**Journal of Education and Educational Research**
ISSN: 2957-9465 | Vol. 5, No. 2, 2023

# Online Exam Cheating Detection Method for Programming Courses

**Faguo Zhou**

School of Artificial Intelligence, China University of Mining and Technology-Beijing, China

**Abstract:** Online exam cheating has always been a problem that the field of education and teaching and the teaching department focus on and research. In order to effectively detect cheating or calculate the probability of cheating, this paper proposes an algorithm for calculating the probability of cheating in the exam based on the rules and codes similarity. According to the current situation of programming courses and the actual situation of the exam, it can be judged whether there is cheating. The implementation of the cheating detection algorithm can curb the plagiarism phenomenon of online exams and purify the discipline of the test style.

**Keywords:** Online Exam; Codes Similarity; Cheating Detection.

## 1. Introduction

In the field of natural language processing, similarity calculation is a basic and core research content, which has long been a hot spot and difficult point for natural language processing researchers [1]. At the same time, similarity calculation has a wide range of application prospects, such as information retrieval, question answering robots, paper anti-plagiarism detection [2][3], etc. In recent years, due to the impact of the coronavirus pandemic (COVID-19), online examinations have become a common way to evaluate courses. How to detect and avoid irregularities or cheating in the online examination process, so as to ensure the fairness and impartiality of the online examination, has become a very important research topic.

## 2. The Importance of Online Exams

According to the "Overall Plan for Deepening the Reform of Education Evaluation in the New Era" issued by the State Council of the People's Republic of China, emphasizing the construction principle of "improving outcome evaluation, strengthening process evaluation, exploring value-added evaluation, improving comprehensive evaluation, and improving the scientificity, professionalism and objectivity of educational evaluation", many colleges and universities, including this university, adopt "multi-stage and multi-goal" teaching tests in the teaching process, and timely feedback the results and effects of relevant tests to both teaching parties. The key to the full implementation of the developmental assessment mechanism lies in the transformation of the examination method, that is, from the traditional examination method to the online examination method. However, this will involve the solution of a series of key issues such as the construction of the test question bank, the construction of the online examination system, the examination appointment management system, and the information statistics and teaching feedback system.

For practical exams, especially programming courses, data structures and algorithms, Online Judge is currently unable to test other types of questions. Meanwhile, during the examination, program fill-in-the-blank, function design, class design, algorithm design and other questions, due to the subjectivity of the teacher and the limitation of the reference answer, may lead to the situation that the student fills in correctly and the student cannot score or underscores the question due to the wrong marking error, and there may also be cases where the marking teacher gives extra points out of sympathy. In order to avoid this situation, for programming courses (C/C++/Java/Python, etc.), data structure and algorithm design, compilation principles, operating systems and other courses involve subjective questions, but can be objectively evaluated, more and more programs based on real-time evaluation online examination (experimental assistance) system is introduced, and the most basic and classic is the program design online evaluation system. It is commonly used in programming competitions such as the ICPC (International Collegiate Programming Competition).

Online Judge System is an online system used to evaluate the correctness of computer programs in real time. Users can submit the source code of a variety of computer language programs (C/C++/Java/Python, etc.) online, and the system will evaluate the source code online to verify its correctness. The online evaluation system can provide students in the school with a programming competition and practice platform, and store various programming topics; It can promote the development of computer teaching and provide evaluation and practice systems for computer programming experiments and practical courses.

## 3. Online Exam Cheating Detection Method

For programming courses, the main submissions for competitions, experiments, or exams are code. As cheaters constantly update their cheating methods, such as changing variable names in the program or defining more redundant variables, making cheating detection increasingly difficult [4] [5]. The ranking and submission records of a C programming test ranking and submission records are shown in Figure 1 below.

| Rank | Solved | Penalty | A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 8 | 06:45:41 | 00:10:59 | 00:14:52 | 00:17:14 | 01:21:58(-1) | 00:22:37 | 02:11:17 | 00:29:41 | 01:17:03 |
| 2 | 8 | 08:41:32 | 00:20:09 | 00:23:01 | 00:25:09 | 01:14:04(-5) | 00:47:49 | 01:46:16 | 00:58:49 | 01:06:15 |
| 3 | 8 | 10:21:20 | 00:12:00 | 00:08:07 | 00:09:23 | 02:18:50 | 01:13:38 | 03:19:55 | 01:06:53 | 01:52:34 |
| 4 | 8 | 10:25:35 | 00:05:56 | 00:12:30 | 00:18:47 | 02:25:10(-1) | 00:47:13 | 03:27:15 | 00:55:41 | 01:53:03 |
| 5 | 8 | 10:50:17 | 00:14:41(-1) | 00:31:36(-2) | 00:08:34 | 01:35:22(-4) | 00:09:37 | 02:36:13 | 00:11:05 | 03:03:09 |
| 6 | 8 | 13:00:07 | 00:03:19 | 00:05:35 | 00:14:27 | 03:03:16(-2) | 00:55:30(-2) | 03:19:34 | 01:14:53(-1) | 02:03:33(-1) |
| 7 | 8 | 14:26:29 | 00:26:59(-1) | 00:42:24 | 00:57:32 | 01:40:41 | 02:01:04 | 03:06:51 | 02:12:35 | 02:58:23 |
| 8 | 8 | 16:48:56 | 01:17:53(-2) | 01:43:48(-1) | 01:41:01 | 02:04:34 | 02:01:03 | 02:46:07 | 01:21:32 | 02:52:58 |
| 9 | 8 | 17:03:35 | 00:33:53 | 00:57:06 | 01:30:04 | 02:00:57(-2) | 02:05:21 | 03:02:14 | 02:39:09 | 03:34:51 |
| 10 | 8 | 17:09:01 | 00:50:00 | 01:20:05(-1) | 01:35:50 | 02:18:01 | 02:44:41(-1) | 03:18:24 | 01:01:06 | 03:20:54 |
| 11 | 8 | 19:41:35 | 01:10:45(-2) | 01:25:18 | 01:54:24 | 02:18:55 | 02:52:47 | 03:22:50(-1) | 02:15:01 | 03:21:35 |
| 12 | 8 | 20:16:49 | 00:34:24(-6) | 00:42:01 | 00:48:32 | 03:13:38(-3) | 03:19:49 | 03:20:40(-1) | 01:56:06 | 02:41:39(-1) |
| 13 | 8 | 20:25:10 | 02:12:34 | 02:33:38(-5) | 00:19:21 | 03:01:42(-4) | 01:41:40(-3) | 01:59:18(-1) | 00:05:28 | 03:31:29(-2) |
| 14 | 7 | 08:53:40 | 00:19:00 | 00:28:39 | 00:33:45 | 02:44:08(-1) | 00:54:27 | | 01:26:00 | 02:07:41 |
| 15 | 7 | 09:39:07 | 00:09:50(-3) | 00:13:49 | 00:21:27 | (-4) | 00:55:21 | 03:20:10 | 01:09:38 | 02:08:52(-1) |
| 16 | 7 | 09:40:18 | 00:13:56 | 00:17:10 | 00:23:24 | 02:32:10(-2) | 01:12:53(-2) | | 01:15:53 | 02:04:52(-1) |
| 17 | 7 | 10:00:42 | 00:16:05(-1) | 00:17:39 | 00:24:17 | 02:25:31(-4) | 01:19:16 | | 01:40:10 | 01:57:44 |
| 18 | 7 | 11:19:29 | 00:08:40 | 00:22:03 | 00:32:05 | 01:51:05 | 03:02:05(-2) | | 02:03:15 | 02:40:16 |
| 19 | 7 | 11:44:57 | 00:17:17(-1) | 00:14:23 | 00:33:44 | 01:50:23 | 02:57:54(-3) | | 01:48:27 | 02:42:49 |
| 20 | 7 | 12:14:51 | 00:52:33(-1) | 01:26:00 | 01:12:16 | 01:43:02(-1) | 02:04:25 | 02:44:17 | 01:32:18 | |
| 21 | 7 | 13:17:01 | 01:47:00(-1) | 01:44:45 | 01:11:24 | 01:55:41(-3) | 01:19:00 | | 01:27:13 | 02:31:58 |
| 22 | 7 | 14:43:35 | 00:46:08 | 01:16:00 | 01:25:59 | 02:15:21(-2) | 02:47:08 | 03:05:29 | 02:27:30 | |
| 23 | 7 | 15:00:50 | 01:20:06(-3) | 00:52:54 | 00:55:26 | 02:40:26 | 02:23:45 | | 01:53:14 | 03:34:59(-1) |
| 24 | 7 | 16:06:44 | 00:48:54 | 01:34:00 | 01:32:14 | 03:05:19(-1) | 02:14:32(-1) | | 02:47:34 | 03:24:11 |
| 25 | 7 | 17:03:27 | 00:28:20 | 00:34:30 | 00:39:30 | 02:49:55(-1) | 03:29:05(-3) | 03:39:21 | 03:22:46(-2) | |
| 26 | 7 | 17:58:41 | 02:43:24 | 02:07:33 | 02:07:00 | 02:40:27 | 02:41:40 | (-2) | 02:39:52(-1) | 02:38:45 |
| 27 | 7 | 18:57:37 | 00:56:42(-4) | 02:35:06(-3) | 02:01:10(-4) | 03:29:47 | 02:50:01 | | 01:28:30 | 01:56:21 |
| 28 | 6 | 08:25:30 | 00:14:45 | 00:23:32 | 00:37:32 | 02:24:24 | 02:32:25 | | 02:12:52 | |
| 29 | 6 | 08:43:56 | 00:20:21 | 00:40:07 | 00:44:54 | | 03:02:23 | | 01:02:23 | 02:53:48 |
| 30 | 6 | 09:40:41 | 01:15:40 | 01:17:47(-1) | 00:14:24(-1) | 01:36:53(-1) | 02:27:35 | | 01:48:22 | |

**Figure 1.** C Programming Test Ranking and Submission Records

## 3.1. Fastest Submission Time Interval

In this paper, the average fastest submission time interval of the top N students (such as 5 for N) or the top K% (10 for K) in the exam is taken as the benchmark fastest time interval. If the fastest submission time interval of the students ranked after the exam is much smaller than the benchmark time interval, then the students may have plagiarized. The calculation of the time probability *timeSim* for recording plagiarism is as follows:

$$timeSim = 1 - \frac{individual\ fastest\ submission\ time\ interval}{benchmark\ time\ interval}$$

If the calculated result is less than 0, it is marked as 0.

For example, in Figure 1, the average fastest submission time interval for the top 5 students is 153 seconds, while the student ranked 26th has a minimum submission time interval of 33 seconds. Using this formula, the plagiarism probability for this student is calculated as 1 - 33/153 = 78.4%.

## 3.2. Number of Problems Solved in a Short Time

The fastest submission time interval alone cannot fully reflect the cheating situation. Based on this, the paper puts forward the number of short time (such as 10 minutes), and record the top N (such as N takes 5) or the top K% (K takes 10) ranking students' average short time passing number as the benchmark short time solved questions, ranking after the test and more than this number, there is a high probability of plagiarism, at this time record plagiarism probability is 1, otherwise it is 0. Take this plagiarism probability as *numSim*, the calculation method is as follows:

$$numSim = \begin{cases} 1 & if\ individual\ solved\ questions\ greater\ than\ base \\ 0 & otherwise \end{cases}$$

As shown in Figure 1, the average number of consecutive questions passed by the top 5 students within 10 minutes is 2. While the students ranked 26th after passed the H question in 2 hours, 38 minutes and 45 seconds after the test began, passed the G, D, E and A four questions in 10 minutes, which had a greater probability of cheating.

## 3.3. Jaccard Similarity

Jaccard similarity [6] (Jaccard coefficient) is a term named after Paul Jaccard and is used to measure the similarity between sets. It is defined as the size of the intersection divided by the size of the union of two sets. This concept has been extended to multisets, where duplicate elements are treated as weights. In this paper, all words that appear in the program, such as keywords, variables, constants, function names, preprocessor directives, etc. are recorded.

In this paper, Jaccard similarity is used to describe one aspect of program similarity, remember *words*(*P*) represents the total number of words appearing in the program *P*, *samewords* ($P_1$, $P_2$) represents the number of identical words appearing in the program $P_1$ and in the program $P_2$, then the Jaccard similarity based on words of the program is calculated using the following formula:

$$wordSim(P_1, P_2) = \frac{samewords(P_1,\ P_2)}{words(P_1) + word(P_2) - samewords(P_1,\ P_2)}$$

## 3.4. Code Length Similarity

From the code length to mark the similarity of the program code, from the language morphology, can reflect the similarity of the program shape to a certain extent. The calculation method is as follows:

$$lenSim = 1 - \left| \frac{len(P_1) - len(P_2)}{maxlen\ (P_1,\ P_2)} \right|$$

Where len($P_i$) represents the length of the program code after removing the program code $P_i$ all whitespace characters (including spaces, tabs, and line breaks),the length of the program $P_i$, i=1, 2; maxlen($P_1$,$P_2$) means to find the larger

length of two programs $P_1$ and $P_2$.

### 3.5. Code Content Similarity

This paper uses a common vector space model-based method [7] to calculate program code similarity, all keywords, variables, constants, function names, preprocessing instructions, etc. that appear in the program are processed as words, without parsing the program, and the program is represented as the following vector:

$$vp = [t_1, t_2, \cdots, t_n]$$

Where, $t_i$ represents the frequency of occurrence of each word, $i$=1, 2, …, $n$. then the similarity of the two program codes can be expressed by using two vectors cosine, the calculation formula is as follows:

$$proSim(vp_1, vp_2) = \frac{\sqrt{\sum_{i=1}^{n} t_{1i} t_{2i}}}{\sqrt{\sum_{i=1}^{n} t_{1i}^2} \cdot \sqrt{\sum_{i=1}^{n} t_{2i}^2}}$$

### 3.6. Calculation of Cheating Probability

Based on the rules defined above and various similarity calculation methods, the probability of a student cheating in an exam can be calculated using the following formula:

$$cheating_{prob} = \lambda_1 timeSim + \lambda_2 numSim + \lambda_3 lenSim + \lambda_4 wordSim + \lambda_5 proSim$$

Where,$\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 + \lambda_5 = 1$, the result is a number between 0 and 1, with the closer to 1, the greater the probability of cheating. According to the role of each sub-item in cheating detection and the actual situation analysis of the exam, the similarity of code content and word-based Jaccard similarity are more important, so in the actual calculation of cheating probability, it is often taken $\lambda_5 \geq 0.4 \geq \lambda_4 \geq 0.2 \geq \lambda_3 \geq \lambda_2 \geq \lambda_1$.

In this experiment, taking $\lambda_5 = 0.5$, $\lambda_4 = 0.2$, $\lambda_3 = \lambda_2 = \lambda_1 = 0.1$, it is calculated that the cheating probability of the student ranked 26 in Figure 1 is 96.8%, and the calculated result is basically consistent with the actual situation, which has greater credibility.

## 4. Conclusion

Cheating detection has certain application prospects in programming competitions, programming course experiments and exams, and software copyright code duplication detection, among other fields. In this paper, a method based on rules and similarity is proposed to calculate the probability of cheating in online exams for programming courses. Experiments have also verified the effectiveness of the method. Examinations are not the goal; they are a means to promote learning. We will continue to design and develop people-oriented teaching programs based on relevant courses. In the teaching mode, teaching process and assessment methods to carry out in-depth reform, give full play to the deep integration of computer science and artificial intelligence technology, solid teaching and learning work, fundamentally eliminate and eliminate cheating, give play to the excellent learning style, purify the examination style and discipline.

## Acknowledgments

This paper is partially supported by Key Project of Undergraduate Education and Teaching Reform of China University of Mining and Technology-Beijing (No. J21ZD14).

## References

[1] Zhou Faguo, Yang Bingru. A New Method for Sentence Similarity Calculation and Its Application in Question-Answering Systems[J]. Computer Engineering and Applications, 2008, 44(1): 165-167.

[2] Zheng Dongmei, Li Yulin. Analysis of Undergraduate Thesis Plagiarism from the Perspective of Paper Detection[J]. Modern Education, 2018, 5(30): 351-352.

[3] Li Dongyan, Fang Ruochen, Xu Kai et al. Research on Anti-Plagiarism Algorithm Based on Similarity Matching[J]. Journal of North China Institute of Science and Technology, 2016, 13(5): 60-64.

[4] Firuz K, Hana S, David C S. Machine learning based approach to exam cheating detection[J]. PloS one,2021,16(8).

[5] Sanaa K, Abdu G. Towards effective and efficient online exam systems using deep learning-based cheating detection approach[J]. Intelligent Systems with Applications, 2022,16.

[6] Hyeok S P, Kyoungok K. Collaborative filtering recommendation system based on improved Jaccard similarity [J]. Journal of Ambient Intelligence and Humanized Computing, 2023,14(8).

[7] Sulaeman O R, Gata W, Wahyudi E, et al. Information Retrieval System to Find Articles and Clauses in UUD 1945 Using Vector Space Model Method[J]. Journal of Physics: Conference Series, 2020,1471.