**PEAP 4 ( 6-weeks online) 2023**

**CELE Position Analysis Paper**

**Deng Wei**

20619258

6 weeks online

Schools of Computer Science

MSc Computer Science

Permission given to use this paper: No

Word Count: 1870

**Will AI programming assistant make traditional programming jobs redundant?**

**18th August 2024**

**Abstract:**

With the emergence of some mainstream AI programming assistants (ChatGPT, Github Copilot), programming practitioners can generate code faster to some extent. This article mainly studies the background of AI programming assistants, the current mainstream usage scenarios, and their shortcomings. Research results show that although AI assistants can help users quickly generate code snippets, there are still some inevitable flaws in the generated code. This article first proposes three academic results on the benefits of AI programming and raises reasonable academic doubts about two of them. Then, this article cites three other academic groups on the correctness, performance, and maintainability of AI-generated code. The maintainable research results discuss some key shortcomings of AI programming that cannot be ignored. These research results mean that in the short term, human-centered programming jobs are still essential. The purpose of this article is to demonstrate the limitations of AI assistants' programming capabilities through existing academic research, journal articles, and other resources, and to discuss the future relationship between AI and users. At the end of the article, the author looks forward to an ideal relationship between AI programming and programming users in the future, namely pair programming, which refers to the interdependence and inseparability between AI programming assistants and programming practitioners.

**Contents**

[Introduction 1](#_Toc737581945)

[1. AI-Assisted Programming: Transformative Impacts 1](#_Toc1264139131)

[1.1. The benifit from AI programming assitants 1](#_Toc1330654946)

[1.2. Some reasons for choosing AI programming assitants 2](#_Toc422977564)

[1.3. Some ares might become redundant due to AI 2](#_Toc813046578)

[2. Limitations of AI programming assistants 3](#_Toc435390585)

[2.1. Limitations of the accuracy of AI assistant generated code 3](#_Toc1132776766)

[2.2. Poor performance of AI assistant generated code 3](#_Toc1136575507)

[2.3. Low maintenance of AI assistant generated code 4](#_Toc557506084)

[Conclusion 4](#_Toc533601927)

[References 6](#_Toc355877217)

**Will AI make traditional programming jobs redundant?**

**Key words:**

Artificial intelligence, AI programming assistants, GitHub Copilot, ChatGPT, Programming jobs

# Introduction

Nowadays (2024), some AI programming assistants have emerged as powerful platforms for programmers to generate code quickly. A recent scholarly investigation has demonstrated that developers who use GitHub Copilot report a median of 30.5% of their code being written with help from the tool (Kuhail et al. 2023). A study initiated by Myers et al. (2023) also pointed out that by giving feedback to the tool to adjust or customize the model, developers hope to enhance their experience with AI programming assistants. They also hope to gain a deeper understanding of programming languages, APIs, and code context by using these tools. Does this mean that traditional programming jobs are becoming subordinate to AI?

Such expositions are unsatisfactory because they ignored the importance of the unique creative thinking of programmers. Their unique job creation ability still makes it difficult for jobs in the software development field to be replaced in the short term. A studies conducted by Sasikumar hold the view that for jobs of lower and medium complexity levels, ChatGPT performed above average in terms of run-time and memory use; but, when faced with increasingly difficult tasks, its performance decreased (Kuhail et al. 2023). These results would seem to suggest that AI plays more of a mentoring role and continues to empower programming- related jobs rather than make people its vassals..

# AI-Assisted Programming: Transformative Impacts

## **The benifit from AI programming assitants**

With the continuous development of AI intelligent technology, AI programming assistants developed by some companies in cutting-edge technological fields have emerged. These AI assistants currently occupy the mainstream market, including GitHub Copilo and ChatGPT.

As Copilot has access to so many open-source projects, it can leverage larger code bases across a greater variety of programming languages. As per the study conducted by Yetistiren and his colleagues, Copilot successfully produced legitimate code for 91.5 percent of the problems found in the HumanEval problem dataset (Yetistiren et al. 2022). Furthermore, the most well-known AI tool currently is ChatGPT developed by OpenAI, which is an AI tool based on context generation capabilities that has found utility across various sectors such as programming. ChatGPT 3.5 (its latest large model version) can help programmers become more productive by creating better-performing, shorter, and more readable code (Dos Santos et al. 2024). It can be seen from the research of Dos Santos and his colleagues that AI software tools are gradually penetrating into all aspects of programming work, including coding software education, as Students have a faster way to get started with programming. According to Jalil et al. ChatGPT may present a benefit or a risk to conventional teaching methods, particularly when it comes to resolving test questions for software testing education (Dos Santos et al. 2024, cited in Jalil et al. 2023).

## Some reasons for choosing AI programming assitants

According to Smith et al. (2023), the main reasons developers employ AI programming assistants are to minimize keystrokes, expedite work completion, and help them remember syntax (Myers et al. 2023). The research has also shown some successful use cases, including quality assurance, learning and documentation. Participants reported using AI programming assistants for quality assurance such as generating useful log message and producing test cases quickly, and they use these tools to consider edge cases. The second major finding was that participants can also utilize these tools when learning new libraries and new programming languages. The participants said it is especially useful when a project uses complex programming language since the codebase from AI is a polyglot project with Java, Golang and CPP implementations (Myers et al. 2023). All kinds of phenomena are explaining to us that AI is freeing up some work that should be performed by programmers.

However, the main weakness with this theory is that Myers and his colleagues ignored the code correctness. According to Brandon’s (2023) findings, Copilot will still produce erroneous code, just like ChatGPT produces incorrect statements, so humans must still be in the loop. These models don't really reason at a deep level and there isn't a clear path to getting them there. It is for that reason that Brandon thinks programmers will be employed for a long time, but the efficiency will improve dramatically (Brandon 2023).

## Some ares might become redundant due to AI

Although there are many different kinds of programmers, including front-end developers, back-end developers, and testers, Upwork claims that the greatest danger to QA and beta testers comes from artificial intelligence, which allows programmers to test their applications more rapidly and precisely. The popularity of no-code website user interface generators like Wix has already resulted in a decline in the need for front-end web developers. On the other hand, there is a greater need than ever for graphic designers to produce fresh web interfaces. Because artificial intelligence algorithms are becoming more efficient, there is a gradual decrease in the need for debuggers. Any monotonous, uninspired programming work is already being replaced or soon will be (GARG 2023).

However, one problem with AI programming assistants is that it rely on existing data, which allows creative programmers to shine through. Also, the communication skills of a programmer will also play an essential role as companies will want to hire programmers who can talk to clients and better understand their needs. Exhibiting innovation, creativity, and understanding of Artificial Intelligence on an interviewer‘s resume might help them get hired easily (Myers et al. 2023).

# Limitations of AI programming assistants

## Limitations of the accuracy of AI assistant generated code

As for computing-related fields such as programming, ChatGPT showed a struggling performance when generating code fragments in hard problems (Hampus and Vilma 2023). In a survey by Hampus and Vilma, they used three iterative runs to try to get ChatGPT to solve program problems. As an objective metric for this evaluation, a success rate that was correlated with the number of iterations that ChatGPT took to solve a problem was employed. Research results show that when dealing with code problems of various difficulties, the success rate of ChatGPT was 90% in the first iteration, improved in the second iteration, reaching 97.78%, and stagnated at 97.78% in the subsequent third iteration. Then, they divided the variables into three groups according to the difficulty of the code, corresponding to low, medium, and high-difficulty codes, and observed the success rates of the three iterations under different difficulties. The research results found that they are similar in solving medium and low-code problems. Even if the code problem is not solved perfectly in the first iteration (96.67% correct rate), the correct rate can still be guaranteed in the next two iterations to 100%. However, difficult code problems are another situation. The accuracy rate in the first iteration is 76.67% and then stagnates at 93.33% in subsequent iterations (Hampus and Vilma 2023). One possible implication of this is that it is possible for ChatGPT to help programmers to tagle some programming issues such as generating a simple code fragment, as for for complex procedural domain problems, human intervention is still required.

## Poor performance of AI assistant generated code

In addition to the accuracy of the code, its performance is also an important indicator, including memory usage and run-time performance. Hampus and Vilma also used three iterations to study the memory usage and runtime of the generated code to test the code generated by the AI assistant. It is indicated by the research content that with percentiles ranging from 47% to 59% for memory usage and from 56% to 59% for run-time across a range of difficulty levels, ChatGPT showed above-average performance. However, when presented with more difficult problems, ChatGPT's capacity to generate accurate and effective code decreases. Notably, even when the research team provided more context, the proportion of ChatGPT producing correct results improved, but at the cost of poor performance in terms of runtime and memory usage (Hampus and Vilma 2023). The evidence from this study suggests that under a limited number of iterations, the code generated by the AI assistant still has flaws in program performance, which also makes the programming ability of programmers difficult to become redundant in a short time.

## Low maintenance of AI assistant generated code

The security and maintainability are one of the important criteria for judging whether a code is excellent. According to recent reports, the most frequent reasons why users of these tools give upon using outputted code are that the code does not perform the correct action or it does not meet functional or nonfunctional requirements (Kuhail et al. 2023). Another study held by Yetiştiren and his classmates also revealed the high maintenance costs of code generated by AI assistants. In this experiment, they used the SonarQube code checker to obtain maintainability results. Then, they compared the code smells of code generated by three different artificial intelligence programming assistants-ChatGPT, Copilot and Codewhispererde. It can be seen from the statistical chart: that two problems had three code smells, three problems had two, and fourteen problems had just one. For the problems with at least one smell, the average technical debt was 9.1 minutes, and the estimated total time to solve all smells was 172 minutes (Yetiştiren et al. 2023). The evidence of low maintainability can be clearly seen in the case of AI’s abuse of code naming: In the AI-generated code snippet shown by Yetiştiren (2023), we can see that SonarQube found that the variable naming is inappropriate. The function was named using the camel case method(which is often used in Java programming), and the snake case should be used (Python programming conventions) methods (Yetiştiren et al. 2023). Overall, these results indicate that AI programming assistants still have a long way to go in terms of code maintainability.

# Conclusion

The aim of the present research was to examine will AI makes programmers redundant. In this study, we investigated the useability of AI assistant programming, such as GitHub Copilot. These experiments confirmed that AI programming assistants have indeed had a huge impact on the work content of traditional programmers. Some things that previously required programmers to do manually no longer need to be done by them. This also brings unprecedented challenges to this position: will low-tech programming jobs be replaced by AI?

In the case of the rapid development of AI, it does have an impact on people in the labor market, whether it is positive or negative. It is likely that AI technology has begun to integrate into all aspects of our lives, and it is bringing changes to our traditional labor market, the future world, will be the era of AI and labor coexistence, it will be a long-term game, seeking a balance (between AI and human) point will be a very good choice. In the short term, AI still has flaws in human emotions, and creative thinking, which also shows that most human work cannot be managed by AI at present.

A limitation of this study is that this study lacks the timeliness of the experimental content, that is, the update and iteration speed of the AI programming assistant is fast enough, which may cause some problems that arise in the article, such as the maintainability of the code, to be quickly solved or create new challenges.

# References

Brandon, P. (2023) *Can AI Tools like ChatGPT Replace Computer Programmers* [online] available from <<https://www.proquest.com/docview/2785489253>> [1 Aug 2024]

Garg, K. (2023) *Has Artificial Intelligence Made Jobs of Programmers and Developers Redundant?* [online] available from <<https://www.theteenmagazine.com/has-artificial-intelligence-made-jobs-of-programmers-and-developers-redundant>> [28 Jul 2024]

Hampus, E. and Vilma, H. (2023) *Can Artificial Intelligence Replace Humans in Programming?* [online] available from <[https://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1776700&dswid=812](https://www.diva-portal.org/smash/record.jsf?pid=diva2:1776700&dswid=812)> [14 Aug 2024]

Kuhail, M.A., Alturki, N., Alramlawi, S., and Alhejori, K. (2022) ‘Interacting with Educational Chatbots: A Systematic Review’. *Education and Information Technologies* [online] 28 (1). available from <<https://link.springer.com/article/10.1007/s10639-022-11177-3>> [13 Aug 2024]

Liang, J.T., Yang, C., and Myers, B.A. (2024) *A Large-Scale Survey on the Usability of AI Programming Assistants: Successes and Challenges*. [online] available from <<https://dl.acm.org/doi/abs/10.1145/3597503.3608128>> [12 Aug 2024]

Silva, C.A.G. da, Ramos, F.N., de Moraes, R.V., and Santos, E.L. dos (2024) ‘ChatGPT: Challenges and Benefits in Software Programming for Higher Education’. *Sustainability* [online] 16 (3), 1245. available from <<https://www.mdpi.com/2071-1050/16/3/1245?fbclid=IwAR0ggeX3AcWwhqNWFqbjOUfI5v54QlE-YXpVcGfphac5TDDW3Y32_ojcsqM>> [12 Aug 2024]

Yetiştiren, B., Özsoy, I., Ayerdem, M., and Tüzün, E. (2023) ‘Evaluating the Code Quality of AI-Assisted Code Generation Tools: An Empirical Study on GitHub Copilot, Amazon CodeWhisperer, and ChatGPT’. *ArXiv* [online] available from <<https://arxiv.org/abs/2304.10778>> [14 Aug 2024]

Yetistiren, B., Ozsoy, I., and Tuzun, E. (2022) ‘Assessing the Quality of GitHub Copilot’s Code Generation’. *ACM* [online] available from <<https://dl.acm.org/doi/abs/10.1145/3558489.3559072>> [1 Aug 2024]