

<Conference abbreviation>

<Conference Series name>

<Volume number and Year> <DOI Number>

The Perceived Efficacy of AI-Assisted Debugging versus Manual Debugging Among Novice Python Developers

¹Ayantika Das and ²Anupam Mondal

¹Brainware University, Barasat, West Bengal, dasayantika97@gmail.com

²Institute of Engineering and Management, Salt Lake, West Bengal, anupam.mondal@iem.edu.in

Abstract.

This study examines how Artificial Intelligence (AI) coding assistants affect the debugging self-confidence and mental load of novice Python programmers. A within-subjects survey was conducted with 75 valid participants, with the aim of comparing the manual debugging of syntax and logical errors with AI-assisted debugging. The Likert scale was used to assess confidence and a modified NASA-TLX scale was used to assess mental effort. The results suggest that the participants experienced less ease with the usage of AI and greater confidence in that usage, especially with regards to simple syntax corrections. The results indicate that AI can assist in debugging, but human judgment is still a vital part of reasoning tasks.

Keywords. AI Coding Assistants, Debugging Confidence, Cognitive Load, Python Programming.

1. INTRODUCTION

The advent of Large Language Models (LLMs) and AI Coding Assistants has transformed the way developers write, review, and debug code. LLMs and AI Coding Assistants have changed the coding landscape, influencing the way developers write, review and debug programs. With the advent of tools like GitHub Copilot and ChatGPT, coding, refactoring, and fixing bugs are being streamlined and made more commonplace in developers' workflows. Programming skill is no longer just about being able to write code from memory but also to assess and control code produced by AI. This convenience and speed creates a big question: is it possible for AI to contribute to the learning process and/or make cognitive learning, in the construction of good debugging skills, easier? Debugging is an essential part of software engineering, which aids in discerning the flow, logic, and error structure of programs. It is significant for the fact that the errors may be syntactic (rule-based) or logical (more abstract thinking). However, most of the published literature does not clearly differentiate between these two types of errors and regards debugging as a unitary activity. Moreover, there are less studies which look at the effect of AI assistance on confidence and perceived mental effort in the same debugging task. In this paper, the participants are referred to as "novice Python developers" who have a limited amount of experience in debugging and have only basic knowledge of Python programming. The study explores the differences in confidence and mental effort between programmers for syntactic and logical

errors when relying on AI assistance. RQ1: Does AI assistance have a correlation with debugging confidence? RQ2: Does AI help decrease the perceived effort? RQ3: Is there a difference between the effects of syntactic errors and logical errors?

2. RELATED WORK

Initial research on how AI coding assistants affect developers focused on productivity; more recent studies have investigated how these tools influence developers' trust, behavior, and learning. As seen in [1], on GitHub Copilot and similar research projects on neural code completion, AI support can help increase the speed of a given task and its short-term efficiency, but can also cause overreliance and a decrease in verification. In computing education, the use of AI in coding environments has been discussed [2], suggesting that such tools might enable quicker task completion in computing education and minimize human engagement with syntax and reasoning [3]. Other studies have examined the level of trust in AI-generated code and the need for human oversight in case of adopting the AI-generated suggestions [4], especially in cases where the task is conceptual rather than pattern matching [5]. Further, cognitive-load research shows that debugging is not a simple process [6], with varying cognitive states requiring different kinds of debugging support, which means that the benefits of AI debugging support will differ depending on the type of debugging [7]. But most previous studies still neglect to consider debugging as a single activity or just consider productivity outcomes [8,9]. In the subsequent studies neither the syntactic errors nor the logical errors were separated, nor were both aspects of the mental effort and the level of confidence measured in one study. To address this need, we fill the gap between manual and AI debugging by comparing the two in the context of two distinct categories of errors, and examine novice Python developers' self-reported confidence and mental effort.

3. METHODOLOGY

This study used a within-subject quantitative design using a survey method to explore how using AI coding assistants affects confidence in debugging and mental load. The survey included numerical and qualitative responses, scenario-based debugging tasks, ratings questions and open-ended reflection questions.

3.1 Study Design: All participants were given the same debugging scenarios to test, each with a different set of conditions: manuals and AI assistance. In both conditions the same participants were used in order to be able to compare the amount of effort and confidence in both the syntactic and the logical error cases. The order of the conditions was held constant (first manual debugging, then AI-assisted debugging), in order to create a cognitive baseline before the introduction of the AIs.

3.2 Participant Selection: Participants were identified through the academics and coding communities online. The inclusion criteria were that the developers had to have some basic knowledge of Python and also have little experience in debugging ("novice" developers). Following an attention check which discarded problematic responses, 75 valid participants (N=75) remained. Demographically, 86.6% were between the ages of 16-25, and had a variety of educational levels (37.3% had received a Bachelor's degree or higher, 34.6% had high school or less).

3.3 Instrument Development: There were four parts of the survey: demographics, debugging scenarios, mental-effort rating, and reflection. The mental effort section was based on the NASA-TLX and a simplified version with 7-point scale was used to represent the overall cognitive load. The items measuring the confidence of self-reported debugging were new items that were created specifically for this study and measured the confidence on a 5-point Likert scale.

3.4 Analytical Framework: The data collected in the survey is ordinal Likert data, therefore the data analysis used is non-parametric statistical methods which are robust to the ordinality of the data being collected, namely Wilcoxon signed-rank tests, which compare the data manual and the data using AI in the survey. Cohen's d was also calculated to give effect sizes of these changes, too. Qualitative responses were analysed qualitatively and categorised into themes and patterns of AI support and reliance using thematic analysis.

4. RESULTS

The survey yielded 75 valid responses, all of whom passed the attention check criteria, resulting in a validated dataset of $N = 75$.

4.1 Quantitative Analysis: The quantitative analysis revealed that there is a significant difference in the confidence and mental efforts of the developers during the debugging process from using AI. The mean score was higher in the AI-assisted condition (3.95) than in the manual condition (3.11) (Wilcoxon $p < .001$, Cohen's $d = 0.68$) as illustrated in Figure 1. There was a significant reduction in cognitive effort for mental effort as the mean scores in the AI-assisted condition was significantly lower than the manual condition (Wilcoxon $p < .001$, Cohen's $d = 0.98$) as seen in Figure 2.

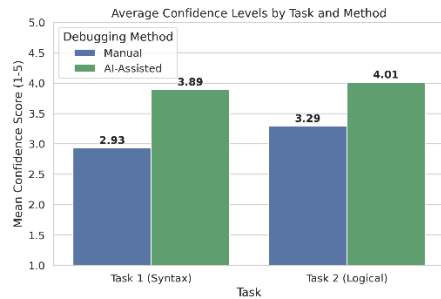


Figure 1: Mean confidence scores: Manual vs. AI-assisted debugging

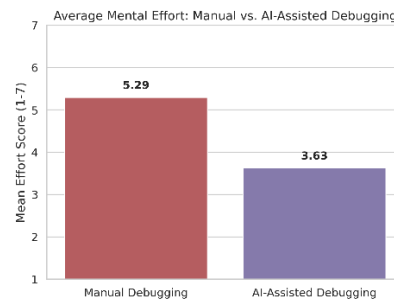


Figure 2: Perceived mental effort scores: Manual vs. AI-assisted debugging

4.2 Qualitative Findings: The open-ended responses were categorized into three recurring themes, summarized in Table 1 below.

Table 1: Themes and Their Representative Quotes

Theme	Description	Representative Quote
Efficiency	AI reduces time spent on syntax fixes.	"It fixes my syntax errors instantly."
Cognitive Offloading	Reduced effort leads to less deep analysis.	"I trust the AI even if I don't get the logic."
Interpretability	Difficulty understanding AI-generated code.	"AI provides solutions I don't fully understand."

5. DISCUSSION

The findings show that there's a stark difference in the attitudes of novice Python developers towards using AI tools to aid in debugging. Although AI can alleviate the cognitive load and speed up the correction of common syntax mistakes through cognitive offloading, there is an important consideration here: the potential for AI to increase efficiency also comes with a cost. Qualitative feedback shows that novices are inclined to blindly follow the solutions that AI provide without understanding the reasoning, which is especially the case in reasoning intensive tasks. This dependency creates an "AI Dependency Gap" that means that there is the potential to grow their lack of core debugging intuition over time. Thus, AI verification should be used in addition to and in support of human-in-the-loop, not instead of. This study was conducted with a relatively large sample size (N=75) that can be used to support the trends revealed in the self-reported behaviors, but a limitation of this study is that the study was largely conducted based on perceptions. Further studies are needed to include objective performance indicators (time to resolution, compilation success, eye-tracking, etc.) that can be used to assess the correlation between this artificially induced confidence and actual debugging skills.

6. CONCLUSION

From the results of this research, it can be concluded that the AI coding assistant system can relieve the developer's mental burden in the debugging process and enhance his/her confidence in debugging process, especially for syntactic errors. But all of the same tools might link to overtrust in logical thinking activities that form what I call the "AI Dependency Gap". The findings suggest that AI tools are not meant to replace human judgment, but rather to be used as a support to it in the context of debugging. Developers will have to follow-up on AI-generated solutions and have to continue practicing logical problem-solving for continuous improvement of their skills.

7. REFERENCES

- [1] Peng, S., Kalliamvakou, E., Cihon, P., & Demirer, M. (2023). "The Impact of AI on Developer Productivity: Evidence from GitHub Copilot." *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*.
- [2] Daniotti, S., Wachs, J., Feng, X., & Neffke, F. (2026). Who is using AI to code? Global diffusion and impact of generative AI. *Science*, eadz9311.
- [3] Crane, L. D., & Soto, P. E. (2026). AI and Coder Employment: Compiling the Evidence.
- [4] Nazir, M., & Arif, Y. (2026). Is AI Code Generation Undermining Developers' Problem-Solving Skills?. *ICCK Journal of Software Engineering*, 2(1), 1-10.
- [5] Määttä, S. (2024). Generative artificial intelligence in programming: A scoping review of cognitive aspects.
- [6] Alenezi, M., & Akour, M. (2025). Ai-driven innovations in software engineering: a review of current practices and future directions. *Applied Sciences*, 15(3), 1344.
- [7] Anushree, B., Devika, M. S., & Achala, R. V. (2025). SyntaxMentor: An AI-Augmented Debugging Assistant for Novice Programmers. *INTERNATIONAL JOURNAL OF ENGINEERING DEVELOPMENT AND RESEARCH*, 13(4), 353-358.
- [8] Paradis, E., Grey, K., Madison, Q., Nam, D., Macvean, A., Meimand, V., ... & Chandra, S. (2024). *How Much Does AI Impact Development Speed? An Enterprise-based Randomized Controlled Trial.* *ArXiv*.
- [9] Kosch, T., Karolus, J., Zagermann, J., Reiterer, H., Schmidt, A., & Woźniak, P. W. (2023). A survey on measuring cognitive workload in human-computer interaction. *ACM Computing Surveys*, 55(13s), 1-39.