OJ systems. They have presented to learn topics automatically and also considered the difficulty level of problems. They have proceeded with a two-mode hidden Markov topic model for learning topics and problem levels by using large-scale OJ datasets. Also, they have demonstrated skill topic extraction, expertise competition, and problem recommendation for improving programming learning.

In addition, some machine learning-based models have been developed in various studies [12]–[15] to support programming learning and education.

Our approach is different from that of existing works proceedings with the impact of language skills in programming education to improve programming learning by analyzing the submission logs and verdict. The best of our knowledge, no research has been done to address this issue.

III. Data Used for the Experiment

We have conducted this research by using the data from the Aizu Online Judge (AOJ) systems. In this section, AOJ and its general features are introduced. The AOJ system [16], [17] is a very popular OJ system in Japan as well as worldwide. It contains around 3,000 problems, about 100,000 registered users, 6 million source codes, and submission logs and started its functionality in 2004 [18]. In addition, the source code of the AOJ system has been used in IBM's research project "Project CodeNet" [19]. Currently, the AOJ system is officially used for programming and algorithm-related courses (e.g., Introduction to Programming I and II, Algorithm and Data Structure, Datasets and Queries, and some other courses) at the University of Aizu, Japan. In this paper, the problem-solving data of the algorithm and data structures course are used. The targeted or main dataset as a total value of 3,932,501 is used, among them, a value of 825,401 data are used related to Algorithm and Data Structure I (ALDS1) course. Five problems of ALDS1 are selected and the total attempting value for problem-solving data is 343,205 and then selected the unique user as a value of 26,338 who solved all the five problems. Finally, 10 users have picked up for our analysis.

IV. Proposed Approach

In this section, we describe our proposed approach for the analysis which is shown in Figure 1. In the proposed approach, we proceeded by following four stages.

- In stage 1, as a good source of the huge amount of programming-related data, for the experiment, the data are collected from the AOJ.
- The data have been processed in stage 2. A total of about 4 million data are considered from AOJ. Then the data from course ALDS1 is selected for the experiment. As a part of processing and to get efficient data, only *U*ser_id, *P*roblem_id, used *L*anguages used for problem-solving, and *V*erdict is taken into account among many attributes. As five problems of the ALDS1 course has selected which are shown in Table I

TABLE I
PROBLEMS LIST

Course	Problem Id	Problem Name
ALDS1	ALDS1_1_A	Insertion Sort
	ALDS1_1_B	Greatest Common Divisor
	ALDS1_1_C	Prime Numbers
	ALDS1_1_D	Maximum Profit
	ALDS1_2_A	Bubble Sort

After that, 10 users are selected randomly who solved all the five problems and collected their problem-solving logs for the experiment.

- In stage 3, all the 10 user's total attempts, total unique languages used for the problem-solving, verdict (Accepted, Wrong Answer), and languages used for the acceptance of all 5 problems are considered.

where,

U= User, P= Problem, L= Language used for Problem-Solving, A= Acceptance, W= Wrong Verdict, TA= Total Attempt.

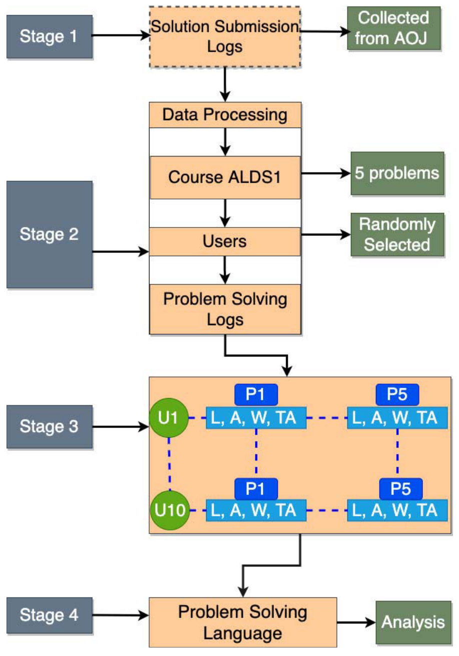

Fig. 1. Overviw of the Proposed Approach.

- In stage 4, we have focused to do an analysis by considering two points such as acceptance rate based on AOJ and acceptance rate based on languages used. For the first case, the equation for the calculation of the general acceptance rate according to AOJ is as follows: for user 1,

$$U_1 = \frac{AC}{TA} \tag{1}$$

Where, AC= Acceptance rate, TA= Total Attempts.

Thus a more general equation for the acceptance rate calculation of all users can be described as follows:

$$U_n = \sum_{i=1}^{n} \frac{AC_i}{TA_i} \tag{2}$$

After that, the acceptance rate based on the languages used for the problem-solving is calculated as follows:

$$U_1 = \frac{\frac{AC}{TA} + \frac{LU}{TUL}}{2} \tag{3}$$

and

$$U_n = \sum_{i=1}^{n} \frac{\frac{AC_i}{TA_i} + \frac{LU_i}{TUL_i}}{2} \tag{4}$$

Where, LU= Language Used, TUL= Total Unique Language Used.

The term TUL used in Equations 3 and 4 refers to unique languages that are frequently used to solve the problem by all 10 users which is including C, C++, Python, and Java.

TABLE II
STATISTICAL INFORMATION OF SELECTED USERS

User Id	Problem Id	Total Attempts	Language Used	Acceptance	Language Used for Acceptance
U_1	ALDS1_1_A	2	1(C)	1	C
	ALDS1_1_B	1	1(C)	1	C
	ALDS1_1_C	5	1(C)	1	C
	ALDS1_1_D	8	1(C)	2	C
	ALDS1_2_A	1	1(Python3)	1	Python3
U_2	ALDS1_1_A	7	1(C)	1	C
	ALDS1_1_B	1	1(C)	1	C
	ALDS1_1_C	1	1(C)	1	C
	ALDS1_1_D	4	1(C)	1	C
	ALDS1_2_A	2	1(C)	1	C
U_3	ALDS1_1_A	11	3(C, C++, Python3)	3	C(1), C++(1), Python3(1)
	ALDS1_1_B	3	2(C, Python3)	2	C(1), Python3(1)
	ALDS1_1_C	2	2(C, Python3)	1	C
	ALDS1_1_D	4	2(C, Python3)	-	-
	ALDS1_2_A	11	2(C, Python3)	2	Python3
U_4	ALDS1_1_A	4	2(Java, Python3)	3	Java(1), Python3(2)
	ALDS1_1_B	2	1(Python3)	2	Python3
	ALDS1_1_C	11	3(Java, Python, Python3)	8	Java(1), Python3(7)
	ALDS1_1_D	6	3(C++, Java, Python3)	4	C++(1), Java(1), Python3(2)
	ALDS1_2_A	9	3(C++, Java, Python3)	5	C++(1), Java(2), Python3(2)
U_5	ALDS1_1_A	7	2(C, C++)	3	C(2), C++(1)
	ALDS1_1_B	1	1(C)	1	C
	ALDS1_1_C	1	1(C)	1	C
	ALDS1_1_D	1	1(C)	1	C
	ALDS1_2_A	1	1(C)	1	C
U_6	ALDS1_1_A	19	1(C)	1	C
	ALDS1_1_B	1	1(C)	1	C
	ALDS1_1_C	1	1(C)	1	C
	ALDS1_1_D	1	1(C)	1	C
	ALDS1_2_A	1	1(C)	1	C
U_7	ALDS1_1_A	16	2(C, C++)	1	C++
	ALDS1_1_B	1	1(C++)	1	C++
	ALDS1_1_C	5	1(C++)	1	C++
	ALDS1_1_D	1	1(C++)	1	C++
	ALDS1_2_A	1	1(C++)	1	C++
U_8	ALDS1_1_A	10	1(C)	4	C
	ALDS1_1_B	2	1(C)	2	C
	ALDS1_1_C	5	1(C)	3	C
	ALDS1_1_D	1	1(C)	1	C
	ALDS1_2_A	6	1(C)	3	C
U_9	ALDS1_1_A	13	1(C)	4	C
	ALDS1_1_B	4	1(C)	3	C
	ALDS1_1_C	2	1(C)	2	C
	ALDS1_1_D	14	1(C)	4	C
	ALDS1_2_A	4	1(C)	2	C
U_{10}	ALDS1_1_A	5	1(C)	2	C
	ALDS1_1_B	2	2(C, C++)	2	C(1), C++(1)
	ALDS1_1_C	20	2(C, C++)	4	C
	ALDS1_1_D	1	1(C)	-	-
	ALDS1_2_A	3	2(C, C++)	3	C(2), C++(1)

V. Experimental Results and Analysis

In this section, the experimental results are presented, and based on the result, the features and impact of the programming language skills have been analyzed. The acceptance rate is calculated according to Equation 2 and the detailed information including total attempts, the number of languages used for problem-solving, acceptance, and the number of solutions based on the individual language used by all 10 users are shown in the Table II. According to Table II, the overall average acceptance rate of all users is 39.95%.
Next, we have considered the single and multiple languages used for problem-solving.

TABLE III
Single Language Used for Solving

User	Acceptance rate based on AOJ	Acceptance Rate based on Language
U_2	33.3%	29.16%
U_6	21.73%	23.36%
U_8	54.16%	39.58%
U_9	40.54%	32.77%
Average	37.43%	31.21%

In Table III the acceptance rate based on AOJ and based on the languages used for solutions are calculated. According to Table II the users U_2, U_6, U_8, and U_9 are used single programming language to solve problems. Equations 1 and 3 are used to calculate user's AOJ and language based acceptance rate. The average AOJ acceptance rate is 37.43% and the average acceptance rate based on language is 31.21%. The comparison is shown in Figure 2. In the case of a single language used, the average acceptance rate of AOJ is better than the acceptance rate based on the language.

From Table II, six users U_1, U_3, U_4, U_5, U_7, and U_{10} have used multiple languages to solve the problem is shown in Table IV. Using the Equations 1 and 3, the acceptance rate for both AOJ-based and language-based are calculated.

TABLE IV
Multiple Language Used for Solving

User	Acceptance rate based on AOJ	Acceptance Rate based on Language
U_1	35.29%	42.64%
U_3	25.80%	50.40%
U_4	68.75%	71.87%
U_5	63.63%	56.81%
U_7	20.83%	36.41%
U_{10}	35.48%	42.74%
Average	41.63%	50.14%

The average AOJ acceptance rate is 41.63%. On the other hand, in the case of used language for problem-solving, the average acceptance rate is 50.14%. The comparison between AOJ and language based acceptance rates are shown in Figure 3.

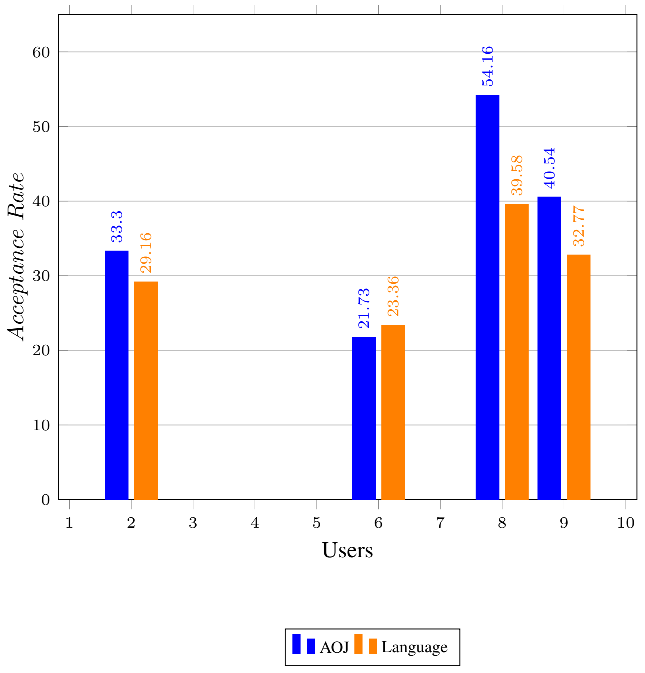

Fig. 2. Comparisons of acceptance rate between AOJ and Language for single language used

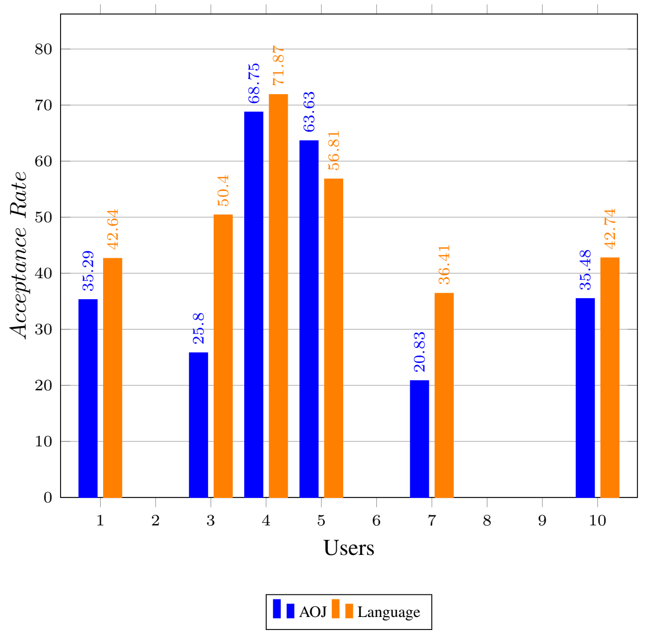

Fig. 3. Comparisons of acceptance rate between AOJ and Language for multiple languages used

Based on the experimental results according to single and multiple languages used for problem-solving, the acceptance rate is higher for the users who have used multiple languages than the users who have used single language in both AOJ-based and language-based cases. The result is shown in Table V.

TABLE V
COMPARISON OF SINGLE AND MULTIPLE USED LANGUAGE BASED ON ACCEPTANCE RATE

Used Language	Acceptance rate based on AOJ	Acceptance Rate based on Language
Single	37.43%	31.21%
Multiple	41.63%	50.14%

The illustration of acceptance rate comparison between language-based and AOJ-based for single and multiple languages used for the solution is shown in Figure 4.

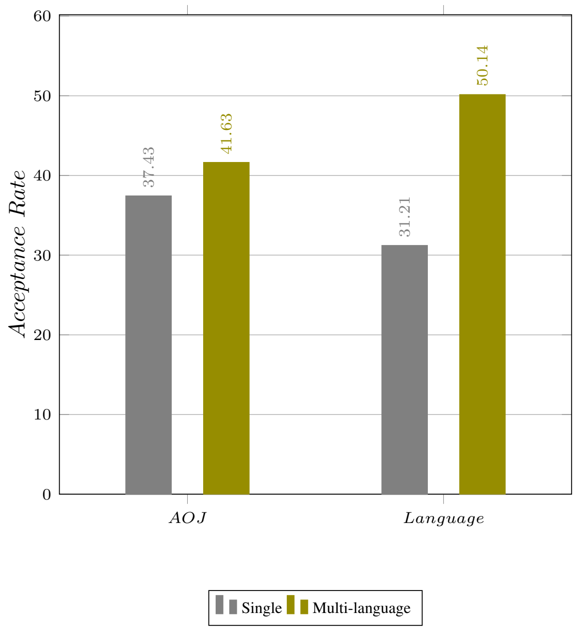

Fig. 4. Comparisons of acceptance rate between AOJ and Language

Based on the analysis, our realization is that users who have familiar with multiple programming languages for problem-solving have a higher ability to solve programming problems. Although the acceptance rate is higher for the multiple programming languages used in solutions, some interesting analyses also observed which are as follows:

Table II shows that most of the users (among 10) have used one language (e.g., C) to solve the problem. With proceeding to the subsequent problems, different programming languages have been used (e.g., C++, Python, and Java) for problem-solving. For example, the users (e.g., U_6, U_8) who have used a single language for problem-solving, the attempt rate is reduced in solving next problems. It can be seen that language skills are improving in subsequent problem-solving. In the case of multiple languages used, users are not maintained a particular set of programming languages. For example, U_{10} and U_7 have used C and C++. On the other hand, U_3 and U_4 have used C, C++, Python, and Java.

In addition, 10 more users' data were analyzed and the results are shown in Table VI. The results for these additional 10 users are similar to those in the previous discussion. When only one language was used to solve the problem, the average acceptance rate by the AOJ was 34.81% and the acceptance rate by language was 29.90%. When more than one language was used to solve the problem, the acceptance rate by AOJ was 47.39% and the acceptance rate by language was 54.18%. For both AOJ and language-based cases, the acceptance rate was higher when more than one language was used to solve the problem.

Apart from the above discussion, the learning orders of programming languages can be impactful. We have found some language orders such as C to Python and C to C++. In order to draw a solid conclusion about learning languages order, a comprehensive data analysis is required. The preliminary results and analysis of the current study can be a good basis for the future research to determine the language learning order.

TABLE VI
INFORMATION OF ADDITIONAL 10 USERS

Single language used for solving			Multiple language used for solving		
User	Acceptance rate based on AOJ	Acceptance rate based on Language	User	Acceptance rate based on AOJ	Acceptance rate based on Language
U_{11}	33.3%	29.16%	U_{12}	46.42%	48.21%
U_{13}	21.21%	23.10%	U_{15}	26.08%	38.04%
U_{14}	21.73%	23.36%	U_{16}	52.58%	66.27%
U_{19}	50%	37.5%	U_{17}	83.3%	66.6%
U_{20}	47.82%	36.41%	U_{18}	28.57%	51.78%
Average	**34.81%**	**29.90%**	-	**47.39%**	**54.18%**

VI. Conclusion and Future Work

In this paper, a statistical analysis has been presented for exploring the effects of programming language skills in programming learning. In a computer programming course (e.g., algorithm and data structure) data from AOJ have been selected for analysis. Experiments performed using the data of 5 problems and randomly selected 10 users. The analysis has been conducted based on single and multiple languages used for problem-solving including the acceptance rate for both cases. Based on the results, we have realized that multiple languages used for the solution have a higher impact on the acceptance rate than the single language used. These analyses can effectively contribute to the improvement of students, novice programmers as well as overall programming learning. In future work, we will consider more data (e.g., users and problems) for exploring additional features and co-relation using the machine learning model.

References

[1] M. M. Rahman, Y. Watanobe, R. U. Kiran, T. C. Thang and I. Paik, "Impact of Practical Skills on Academic Performance: A Data-Driven Analysis," in IEEE Access, vol. 9, pp. 139975-139993, 2021, doi: 10.1109/ACCESS.2021.3119145.

[2] R. Yera and L. Martínez, "A recommendation approach for programming online judges supported by data preprocessing techniques," Appl Intell 47, 277–290 (2017). https://doi.org/10.1007/s10489-016-0892-x

[3] M. A. Revilla, S. Manzoor and R. Liu, "Competitive learning in informatics: The UVA online judge experience", Olympiads Informat., vol. 2, no. 10, pp. 131-148, 2008.

[4] M. M. Rahman, Y. Watanobe, T. Matsumoto, R. U. Kiran and K. Nakamura, "Educational Data Mining to Support Programming Learning Using Problem-Solving Data," in IEEE Access, vol. 10, pp. 26186-26202, 2022, doi: 10.1109/ACCESS.2022.3157288.

[5] C. A. Higgins, G. Gray, P. Symeonidis, and A. Tsintsifas. 2005. "Automated assessment and experiences of teaching programming", J. Educ. Resour. Comput. 5, 3 (September 2005), 5–es. https://doi.org/10.1145/1163405.1163410

[6] I. Mekterović, L. Brkić, B. Milašinović and M. Baranović, "Building a Comprehensive Automated Programming Assessment System," in IEEE Access, vol. 8, pp. 81154-81172, 2020, doi: 10.1109/ACCESS.2020.2990980.

[7] N.A. Rashid, L.W. Lim, O.S. Eng, T.H. Ping, Z. Zainol, O. Majid, (2016). "A Framework of an Automatic Assessment System for Learning Programming", in Advanced Computer and Communication Engineering Technology. Lecture Notes in Electrical Engineering, vol 362. Springer, Cham. https://doi.org/10.1007/978-3-319-24584-3_82

[8] T. Saito and Y. Watanobe. (2020), "Learning Path Recommendation System for Programming Education Based on Neural Networks," International Journal of Distance Education Technologies. 18. 36-64. 10.4018/IJDET.2020010103.

[9] L. Belcastro, R. Cantini, F. Marozzo, et al. "Programming big data analysis: principles and solutions," J Big Data 9, 4 (2022). https://doi.org/10.1186/s40537-021-00555-2

[10] R. Y. Toledo, Y. C. Mota, and L. Martinez, (2018). "A Recommender System for Programming Online Judges Using Fuzzy Information Modeling," Informatics. 5. 10.3390/informatics5020017.

[11] W. X. Zhao, W. Zhang, Y. He, X. Xie, and J. Wen. 2018. "Automatically Learning Topics and Difficulty Levels of Problems in Online Judge Systems," ACM Trans. Inf. Syst. 36, 3, Article 27 (July 2018), 33 pages. https://doi.org/10.1145/3158670

[12] M. M. Rahman, Y. Watanobe, R. U. Kiran, K. Nakamura. (2021). "A Novel Rule-Based Online Judge Recommender System to Promote Computer Programming Education," In: Fujita, H., Selamat, A., Lin, J.CW., Ali, M. (eds) Advances and Trends in Artificial in IEA/AIE 2021, vol 12799. Springer, Cham. https://doi.org/10.1007/978-3-030-79463-7_2

[13] M. M. Rahman, Y. Watanobe, K. Nakamura, "Source Code Assessment and Classification Based on Estimated Error Probability Using Attentive LSTM Language Model and Its Application in Programming Education," Appl. Sci. 2020, 10, 2973. https://doi.org/10.3390/app10082973

[14] Y. Watanobe, M. M. Rahman, R. Kabir, and M. F. I. Amin. "Identifying algorithm in program code based on structural features using cnn classification model," Applied Intelligence, 2022.

[15] M. M. Rahman, Y. Watanobe, K. Nakamura, "A Neural Network Based Intelligent Support Model for Program Code Completion," in Scientific Programming, Vol. 2020, Article Id. 7426461. https://doi.org/10.1155/2020/7426461

[16] Y. Watanobe, Aizu Online Judge, May 2020, [online] Available: https://onlinejudge.u-aizu.ac.jp/home

[17] Aizu Online Judge: Developers Site (API), Dec. 2019, [online] Available: http://developers.u-aizu.ac.jp/index

[18] Y. Watanobe, M. M. Rahman, T. Matsumoto, U. K. Rage, and P. Ravikumar, "Online Judge System: Requirements, Architecture, and Experiences", International Journal of Software Engineering and Knowledge Engineering, Vol. 32, No. 4, pp. 1-30, 2022. doi: 10.1142/S0218194022500346

[19] Machines (IBM): Project CodeNet, 2021, [online] Available: https://github.com/IBM/Project_CodeNet