ChatGPT vs. Stack Overflow: An Exploratory Comparison of Programming Assistance Tools

Jinrun Liu*, Xinyu Tang*, Linlin Li, Panpan Chen, and Yepang Liu†
Southern University of Science and Technology, Shenzhen, Guangdong, China
12011216@mail.sustech.edu.cn, 12011439@mail.sustech.edu.cn, lill3@mail.sustech.edu.cn,
11910236@mail.sustech.edu.cn, liuyp1@sustech.edu.cn

Abstract—Programmers often seek help from Q&A websites to resolve issues they encounter during programming. Stack Overflow has been a widely used platform for this purpose for over a decade. Recently, revolutionary AI-powered platforms like ChatGPT have quickly gained popularity among programmers for their efficient and personalized programming assistance via natural language interactions. Both platforms can offer valuable assistance to programmers, but it's unclear which is more effective at enhancing programmer productivity. In our paper, we conducted an exploratory user study to compare the performance of Stack Overflow and ChatGPT in enhancing programmer productivity. Two groups of students with similar programming abilities were instructed to use the two platforms to solve three different types of programming tasks: algorithmic challenges, library usage, and debugging. During the experiments, we measured and compared the quality of code produced and the time taken to complete tasks for the two groups. The results show that, concerning code quality, ChatGPT outperforms Stack Overflow significantly in helping complete algorithmic and library-related tasks, while Stack Overflow is better for debugging tasks. Regarding task completion speed, the ChatGPT group is obviously faster than the Stack Overflow group in the algorithmic challenge, but the two groups have a similar performance in the other two tasks. Additionally, we conducted a post-experiment survey with the participants to understand how the platforms have helped them complete the programming tasks. We analyzed the questionnaires to summarize ChatGPT and Stack Overflow's strengths and weaknesses pointed out by the participants. By comparing these, we identified the reasons behind the two platforms' divergent performances in programming assistance.

Keywords–ChatGPT; Stack Overflow; programming; user study

1. Introduction

Programmers often encounter situations where they need help and guidance to complete their tasks. This is where question-and-answer (Q&A) websites come into play. These websites provide a platform for programmers to ask questions and receive answers from other programmers who have faced similar challenges. Some of the most popular Q&A websites include Stack Overflow [16], GitHub [17], and programming-related subreddits hosted by Reddit [18]. Among these websites, Stack Overflow stands out as the most widely used platform with millions of active users worldwide [12]. It has over 21 million registered users and over 5.5 million visits per day [12]. However, the recent emergence of AI copilots, such as ChatGPT [19], has gained 100 million active users in just two months [15]. ChatGPT is a language model developed by OpenAI that can assist programmers by providing assistance, suggestions and generating code snippets.

There are various existing research focused on user studies of Stack Overflow [20], [21], [36], [37], [38], and user studies of language models for programming assistance [22], [23], [40], [39], [41]. However, there are very few articles comparing ChatGPT and Stack Overflow. Delile et al. [24] conducted a comparative analysis between the responses provided by Stack Overflow users and the responses generated by ChatGPT for the extracted questions. Programmers often face tight deadlines, requiring them to complete tasks efficiently and effectively. Both Stack Overflow, a widely used traditional developer Q&A website, and ChatGPT, a new-generation AI-powered Q&A platform, can offer valuable assistance to programmers. However, there is currently no existing work that studies which platform is more effective in helping enhance programmer productivity. When considering productivity, we take into account two factors: the quality of the code produced and the speed of completing tasks. Code quality is essential for programmers. Writing high-quality code with fewer bugs not only saves time and effort in fixing issues but also improves the reliability and stability of the software. Besides, the speed of completing tasks is often a critical factor in time-sensitive projects and rapid development allows programmers to deliver features quickly to meet market demands. The main objective of our paper is to compare ChatGPT and Stack Overflow in order to address the following two research questions:

- RQ1: Which platform is better at enhancing programmers' code quality?
- RQ2: Which platform is better at improving programmers' speed of completing tasks?

We conducted an exploratory user study to answer these two research questions. To design a controlled experiment, one direct approach is to have the same group of participants use both Q&A platforms to perform identical tasks in a

*These two authors contributed equally to this work.
† Yepang Liu is affiliated with both the Department of Computer Science and Engineering and the Research Institute of Trustworthy Autonomous Systems. He is the corresponding author of this paper.

2693-9371/23/$31.00 ©2023 IEEE
DOI 10.1109/QRS-C60940.2023.00105

consistent environment. However, as tasks or task types are one-time occurrences, independent replication is not feasible. So our idea is to let two groups of participants with similar programming abilities using ChatGPT and Stack Overflow to solve three different types of programming tasks: algorithmic challenges, library usage, and debugging. Our experiment involved 44 participants, with 23 participants in the ChatGPT group and 21 participants in the Stack Overflow group. We then compare the two platforms by respectively measuring the quality of code the two groups produced and the time they spent to complete the tasks. To ensure that the two groups had similar coding abilities, we conducted a pre-experiment questionnaire to assess the coding skills and experience of the participants and carefully considered the responses to form the groups. During the experiment, to control variables, we ensured that both groups of participants completed the tasks within the same software and hardware environment and provided them with the same instructional documentation for both platforms, adhering to identical time limits. We recorded participants' actions throughout the experiment by capturing the screens. Finally, the code quality and task completion time were assessed using the objective evaluation criteria to compare the two platforms. It should be noted that for assessing code quality, we design various test cases to evaluate different aspects of the code, such as correctness, performance, etc. In the end, we gauge the quality of the code based on the number of test cases it successfully passes.

To gain the experiences of participants while using these two platforms, we distributed a post-experiment questionnaire and collected their responses. This questionnaire contains a rating item for ChatGPT (or Stack Overflow), as well as an open-ended question section about its strengths and weaknesses. We extracted key points from the questionnaire responses, categorized them, and arranged them in order of frequency to provide insights and answer the following research question.

- RQ3: What are the underlying reasons for the different performances of ChatGPT and Stack Overflow in assisting programmers?

In terms of code quality, the experimental results demonstrate that the average code scores of the ChatGPT group are significantly higher than those of the Stack Overflow group in algorithm tasks (38.7 vs. 5.0) and library tasks (56.2 vs. 27.4). However, in debugging tasks, the ChatGPT group scored lower than the Stack Overflow group (65.9 vs. 84.2). To assess task completion speed, for algorithm tasks, we recorded time at three points: first successful code execution, first test case passed and submission. For library tasks, we additionally observed the time of successful library installation. For debugging tasks, we observed the time taken to fix each individual bug. By comparing these time points, we observed that the ChatGPT group is obviously faster than the Stack Overflow group in the algorithmic challenge, but the two groups have a similar performance in the other two tasks. Through the analysis of post-experiment survey responses, we found that ChatGPT is obviously better than Stack Overflow in algorithm and library tasks due to the fact that ChatGPT can quickly generate code and provide ideas, while Stack Overflow lacks task-related questions and answers. Stack Overflow has an advantage in debugging tasks due to its expertise in solving explicit exceptions and providing helpful links.

In summary, our work makes three major contributions:

- To the best of our knowledge, we conducted the first exploratory user study to compare the performance of ChatGPT and Stack Overflow in enhancing programmer productivity via both quantitative and qualitative analyses.
- We conducted a post-experiment survey to understand participants' perspectives on both platforms. By analyzing these responses, we identified the underlying reasons behind the two platforms' divergent performances in programming assistance.
- We summarized ChatGPT and Stack Overflow's strengths and weaknesses pointed out by the participants. The results may inspire future research to improve the performance of programming assistance tools and advance the state of the art in this field.

2. Background

2.1 ChatGPT

OpenAI [43] has published the GPT series since 2018, which is used to generate text. GPT-3 [45] gained attention for its impressive language generation capabilities. However, it had limitations when it came to conversational interactions. To address this, OpenAI developed ChatGPT [5] by fine-tuning GPT-3 with a conversational dataset and reinforcement learning from human feedback (RLHF) [44]. This makes ChatGPT particularly effective for conversational interactions. Benefit from the broad knowledge base, ChatGPT offers several key features for programmers: (1) ChatGPT can generate code snippets based on the given requirements; (2) It can assist in error debugging by identifying common mistakes or potential issues; (3) It can offer guidance for enhancing the algorithm by optimizing code efficiency; (4) It can help with utilizing specific APIs or libraries; (5) It can offer advice on coding bets practices and design patterns.

2.2 Stack Overflow

Stack Overflow [11] is a popular technical question-and-answer community that provides a platform for programmers to exchange information and solve technical problems. Created by Joel Spolsky and Jeff Atwood in 2008, it has become an essential part of the global programmer community. The main function of Stack Overflow is to provide a platform for programmers to ask technical questions and seek solutions from other members of the community. Users can also vote and edit answers to find the best solution. It offers features like user profiles, rankings, tags, and search to help users find information and build reputations.

3. Approach

Our approach can be divided into two parts: a comparative experiment and a post-experiment survey. The comparative

experiment aims to quantitatively compare the effectiveness of ChatGPT and Stack Overflow in enhancing programmer productivity. The post-experiment survey is conducted to identify the underlying reasons for the divergent performances of these two platforms via qualitative analysis.

3.1 The Comparative Experiment

Our experiment involves two groups with similar programming abilities completing three types of tasks using ChatGPT and Stack Overflow, respectively, in the same environment. We then measure and compare the code quality and completion speed of these two groups. Before conducting the experiments, we administered a pre-experiment questionnaire to assess the coding skills and experience of the participants. In the following, we will first describe the six tasks in detail. Then, we will explain the pre-experiment questionnaire design, recruitment and grouping process for participants. Finally, we will discuss the experimental procedure.

3.1.1 Tasks Used in the Experiment: Implementing algorithms [33], calling library functions [34], and debugging existing code [35] are most common scenarios in real-life software development. As such, we have decided to design tasks that cover these three scenarios. The quality of an algorithm has a significant impact on the performance and efficiency of software development. Selecting an appropriate algorithm and optimizing it are crucial in software development. For this scenario, we selected three algorithm tasks with increasing levels of difficulty. Calling library functions is essential in software development. It simplifies the process and improves code maintainability and reusability. For this scenario, we selected a typical library-related task. Debugging is essential for software development. It ensures the high quality and reliability of software. We designed two debugging tasks, one involving fixing logical bugs and the other related to library utilization bugs.

We designed six tasks that correspond to the software development scenarios described above, which require the participants to complete them. Table 1 presents the detailed information of these six tasks, including task descriptions, specific instructions (input, output, constraints), and possible solutions. Table 2 shows the criteria we used to evaluate task completion.

3.1.2 The Pre-experiment Questionnaire: There are two purposes for conducting a questionnaire survey to assess the programming ability of participants: 1) to assign tasks to these participants, ensuring that they have the required knowledge and ability to complete the tasks assigned to them; 2) to ensure that the ChatGPT and Stack Overflow groups assigned the same task have similar programming abilities.

The questionnaire was designed to collect basic information and assess the programming proficiency of the participants. As shown in Table 3, it consisted of nine questions covering five aspects: basic information, programming experience, programming languages, project impact, and a specific programming skill. Q1-Q3 were employed to collect basic information, including name, grade, as well as completion status and grades of four courses: Java Programming (JAVA), Data Structure and Algorithm Analysis (DSAA), Algorithm Design, and Artificial Intelligence (AI), which were related to our tasks. Q4 gathered participants' number of years of programming experience, while Q5 inquired about the programming languages they were familiar with and their proficiency level. Q6-Q8 collected information on the number of projects, followers, and stars on participants' GitHub profiles, helping us understand their project impact. Q9 aimed to assess participants' familiarity with Maven [32], which will help us determine the participants for task 4, which is related to Maven. Notice that our questionnaire was designed according to the guidelines outlined in *Guidelines for Conducting Surveys in Software Engineering v. 1.1* [13] to ensure its reliability.

3.1.3 Recruitment and Grouping of Participants: To recruit participants interested in completing programming tasks using ChatGPT and Stack Overflow, we offered prizes and designed an advertisement for the experiment. The ads were placed on social media platforms to attract a sufficient number of individuals to sign up for the experiment. Finally, we received 44 registrations. Among these participants, there were 9 females and 35 males, including 9 freshmen, 11 sophomores, 18 juniors, 5 seniors, and 1 graduate student. Notably, the freshmen had limited programming experience, primarily with Java, while the rest were pursuing computer science or related majors, with more extensive programming backgrounds. In the following, we will first describe how we assigned tasks to the participants and then introduce how they were assigned to either the ChatGPT or Stack Overflow subgroup.

Since Task 4 and Task 6 not only require participants to have programming experience but also have specific programming skills, Task 4 requires participants to have proficiency in using Maven, while Task 6 requires participants to have the ability to train neural network models, we will first find suitable participants for these two tasks. Assignment of participants to Task 4 is determined by their response to Q9. If they have not used Maven, they will not be assigned to Task 4. A total of 21 participants are eligible to be assigned to this task. Assignment of participants to Task 6 is determined by their response to Q2. Participants who are less than a junior, indicating that they have not studied artificial intelligence courses, cannot be assigned to Task 6. There are 13 participants who are eligible to be assigned to this task. Following a discussion between three researchers from our team, we selected 6 eligible participants with comparable programming abilities to complete Task 6. Programming abilities were obtained by analyzing their responses to Q3-Q8 in the questionnaire. From the remaining 16 eligible participants, we then selected another 6 with comparable abilities to complete Task 4.

All the remaining participants were assigned to the algorithm-related Task 1, 2, 3 and 5. After a discussion among three researchers from our team, we ranked the four tasks by difficulty level, from easiest to hardest, as follows: Task 1, Task 2, Task 5, and Task 3. To ensure that the remaining participants were assigned appropriate tasks, we further discussed their responses to Q3-Q8, and ranked their programming abilities. Finally, we allocated tasks based on the principle of assigning

Table 1. Six tasks used in the comparative experiment (the numbers 1 to 6 in the table represent the names of the tasks as follows: “String matching [25]”; ‘The longest non-increasing subsequence [26]”; “Number of primes [27]”; “Extract text from images using OpenCV [29] and Tesseract OCR libraries [28]”; “Greatest rectangle area [30]”; “The wine classifier [31]”)

Types	Tasks	Contents	
Algorithm	1	*Description:*	Given a template string of length n and k strings of length up to L, find the string that has the longest common substring with the template string, and find the length of that longest common string
		Instructions:	**Input:** The first line is the template string. The second line is the value of k. From the third line to the $(k+2)$th line, each line has a string. **Output:** The length of the longest common substring. **Constraints:** (1) Every string consists of only lowercase English letters; (2) $1 \leq k \leq 10^9$; 3. $1 \leq L \leq 10^2$ (L is the length of each string);
		Solutions:	(1) Brute Force: $\mathcal{O}(L*n*k)$; (2) KMP: $\mathcal{O}((L+n)*k)$;
	2	*Description:*	Given an array, find the longest non-increasing subsequence and the longest increasing subsequence in the sequence.
		Instructions:	**Input:** A sequence composed with integers. **Output:** The first line is the length of the longest non-increasing subsequence and the second line is the length of longest increasing subsequence. **Constrains:** (1) For the first half of the evaluation data, the length of array is less than 10^4; (2) For the rest evaluation data, the length of array is less than 10^5; (3) For all data, the length of array is a integer and less than 5×10^4;
		Solutions:	(1) Dynamic Programming: $\mathcal{O}(n^2)$; (2) Dynamic Programming plus Binary Search: $\mathcal{O}(nlogn)$;
	3	*Description:*	Given an integer n, calculate the value of π (n). π(n) represents the number of primes in the range 1 to n.
		Instructions:	**Input:** A integer n. **Output:** The number of primes between 1 to n. **Constrains:** $n \leq 10^{13}$.
		Solutions:	(1) Brute Force: $\mathcal{O}(n^2)$; (2) Eratosthenes: $\mathcal{O}(nloglogn)$; (3) Eyler: $\mathcal{O}(n)$; (4) Meissel-Lehmer: $\mathcal{O}(n^{2/3}/log^2 n)$;
Library	4	*Description:*	Write a program that aims to extract text from images as accurately as possible.
		Instructions:	**Input:** Images including text. **Output:** The recognized text. **Constraints:** Texts used for testing is in English or Chinese.
		Solutions:	Implement image recognition using OpenCV and Tesseract OCR libraries, which can extract text from images.
Debugging	5	*Description:*	Given an array of integers ‘heights‘ representing the histogram’s bar height where the width of each bar is ‘1‘, return the area of the largest rectangle in the histogram. We provided a code with errors, please find the errors in the code and correct them.
		Instructions:	**Input:** An array of integers representing the heights. **Output:** The number of areas of the largest rectangle in the histogram. **Constraints:** (1) $1 \leq heights.length \leq 10^5$; (2) $0 \leq heights[i] \leq 10^4$;
		Solutions:	Analyse the code we have provided and fix the three logical errors in it to implement the complete monotonic stack algorithm.
	6	*Description:*	Given the dataset data.csv, you are asked to modify the code to implement the J48 classifier to predict the quality of wine.
		Instructions:	**Input:** A CSV file, with wine’s features (fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulfates, alcohol, quality). **Output:** The confusion matrix, accuracy, and the size of training and testing sets. **Constraints:** The training and testing sets should be delimited by yourself, in a ratio of 8:2.
		Solutions:	Analyse the code we have provided and fix three of the running errors and two of the logic errors to implement the J48 classifier. In addition, the final accuracy of the classifier should be more than 85% to prove that the code was correctly corrected.

Table 2. Evaluation criteria for the six tasks used in the comparative experiment

Types	Evaluation
Algorithm	The total score for each task is 100 points. Task 1 has 10 test cases, each worth 10 points. Tasks 2 and 3 have 20 test cases each, with each test case worth 5 points. The time limit for each test case is 1 second, and the space limit is 128 MB.
Library	Scoring is divided into two parts, totaling 100 points. First part: 10 points for successfully installing the library. Second part: 90 points, consisting of 8 test cases, each associated with an image. Participants need to extract 494 English or Chinese characters. The score for this part is $\frac{x}{494} \times 90$. The final score is the sum of scores from both parts of the library section.
Debugging	For Task 5, we set up 10 test cases, with each passed test case awarding 10 points. For Task 6, points are allocated based on the importance of fixing each bug: two bugs are worth 10 points each, and the other four are worth 20 points each.

more challenging tasks to participants with higher abilities. As a result, we assigned 10 participants to Task 1, 9 to Task 2, 7 to Task 3, and 6 to Task 5. Since all participants had used both tools and were equally familiar with each of them, we divided each group equally into ChatGPT and Stack Overflow subgroups through a randomized process. Ultimately, for Tasks 1-6, the number of participants in the ChatGPT group and the Stack Overflow group were respectively: (5, 5); (5, 4); (3, 3); (3, 3); (4, 3); (3, 3).

3.1.4 Experimental Procedure: During the experiment, participants used the same software and hardware environment. Participants were given identical task descriptions, tool usage manuals, and initial code. They had a limited time of 1 hour to complete the tasks without seeking assistance from any platform other than ChatGPT or Stack Overflow. We recorded participants’ computer screens to monitor compliance with our guidelines and track task completion times.

3.2 The Post-experiment Questionnaire

The post-experiment questionnaire aims to investigate the underlying reasons for the differential effects of the two platforms in assisting programming. The questionnaire consists of

Table 3. Pre-experiment questionnaire

#	Questions
1	What is your name?
2	What is your grade?
3	Have you completed the following courses: Java (CS102A/CS109), DSAA (CS203), Algorithm Design (CS208), and AI (CS303)? If yes, please provide your grades.
4	How many years of programming experience do you have?
5	Please provide a list of programming languages you are familiar with, ranked in order of familiarity.
6	How many open-source projects have you contributed to?
7	How many followers do you have on Github?
8	How many stars do you have in total on Github?
9	Have you ever used maven to import a library in Java? Please rate your familiarity on a scale of 1-5.

three questions, one multiple-choice question and two open-ended questions, as shown in Table 4. Q1 asks participants to rate the level of assistance provided by ChatGPT or Stack Overflow in completing their tasks, on a scale from 1 to 5. Q2 and Q3 ask participants to describe the specific ways in which ChatGPT or Stack Overflow has been helpful to them and to identify any aspects in which they were not satisfactory. By analyzing the answers to Q1, we can obtain a direct comparison between ChatGPT and Stack Overflow in terms of assistance provided. Through the analysis of participants' responses to Q2 and Q3, we can gather their perceptions of the strengths and weaknesses of the two platforms. To ensure participants remembered experiment details, we immediately conducted a questionnaire survey after they finished the programming tasks. There was no time limit for the questionnaire.

4. Results

4.1 RQ1: Comparison of Code Quality

For the six tasks, we evaluated the code produced by both the ChatGPT and Stack Overflow groups using the criteria outlined in Table 2. The comparison of code quality scores between the two groups is presented in Table 5. The two "*Avg*" columns in the table display the average scores for the ChatGPT and Stack Overflow groups, respectively. Notice that the maximum score for each of six tasks was set to 100 points. In the table, a score of 0 with an underline indicates that the code submitted by the participant could not be successfully executed, whereas a score of 0 without an underline indicates that the submitted code was able to run but failed in all test cases. Based on the analysis of the data, it can be concluded that for the algorithm tasks, the scores of the ChatGPT group are significantly higher than those of the Stack Overflow group (72.0 vs. 0.0; 15.8 vs. 0.0; 28.3 vs. 15.0), as well as the average scores (38.7 vs. 5.0). In the library usage task, the ChatGPT group's score is notably higher than that of the Stack Overflow group (56.2 vs. 27.4). However, for debugging tasks, ChatGPT's performance is inferior to that of the Stack Overflow group on Task 5 (70.0 vs. 90.0) and Task 6 (61.7 vs. 78.3). The average scores of the two groups are 65.9 and 84.2. From this table, we also observed that all participants in the ChatGPT group successfully produced runnable code, while 28.6% (6/21) of participants in the Stack Overflow group had code that failed to run.

Answer to RQ1: The ChatGPT group significantly outperformed the Stack Overflow group in algorithm tasks (38.7 vs. 5.0) and library tasks (56.2 vs. 27.4). However, in debugging tasks, the ChatGPT group scored lower than the Stack Overflow group (65.9 vs. 84.2).

Observation 1: The ChatGPT group had a much lower rate of creating non-runnable code than the Stack Overflow group (0% vs. 28.6%).

4.2 RQ2: Comparison of Task Completion Speed

During the experiment, we recorded participants' screen activities as they completed tasks. We manually reviewed the recorded videos afterward to determine when participants completed each task. For algorithm tasks, we recorded three timestamps: the first successful code execution, the first successful test case passed, and the point when participants stopped making changes to their code (submission). The first code execution indicates that the participant's code is ready for testing. Once the code passes the first test case, it is considered mature and relatively complete. The time of submission shows the total time invested by participants. We also calculated the time interval between the first code execution and the first test case passed, which indicates the time taken to modify the code to pass the test case. For library tasks, we also observed the timestamp of successful library installation. For debugging tasks, we noted the time taken to fix each individual bug.

The Comparison of timestamps for three algorithm tasks was shown in Table 6. It's important to note that our default start time for each task was 00:00:00, and the end time was set at 01:00:00. Task 2 has two "first test case passed" time points, as it includes two subtasks. Furthermore, it should be noted that if a participant's code failed to run successfully during the experiment, the "first program execution" time point was set to 01:00:00 and indicated with an underline in the table. Similarly, if a participant's code did not pass the test cases, the "first test case passed" time point was set to 01:00:00 and also marked with an underline. During our experiment, one participant from the Stack Overflow group abandoned Task 2 midway, which resulted in the inability to accurately collect their corresponding time point data. In Table 6, we denote this with a "-" symbol. Similarly, for cases where the "first test case passed" time point data is indicated as underlined 01:00:00, we were unable to calculate the time interval data, and in the table, we use "-" to represent this. Table 7 shows the comparison of timestamps for the library-related task. The underlined numbers and "-" in the table convey the same

Table 4. Post-experiment questionnaire

#	Questions	Notes
1	How helpful do you think ChatGPT (or Stack Overflow) has been for you? (1. Not helpful, 2. Somewhat helpful, 3. Moderately helpful, 4. Quite helpful, 5. Very helpful)	Rating
2	Please specify in which ways ChatGPT (or Stack Overflow) has been helpful for you? (short answer question)	Strengths
3	Please specify in what aspects ChatGPT (or Stack Overflow) is not satisfactory? (short answer question)	Weaknesses

Table 5. Comparison of code quality between ChatGPT group and Stack Overflow group

Types	Tasks	Scores (out of 100)											
		ChatGPT group					*Avg*	Stack Overflow group					*Avg*
Algorithm	1	0	80	80	100	100	72.0	0	0	0	0	0	0.0
	2	2	2	5	20	50	15.8	0	0	0	0		0.0
	3	5	40	40			28.3	5	5	35			15.0
Library	4	10.0	78.1	80.5			56.2	0	0	82.3			27.4
Debugging	5	0	90	90	100		70.0	90	90	90			90.0
	6	45	60	80			61.7	55	80	100			78.3

meanings as in Table 6. Table 8 shows the comparison of times spent for the two debugging tasks. Notice that one participant from the ChatGPT group abandoned Task 5, in the table, we used "-" to indicate the relevant data for this participant.

For algorithm tasks, our analysis of Table 6 reveals that the ChatGPT group consistently precedes the Stack Overflow group at all three timestamps. Notably, the ChatGPT group achieved a clear lead in completing the first program execution ahead of the Stack Overflow group, and completed the first test case passed and submission slightly before the Stack Overflow group. We observed a significant difference in the interval between "first program execution" and "first test case passed" for the algorithm tasks, with the ChatGPT group having a much larger interval compared to the Stack Overflow group. This indicates that participants using ChatGPT require more time for code refinement and modifications following the successful execution of their code. For the library-related task and debugging tasks, we observed that there were no significant differences between the ChatGPT and Stack Overflow groups in terms of time spent or at various timestamps.

Answer to RQ2: For algorithm tasks, the ChatGPT group significantly precedes the Stack Overflow group at all three timestamps. For the library-related task and debugging tasks, there were no significant differences between the ChatGPT and Stack Overflow groups at various timestamps or in terms of time spent.

Observation 2: The time interval between "first program execution" and "first test case passed" is notably longer for the ChatGPT group compared to the Stack Overflow group. This indicates that participants using ChatGPT require more time for code refinement and modifications following the successful execution of their code.

4.3 RQ3: Reasons for Differential Effects of ChatGPT and Stack Overflow

To investigate the differing performance of ChatGPT and Stack Overflow in helping programmers generate high-quality code and complete tasks quickly, we conducted a post-experiment questionnaire survey with 44 participants. Of these, 23 were in the ChatGPT group, and 21 were in the Stack Overflow group. In the questionnaire, participants were asked to rate chatGPT (or Stack Overflow) and provide their perceptions of its strengths and weaknesses. It should be noted that a higher score indicates that participants perceive the tool to be more helpful.

We calculated the frequency of ratings from 1 to 5 for the two platforms, the results are presented in Table 9. In the table, "1s count" refers to the total number of participants who rated with a score of 1, and columns 3 to 6 follow the same interpretation. The "Avg score" column calculates the average scores for both platforms, with ChatGPT scoring 4.0 and Stack Overflow scoring 2.5. ChatGPT's average score is significantly higher than that of Stack Overflow.

We processed the responses to the two open-ended questions regarding the strengths and weaknesses of ChatGPT (or Stack Overflow) in the following manner: (1) we extracted key points from each participant's response; (2) for each platform's strengths and weaknesses, we grouped similar key points together and calculated their frequencies. Note that the total count of key points may not match the number of participants due to multiple viewpoints expressed in the questionnaire. (3) starting from the features of ChatGPT and Stack Overflow, we divided users' feedback on them into 2 and 4 main aspects respectively, and classified the key points obtained into these main aspects. Since the usage of ChatGPT is to provide a demand and then get an answer, we divided users' feedback on it into two main aspects: answer quality and

Table 6. Comparison of timestamps for three algorithm tasks

Tasks	Key process points	Timestamps (hh:mm:ss) / Time interval (hh:mm:ss)											
		ChatGPT group					*Avg*	Stack Overflow group					*Avg*
1	First program execution	00:05:47	00:20:43	00:50:00	00:13:43	00:20:08	00:22:04	00:31:01	01:00:00	00:43:49	00:58:57	01:00:00	00:50:45
	Interval	00:54:13	00:22:56	00:10:00	00:14:25	00:03:01	00:24:10	00:28:59	-	00:00:28	00:00:00	-	00:11:47
	First test case passed	01:00:00	00:43:39	01:00:00	00:28:08	00:23:09	00:46:14	01:00:00	01:00:00	00:44:17	00:58:57	01:00:00	00:56:39
	Submission	01:00:00	00:47:46	01:00:00	00:41:24	00:29:45	00:47:47	01:00:00	01:00:00	00:45:45	00:59:29	01:00:00	00:57:03
2	First program execution	00:40:36	00:13:42	00:27:57	00:15:24	00:30:08	00:24:55	-	01:00:00	00:47:36	00:58:54		00:54:34
	Interval	00:13:43	00:05:51	00:32:03	00:14:25	00:00:00	00:13:12	-	-	00:12:24	00:01:06		00:06:45
	First test case passed	00:54:07	00:19:33	01:00:00	00:29:49	00:30:08	00:38:43	-	01:00:00	01:00:00	01:00:00		00:59:04
		01:00:00	00:25:05	01:00:00	00:29:49	00:49:56	00:44:58	-	01:00:00	01:00:00	01:00:00		00:59:04
	Submission	01:00:00	00:31:05	01:00:00	00:45:26	00:50:32	00:49:25	-	00:57:12	01:00:00	01:00:00		00:59:04
3	First program execution	00:04:42	00:27:23	00:15:37			00:15:54	00:23:31	00:37:04	00:33:19			00:31:18
	Interval	00:01:11	00:00:00	00:20:09			00:07:07	00:01:16	00:02:07	00:01:59			00:01:47
	First test case passed	00:05:53	00:27:23	00:35:46			00:23:01	00:24:47	00:39:11	00:35:18			00:33:05
	Submission	00:59:25	00:56:15	01:00:00			00:58:33	00:24:50	01:00:00	01:00:00			00:48:17

Table 7. Comparison of timestamps for the library-related task

Tasks	Key process points	Timestamps (hh:mm:ss) / Time interval (hh:mm:ss)							
		ChatGPT group			*Avg*	Stack Overflow group			*Avg*
4	Library installation	00:45:21	00:53:04	00:23:42	00:38:23	01:00:00	01:00:00	00:12:59	00:44:20
	Interval	00:14:39	00:03:44	00:00:08	00:06:10	-	-	00:00:15	00:00:15
	First program execution	01:00:00	00:56:48	00:23:50	00:46:53	01:00:00	01:00:00	00:13:14	00:44:25
	Interval	-	00:00:00	00:00:00	00:00:00	-	-	00:00:00	00:00:00
	First test case passed	01:00:00	00:56:48	00:23:50	00:46:53	01:00:00	01:00:00	00:13:14	00:44:25
	Submission	01:00:00	01:00:00	00:41:15	00:53:45	01:00:00	01:00:00	00:56:14	00:58:45

Table 8. Comparison of times for the two debugging tasks

Tasks	Bugs	Time spent (hh:mm:ss)								
		ChatGPT group				*Avg*	Stack Overflow group			*Avg*
5	Empty Stack 1	01:00:00	00:14:25	00:21:50	-	00:32:05	00:15:00	00:45:28	00:09:52	00:23:27
	Division by Zero 1	01:00:00	01:00:00	01:00:00	-	01:00:00	01:00:00	01:00:00	01:00:00	01:00:00
	Clear Stack	01:00:00	00:33:46	00:23:27	-	00:39:04	00:15:30	00:45:10	00:40:40	00:33:47
	Empty Stack 2	01:00:00	00:14:59	00:23:54	-	00:32:58	00:15:15	00:45:38	00:10:10	00:23:41
	Division by Zero 2	01:00:00	01:00:00	01:00:00	-	01:00:00	01:00:00	01:00:00	01:00:00	01:00:00
6	File Path	00:07:09	00:06:33	00:00:31		00:04:44	00:04:33	00:00:39	00:06:49	00:04:00
	Set Class Index	00:29:58	00:11:13	00:07:30		00:16:14	00:14:20	00:03:38	00:45:38	00:21:12
	Size of Training Set	01:00:00	01:00:00	01:00:00		01:00:00	01:00:00	01:00:00	00:36:41	00:52:14
	Numeric to Normal	00:40:11	00:46:55	00:36:35		00:41:14	00:40:46	00:58:00	00:54:48	00:51:11
	Remove Codes	01:00:00	01:00:00	00:21:58		00:47:19	01:00:00	00:27:47	01:00:00	00:49:16
	Accuracy	01:00:00	00:54:57	00:39:02		00:51:20	00:41:58	00:58:00	01:00:00	00:53:19

Table 9. Comparison of ratings between the two platforms

Platforms	1s count	2s count	3s count	4s count	5s count	Avg score
ChatGPT	0	2	5	7	9	4.0
Stack Overflow	3	8	7	3	0	2.5

user experience. Since the usage of Stack Overflow is to search for a question and then find a suitable answer to get solution, we divided users' feedback on it into four main aspects: number of questions, number of answers, answer quality, and user experience. Notice that the analysis process of

Table 10. Strengths and weaknesses of ChatGPT and Stack Overflow analyzed from the post-experiment questionnaires

Platforms	Aspects	S/W	Key points with frequencies and examples of original participant responses
ChatGPT	Quality of answers	Strengths	**details to write code (6):** "Assistance in syntax, methods, or other technical details that I cannot remember."
			provide algorithm templates (6): "ChatGPT can provide algorithms for classic problems."
		Weaknesses	**wrong and expired answers (25):** "Links given are expired."; "The code is wrong."
			cannot handle uncommon problems (1): "At present, it still lacks the ability to create new algorithms and can only use known algorithms, making it difficult to handle some rare problems."
	User experiences	Strengths	**provide ideas (15):** "It can provide ideas to solve the problem exactly."
			generate and explain code (16): "It can generate code and give a clear structure."
			help debug (3): "Returning samples to GPT would enable it to perform automatic error correction."
			help describe questions (1): "ChatGPT possesses strong text comprehension ability, and sometimes users may not be fully aware of the problems they need to search. However, ChatGPT can filter out related questions that users may be interested in while answering the question."
		Weaknesses	**need to understand code for code refinement (7):** "The code requires understanding before debugging."
			may mislead (2): "The code does not match the results of the test cases, which can cause confusion. "
Stack Overflow	Number of questions	Weaknesses	**searchable questions are not enough (5):** "I can only find answers to solved problems."
	Number of answers	Strengths	**extra knowledge (3):** "Some answers may expand on related knowledge."
		Weaknesses	**answers are not enough (11):** "The assistance for basic syntax is weak."
	Quality of answers	Strengths	**solve explicit exceptions (3):** "Solutions to exceptions can be easily obtained."
			useful links (7): " The GitHub links or Apache Maven repository instructions provided are useful."
		Weaknesses	**few and scattered codes (1):** "The codes are fragmented, cannot be combined due to incompatibility."
			bad answers (3): "The code may contain errors and it can be difficult to determine."
			no detailed explanation (6): "Some solutions to the problem may not have detailed explanations. "
	User experiences	Strengths	**communication (1):** "It provides a platform to communicate."
		Weaknesses	**cannot provide ideas (6):** "It is of no help in some algorithmic problems."
			inapposite order of answers (2): "The answer I need may be ranked relatively low in the search results."
			need search skills (5): "The search terms may not be precise , resulting in answers that do not apply."

this questionnaire was carried out through multiple discussions among the three authors among us to reach a consensus.

Table 10 presents a comparison of the strengths and weaknesses of the two platforms. We selected the top two key points with the highest frequencies, which we consider to represent points of consensus among participants. Our findings reveal that participants perceived the key strengths of ChatGPT to be "generate and explain code" and "provide ideas" while its weaknesses include "wrong and expired answers" and "need to understand code for code refinement". As for Stack Overflow, its key strengths encompass "useful links", "extra knowledge", and "solve explicit exceptions", while its weaknesses involve "answers are not enough", "no detailed explanation" , and "cannot provide ideas". In comparing code quality, ChatGPT outperforms Stack Overflow in algorithmic and library-related tasks. This is because participants can easily generate code and obtain insights from ChatGPT. On the other hand, Stack Overflow lacks clear questions and answers, making it difficult for participants to gain problem-solving insights. This also explains why ChatGPT shows a significantly faster completion time in algorithmic tasks compared to Stack Overflow. For debugging tasks, Stack Overflow is better due to its ability to solve explicit exceptions and provide useful links for resolution methods, which ChatGPT lacks. The reason why ChatGPT users take more time for code refinement may be due to errors and outdated information in its answers.

We also made an interesting observation where 3 participants mentioned that using Stack Overflow allowed them to gain more knowledge during the problem-solving process , 7 participants complained about the need to understand the generated code from ChatGPT . This may suggest that relying too heavily on ChatGPT may limit problem-solving abilities and hinder creativity. It could explain why Stack Overflow performs better than ChatGPT in debugging tasks.

Answer to RQ3: ChatGPT is obviously better than Stack Overflow in algorithm and library tasks due to the fact that ChatGPT can quickly generate code and provide ideas, while Stack Overflow lacks task-related questions and answers. Stack Overflow excels in debugging tasks due to its expertise in solving explicit exceptions and providing helpful links.

Observation 3: Excessive reliance on ChatGPT may potentially limit our problem-solving thought, thereby diminishing certain aspects of creativity.

5. Discussions

Threats to validity: (1) Assessment of programming abilities: We evaluated participants' programming skills using surveys

and discussions with the authors, which could affect the validity of the results. To minimize this, we assessed participants' abilities in various aspects through the questionnaire and the three authors reached a consensus to reduce subjective bias; (2) Random assignment of participants: In the experiment, participants were randomly assigned to two groups. This could affect the validity of the results if there were differences in their familiarity with the two platforms. To mitigate this, we provided guidance manuals for either ChatGPT or Stack Overflow to each participant.

Limitations: Our work has the following limitations: (1) The relatively small sample size of 44 participants in our user study; (2) The participants were all computer science students, which may lack representativeness for the broader programmer population; (3) The three types of programming tasks we designed may not cover all possible usage scenarios. We intend to address these limitations in our future work.

6. Related Works

User studies of Stack Overflow: In order to assess the extent of assistance provided by Stack Overflow to programmers, numerous researchers have conducted user studies. Dondio et al. [20] conducted a user study aimed at evaluating the impact of using Stack Overflow as a supplementary plugin to enhance the academic performance of students. Lo et al. [21] discovered limitations of Stack Overflow in MATLAB code questions. Fangl et al. [37] focused on a specific politeness strategy-expressing gratitude-in Q&A sites, finding that gratitude expressions in comments can motivate answerers to generate higher-quality content. Wijekoon et al. [38] analyzed the global user distribution and contribution of Stack Overflow.

User studies of language models for code assistance: To assess the effectiveness and feasibility of AI-powered code assistance tools, researchers have conducted several user studies. Perry et al. [22] conducted a user study and found that users with access to the assistant were more likely to introduce security vulnerabilities. Sandoval et al. [23] conducted a user study showing that AI-assisted users produced critical security bugs at a rate no more than 10% higher than the control group. Vaithilingam et al. [40] found Copilot generates code a lot quicker than typing or finding it from other sources but it is often buggy. Imai [39] tasked a group of developers (N=21) to implement code for a 'minesweeper' game. The study concluded that Copilot tended to result in more lines of code than the human-based pair-programming in the same amount of time. However, the quality of code produced by Copilot was lower. A study by Ziegler et al. [41] from GitHub examines user perspectives on productivity during usage of GitHub Copilot. Users felt Copilot had a more beneficial effect on their productivity than a negative one.

Comparison of programming assistance platforms: Delile et al. [24] compared responses from Stack Overflow users and ChatGPT. They found that ChatGPT provides correct responses for approximately 56% of questions. However, Stack Overflow answers are slightly more accurate for the remaining responses. Samia Kabir et al. [46] investigated ChatGPT's responses to software engineering queries. They found that 52% of the answers were inaccurate and 77% were verbose. However, users still preferred ChatGPT's responses 39.34% of the time because they were comprehensive and well-written. Our study differs from the article by focusing on the various effects of platforms on boosting programmer productivity.

7. Conclusion

We conducted a user study to compare ChatGPT and Stack Overflow's performance in assisting programmers. ChatGPT outperformed Stack Overflow in algorithm and library tasks, while Stack Overflow was better for debugging tasks. We also surveyed participants to understand the factors influencing the different performances of the two platforms.

Acknowledgment

This work is supported by the National Natural Science Foundation of China (Grant No. 61932021).

References

[1] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, *et al.*, "Training language models to follow instructions with human feedback," *arXiv preprint arXiv:2203.02155*, 2022.

[2] W. B. Knox and P. Stone, "Augmenting reinforcement learning with human feedback," in *ICML 2011 Workshop on New Developments in Imitation Learning (July 2011)*, vol. 855, p. 3, 2011.

[3] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[4] S. Greengard, "Chatgpt: Understanding the chatgpt ai chatbot," 2022.

[5] OpenAI, "Introducing chatgpt," 2022.

[6] L. Tung, "Chatgpt can write code. now researchers say it's good at fixing bugs, too," 2023.

[7] J. Vincent, "Openai's new chatbot can explain code and write sitcom scripts but is still easily tricked," 2022.

[8] J. Atwood, "Introducing stackoverflow.com," 2008.

[9] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[10] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[11] J. Spolsky and J. Atwood, "Introducing stack overflow." https://blog.stackoverflow.com/2008/09/introducing-stack-overflow/, 2008.

[12] "All sites – stack exchange." https://stackexchange.com/sites. Accessed on 26 March 2023.

[13] J. Linaker, S. M. Sulaman, M. Höst, and R. M. de Mello, "Guidelines for conducting surveys in software engineering v. 1.1," *Lund University*, vol. 50, 2015.

[14] A. Joshi, S. Kale, S. Chandel, and D. K. Pal, "Likert scale: Explored and explained," *British journal of applied science & technology*, vol. 7, no. 4, p. 396, 2015.

[15] B. Maurer, "Geo p. tech, ai chatbot geotechnical engineer: How ai language models like "chatgpt" could change the profession,"

[16] "Stack overflow." https://blog.stackoverflow.com.

[17] "Github." https://github.com/.

[18] "Reddit." https://www.reddit.com/.

[19] "Chatgpt." https://chat.openai.com/.

[20] P. Dondio and S. Shaheen, "Is stackoverflow an effective complement to gaining practical knowledge compared to traditional computer science learning?," in *Proceedings of the 11th International Conference on Education Technology and Computers*, ICETC '19, (New York, NY, USA), p. 132–138, Association for Computing Machinery, 2020.

[21] M. Naghashzadeh, A. Haghshenas, A. Sami, and D. Lo, "How do users answer matlab questions on q&a sites? a case study on stack overflow and mathworks," in *2021 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pp. 526–530, 2021.

[22] N. Perry, M. Srivastava, D. Kumar, and D. Boneh, "Do users write more insecure code with ai assistants?," 2022.

[23] G. Sandoval, H. Pearce, T. Nys, R. Karri, S. Garg, and B. Dolan-Gavitt, "Lost at c: A user study on the security implications of large language model code assistants," 2023.

[24] Z. Delile, S. Radel, J. Godinez, G. Engstrom, T. Brucker, K. Young, and S. Ghanavati, "Evaluating privacy questions from stack overflow: Can chatgpt compete?," 2023.

[25] "String matching." https://leetcode.cn/problems/qJnOS7/.

[26] "P1020 [noip1999 junior division] missile interception." https://www.luogu.com.cn/problem/P1020.

[27] "P7884 meissel–lehmer algorithms." https://www.luogu.com.cn/problem/P7884.

[28] "Tesseract open source ocr engine." https://github.com/tesseract-ocr/tesseract.

[29] "Opencv." https://opencv.org/.

[30] "largest-rectangle-in-histogram." https://leetcode.cn/problems/largest-rectangle-in-histogram/.

[31] "Wine-quality-dataset." https://github.com/shrikant-temburwar/Wine-Quality-Dataset.

[32] "Maven." https://maven.apache.org/.

[33] "The path of software development craftsmanship." https://dmitripavlutin.com/the-path-of-software-development-craftsmanship/.

[34] Q. He, B. Li, F. Chen, J. Grundy, X. Xia, and Y. Yang, "Diversified third-party library prediction for mobile app development," *IEEE Transactions on Software Engineering*, vol. 48, no. 1, pp. 150–165, 2022.

[35] "What is debugging? (plus 8 important strategies to try)." https://www.indeed.com/career-advice/career-development/debugging.

[36] S. Wang, T.-H. Chen, and A. E. Hassan, "How do users revise answers on technical q&a websites? a case study on stack overflow," *IEEE Transactions on Software Engineering*, vol. 46, no. 9, pp. 1024–1038, 2020.

[37] Y. Fangl, T. Lu, P. Zhang, H. Gu, and N. Gu, "Exploring the effect of politeness on user contribution in q&a sites: A case study of stack overflow," in *2018 IEEE 22nd International Conference on Computer Supported Cooperative Work in Design ((CSCWD))*, pp. 713–718, 2018.

[38] H. Wijekoon and V. Merunka, "Patterns of user participation and contribution in global crowdsourcing: A data mining study of stack overflow," in *Proceedings of the 10th International Conference on Information and Communication Technologies in Agriculture, Food and Environment (HAICTA 2022), Athens, Greece, September 22-25, 2022* (A. Theodoridis and S. Koutsou, eds.), vol. 3293 of *CEUR Workshop Proceedings*, pp. 143–150, CEUR-WS.org, 2022.

[39] S. Imai, "Is github copilot a substitute for human pair-programming? an empirical study," in *Proceedings of the ACM/IEEE 44th International Conference on Software Engineering: Companion Proceedings*, ICSE '22, (New York, NY, USA), p. 319–321, Association for Computing Machinery, 2022.

[40] P. Vaithilingam, T. Zhang, and E. L. Glassman, "Expectation vs. experience: Evaluating the usability of code generation tools powered by large language models," in *Extended Abstracts of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI EA '22, (New York, NY, USA), Association for Computing Machinery, 2022.

[41] A. Ziegler, E. Kalliamvakou, X. A. Li, A. Rice, D. Rifkin, S. Simister, G. Sittampalam, and E. Aftandilian, "Productivity assessment of neural code completion," in *Proceedings of the 6th ACM SIGPLAN International Symposium on Machine Programming*, MAPS 2022, (New York, NY, USA), p. 21–29, Association for Computing Machinery, 2022.

[42] "Intellij idea." https://www.jetbrains.com/zh-cn/idea/.

[43] "Openai." https://openai.com/.

[44] P. Christiano, J. Leike, T. B. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," 2023.

[45] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," 2020.

[46] S. Kabir, D. N. Udo-Imeh, B. Kou, and T. Zhang, "Who Answers It Better? An In-Depth Analysis of ChatGPT and Stack Overflow Answers to Software Engineering Questions," *arXiv preprint arXiv:2308.02312*, 2023.