Is Stack Overflow Obsolete? An Empirical Study of the Characteristics of ChatGPT Answers to Stack Overflow Questions

Samia Kabir
Purdue University
West Lafayette, USA
kabirs@purdue.edu

David N. Udo-Imeh
Purdue University
West Lafayette, USA
dudoimeh@purdue.edu

Bonan Kou
Purdue University
West Lafayette, USA
koub@purdue.edu

Tianyi Zhang
Purdue University
West Lafayette, USA
tianyi@purdue.edu

ABSTRACT

Q&A platforms have been crucial for the online help-seeking behavior of programmers. However, the recent popularity of ChatGPT is altering this trend. Despite this popularity, no comprehensive study has been conducted to evaluate the characteristics of ChatGPT's answers to programming questions. To bridge the gap, we conducted the first in-depth analysis of ChatGPT answers to 517 programming questions on Stack Overflow and examined the correctness, consistency, comprehensiveness, and conciseness of ChatGPT answers. Furthermore, we conducted a large-scale linguistic analysis, as well as a user study, to understand the characteristics of ChatGPT answers from linguistic and human aspects. Our analysis shows that 52% of ChatGPT answers contain incorrect information and 77% are verbose. Nonetheless, our user study participants still preferred ChatGPT answers 35% of the time due to their comprehensiveness and well-articulated language style. However, they also overlooked the misinformation in the ChatGPT answers 39% of the time. This implies the need to counter misinformation in ChatGPT answers to programming questions and raise awareness of the risks associated with seemingly correct answers.

CCS CONCEPTS

• **Human-centered computing → Empirical studies in HCI**; • **Software and its engineering**; • **General and reference → Empirical studies**;

KEYWORDS

stack overflow, q&a, large language model, chatgpt, misinformation

ACM Reference Format:
Samia Kabir, David N. Udo-Imeh, Bonan Kou, and Tianyi Zhang. 2024. Is Stack Overflow Obsolete? An Empirical Study of the Characteristics of ChatGPT Answers to Stack Overflow Questions. In *Proceedings of the CHI Conference on Human Factors in Computing Systems (CHI '24), May 11–16, 2024, Honolulu, HI, USA*. ACM, New York, NY, USA, 17 pages. https://doi.org/10.1145/3613904.3642596

CHI '24, May 11–16, 2024, Honolulu, HI, USA

ACM ISBN 979-8-4007-0330-0/24/05
https://doi.org/10.1145/3613904.3642596

1 INTRODUCTION

Programmers often resort to online resources for a variety of programming tasks, e.g., API learning, bug fixing, comprehension of code or concepts, etc. [70, 75, 86]. A vast majority of these help-seeking activities include frequent engagement with community Q&A platforms such as Stack Overflow (SO) [69, 70, 84, 86]. The emergence of *Large Language Models (LLMs)* has demonstrated the potential to transform the online help-seeking patterns of programmers. In November 2022, ChatGPT [61] was released and quickly gained significant attention and popularity among programmers. There have been increasing debates about whether and when ChatGPT would replace prominent search engines and Q&A forums among researchers and industrial practitioners [22, 68].

Despite the rising popularity of ChatGPT, there are also many increasing concerns. Previous studies show that *LLMs* can acquire factually incorrect knowledge during training and propagate the incorrect knowledge to generated content [9, 33, 35, 39, 56]. Besides, *LLMs* often generate fabricated texts that mimic truthful information and are hard to recognize, especially for users who lack the expertise [14, 21, 29]. Like other *LLMs*, ChatGPT is also plagued with these issues [15, 41, 50, 58]. The prevalence of misinformation, which can easily mislead users, has prompted Stack Overflow to impose a ban on answers generated by ChatGPT [64].

Recent studies have compared ChatGPT to human experts in legal, medical, and financial domains [34, 41]. To the best of our knowledge, no comprehensive analysis has been conducted to investigate ChatGPT's capability to answer programming questions, especially the quality and characteristics of ChatGPT answers in comparison to human answers. If misinformation is prevalent in ChatGPT answers and is hard to recognize, it may inevitably lead to suboptimal design choices and software defects. In the long term, this may jeopardize the quality and robustness of software and cyberinfrastructure in our society, affecting a broader population beyond programmers.

This work aims to bridge the gap by adopting a mixed-methods research design [48] with a combination of manual analysis, linguistic analysis, and user studies to compare human answers and ChatGPT answers to programming questions on Stack Overflow

(SO). Specifically, we performed stratified sampling to collect ChatGPT answers to 517 SO questions with different characteristics (e.g., popularity, question types, recency, etc.). The sample size is statistically significant with a 95% confidence level and 5% margin of error. We manually analyzed ChatGPT answers and compared them with the accepted SO answers written by human programmers. In addition to correctness, we assessed the consistency, comprehensiveness, and conciseness of ChatGPT answers. We found that 52% of ChatGPT answers contain misinformation, 77% of the answers are more verbose than human answers, and 78% of the answers suffer from different degrees of inconsistency to human answers.

Furthermore, to examine how the linguistic features of ChatGPT answers differ from human answers, we conducted a large-scale linguistic analysis on 2000 randomly sampled SO questions. Specifically, we run Linguistic Inquiry and Word Count (LIWC) [66] and sentiment analysis on ChatGPT answers and human answers. Our results show that ChatGPT uses more formal and analytical language and portrays less negative sentiment.

Finally, to capture how different characteristics of the answers influence programmers' preferences between ChatGPT and SO, we conducted a user study with 12 programmers. The study results show that participants' overall preferences, correctness ratings, and quality ratings were more leaning toward human answers from Stack Overflow. However, participants still preferred ChatGPT answers 35% of the time and overlooked misinformation in the answers 39% of the time. When asked why they preferred ChatGPT answers even when they were incorrect, participants suggested the comprehensiveness and articulated language structures of the answers as reasons for their preference, which is consistent with our linguistic analysis result.

Our manual analysis, linguistic analysis, and user study collectively demonstrate that while ChatGPT performs remarkably well in many cases, it frequently makes errors and unnecessarily prolongs its responses. However, ChatGPT answers have richer linguistic features, leading some users to prefer ChatGPT answers over human answers and sometimes overlook the underlying incorrectness and inconsistencies in ChatGPT answers. Our in-depth analysis points towards several challenges and risks of using ChatGPT in programming and also highlights several opportunities for designing new interaction and computational methods to counter misinformation generated by ChatGPT.

To conclude, this paper makes the following contributions:

- We conducted an in-depth analysis of the correctness and quality of ChatGPT answers across four distinct quality aspects for various types of SO question posts.
- We performed a large-scale analysis of the linguistic characteristics of ChatGPT answers and identified distinct linguistic features that are prominent in ChatGPT answers.
- We investigated how real programmers consider answer correctness, quality, and linguistic features when choosing between ChatGPT and Stack Overflow (SO) through a within-subjects user study.
- We provided a comprehensive discussion of the design implications, emphasized the risks of misinformation, and outlined future directions aimed at detecting and mitigating misinformation in AI-assisted programming.
- We made our data and codebooks publicly available at https://github.com/SamiaKabir/ChatGPT-Answers-to-SO-questions to foster future research in this direction.

The rest of the paper is organized as follows. Section 2 describes the related work. Section 3 describes the research questions. Section 4 describes the data collection process and the methodology of our mixed-methods study. Sections 5, 6, and 7 describe the analysis results of our manual analysis, linguistic analysis, and user study respectively. Section 8 discusses the implications of our findings and future research directions. Section 9 describes the limitations of this work. Section 10 concludes this work.

2 RELATED WORK

2.1 Misinformation Generated by LLMs

Previous studies have shown that content generated by *LLM*s may contain hallucinations and misinformation [14, 49, 85]. Some recent work has investigated *LLM*s' capability to generate fake news, images, and videos [27, 35, 89] about a multitude of social phenomena such as politics, elections, disease, economics, etc. These types of misinformation have the power to mislead and misguide people and can potentially impair the normal functionalities of society and cause chaos [33, 45, 49, 79]. Specifically, several studies have investigated the power of AI-generated texts in deceiving people [17, 54]. Zhou et al. [89] further highlight the risks by showing how traditional misinformation detection and mitigation methods often fail to identify misinformation generated by state-of-the-art *LLM*s.

Since its release in November 2022, ChatGPT has surpassed other *LLM*s in popularity among general users. The usability and effectiveness of ChatGPT have been examined in different domains, such as law, medicine, and finance [41]. Like other *LLM*s, ChatGPT also fabricates facts and generates low-quality or misleading information [15, 41, 50, 58]. However, to the best of our knowledge, no studies have investigated the characteristics and human perception of misinformation in ChatGPT's answers to programming questions. Our work aims to bridge this gap with a combination of manual analysis, linguistic analysis, and user studies.

2.2 Help-Seeking Behavior of Programmers

The proliferation of social media and Q&A platforms for programming have immensely shaped the online help-seeking behavior of programmers [1, 76, 81, 84]. Treude et al. [81] investigated the role and benefits of a popular Q&A platform—Stack Overflow (SO)—and found that Stack Overflow is highly effective in code reviews and answering conceptual questions. Through a mixed-methods study, Mamykina et al. [55] show that the chance of getting quick answers from the SO community is high. Despite the popularity and effectiveness of Stack Overflow, several concerns have been raised. For instance, since Q&A platforms are not integrated with IDEs, developers have to constantly switch between their IDEs and Q&A platforms, which may interrupt developers' workflow and impair their performance persistence [6, 76, 84]. Another concern is the presence of toxicity and negative sentiment in people's answers and comments on Stack Overflow. Calefato et al. [19] found that the presence of positive and negative sentiment contributes towards the upvotes and downvotes of SO answers respectively. In a follow-up study, they found that novices and student programmers often encounter arrogant and rude comments on Stack Overflow, which discourages them from posting questions [20]. Asaduzzaman et al. [4] also found that the presence of toxicity and negative emotions in SO answers can discourage follow-up discussions on Stack Overflow. Our study also confirms this, since users prefer ChatGPT answers due to their politeness and positive sentiment.

2.3 Human-AI Collaboration in Programming

Recent studies show that AI pair-programming tools such as GitHub Copilot [38] have shifted developers' behavior from code writing to code understanding and can improve developer productivity [12, 44]. The online help-seeking behavior of developers is also changing along with other behavior shifts. Developers often use Copilot to get quick code suggestions and only turn to web searches to access the documentation or verify the suggestions [7, 73, 82]. However, recent studies show that Copilot often generates code with errors, which can become a liability for programmers [24]. Furthermore, programmers also need to debug and fix those errors and make other modifications in order to integrate generated code into their program context, which may, in turn, impair their productivity [82].

ChatGPT has gained popularity among programmers of all levels since its release in November 2022. One of the main advantages of ChatGPT over GitHub Copilot is that ChatGPT works as a conversational chatbot that allows users to ask questions and give feedback beyond code completion. For instance, programmers can ask a conceptual question about a data type used in a program, ask for a code explanation, and ask how to fix an error message [80]. Recently, GitHub announced GitHub Copilot X, which integrates GPT-4, a more advanced version of the *LLM* behind ChatGPT, into Copilot [36, 37]. Despite the popularity of ChatGPT among programmers, to the best of our knowledge, there is still no in-depth and comprehensive analysis of the characteristics and quality of ChatGPT answers to programming questions. We bridge this research gap by empirically studying ChatGPT answers to programming questions on Stack Overflow.

3 RESEARCH QUESTIONS

This section describes the research questions investigated in this work and the rationale of each research question. The findings of these research questions will deepen our understanding of the characteristics and human perception of ChatGPT answers. They will also shed light on the challenges and risks of using ChatGPT-generated answers for programming and inform the design of new interactive and computational methods to counter misinformation generated by ChatGPT.

- *RQ1. How do ChatGPT answers differ from SO answers in terms of correctness and quality?* Previous work [21, 43, 50] has shown that *LLMs* such as ChatGPT are prone to hallucination and may generate content with low quality. Therefore, we want to assess and quantify the correctness and different quality aspects (e.g., consistency, conciseness, comprehensiveness) of ChatGPT answers to programming questions.
- *RQ2. What are the fine-grained issues associated with each of the correctness and quality aspects?* While RQ1 aims to provide a quantification of the correctness and quality of ChatGPT answers, RQ2 aims to conduct an in-depth, qualitative analysis and develop a taxonomy of the issues in ChatGPT answers. For instance, we are interested in finding out the common symptoms of hallucinations, e.g., conceptual errors, code errors, terminology errors, etc.
- *RQ3. Do the types of SO questions affect the quality of ChatGPT answers?* Previous studies [3, 51] show that linguistic forms of human answers on Stack Overflow vary based on the types of programming questions. For example, *How-to* questions have step-by-step answers, while *conceptual* questions contain descriptions and definitions. We seek to understand if the types of programming questions influence the characteristics of ChatGPT answers in a similar manner.
- *RQ4. Do the language structure and attributes of ChatGPT answers differ from SO answers?* Previous studies [89] show that human-crafted misinformation and machine-generated misinformation have distinct linguistic features, which can facilitate misinformation detection. Prior work [8] has also shown a relationship between linguistic characteristics and the acceptance of Stack Overflow answers. Inspired by these findings, we want to investigate the distinct linguistic characteristics of ChatGPT answers and how they compare to accepted SO answers written by human programmers.
- *RQ5. Do the underlying sentiment of ChatGPT answers differ from SO answers?* Previous studies [57, 67] discuss the harmful effect of toxicity or negative tone in online discussions. Prior work [19] also shows the role of underlying sentiment in the acceptance of SO answers. Therefore, we seek to analyze the sentiment of ChatGPT answers and compare it to accepted answers on Stack Overflow.
- *RQ6. Can programmers differentiate ChatGPT answers from human answers?* We are curious about whether programmers can discern machine-generated answers from human-written answers and what kinds of heuristics they employ to make the decision. Investigating these heuristics is important since it helps identify good practices that can be adopted by the programmer community and inform the design of automated mechanisms.
- *RQ7. Can programmers identify misinformation in ChatGPT answers?* Understanding how programmers identify misinformation in ChatGPT answers is important as it can provide insights about effective mechanisms to counter misinformation. If programmers can identify the misinformation properly, we expect to find out the techniques of identification. Otherwise, if programmers struggle to identify misinformation, we expect to find out the challenges.
- *RQ8. Do programmers prefer ChatGPT over Stack Overflow?* Finally, we want to understand the user preference between ChatGPT and human-generated answers based on the correctness, quality, and linguistic characteristics of the answer.

4 METHODOLOGY

We adopted a mixed-methods research design to answer the research questions in Section 3. Specifically, to answer RQ1-RQ3, we conducted an in-depth manual analysis through open coding and thematic analysis (Section 4.2). To answer RQ4 and RQ5, we conducted a large-scale linguistic analysis and sentiment analysis using automated methods (Section 4.3). To address RQ6 to RQ8, we conducted user studies followed by semi-structured interviews with 12 participants (Section 4.4). The following sections provide a detailed description of each method.

4.1 Data Collection

4.1.1 ***SO Question Collection.*** We consider three characteristics of programming questions—question popularity, posting time, and question type. We adopted a stratified sampling strategy to collect a balanced set of SO questions that fall into different categories w.r.t. their popularity, posting time, and question type. Table 1 shows the distribution of the sampled questions. We describe the sampling procedure below.

First, we collected all questions in the SO data dump (March 2023) [30] and ranked them by their view counts. We used view counts as the popularity metric of SO questions. We selected three categories of questions—the top 10% of questions in the view count ranking (*Highly Popular*), the questions in the middle (*Average Popular*), and the bottom 10% in the ranking (*Unpopular*).

Second, from the three categories of questions above, we moved on to categorize them by their recency. We split questions in each popularity category into two recency categories—questions posted before the release of ChatGPT (November 30, 2022) as *Old*, and questions posted after that time as *New*. We selected the release date of ChatGPT to evaluate how the answer characteristics of ChatGPT reflect the presence or absence of specific knowledge in ChatGPT's training data.

Third, for question types, based on the literature [3, 25, 52, 81], we focused on three common question types—*Conceptual, How-to*, and *Debugging*. We followed prior work [46, 51] and trained a Support Vector Machine (SVM) classifier to predict the type of a SO question based on the question title. The classifier achieves an accuracy of 78%, which is comparable to prior work. Then, we used this classifier to predict the question type of SO questions in each category of questions obtained from the two previous steps.

In the end, we randomly sampled the same number of questions from each category along the three aspects. Given that the question type classifier may not be accurate, we manually validated the question type of each sample and discarded those with the wrong types. We ended up with 517 sampled questions, as shown in Table 1. Additionally, we randomly sampled another set of 2000 questions from the SO data dump for linguistic analysis. Since all collected questions are originally in HTML format, we removed HTML tags and stored them as plain text with their metadata (e.g., tags, view count, types, etc.) in CSV files.

4.1.2 ***ChatGPT Answer Collection.*** For each of the 517 SO questions, the first two authors manually used the SO question's title, body, and tags to form one question prompt[1] and fed that to the free version of ChatGPT, which is based on GPT-3.5. We chose the free version of ChatGPT because it captures the majority of the target population of this work. Since the target population of this research is not only industry developers but also programmers of all levels, including students and freelancers around the world, the free version of ChatGPT has significantly more users than the paid version, which costs a monthly rate of 20 US dollars. The ChatGPT-generated answers are stored in CSV files. Since ChatGPT stores the history of previous input and output of a session, a new chat session was started before feeding each question prompt to ChatGPT. For the additional 2000 SO questions, we developed an automated script to prompt ChatGPT with the gpt-3.5-turbo API. For each question, this script automatically extracted and concatenated its title, body, and tags based on the prompt template and stored ChatGPT answers in CSV files. Each new prompt was conducted via a new API call, which cleared the context history of previous prompts.

[1]Example prompts are included in the Supplementary Material.

4.2 Manual Analysis

In this section, we describe the manual analysis procedure for the 517 ChatGPT answers.

4.2.1 ***Open Coding Procedure.*** To assess the quality and correctness of ChatGPT answers (RQ1), we used a standard NLP data labeling process [72, 88] to label the ChatGPT answers at the sentence level. Over the course of five weeks, the first three authors met six times to generate, refine, and finalize the codebooks to annotate the ChatGPT answers. First, the first two authors familiarized themselves with the data. Each author independently labeled five ChatGPT answers at the sentence level and took notes about their observations. The two authors met to review their labeling notes and performed thematic analysis [16, 40] to categorize the labels into four themes—*Correctness*, *Consistency*, *Comprehensiveness*, *Conciseness*. Then, they developed the initial codebook, relabeled the previous five ChatGPT answers based on the codebook, and met the other co-authors to resolve the disagreements and refine the codebook. After this step, the codebook contained 24 codes in the four themes.

The first two authors then moved on and labeled 20 new ChatGPT answers independently based on the codebook. Since one text span in an answer may suffer from multiple quality issues, the labeling is essentially a multi-label, multi-class classification where labels are not mutually exclusive. Therefore, we cannot use Cohen's Kappa to measure the agreement level between labelers. Instead, we used Fleiss's Kappa [32] score. The initial score was 0.45, which was not high enough to proceed to label more answers. Thus, the authors met again to discuss the labeling. They carefully reviewed each label in the answers and resolved the conflicts. They further refined the codebook by merging redundant codes, improving the definitions of ambiguous codes, and introducing new codes. At the end of this step, there were 21 codes in the codebook.

With the refined codebook, the first two authors re-labeled 10 of the previous 20 answers and confirmed the agreement. Except for disagreement about the definition and usage of 2 codes in *correctness* category, no new disagreement was discovered. At this point, Fleiss's Kappa score was 0.79. Next, the first two authors met the co-authors to review and refine the current codebook and labelings. After this meeting, the codebook was refined to 19 codes.

Finally, the first two authors labeled 20 new ChatGPT answers with the refined cookbook and arrived at a Fleiss's Kappa score of 0.83, which implies substantial agreement. With this codebook, the first two authors split the remaining ChatGPT answers and labeled them separately. The whole labeling process took about 216 person-hours.

Properties of SO questions	Sub-Category	Selection Criteria	# of Questions
Type	Conceptual	(Initial question posts are divided into these three	175
	How-to	sub-categories by implementing an SVM classifier	170
	Debugging	and manually validated afterward.)	172
Popularity	Popular	Highest 10% View Count (Avg. 28750.5)	179
	Average Popular	Average View Count (Avg. 905.3)	165
	Unpopular	Lowest 10% View Count (Avg. 42.1)	173
Recency	Old	Before November 30, 2022 (the release of ChatGPT)	266
	New	After November 30, 2022	251

Table 1: Different properties of SO questions analyzed in the Manual Analysis, sub-categories for each property, and selection criteria for each sub-category of SO questions posts.

4.2.2 *Definitions and Discussion of Codebook.*

The codebook developed in the previous section contains a wide range of fine-grained codes that are used to develop a taxonomy of issues in ChatGPT answers (RQ2). We give a quick overview of these codes below. Section 5 provides more details.

For ***Correctness***, we compared ChatGPT answers with the accepted SO answers and also resorted to other online resources such as blog posts, tutorials, and official documentation. Our codebook includes four types of correctness issues— *Factual, Conceptual, Code*, and *Terminological* errors. Specifically, for incorrect code examples embedded in ChatGPT answers, we identified four types of code errors—*Syntax* errors, *Wrong Logic*, *Wrong API/Library/Function Usage*, and *Incomplete Code*. An answer is considered fully correct if it does not contain any of these errors, i.e., *Factual, Conceptual, Code*, or *Terminological* errors.

For ***Consistency***, we measured the consistency between ChatGPT answers and the accepted human-written answers on Stack Overflow. Note that inconsistency does not imply incorrectness. A ChatGPT answer can be different from an accepted human answer, but it can still be correct. Five types of inconsistencies emerged from the manual analysis—*Factual Inconsistency, Conceptual Inconsistency, Terminological Inconsistency, Coding Inconsistency*, and *Different Number of Solutions* (e.g., ChatGPT provides four solutions where SO gives only one).

For ***Conciseness***, three types of conciseness issues were identified and included in the codebook—*Redundant*, *Irrelevant*, and *Excess* information. *Redundant* sentences reiterate information stated in the question or in other parts of the answer. *Irrelevant* sentences talk about concepts that are out of the scope of the question being asked. And lastly, *Excess* sentences provide information that is not required to understand the answer.

Comprehensiveness is an overall assessment of the entire answer. Thus, the codebook only includes two codes—*Comprehensive*, and *Not Comprehensive*. To consider an answer to be comprehensive, it needs to fulfill two requirements–(1) all parts of the question are addressed in the answer, and (2) a complete solution is provided in the answer.

4.3 Linguistic Analysis

Previous studies show that user preference and acceptance of an SO answer can depend on the underlying emotion, tone, linguistic style, and sentiment in the answer [8, 19, 74]. In this section, we describe the automated methods utilized to determine linguistic features and sentiments of ChatGPT answers.

4.3.1 *Linguistic Characteristics.*

We employed a widely used tool called Linguistic Inquiry and Word Count (LIWC) [66] to analyze the linguistic features of ChatGPT and SO answers. LIWC is a psycholinguistic database that provides a dictionary of validated psycholinguistic lexicons in pre-determined categories that are psychologically meaningful. LIWC counts word occurrence frequencies in each category that holds important information about the emotional, cognitive, and structural components associated with text or speech. LIWC has been used to study AI-generated misinformation [89], emotional expressions in social media posts [53], the success of human answers [8], etc. In our work, we considered the following categories:

- **Linguistic Styles:** We considered four attributes related to linguistic styles—*Analytical Thinking* (complex thinking, abstract thinking), *Clout* (power, confidence, or influential expression), *Authentic* (spontaneity of language), and *Emotional Tone*.
- **Affective Attributes:** Affective attributes capture expressions and features related to emotional status. They include *Affect* (overall emotional expressions, e.g., "happy", "cried"), *Positive Emotion* (e.g., "happy", "nice"), and *Negative Emotion* (e.g., "hurt", "cried").
- **Cognitive Processes:** Cognitive processes represent features that are related to cognitive thinking and processing, e.g., causation, knowledge, insight, etc. For this category, we considered *Insight* (e.g., "think", "know"), *Causation* (e.g., "because"), *Discrepancy* (e.g., "should", "would"), *Tentative* (e.g., "perhaps"), *Certainty* (e.g., "always"), and *Differentiation* (e.g., "but", "else").
- **Drives Attributes:** Drives capture expressions that show the need, desire, and effort to achieve something. For this category, we considered *Drives*, *Affiliation* (e.g., "ally", "friend"), *Achievement* (e.g., "win", "sucess"), *Power* (e.g., "superior"), *Reward* (e.g., "prize", "benefit"), and *Risk* (e.g., "danger", "doubt").
- **Perceptual Attributes:** This category captures the attributes that are related to *Perceive, See, Feel*, or *Hear*.
- **Informal Attributes:** This category captures the causality in everyday conversations. The attributes in this category include *Informal Language, Swear Words, Netspeak* (e.g., "btw, lol"), *Assent* (e.g., "OK", "Yeah"), *Nonfluencies* (e.g., "er", "hmm"), and *Fillers* (e.g., "I mean", "you know").

We used LIWC to compute word frequency in each of the categories for 2000 ChatGPT answers and the corresponding human answers from Stack Overflow. For ease of understanding, we computed the relative differences (*RD*) in linguistic features between 2000 pairs of ChatGPT and SO answers from the computed average word frequencies in each category.

$$RD = \frac{ChatGPT\ \ avg.\ \ frequency - SO\ \ avg.\ \ frequency}{SO\ \ avg.\ \ frequency}$$

4.3.2 ***Sentiment Analysis***. Lexicon-based LIWC evaluates linguistic characteristics based on psycholinguistic features and captures the sentiment of texts only based on overall polarity. Hence, LIWC is insufficient when it comes to capturing the intensity of the polarity [13]. Moreover, LIWC can not capture sarcasm, irony, misspelling, or negation, which is necessary to analyze sentiment in human-written texts on Q&A platforms. Therefore, we employed a machine learning algorithm to further evaluate and compare the underlying sentiment portrayed in the ChatGPT answers and human answers. Specifically, we used a RoBERTa-based sentiment analysis model from Hugging Face [31]. This model is pre-trained on a Twitter corpus and is then finetuned with the 4423 annotated SO posts from Calefato et al. [18]. This well-balanced dataset has 35% posts with positive sentiment, 27% of posts with negative sentiment, and 38% of posts with neutral sentiment.

4.4 User Study

To understand programmers' perception of ChatGPT answers and human answers, we conducted a within-subjects user study with 12 participants. Our goal is to observe how programmers assess those answers and which kind of answers they prefer.

4.4.1 ***Participants***. For the user study, we recruited 12 participants (3 female, 9 male) with programming backgrounds. 7 participants were graduate students, 4 participants were undergraduate students, and 1 participant was a software engineer from the industry. The participants were recruited by word of mouth. Participants rated their programming expertise by answering multiple-choice questions with five options—*Novice, Beginner, Competent, Proficient, and Expert*. Eight participants self-reported as proficient, three as competent, and one as beginner. Since some participants may be modest about their programming skills, we also collected the number of years of programming experience. Four participants had three years of experience, one had four years, one had five years, two had six years, one had seven years, and three had eight years of programming experience. Additionally, we asked participants how often they use ChatGPT and how often they use SO. For ChatGPT, three answered *very often*, three answered *some of the time*, two answered *seldom*, and four answered *never*. For SO, four participants answered *all the time*, five answered *very often*, two answered *some of the time*, and one answered *seldom*.

4.4.2 ***SO Question Selection***. We randomly sampled eight questions from our manual analysis dataset. ChatGPT gave incorrect answers to five questions and correct answers to three questions. One question was about C++, one about PHP, two about HTML/CSS, three about JavaScript, and one about Python.

4.4.3 ***Protocol***. In this user study, we asked participants to complete a sequence of decision-making tasks to verify and assess the quality of machine and human-generated answers to programming questions. The tasks were designed to capture user perception and preference for human and machine-generated answers. In each task, we asked the participants to verify and assess a ChatGPT answer and a human answer to a SO question and rate the correctness and quality of each answer. Moreover, for each task, the participants were asked to mark which answer they preferred and guess which answer was generated by ChatGPT. The step-by-step procedure for each task is described below.

Each user study started with consent collection and an introduction to the study procedure. Then, the participants started the study tasks by reading each SO question and rating their familiarity with the topic asked in the question (5-point Likert Scale [60]). Familiarity with a specific programming topic is not directly related to years or hours of programming experience, as programmers tend to be more familiar with topics they have used more recently. Therefore, we resorted to the self-reporting method. Then, they were presented with an answer to the question. This answer is either generated by ChatGPT or written by a human programmer on Stack Overflow. Then, they were asked to answer a series of 5-point scale survey questions to assess the correctness, comprehensiveness, conciseness, and usefulness of this answer. Then, they were presented with the other answer and asked to answer the same set of survey questions. Then, they were asked to select which answer they prefer, which answer they believe is generated by ChatGPT, and how confident they are about their choices. We repeated this process for all eight SO questions and randomized the order of ChatGPT and human answers for each question. For ease of running this study, all SO questions, answers, survey questions, and instructions were encoded into a Qualtrics survey.[2]

The human answers and ChatGPT answers were presented with the same text format and style (e.g., font type, font size, code format, etc.), so participants could not easily tell them apart just based on formatting and visual styles. Participants were allowed to skip to the next SO question if they were not familiar with the topic of a certain SO question. The order of ChatGPT and human answers was assigned randomly (i.e., not always Answer 1 was ChatGPT answer). Additionally, participants were encouraged to refer to external resources, such as Google search, tutorials, and API documentation, to verify the correctness of the given answers. In the verification process, to prevent participants from running into the same human answer on Stack Overflow or getting the same answer from ChatGPT, the participants were not allowed to search on Stack Overflow, open a Stack Overflow page returned by Google Search, or ask the same question to ChatGPT. Apart from accessing ChatGPT and SO for the same question, participants were allowed to validate the code generated by ChatGPT in any local IDE, online sandbox, or online code editor at their convenience. Each participant was given 20 minutes to examine and rate answers to SO questions. Participants were made aware that finishing all eight questions was not required and were encouraged to aim for comprehensiveness and quality instead of the number of examined answers. All participants used up the given 20 minutes of time in the study. On average, participants assessed the correctness and quality of the answers to 5 questions.

4.4.4 ***Semi-Structured Interview***. The survey was followed by a lightweight semi-structured interview. Each interview took about 10 minutes on average. During the interview, we reviewed the participant's responses to the survey together with the participant

[2]The survey is included in Supplementary Material

and asked them why they preferred one answer over the other. Then, we asked the participants about their heuristics to identify the ChatGPT answer before revealing the correct answer to them. If the participants were correct, we asked a follow-up question about the characteristics of ChatGPT answers that influenced their decision. Lastly, we asked how they determined the incorrect information in an answer. We also asked follow-up questions such as why they failed to identify some misinformation, what the main challenges were in verifying the correctness, what additional tool support they wish to have, etc.[3]

4.4.5 ***Qualitative Analysis of the Interview Transcripts.*** The first author transcribed the audio recordings and labeled all 12 transcripts following the open coding method [42]. The author labeled all insightful responses that mentioned factors related to participants' preferences, the heuristics used by the participants, the obstacles they faced, and the tool support they wished to have. After this step, the author did a thematic analysis [16, 40] to group the low-level labels into high-level patterns and themes. The final codebook for thematic analysis contains 5 themes and 21 patterns.[4] The overall process took about six person-hours.

5 MANUAL ANALYSIS RESULTS

This section presents the results and findings for RQ1-RQ3.

5.1 RQ1: Overall Correctness and Quality

Our results show that, among the 517 ChatGPT answers we labeled, 52% of them contain incorrect information, 78% are inconsistent from human answers, 35% lack comprehensiveness, and 77% contain redundant, irrelevant, or unnecessary information. Moreover, on average, ChatGPT answers and human answers contain 266.43 tokens (σ=87.99) and 213.80 tokens (σ=246.04) respectively. The mean difference of 52.63 tokens is statistically significant (paired t-test: p-value<0.001). Table 2 shows our manual analysis results.

Finding 1

More than half of ChatGPT answers contain incorrect information, 77% of ChatGPT answers are verbose, and 78% of ChatGPT answers contain inconsistencies with human answers. However, ChatGPT answers are comprehensive and cover different aspects of the questions and answers.

5.2 RQ2: A Taxonomy of Fine-Grained Issues in ChatGPT Answers

Our thematic analysis reveals four types of ***incorrectness*** in ChatGPT answers—*Conceptual* (54%), *Factual* (36%), *Code* (28%) and *Terminology* (12%) errors. Note that these errors are not mutually exclusive. Some answers have more than one of these errors. *Factual* errors occur when ChatGPT states some fabricated or untruthful information about existing knowledge, e.g., claiming a certain API solves a problem when it does not, fabricating non-existent links, untruthful explanations, etc. On the other hand, *Conceptual* errors occur if ChatGPT fails to understand the question. For example, the user asked how to use public and private access modifiers, and ChatGPT answered the benefits of encapsulation in C++. *Code* errors occur when the code example in the answer does not work, or cannot provide a desired output. And lastly, *Terminology* errors are related to wrong usages of correct terminology or any use of incorrect terminology, e.g., *perl* as a header of *Python* code.

Specifically, for code errors, our analysis reveals four types of code errors—wrong logic (48%), wrong API/library/function usage (39%), incomplete code (11%), and wrong syntax (2%). Again, some generated code has more than one of these errors. Logical errors are made by ChatGPT when it can not understand the problem, fails to pinpoint the exact part of the problem, or provides a solution that does not solve the problem. For example, in many debugging instances, we found that ChatGPT tried to resolve one part of the given code, whereas the problem lied in another part of the code. One such example is provided in Appendix A. We also observed that ChatGPT often fabricated APIs or claimed certain functionalities that were wrong.

Finding 2

Many answers are incorrect due to ChatGPT's incapability to understand the underlying context of the question being asked. Yet ChatGPT makes fewer factual errors compared to conceptual errors.

Finding 3

ChatGPT rarely makes syntax errors for code answers. The majority of the code errors are due to applying wrong logic or implementing non-existing or wrong API, library, or functions.

The ChatGPT answers that have no statements annotated as factual, conceptual, code, or terminological errors, are considered to be correct. In the manual analysis, we found that 48% of the ChatGPT answers had an absence of any type of fine-grained errors.

Among the answers that are ***Not Concise***, 46% of them have *Redundant* information, 33% have *Excess* information, and 22% have *Irrelevant* information. For *Redundant* information, during our labeling process, we observed that many of the ChatGPT answers repeat the same information that is either stated in the question or stated in other parts of the answers. For *Excess* information, we observed a handful of cases where ChatGPT unnecessarily gives background information such as long definitions, or writes something at the end of the answer that does not add any necessary information to understand the solution. Lastly, many answers contain *Irrelevant* information that is out of context or scope of the question. In answers with conceptual errors, we observed this behavior more often. There are answers that have a combination of more than one of these conciseness issues. An example of a verbose ChatGPT response is provided in Appendix B.

And lastly, for inconsistency with human answers, we found five types of ***Inconsistencies***—*Conceptual* (67%), *Factual* (44%), *Code* (55%), *Terminology* (6%), and *Number of Solutions* (42%). The first four types of inconsistencies occur for the same reason as incorrectness. The only difference is that inconsistency does not always mean incorrectness, as explained in Section 4.2.2. Similar to incorrectness, conceptual inconsistencies are higher than factual inconsistencies. Our observation also reveals that ChatGPT-generated code is very different from human-written code in format, semantics, syntax, and logic. This contributes to the higher number of *Code*

[3]A complete list of interview questions are included in the Supplementary Material.
[4]We have included the codebook in Supplementary Material.

		Correct		Consistent		Comprehensive		Concise	
		Yes	No	Yes	No	Yes	No	Yes	No
Popularity	Popular	0.55	0.45	0.21	0.79	0.64	0.36	0.16	0.84
	Avg. Popular	0.46	0.54	0.22	0.78	0.64	0.36	0.26	0.74
	Not Popular	0.42	0.58	0.25	0.75	0.66	0.34	0.28	0.72
Type	Debugging	0.45	0.55	0.17	0.83	0.63	0.37	0.40	0.60
	How-to	0.47	0.53	0.21	0.79	0.67	0.33	0.13	0.87
	Conceptual	0.48	0.52	0.28	0.72	0.64	0.36	0.16	0.84
Recency	Old	0.53	0.47	0.22	0.78	0.68	0.32	0.17	0.83
	New	0.42	0.58	0.22	0.78	0.61	0.39	0.29	0.71
Overall	_	0.48	0.52	0.22	0.78	0.65	0.35	0.23	0.77

Table 2: Percentage distribution of ChatGPT answers for all 4 correctness and quality issues (Correctness, Consistency, Comprehensiveness, and Conciseness) across 3 properties of question posts (Popularity, Type, and Time). The statistically significant (Pearson's Chi-square Test: p-value<0.05) relations are highlighted in blue.

inconsistencies. The *Number of solutions* inconsistency is also very prominent as ChatGPT often provides many additional solutions to solve a problem.

5.3 RQ3: Effects of Question Type

To evaluate the relationship between question types and ChatGPT answer quality, we calculated the percentage of each label across all categories for each question type. As our data is entirely categorical, we evaluated the statistical significance of the relationship between each question type and each of the four label categories with Pearson's Chi-square test. Table 2 highlights all relationships that are statistically significant (p-value<0.05). Our results show that ***Question Popularity*** and ***Recency*** have a statistically significant impact on the ***Correctness*** of answers. Specifically, answers to popular questions and questions posted before November 2022 (the release date of ChatGPT) have fewer incorrect answers than answers to other questions. This implies that ChatGPT generates more correct answers when it has more information about the question topic in its training data. Although ***Debugging*** questions have more incorrect ChatGPT answers, the difference is not statistically significant. This indicates that ***Question Type*** does not affect the ***Correctness*** of ChatGPT answers.

Additionally, we found a statistically significant relationship between ***Question Type*** and ***Inconsistency***. Since there are often multiple ways to debug and fix a problem, the inconsistencies between human and ChatGPT-generated answers for ***Debugging*** questions are higher, with 83% of ***inconsistent*** answers. Our observation aligns with this result too. While labeling the answers, we found that almost half of the correct ***Debugging*** answers use different logic, API, or library to solve a problem that produces the same output as human answers.

Our results also show that ChatGPT answers are consistently ***Comprehensive*** for all categories of SO questions and do not vary with different ***Question Type, Recency,*** or ***Popularity***.

Moreover, our analysis shows that answers to all kinds of questions, irrespective of the ***Type, Recency,*** and ***Popularity***, are consistently verbose. Yet answers to different kinds of questions indeed have statistical differences in verbosity. Specifically, answers to ***Popular*** questions are ***Not Concise*** 84% of the time, while answers for ***Average*** and ***Not Popular*** questions are ***Not Concise*** 74% and 72% of the time. This suggests that for questions targeting popular topics, ChatGPT has more information on them and adds lengthy details. We found the same pattern for ***Old*** questions. Answers to ***Old*** questions (83%) are more verbose than ***New*** questions (71%). Finally, for ***Question Type***, ***Debugging*** answers are more ***Concise*** (40%) compared to ***Conceptual*** (16%) and ***How-to*** (13%) answers, which are extremely verbose. This is because of ChatGPT's tendency to elaborate definitions for ***Conceptual*** questions and to generate step-by-step descriptions for ***How-to*** questions.

Finding 4

Popularity, Type, and Recency of programming questions affect the correctness and quality of ChatGPT answers. Answers to more *Popular* and *Older* posts are less incorrect and more verbose. *Debugging* answers are more inconsistent but less verbose. *Conceptual* and *How-to* answers are the most verbose.

6 LINGUISTIC ANALYSIS RESULTS

6.1 RQ4: Linguistic Characteristics

Table 3 presents the relative differences in the linguistic features between ChatGPT answers and human answers. As stated in Section 4.3, relative differences capture the normalized difference in word frequencies for each linguistic feature between ChatGPT answers and human answers. Positive relative differences indicate features prominent in ChatGPT answers, and negative relative differences indicate features prominent in human answers. Our result shows several statistically significant linguistic differences between ChatGPT answers and human answers.

First, we found that ChatGPT answers differ from human answers in terms of ***language styles***. ChatGPT answers are found to contain more words related to *analytical thinking and clout expressions*. This indicates that ChatGPT answers communicate a more abstract and cognitive understanding of the answer topic, and the language style is more influential and confident. On the other hand, human answers include fewer words related to *authenticity*, indicating that human answers are more spontaneous and non-regulated.

For ***affective*** attributes that capture emotional status, we found human answers contain more keywords related to emotional status. Though not statistically significant, ChatGPT answers portray more positive emotions, whereas human answers portray significantly more negative emotions than ChatGPT.

Moreover, ChatGPT answers contain significantly more ***drives*** attributes compared to human answers. ChatGPT conveys stronger

Linguistic Features	Rel. Diff.(%)	Linguistic Features	Rel. Diff.(%)
		Drive Attributes	
Language Styles		Drives	9.53***
Analytic	20.65***	Affiliation	16.05**
Clout	13.01***	Achievement	10.85***
Authentic	-38.50***	Power	22.86***
Tone	14.95***	Reward	2.23
		Risk	-7.08
		Perception Attributes	
Affective Attributes		Perception	-26.28***
Affect	-6.53**	See	-34.98***
Positive Emotion	2.09	Hear	-16.50*
Negative Emotion	-34.45***	Feel	7.55
Cognitive Attributes		**Informal Attributes**	
Insight	-8.86**	Informal Language	-53.97***
Causation	23.94***	Swear words	-71.52**
Discrepancy	-35.89***	Netspeak	-60.03***
Tentative	-10.23***	Assent	-11.86
Certainty	-4.23	Nonfluencies	-55.34***
Differentiation	-13.29***	Fillers	-82.85***

Table 3: Relative Linguistic Differences (%) between 2000 pairs of ChatGPT and human answers. Positive numbers indicate higher occurrence frequencies of linguistic features in ChatGPT answers compared to SO, and negative numbers indicate lower occurrence frequencies. Numbers marked with (*) indicate differences that are statistically significant (paired t-test: * means p-value<0.001, ** means p-value<0.01, * means p-value<0.05)**

drives, affiliation, achievement, and *power* in its answers. We observed that many ChatGPT answers include words and phrases, such as "of course I can help you" and "this will certainly fix it." This observation aligns with the higher *drives* attributes in ChatGPT-generated answers. However, ChatGPT answers do not convey risks as much as human answers do. This indicates that human answers on Stack Overflow often warn programmers of the side effects of solutions more than ChatGPT does.

For ***informal*** attributes, human answers are highly informal and casual. On the contrary, ChatGPT answers are very formal and do not make use of swear words, netspeak, nonfluencies, or fillers. In our observation, we rarely saw ChatGPT using a casual conversation style. On the other hand, human answers often had words such as "btw", "I guess", etc. Human answers also contain higher ***perceptual*** and ***cognitive*** keywords than ChatGPT answers. According to the definitions of ***perceptual*** and ***cognitive*** attributes by LIWC (Section 4.3), this indicates that human answers portray more personal observations and insights from human programmers when answering the question.

Finding 5

Compared to human answers, ChatGPT answers are more formal, express more analytic thinking, showcase more efforts towards achieving goals, and exhibit less negative emotion.

6.2 RQ5: Sentiment Analysis

Our results show that, among the 2000 ChatGPT answers, 1707 (85.35%) of them portray positive sentiment, 291 answers (14.55%) portray neutral sentiment, and only 2 answers (0.1%) portray negative sentiment. On the other hand, 1466 of the 2000 SO answers (73.30%) portray positive sentiment, 513 answers (25.65%) portray neutral, and 21 answers (1.05%) portray negative sentiment. To assess the sentiment difference between ChatGPT and SO answers, we performed a McNemar-Bowker test on the sentiments. Since we have paired-nominal data, we opted for the McNemar-Bowker test for testing the goodness of fit when comparing the distribution of counts of each label. The results are statistically significant ($X^2 = 186.84, df = 3, p < 0.001$). Our results show that for 13.90% questions, ChatGPT answers portrayed positive sentiment while human answers portrayed neutral or negative sentiments. On the other hand, only 2 ChatGPT answers portrayed negative sentiment when the human answers were positive or neutral. Our result indicates that ChatGPT shows significantly more positive sentiment compared to human answers.

Finding 6

ChatGPT answers portray significantly more positive sentiments compared to human answers on Stack Overflow.

7 USER STUDY RESULTS

We retrieved 56 pairs of ratings of ChatGPT answers and human answers as rated by 12 participants. Figure 1 presents the average ratings of the two kinds of answers in all four quality aspects. Overall, users found human answers to be more correct (mean rating human: 4.41, ChatGPT: 3.21, Welch's t-test: p-value<0.001), more concise (human: 4.16, ChatGPT: 3.69, Welch's t-test: p-value<0.05), and more useful (human: 4.21, ChatGPT: 3.42, Welch's t-test: p-value<0.01). For comprehensiveness, the average ratings are 3.89 and 3.98 for human answers and ChatGPT answers respectively. However, this result is not statistically significant.

Additionally, our thematic analysis revealed five themes—*Process of differentiating ChatGPT answers from human answers*, *Heuristics of verifying correctness*, *Reasons for incorrect determination*, *Desired support*, and *Factors that influence user preference.* Findings from our quantitative and thematic analysis for each of the research questions are described in the following subsections.

7.1 RQ6: Differentiating ChatGPT answers from human answers

Our study results show that participants successfully identified which one is the machine-generated answer 80.75% of the time and failed only 19.25% of the time (Welch's t-Test, p-value<0.001).

From thematic analysis, we identified the factors that participants found helpful to discern ChatGPT answers from human answers. 6 out of 12 participants reported the writing style of answers to be helpful in identifying the ChatGPT answer. Participant P5 mentioned, *"good grammar"*, and P8 mentioned, *"header, body, summary format"* to be contributing factors for identification. Two other factors are language style (e.g., casual or formal language, format) (10 out of 12 participants) and length (7 out of 12 participants). Additionally, 5 participants found unexpected or impossible errors as a helpful factor in identifying the machine-generated answers. Apart from these, tricks and insights that only experienced people can provide (5 out of 12 participants), and high entropy generation

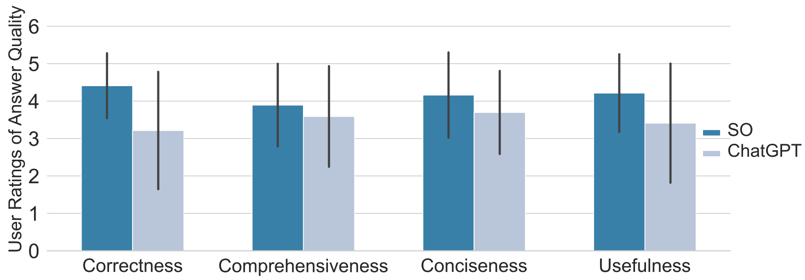

Figure 1: Quality of answers as rated by participants. Difference in *Correctness, Conciseness,* and *Usefulness* are statistically significant (Paired t-Test, p-value<0.05)

(1 out of 12 participants) were two other reported factors. Our result suggests that most participants use language and writing styles, length, and the presence of abnormal errors to determine the source of an answer.

Finding 7

Participants can correctly discern ChatGPT answers from human answers over 80% of the time. They look for factors such as formal language, structured writing, answer length, or unusual errors to decide whether an answer is generated by ChatGPT.

7.2 RQ7: Assessing Answer Correctness

Our study result shows that users could successfully identify the incorrect answers only 60.66% of the time and failed 39.34% of the time (Welch's t-test, p-value<0.05).

When we asked users how they identified incorrect information in an answer, we received three types of responses. 10 out of 12 participants mentioned they read through the answer, tried to find any logical flaws, and tried to assess if the reasoning made sense. 7 participants mentioned they identified the terminology and concepts they were not familiar with and did a Google search, and read documentation to verify the solutions. And lastly, 4 out of 12 users mentioned that they compared the two answers and tried to understand which one made more sense to them. All of the aforementioned verification processes involved assessing the code or part of the answers in external IDEs. All of our participants copied code or tested part of the solution from at least one answer into their local IDE for validation, 9 participants utilized some online code sandbox for validation (e.g., sandbox for HTML, CSS, JS), and 6 participants used the built-in code editor from tutorial sites such as W3Schools as a part of their assessment process.

When a participant failed to correctly identify the incorrect answer, we asked them what could be the contributing factors. 7 out of 12 participants mentioned the logical and insightful explanations, and comprehensive and easy-to-read solutions generated by ChatGPT made them believe it to be correct. 6 participants mentioned lack of expertise to be the reason. However, we ran Pearson's Chi-Square test to evaluate the relationship between overlooking incorrect answers and topic expertise and found no significant relation between these two. P7 and P10 said ChatGPT's ability to mimic human answers made them trust the incorrect answers.

Additionally, participants expressed their desire for tools and support that can help them verify the correctness. 10 out of 12 participants emphasized the necessity of verifying answers generated by ChatGPT before using it. Participants also suggested adding links to official documentation and supporting in-situ execution of generated code to ease the validation process.

Finding 8

Users overlook incorrect information in ChatGPT answers 39.34% of the time due to the comprehensive, well-articulated, and humanoid insights in ChatGPT answers.

7.3 RQ8: Factors for User Preference

Participants preferred SO answers 65.18% of the time. However, participants still preferred ChatGPT answers 34.82% of the time (Welch's t-test, p-value<0.01). Among the ChatGPT preferences, 77.27% of the answers were incorrect.

For *factors that influence user preference*, 10 out of 12 participants mentioned correctness to be the main contributing factor for preference. 8 participants mentioned answer quality (e.g., conciseness, comprehensiveness) as contributing factors. 6 participants mentioned they put emphasis on how insightful and informative the answer is while preferring. 6 participants stated language style to be one of the factors, 2 of these 6 participants preferred the casual, spontaneous language style of human answer, while the other 4 preferred the well-structured and polite language of ChatGPT. P2 mentioned, *"It feels like it's trying to teach me something"*. Finally, 5 participants mentioned the format, look and feel (e.g., highlighting, color scheme) as contributing factors toward preference.

Finding 9

Participants preferred human answers from Stack Overflow more than ChatGPT answers (65.18% of the time). Participants found human answers to be more correct, concise, and useful.

8 DISCUSSION AND FUTURE WORK

In this section, we discuss the implications of our findings and future directions to counter misinformation when using ChatGPT for programming.

8.1 Why Do Users Prefer ChatGPT Responses?

Surprisingly, our user study shows that participants preferred ChatGPT answers 34.82% of the time, though 77.27% of these answers contained misinformation. Furthermore, we observed that participants overlooked a lot of misinformation in ChatGPT answers. Specifically, when ChatGPT answers are not readily verifiable (e.g., requiring execution in an IDE or needing to go through long documentation to validate), users often fail to identify the misinformation and underestimate the degree of incorrectness in the answer.

The follow-up semi-structured interviews revealed that the polite language, articulated and text-book style answers, and comprehensiveness are some of the main reasons that made ChatGPT answers look more convincing, so the participants lowered their guard and overlooked some misinformation in ChatGPT answers. This finding is consistent with previous findings of user preferences over Stack Overflow (SO) posts. Prior work [4, 8, 19, 59] shows that SO users preferred posts that contain illustrations, step-by-step instructions, multiple solutions, and positive sentiments. Our linguistic analysis shows that ChatGPT answers possess many of these linguistic characteristics that SO users appreciate.

Recently, there has been a decline in network traffic to the Stack Overflow website, which was attributed to the rise of ChatGPT [63]. Although our user study does not evaluate what encourages users to ask a question to ChatGPT rather than Stack Overflow in the first place, our findings point to some possible reasons. We believe the fact that users can avoid the embarrassment of posting online and the risk of receiving negative comments but still receive seemingly high-quality answers in a timely manner can be some contributors. Moreover, the interactive feature of ChatGPT makes it easier for users to change prompts and interactively work with the language model to make it generate desired or optimal answers. Using interactivity to rectify errors can be another contribution to ChatGPT's popularity among programmers.

8.2 Where Do Errors in ChatGPT Answers Emerge from?

It is evident from our results that ChatGPT produces incorrect answers more than half of the time. Our observation sheds light on three main reasons for these errors.

Lack of Understanding for Some Programming Concepts. First, 54% of the time, errors are made due to ChatGPT not understanding the concepts mentioned in a question. For example, we found a JavaScript question about a website not showing the File Upload option [62]. Clearly, it is an issue with the front end and User Interface (UI), since the question mentioned "the file upload area not working" and provided JavaScript and HTML code snippets. In this context, "the file upload area" refers to the UI widget to upload a file, rather than the action of uploading a file. ChatGPT did not get this and answered a handful of irrelevant solutions, such as how the file path needs to be set, how to locate the file in your machine, CORS issues, etc. By contrast, the human-written answer suggests adding an appropriate *id* to the file input field in the HTML code. These types of misunderstanding issues contribute to the high number of *Conceptual* errors.

Limited Capability to Understand and Reason Program Semantics. Our manual analysis reveals that while most of the code examples (98%) generated by ChatGPT are syntactically correct, many of them contain incorrect logic (48%) or incorrect API usage (39%). We suspect this is largely due to ChatGPT's limited capability to understand and reason program semantics. In many cases, ChatGPT makes obvious programming mistakes that human programmers barely make. For example, ChatGPT may generate a loop ending condition that is always true or false, e.g. `while(i<0 && i>10)`. Furthermore, the content generation process in ChatGPT is essentially an auto-regressive decoding process guided by the probability distribution at each token prediction step. Thus, ChatGPT cannot foresee the potential outcome or execution result of the generated code. For example, we observed that ChatGPT generated a code example that keeps decreasing a variable in a for loop and eventually leads to a division-by-zero exception in the end. ChatGPT seems unable to understand the consequences or side effects of some code operations and expressions.

Missing or Incorrect Attention to a Programming Question. Since questions asked in SO are long human-written questions with many components involved, ChatGPT often focuses on the wrong part of the question or gives high-level solutions without fully understanding the minute details of a problem. For example, we found an instance where the SO question asked about differences between public, private, and protected access modifiers in Java. However, ChatGPT only focused on the part "access modifiers" ignoring the "difference" part in the question. Therefore, it gives an extremely verbose response that contains the definitions of encapsulation, inheritance, etc., which is not useful in terms of identifying the differences originally asked for.

8.3 What Is at Stake and What Does the Future Hold?

Impact on the Software Industry and Society. We believe that the large number of seemingly correct ChatGPT answers pose high risks to programming practices since they can easily trick programmers into thinking they are correct, especially when programmers lack the expertise or means to verify the correctness. As AI Chain frameworks are getting increasingly popular, it becomes riskier when ChatGPT answers are automatically integrated into downstream AI components with no human involvement and validation. The misinformation will propagate along the AI chain and may have devastating effects on downstream tasks. In the long term, this could jeopardize the quality and robustness of software and cyberinfrastructure in our society, since the misinformation in these answers may lead to suboptimal design decisions and software defects. The repercussions can potentially affect other societal factors, including the safety, security, and trust of the general population.

Impact on STEM Education. Many STEM fields, beyond Computer Science, require students to learn basic programming. Students using ChatGPT for learning materials may be misled into learning incorrect concepts and information. This may even harm the grades or reputation of students. We believe identifying and verifying errors in programming answers require as much expertise as learning and writing code. Hence, learning through the wrong materials has the potential to create a chain of misinformation where the veracity assessment of students and learners will be compromised in the long term.

The Silver Lining. While our manual analysis reveals 52% of the answers are incorrect, 48% answers are completely correct (i.e., no statements in those answers annotated with factual, conceptual, code, or terminological errors), which by no means is an insignificant number. Compared with Stack Overflow, ChatGPT can give immediate answers to users' questions, significantly saving the time and effort of users. Thus, conversational chatbots such as ChatGPT may be considered more convenient than Q&A forums. Programmers of all levels, including students and professional developers, may find it easy and less time-consuming to ask basic programming questions instead of going to instructors, mentors, or even posting on traditional Q&A platforms.

Hence, along with trying to rectify the error and mitigate the risks, steps should be taken to create awareness and adopt new strategies and policies to address the risks associated with incorrect information generated by ChatGPT.

8.4 What Further Actions are Needed to Address Misinformation in ChatGPT?

8.4.1 Limitations of Existing Approaches. Although approaches have been proposed to mitigate hallucinations from *LLM*s [28, 65], they are only applicable to fixing *Factual* errors. Since the root of *Conceptual* errors is not hallucinations but rather a lack of understanding of programming concepts and incapability to reason program semantics, existing approaches for hallucination may not be effective in mitigating conceptual errors.

Most of the existing methods to help *LLM*s understand and reason rely on *Prompt Engineering*. While *Prompt Engineering* can be helpful in probing ChatGPT to understand a problem to some extent [77, 90], they are still insufficient when it comes to injecting reasoning into *LLM*s to solve special cases. Moreover, *Prompt Engineering* is not a sustainable solution and the responsibility largely falls on users.

Furthermore, ChatGPT provides different answers even when prompted with the same questions. This makes the verification process even harder since users cannot deterministically identify the prompts that will always result in correct or optimal solutions. Although lowering the temperature value can help in achieving consistent answers for the same prompts, lower temperature often reduces the quality of answers generated by *LLM*s. Thus, this variability adds another dimension to the challenges already posed by *Prompt Engineering*. Additionally, *Prompt Engineering* implies that to make ChatGPT give the right answer, users need to ask the right question. Thus, overly relying on *Prompt Engineering* to make ChatGPT produce the correct answer shifts the responsibility for AI errors to humans. Hence, we urge that instead of temporary patches such as changing prompts that also make humans somewhat responsible for the errors made by ChatGPT, it is essential to understand the sources and factors of conceptual errors in order to develop sustainable and special-purpose solutions to fix them.

8.4.2 Communicating the level of incorrectness is necessary. The user interface of ChatGPT includes a one-line warning—"*ChatGPT may produce inaccurate information about people, places, or facts.*" However, we believe such a generic warning is insufficient. Each answer should be accompanied by a level of incorrectness and uncertainty in the answer. Moreover, our observations indicate that not all answers have an equal amount of incorrectness—some answers have the majority of parts marked as incorrect, whereas some answers have only a few lines marked as incorrect. Since each incorrect answer differs in the severity of incorrectness, it is vitally important to provide users with the level of incorrectness for each answer. A recent study shows that an *LLM* may know when it is lying [5], which can be leveraged to warn users about the potential errors made by LLMs. However, recent studies [2, 83] also show that only rendering the confidence level is not sufficient to help programmers understand the uncertainty and risks in the generated code. Thus, it is necessary to investigate more effective communication and visualization methods for model uncertainty in programming tasks.

Moreover, for software companies, it is worthwhile to invest in more awareness campaigns and training for software developers. Special training is necessary for software developers so that they can monitor the code bases, readily verify errors in ChatGPT answers, and perform more testing to safeguard errors from sneaking into their codebases. In particular, software developers should be advised to use ChatGPT with more caution and scrutiny for high-stake code blocks and programming tasks.

8.4.3 More rigorous code reviews and testing are needed. Software companies should enforce more rigorous code reviews and software testing methods to source code that is produced with the facilitation of ChatGPT and other AI technologies. Since ChatGPT may make programming mistakes that human programmers barely make, it is important to adapt traditional methods to account for the types of programming mistakes generated by ChatGPT or other *LLM*s. Additionally, it is necessary to have continuous testing and security checking so that incorrect or insecure code can not seep into any part of the software products. Moreover, ChatGPT can be integrated into the testing pipeline as ChatGPT can potentially generate test cases on the fly. Hence, encouraging the integration of testing during the generation process can limit the risk of programming mistakes made by ChatGPT.

8.4.4 Future actions for academics and researchers. Bender and Koller [10] show that any *LLM*s trained only on the form of language can not fully reach the human level of understanding. They argued that to aid *LLM*s in performing natural language understanding, it is imperative to have information in the training data that goes beyond just the form of language, e.g., code paired with several input and correlated output, edge cases, etc. Furthermore, Bender and Gebru et al. [9] argue that increasing the size of language models is not a solution to achieving natural language understanding. We believe one of the main reasons behind the large number of conceptual errors can be attributed to ChatGPT's limitation in performing natural language understanding. Moreover, although existing work [23, 87] shows the challenges and limitations of reasoning in *LLM*s and presents Knowledge Graphs as a powerful method to aid in reasoning, our results highlight the limitation in reasoning when it comes to programming answers or code solutions. Therefore, we urge the attention of the research community for rigorous investigation and mitigation methods to improve the reasoning and understanding capability of *LLM*s, especially in the field of programming.

8.4.5 Implications for code reviewers and teaching staff in STEM classrooms. Previous work [47, 71] shows that linguistic features can be used as a mechanism to identify misinformation and AI-generated content. Our results show that ChatGPT answers have a distinct linguistic structure and communication style when answering programming questions. We believe identifying these distinct linguistic features is essential in situations where users need to differentiate between human and machine-generated answers. For example, in CS classrooms, there is an increasing concern that students are using ChatGPT to solve homework assignments, which impedes learning. Traditional plagiarism tools used in academia often cannot detect ChatGPT-generated answers. By having general knowledge of common language styles of ChatGPT answers (e.g., verbosity, formal language, title-body-summary structure, etc.), teaching staff can be more aware of what to look for. Moreover, plagiarism tools, both AI and non-AI, should incorporate unique linguistic characteristics as factors to classify plagiarised documents. Furthermore, as discussed in the previous subsections, code reviewers must adopt new techniques and tools to take extra precautions so that incorrect and insecure code does not seep into software products. Incorporating the linguistics style of ChatGPT responses while creating these tools and training code reviewers to make them aware of unique linguistic markers can help the software industry install additional safeguards against incorrect code.

8.4.6 New pedagogical methods are necessary. Apart from the software industry, faculty and teaching staff in the educational institute should also make the students aware of the potential risks that come with seemingly correct ChatGPT answers. Moreover, new pedagogical methods should be adopted to incorporate ChatGPT into the curriculum to utilize the incorrectness as a learning tool. For example, in a beginner Python class, students can be given multiple wrong programs generated by ChatGPT and asked to identify the errors in each program. This type of activity can render learning and create awareness at the same time.

8.4.7 Separation of accountability. New policies should be made to separate and distinguish the role of humans and *LLM*s when *LLM*s generate misinformation. As discussed previously, depending on solutions such as *Prompt Engineering* shifts the accountability of misinformation to humans. Furthermore, when AI is involved in the step of decision-making and manufacturing software products, ethical questions such as who will be held accountable for AI's errors come to light [26]. Previous work [11, 78] on responsible AI also highlights the need for accountability of AI systems that are grounded in human rights and ethics. Hence, strict policies should be created to maintain the separation of accountability to protect humans from false accusations and preserve the interests of impacted stakeholders by ensuring responsible use. We believe this work will encourage further research for the informed design of responsible conversational chatbots and for careful policy-making to preserve the rights of stakeholders.

9 LIMITATIONS

One limitation of this work is the subjective nature of the manual analysis. We tried to address this limitation by recruiting multiple labelers, constantly measuring the agreement level among labelers, and adopting an iterative analysis procedure with extensive discussions. Moreover, our user study has limitations concerning other factors such as sample size and participants' own biases. To reduce participants' biases against human or ChatGPT answers, we anonymized the source of the answers during the study and standardized the visual style and format of the answers, e.g., using the same font size, type, code style, etc.

Additionally, this work has used the free version of ChatGPT (GPT-3.5) for acquiring the ChatGPT responses for the manual analysis. Hence, one might argue that the results are not generalizable for ChatGPT since the new GPT-4 (released on March 2023) can perform differently. To understand how differently GPT-4 performs compared to GPT-3.5, we conducted a small analysis on 21 randomly selected SO questions where GPT-3.5 gave incorrect answers. [5] Our analysis shows that, among these 21 questions, GPT-4 could answer only 6 questions correctly, and 15 questions were still answered incorrectly. Moreover, the types of errors introduced by GPT-4 follow the same pattern as GPT-3.5. This tells us that, although GPT-4 performs slightly better than GPT -3.5 (e.g., rectified error in 6 answers), the rate of inaccuracy is still high with similar types of errors. Moreover, this new ChatGPT (also known as ChatGPT plus) is a paid version ($20 per month). Since the target population of this research is not only industry developers but also programmers of all levels, including students and freelancers around the world, the free version of ChatGPT has significantly more users than the paid version which only the privileged population can access. Moreover, $20 per month has a considerably high monetary value for many countries. Hence, for this study, we used the free version (GPT-3.5) so that the results benefit the majority of our target populations. We acknowledge that other *LLM*s can perform differently and we encourage future research to empirically study programming answers generated by other *LLM*s.

Another limitation lies in the prompting strategy adopted by our study. In this work, we did not account for the interactive nature of ChatGPT. In practice, if the initial ChatGPT answer is not satisfactory, programmers can refine their initial prompt or ask follow-up questions to get new answers. We did not consider this, since it required designing specific follow-up questions or prompt refinements for each question under analysis. Furthermore, such interaction is not guaranteed to generate better and more correct answers. Thus, it may require multiple rounds of interaction to improve the answer. This would significantly increase the analysis effort and limit our capability to analyze many different kinds of questions in this study. As a result, we restrict the project scope to only analyze the initial answers generated by ChatGPT. To address this limitation, future work could conduct a small-scale but more focused analysis to investigate how interactivity impacts the correctness of ChatGPT answers.

In this work, we reused the original SO question as the prompt, since the original SO question represents how a programmer may ask the question in a natural conversation. This can be improved with more advanced prompting templates and tricks. However, the design of the prompt is highly dependent on the problem itself and also varies from person to person. Reaching an agreement level for prompt engineering is even more challenging without any established guidelines or studies to follow. To address this limitation, future work could conduct a systematic investigation into how different prompting strategies and tips influence the correctness of ChatGPT answers to different kinds of programming questions.

ChatGPT is inherently stochastic. The same prompt may generate different answers with a moderate temperature setting of 0.8. To

[5] The annotations for GPT-4 are added in the repository https://github.com/SamiaKabir/ChatGPT-Answers-to-SO-questions/blob/main/ChatGPT%20answers%20to%20SO%20questions/Labeler1/Annotations_GPT-4.docx

account for this, one needs to run ChatGPT multiple times with the same prompt for each programming question, manually analyze all answers, and measure the average correctness. If we run ChatGPT 5 times for each question, our analysis effort would be increased by five times and we would not be able to do the study at a satisfiable scale and comprehensiveness level. Thus, we chose to only consider the initial answer generated by ChatGPT.

Finally, we acknowledge that despite our efforts to mitigate the potential issues, some level of human bias and the generalizability limitation still persist. Nonetheless, we hope this study will foster new research in the direction of identification, understanding, rectification, and risk mitigation of errors in *LLM*s for better human-AI collaboration.

10 CONCLUSION

In this paper, we empirically studied the characteristics of ChatGPT answers to SO questions through a combination of manual analysis, linguistic analysis, and user study. Our manual analysis shows that ChatGPT produces incorrect answers more than 50% of the time. Moreover, ChatGPT suffers from other quality issues such as verbosity, inconsistency, etc. Results of the in-depth manual analysis also point towards a large number of conceptual and logical errors in ChatGPT answers. Additionally, our linguistic analysis results show that ChatGPT answers are very formal, and rarely portray negative sentiments. Although our user study shows higher user preference and quality rating for human answers, users make occasional mistakes by preferring incorrect ChatGPT answers based on ChatGPT's articulated language styles, as well as seemingly correct logic that is presented with positive assertions. Since ChatGPT produces a large number of incorrect answers, our results emphasize the necessity of caution and awareness regarding the usage of ChatGPT answers in programming tasks. This work also seeks to encourage further research in identifying and mitigating different types of conceptual and factual errors. Finally, we expect this work will foster more research on transparency and communication of incorrectness in machine-generated answers, especially in the context of programming.

REFERENCES

[1] Rabe Abdalkareem, Emad Shihab, and Juergen Rilling. 2017. What do developers use the crowd for? a study using stack overflow. *IEEE Software* 34, 2 (2017), 53–60.

[2] Mayank Agarwal, Kartik Talamadupula, Stephanie Houde, Fernando Martinez, Michael Muller, John Richards, Steven Ross, and Justin D Weisz. 2020. Quality estimation & interpretability for code translation. *arXiv preprint arXiv:2012.07581* (2020).

[3] Miltiadis Allamanis and Charles Sutton. 2013. Why, when, and what: analyzing stack overflow questions by topic, type, and code. In *2013 10th Working conference on mining software repositories (MSR)*. IEEE, 53–56.

[4] Muhammad Asaduzzaman, Ahmed Shah Mashiyat, Chanchal K Roy, and Kevin A Schneider. 2013. Answering questions about unanswered questions of stack overflow. In *2013 10th Working Conference on Mining Software Repositories (MSR)*. IEEE, 97–100.

[5] Amos Azaria and Tom Mitchell. 2023. The internal state of an llm knows when its lying. *arXiv preprint arXiv:2304.13734* (2023).

[6] Alberto Bacchelli, Luca Ponzanelli, and Michele Lanza. 2012. Harnessing stack overflow for the ide. In *2012 Third International Workshop on Recommendation Systems for Software Engineering (RSSE)*. IEEE, 26–30.

[7] Shraddha Barke, Michael B James, and Nadia Polikarpova. 2023. Grounded copilot: How programmers interact with code-generating models. *Proceedings of the ACM on Programming Languages* 7, OOPSLA1 (2023), 85–111.

[8] Blerina Bazelli, Abram Hindle, and Eleni Stroulia. 2013. On the personality traits of stackoverflow users. In *2013 IEEE international conference on software maintenance*. IEEE, 460–463.

[9] Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big?. In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*. 610–623.

[10] Emily M Bender and Alexander Koller. 2020. Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th annual meeting of the association for computational linguistics*. 5185–5198.

[11] Reuben Binns, Max Van Kleek, Michael Veale, Ulrik Lyngs, Jun Zhao, and Nigel Shadbolt. 2018. 'It's Reducing a Human Being to a Percentage' Perceptions of Justice in Algorithmic Decisions. In *Proceedings of the 2018 Chi conference on human factors in computing systems*. 1–14.

[12] Christian Bird, Denae Ford, Thomas Zimmermann, Nicole Forsgren, Eirini Kalliamvakou, Travis Lowdermilk, and Idan Gazit. 2022. Taking Flight with Copilot: Early insights and opportunities of AI-powered pair-programming tools. *Queue* 20, 6 (2022), 35–57.

[13] Kristof Boghe. 2020. We Need to Talk About Sentiment Analysis. https://medium.com/p/9d1f20f2ebfb.

[14] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258* (2021).

[15] Ali Borji. 2023. A categorical archive of chatgpt failures. *arXiv preprint arXiv:2302.03494* (2023).

[16] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. *Qualitative research in psychology* 3, 2 (2006), 77–101.

[17] Ben Buchanan, Andrew Lohn, and Micah Musser. 2021. *Truth, lies, and automation: How language models could change disinformation.* Center for Security and Emerging Technology.

[18] Fabio Calefato, Filippo Lanubile, Federico Maiorano, and Nicole Novielli. 2018. Sentiment polarity detection for software development. In *Proceedings of the 40th International Conference on Software Engineering*. 128–128.

[19] Fabio Calefato, Filippo Lanubile, Maria Concetta Marasciulo, and Nicole Novielli. 2015. Mining successful answers in stack overflow. In *2015 IEEE/ACM 12th Working Conference on Mining Software Repositories*. IEEE, 430–433.

[20] Fabio Calefato, Filippo Lanubile, and Nicole Novielli. 2018. How to ask for technical help? Evidence-based guidelines for writing questions on Stack Overflow. *Information and software technology* 94 (2018), 186–207.

[21] Boxi Cao, Hongyu Lin, Xianpei Han, Le Sun, Lingyong Yan, Meng Liao, Tong Xue, and Jin Xu. 2021. Knowledgeable or educated guess? revisiting language models as knowledge bases. *arXiv preprint arXiv:2106.09231* (2021).

[22] Alex Castillo. Dec, 2022. [Twitter Post] ChatGPT will replace StackOverflow. https://twitter.com/castillo__io/status/1599255771736604673?s=20.

[23] Xiaojun Chen, Shengbin Jia, and Yang Xiang. 2020. A review: Knowledge reasoning over knowledge graph. *Expert Systems with Applications* 141 (2020), 112948.

[24] Arghavan Moradi Dakhel, Vahid Majdinasab, Amin Nikanjam, Foutse Khomh, Michel C Desmarais, and Zhen Ming Jack Jiang. 2023. Github copilot ai pair programmer: Asset or liability? *Journal of Systems and Software* 203 (2023), 111734.

[25] Lucas BL De Souza, Eduardo C Campos, and Marcelo de A Maia. 2014. Ranking crowd knowledge to assist software development. In *Proceedings of the 22nd International Conference on Program Comprehension*. 72–82.

[26] Advait Deshpande and Helen Sharp. 2022. Responsible AI Systems: Who are the Stakeholders?. In *Proceedings of the 2022 AAAI/ACM Conference on AI, Ethics, and Society*. 227–236.

[27] Renee DiResta. 2020. AI-Generated Text Is the Scariest Deepfake of All. https://www.wired.com/story/ai-generated-text-is-the-scariest-deepfake-of-all/.

[28] Qingxiu Dong, Damai Dai, Yifan Song, Jingjing Xu, Zhifang Sui, and Lei Li. 2022. Calibrating factual knowledge in pretrained language models. *arXiv preprint arXiv:2210.03329* (2022).

[29] Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, Eduard Hovy, Hinrich Schütze, and Yoav Goldberg. 2021. Measuring and improving consistency in pretrained language models. *Transactions of the Association for Computational Linguistics* 9 (2021), 1012–1031.

[30] Stack Exchange. 2023. Stack Exchange Data Dump. https://archive.org/details/stackexchange.

[31] Hugging Face. 2022. Hugging Face – The AI community building the future. https://huggingface.co/.

[32] Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin* 76, 5 (1971), 378.

[33] Dilrukshi Gamage, Piyush Ghasiya, Vamshi Bonagiri, Mark E Whiting, and Kazutoshi Sasahara. 2022. Are deepfakes concerning? analyzing conversations of deepfakes on reddit and exploring societal implications. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*. 1–19.

[34] Catherine A Gao, Frederick M Howard, Nikolay S Markov, Emma C Dyer, Siddhi Ramesh, Yuan Luo, and Alexander T Pearson. 2022. Comparing scientific abstracts generated by ChatGPT to original abstracts using an artificial intelligence output detector, plagiarism detector, and blinded human reviewers. *BioRxiv* (2022),

2022–12.

[35] Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A Smith. 2020. Realtoxicityprompts: Evaluating neural toxic degeneration in language models. *arXiv preprint arXiv:2009.11462* (2020).

[36] GitHub. 2023. GitHub Copilot X: The AI-powered developer experience. https://github.blog/2023-03-22-github-copilot-x-the-ai-powered-developer-experience/.

[37] GitHub. 2023. Introducing GitHub Copilot X. https://github.com/features/preview/copilot-x.

[38] Inc. GitHub. 2023. GitHub Copilot · Your AI pair programmer. https://github.com/features/copilot.

[39] Ben Goodrich, Vinay Rao, Peter J Liu, and Mohammad Saleh. 2019. Assessing the factual accuracy of generated text. In *proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 166–175.

[40] Greg Guest, Kathleen M MacQueen, and Emily E Namey. 2011. *Applied thematic analysis*. sage publications.

[41] Biyang Guo, Xin Zhang, Ziyuan Wang, Minqi Jiang, Jinran Nie, Yuxuan Ding, Jianwei Yue, and Yupeng Wu. 2023. How close is chatgpt to human experts? comparison corpus, evaluation, and detection. *arXiv preprint arXiv:2301.07597* (2023).

[42] Beverley Hancock, Elizabeth Ockleford, and Kate Windridge. 2001. *An introduction to qualitative research*. Trent focus group London.

[43] Yichong Huang, Xiachong Feng, Xiaocheng Feng, and Bing Qin. 2021. The factual inconsistency problem in abstractive text summarization: A survey. *arXiv preprint arXiv:2104.14839* (2021).

[44] Saki Imai. 2022. 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*. 319–321.

[45] Md Saiful Islam, Tonmoy Sarkar, Sazzad Hossain Khan, Abu-Hena Mostofa Kamal, SM Murshid Hasan, Alamgir Kabir, Dalia Yeasmin, Mohammad Ariful Islam, Kamal Ibne Amin Chowdhury, Kazi Selim Anwar, et al. 2020. COVID-19–related infodemic and its impact on public health: A global social media analysis. *The American journal of tropical medicine and hygiene* 103, 4 (2020), 1621.

[46] Srinivasan Iyer, Ioannis Konstas, Alvin Cheung, and Luke Zettlemoyer. 2016. Summarizing source code using a neural attention model. In *54th Annual Meeting of the Association for Computational Linguistics 2016*. Association for Computational Linguistics, 2073–2083.

[47] Shan Jiang and Christo Wilson. 2018. Linguistic signals under misinformation and fact-checking: Evidence from user comments on social media. *Proceedings of the ACM on Human-Computer Interaction* 2, CSCW (2018), 1–23.

[48] R Burke Johnson and Anthony J Onwuegbuzie. 2004. Mixed methods research: A research paradigm whose time has come. *Educational researcher* 33, 7 (2004), 14–26.

[49] Will Knight. 2021. AI Can Write Disinformation Now—and Dupe Human Readers. https://www.wired.com/story/ai-write-disinformation-dupe-human-readers/.

[50] Jan Kocoń, Igor Cichecki, Oliwier Kaszyca, Mateusz Kochanek, Dominika Szydło, Joanna Baran, Julita Bielaniewicz, Marcin Gruza, Arkadiusz Janz, Kamil Kanclerz, et al. 2023. ChatGPT: Jack of all trades, master of none. *Information Fusion* (2023), 101861.

[51] Bonan Kou, Muhao Chen, and Tianyi Zhang. 2023. Automated Summarization of Stack Overflow Posts. In *45th IEEE/ACM International Conference on Software Engineering, ICSE 2023, Melbourne, Australia, May 14-20, 2023*. IEEE, 1853–1865.

[52] Bonan Kou, Yifeng Di, Muhao Chen, and Tianyi Zhang. 2022. SOSum: a dataset of stack overflow post summaries. In *Proceedings of the 19th International Conference on Mining Software Repositories*. 247–251.

[53] Adam DI Kramer. 2012. The spread of emotion via Facebook. In *Proceedings of the SIGCHI conference on human factors in computing systems*. 767–770.

[54] Sarah Kreps, R Miles McCain, and Miles Brundage. 2022. All the news that's fit to fabricate: AI-generated text as a tool of media misinformation. *Journal of experimental political science* 9, 1 (2022), 104–117.

[55] Lena Mamykina, Bella Manoim, Manas Mittal, George Hripcsak, and Björn Hartmann. 2011. Design lessons from the fastest q&a site in the west. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. 2857–2866.

[56] Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. *arXiv preprint arXiv:2005.00661* (2020).

[57] Courtney Miller, Sophie Cohen, Daniel Klug, Bogdan Vasilescu, and Christian Kastner. 2022. "Did you miss my comment or what?" understanding toxicity in open source discussions. In *Proceedings of the 44th International Conference on Software Engineering*. 710–722.

[58] Sandra Mitrović, Davide Andreoletti, and Omran Ayoub. 2023. Chatgpt or human? detect and explain. explaining decisions of machine learning model for detecting short chatgpt-generated text. *arXiv preprint arXiv:2301.13852* (2023).

[59] Seyed Mehdi Nasehi, Jonathan Sillito, Frank Maurer, and Chris Burns. 2012. What makes a good code example?: A study of programming Q&A in StackOverflow. In *2012 28th IEEE International Conference on Software Maintenance (ICSM)*. IEEE, 25–34.

[60] Geoff Norman. 2010. Likert scales, levels of measurement and the "laws" of statistics. *Advances in health sciences education* 15 (2010), 625–632.

[61] OpenAI. 2023. ChatGPT: Optimizing Language Models for Dialogue. https://openai.com/blog/chatgpt.

[62] Stack Overflow. 2023. File Upload Area. https://stackoverflow.com/questions/76003368/file-upload-area.

[63] Stack Overflow. 2023. Insights into Stack Overflow's traffic. https://stackoverflow.blog/2023/08/08/insights-into-stack-overflows-traffic/.

[64] Stack Overflow. 2023. Temporary policy: Generative AI (e.g., ChatGPT) is banned. https://meta.stackoverflow.com/questions/421831/temporary-policy-generative-ai-e-g-chatgpt-is-banned.

[65] Baolin Peng, Michel Galley, Pengcheng He, Hao Cheng, Yujia Xie, Yu Hu, Qiuyuan Huang, Lars Liden, Zhou Yu, Weizhu Chen, et al. 2023. Check your facts and try again: Improving large language models with external knowledge and automated feedback. *arXiv preprint arXiv:2302.12813* (2023).

[66] James W Pennebaker, Ryan L Boyd, Kayla Jordan, and Kate Blackburn. 2015. *The development and psychometric properties of LIWC2015*. Technical Report.

[67] Huilian Sophie Qiu, Yucen Lily Li, Susmita Padala, Anita Sarma, and Bogdan Vasilescu. 2019. The signals that potential contributors look for when choosing open-source projects. *Proceedings of the ACM on Human-Computer Interaction* 3, CSCW (2019), 1–29.

[68] Quora. 2023. Will ChatGPT replace Stack Overflow? https://www.quora.com/Will-ChatGPT-replace-Stack-Overflow.

[69] Md Masudur Rahman, Jed Barson, Sydney Paul, Joshua Kayani, Federico Andrés Lois, Sebastián Fernandez Quezada, Christopher Parnin, Kathryn T Stolee, and Baishakhi Ray. 2018. Evaluating how developers use general-purpose web-search for code retrieval. In *Proceedings of the 15th International Conference on Mining Software Repositories*. 465–475.

[70] Nikitha Rao, Chetan Bansal, Thomas Zimmermann, Ahmed Hassan Awadallah, and Nachiappan Nagappan. 2020. Analyzing web search behavior for software engineering tasks. In *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, 768–777.

[71] Hannah Rashkin, Eunsol Choi, Jin Yea Jang, Svitlana Volkova, and Yejin Choi. 2017. Truth of varying shades: Analyzing language in fake news and political fact-checking. In *Proceedings of the 2017 conference on empirical methods in natural language processing*. 2931–2937.

[72] Filipe Rodrigues, Francisco Pereira, and Bernardete Ribeiro. 2014. Sequence labeling with multiple annotators. *Machine learning* 95 (2014), 165–181.

[73] Steven I Ross, Fernando Martinez, Stephanie Houde, Michael Muller, and Justin D Weisz. 2023. The programmer's assistant: Conversational interaction with a large language model for software development. In *Proceedings of the 28th International Conference on Intelligent User Interfaces*. 491–514.

[74] Ben Shneiderman. 1980. *Software psychology: Human factors in computer and information systems (Winthrop computer systems series)*. Winthrop Publishers.

[75] James Skripchuk, Neil Bennett, Jeffrey Zhang, Eric Li, and Thomas Price. 2023. Analysis of Novices' Web-Based Help-Seeking Behavior While Programming. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*. 945–951.

[76] Margaret-Anne Storey, Christoph Treude, Arie Van Deursen, and Li-Te Cheng. 2010. The impact of social media on software engineering practices and tools. In *Proceedings of the FSE/SDP workshop on Future of software engineering research*. 359–364.

[77] Hendrik Strobelt, Albert Webson, Victor Sanh, Benjamin Hoover, Johanna Beyer, Hanspeter Pfister, and Alexander M Rush. 2022. Interactive and visual prompt engineering for ad-hoc task adaptation with large language models. *IEEE transactions on visualization and computer graphics* 29, 1 (2022), 1146–1156.

[78] Mohammad Tahaei, Marios Constantinides, Daniele Quercia, Sean Kennedy, Michael Muller, Simone Stumpf, Q Vera Liao, Ricardo Baeza-Yates, Lora Aroyo, Jess Holbrook, et al. 2023. Human-Centered Responsible Artificial Intelligence: Current & Future Trends. In *Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems*. 1–4.

[79] Rob Toews. 2020. Deepfakes Are Going To Wreak Havoc On Society. We Are Not Prepared. https://www.forbes.com/sites/robtoews/2020/05/25/deepfakes-are-going-to-wreak-havoc-on-society-we-are-not-prepared/?sh=392a1cb07494.

[80] Christopher Tozzi. 2023. GitHub Copilot vs. ChatGPT: Which Tool Is Better for Software Development? https://www.itprotoday.com/development-techniques-and-management/github-copilot-vs-chatgpt-which-tool-better-software.

[81] Christoph Treude, Ohad Barzilay, and Margaret-Anne Storey. 2011. How do programmers ask and answer questions on the web?(nier track). In *Proceedings of the 33rd international conference on software engineering*. 804–807.

[82] Priyan Vaithilingam, Tianyi Zhang, and Elena L Glassman. 2022. Expectation vs. experience: Evaluating the usability of code generation tools powered by large language models. In *Chi conference on human factors in computing systems extended abstracts*. 1–7.

[83] Helena Vasconcelos, Gagan Bansal, Adam Fourney, Q Vera Liao, and Jennifer Wortman Vaughan. 2023. Generation probabilities are not enough: Exploring the effectiveness of uncertainty highlighting in AI-powered code completions. *arXiv preprint arXiv:2302.07248* (2023).

[84] Bogdan Vasilescu, Vladimir Filkov, and Alexander Serebrenik. 2013. Stackoverflow and github: Associations between software development and crowdsourced knowledge. In *2013 International Conference on Social Computing*. IEEE, 188–195.

[85] Mika Westerlund. 2019. The emergence of deepfake technology: A review. *Technology innovation management review* 9, 11 (2019).

[86] Xin Xia, Lingfeng Bao, David Lo, Pavneet Singh Kochhar, Ahmed E Hassan, and Zhenchang Xing. 2017. What do developers search for on the web? *Empirical Software Engineering* 22 (2017), 3149–3185.

[87] Wenhan Xiong, Thien Hoang, and William Yang Wang. 2017. Deeppath: A reinforcement learning method for knowledge graph reasoning. *arXiv preprint arXiv:1707.06690* (2017).

[88] Jinghang Xu, Wanli Zuo, Shining Liang, and Xianglin Zuo. 2020. A review of dataset and labeling methods for causality extraction. In *Proceedings of the 28th International Conference on Computational Linguistics*. 1519–1531.

[89] Jiawei Zhou, Yixuan Zhang, Qianni Luo, Andrea G Parker, and Munmun De Choudhury. 2023. Synthetic lies: Understanding ai-generated misinformation and evaluating algorithmic and human solutions. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. 1–20.

[90] Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2022. Large language models are human-level prompt engineers. *arXiv preprint arXiv:2211.01910* (2022).

A EXAMPLE OF INCORRECT CHATGPT RESPONSE

Stack Overflow Question

How to end program in a do-while (C++)

Here is my code. I am trying to get the entire program to end if it goes into the second if statement inside the do-while loop. But every time I run it, it crashes. I am not sure what I am doing wrong.

```
#include <iostream>
using namespace std;

int main() {
    int myData[10];
    for(int i=0;i<10;i++){
        myData[i] = 1;
        cout<<myData[i];
    }

    do{
        int i;
        cout<<endl<<"Input index: ";
        cin>> i;

        int v;
        cout<<endl<<"Input value: ";
        cin>>v;

        if(i>=0||i<10){
            myData[i]=v;
            for(int i=0;i<10;i++){
                cout<<myData[i]<<" ";
            }
        }

        if (i<0||i>=10){
            cout<<"Index out of range. Exit.";
            return 0;
        }

    }while(1);
}
```

ChatGPT Answer

The problem with your code is that the second if statement inside the do-while loop will always be executed because you are checking if i is less than 0 or greater than or equal to 10, but not both. In the case where i is equal to 10, both conditions will be true and the program will print "Index out of range. Exit." and then end. To fix this, you should change the if statement to use the logical and operator (&&) instead of the logical or operator (||):

C Code

```
if (i <0 && i >=10)
{
    court << "Index out of range. Exit."; << return 0;
}
```

With this change, the second if statement will only be executed if i is less than 0 and greater than or equal to 10, which is not possible.

Stack Overflow Answer

```
if (i >= 0 || i < 10) {
```

Think about which numbers are either greater than zero or less than ten. I'm sure you realise that is true of all numbers. What you meant to write is

```
if (i >= 0 && i < 10) {
```

This explains your crash, you are accessing the myData array with an index that is outside the array bounds.
It's very common for beginners to get && and || confused, especially where there is negation involved as well.

B EXAMPLE OF VERBOSE CHATGPT RESPONSE

Stack Overflow Question

JavaScript check if variable exists (is defined/initialized)

Which method of checking if a variable has been initialized is better/correct? (Assuming the variable could hold anything (string, int, object, function, etc.))

```
if (elem) {
```

or

```
if (typeof elem !== 'undefined') {
```

or

```
if (elem != null) {
```

ChatGPT Answer

The method of checking if a variable has been initialized depends on the specific use case and the expected value of the variable. Here's an overview of each method:

if (elem) checks if elem is "truthy", meaning that it has a value that evaluates to true in a boolean context. This includes all non-falsy values, such as numbers (except 0), strings, and objects. However, this method will also return true for empty strings, zero, NaN, null, and undefined, which may not be desirable in all cases.

if (typeof elem !== 'undefined') checks if elem has been defined. This method is more precise than the first one, as it only returns true if elem has been declared in the current scope, even if it has been assigned a falsy value. However, it can be more verbose and may not be necessary in all cases.

if (elem != null) checks if elem is not null or undefined. This method is useful if you want to exclude null and undefined values, but may return true for other falsy values such as empty strings and zero.

In general, it's a good practice to be explicit about the expected value of a variable and use the appropriate method to check for it. For example, if you're checking for the existence of an object property, you may want to use if ('propertyName' in object) to avoid potential errors with falsy values.

Stack Overflow Answer

You want the typeof operator. Specifically:

```
if (typeof variable !== 'undefined') {
    // the variable is defined
}
```