"A Large-Scale Survey on the Usability of AI Programming Assistants: Successes and Challenges"

by Jenny T. Liang, Chenyang Yang, and Brad A. Myers

Reviewed By: Abdur Rafay Saleem

Paper Summary:

This paper presents the results of a large-scale exploratory qualitative study that surveyed 410 developers to investigate their usage practices of AI programming assistants like GitHub Copilot and uncover important usability challenges they face. Through a mix of quantitative and qualitative analysis, the authors found that developers primarily use these tools for convenience - to reduce key-strokes, finish tasks faster, and recall syntax. The tools were most successfully used for generating repetitive code or code with simple logic. However, developers resonate less with using the tools for brainstorming solutions. Key reasons developers don't use the tools are that they don't generate code meeting functional/non-functional requirements and the difficulty controlling the tools. The most prominent usability issues were not knowing what inputs influence the outputs, giving up on using the generated code, and trouble controlling the tool. Participants wanted to improve the tools through direct user feedback and teaching the models to better understand code context. The findings provide implications for both users and creators of AI programming assistants.

Critical Evaluation Criteria:

A. Soundness of Approach

The study employs an appropriate mixed-methods approach, using a survey to gather both quantitative and qualitative data at scale from a large, diverse sample of 410 developers. The survey questions were piloted with developers and reviewed by an external researcher to ensure clarity and topic coverage. Established best practices for statistical analysis of survey data were followed. The qualitative analysis involved multiple rounds of open coding by the two authors, leading to a well-developed and unanimous codebook. The approach appears technically sound for the exploratory nature and scale of the research.

One limitation is potential memory bias, as participants had to recall experiences. The authors tried to mitigate this by grounding responses to a specific project. There could also be some sampling bias, as participants were recruited from GitHub repos related to AI programming assistants, so it may over-represent enthusiasts of these tools compared to the general developer population.

B. Novelty

This study stands out as a comprehensive large-scale investigation into the real-world usability practices and challenges with AI programming assistants across a diverse population of developers. While prior smaller studies revealed some potential usability issues, this work quantifies the prevalence and importance of those challenges in practice. It also uncovers new themes around developers' mental models and desired

improvements for these tools. The findings on using AI assistants for recall and learning are novel use cases compared to prior work.

C. Clarity of Relation with Related Work

The paper provides good coverage of relevant work, including prior user studies on AI programming assistants (both RNN and transformer-based), theories of assistant usage modes, design recommendations, and studies on how developers traditionally learn and recall syntax. The authors clearly position their work as validating, extending or revealing limitations of prior findings through their larger-scale study and broader focus on usability. They describe how their results support, augment, or raise new questions compared to previous work.

D. Quality of Evaluation & Results

The scale of the study (410 developers) and diversity of the sample (geography, gender, programming experience and context) lends confidence to the generalizability and reliability of the results. The figures and statistics for Likert-scale questions are presented following best practices. The qualitative codes are well-described with clear examples. Overall the evaluation appears rigorous, and the results align with and extend findings from prior studies.

However, the self-reported frequency estimates may not be fully accurate, so in-situ telemetry would be valuable to collect in future work. Also, while the study had a large overall sample size, the number of participants for some specific survey branches (e.g. reasons for giving up on code) are not reported, so it's unclear if the sample was always large enough to draw reliable conclusions.

E. Ability to Replicate

The paper provides the full survey instrument and codebooks in supplementary material to enable study replication, which is a strong plus. The participant recruitment and screening methodology is documented in detail. The analysis methods are thoroughly described.

The only gap is the specifics of the GitHub repos and screening criteria used aren't provided, which would be needed to closely replicate the sampling. The fast-moving nature of AI programming assistants means replication on newer versions of tools may yield different results.

F. Quality of Presentation

The motivation, research questions, methodology, and results are presented in a logical, easy-to-follow structure. Figures, tables and headings make the results skimmable. The writing style is academic but accessible. The key findings and implications are clearly summarized and connected back to prior work. Overall the paper is well-organized and written.

The results section is quite long and at times feels like a laundry list of codes and statistics. Perhaps a high-level summary early on, with the detailed results moved to an appendix, could improve the narrative flow. There are also a few instances of casual language (e.g. "by storm") that could be tightened up.

Conclusion:

In conclusion, this paper makes a valuable contribution to the HCI and software engineering literature by providing a comprehensive, large-scale analysis of developers' real-world practices and challenges with the usability of AI programming assistants. The mixed-methods approach is rigorous, though some of the sampling and evaluation limitations should be probed further in future work. The results validate and extend prior findings while revealing new themes to guide the design of future AI programming aids. With a few tweaks to the organization and presentation, this paper is well positioned to spur productive conversations and follow-up research in the software engineering community.