Evaluating the Quality of Artificial-Intelligence-Generated Code in Fixing Technical Debt and Merge Conflicts

Gustavo Vale and Eduardo Figueiredo
Federal University of Minas Gerais (UFMG)

Context. Software development is a collaborative activity, often involving distributed teams working on shared codebases [1]. Version control systems facilitate coordination by tracking code changes over time and allowing concurrent development tasks, such as bug fixing and feature additions [2]. However, ***merge conflicts*** - a frequent issue when integrating concurrent changes - and ***technical debts*** - a collection of design or implementation constructs that deteriorate the software quality in the middle-term - can disrupt workflows, reduce productivity, and lead to time-consuming, and error-prone resolution efforts [3]. These issues significantly impact the maintainability, security, and stability of software systems, making them critical areas for improvement [3,4]. On the other hand, AI coding tools have gained the attention of practitioners since these tools support them on daily tasks, such as bug fixing, code refactoring, and feature implementation.

Problem. The central problem is that, despite the growing use of AI coding tools, their effectiveness in reducing technical debt and handling merge conflicts has not been systematically studied.

Goals. This research aims to evaluate the effectiveness and quality of AI-generated code in addressing merge conflicts and technical debts – two major challenges in real-world software projects. As AI-powered tools like GitHub Copilot and OpenAI Codex become integrated into developer workflows, their ability to provide reliable solutions for these problems must be rigorously assessed.

Hypothesis. Our hypothesis is that AI-generated code can significantly reduce the time and effort required to resolve merge conflicts and address technical debts. However, there are concerns regarding the quality and security of such code, including the potential introduction of vulnerabilities.

Methods. Preliminary studies have shown that while AI algorithms can predict merge conflicts [4,5], a combination of social and technical metrics is essential for optimal performance. This research project will build on these findings by conducting empirical studies to evaluate AI-generated solutions using real-world datasets and established security frameworks like CWE and OWASP.

Expected Outcomes. The results will provide valuable insights into the capabilities and limitations of AI tools in software engineering, offering practical recommendations for developers and informing the development of more robust AI-assisted coding environments.

References

[1] M. Joblin, S. Apel, and W. Mauerer. “Evolutionary trends of developer coordination: a network approach.” In: **Empirical Software Engineering (EMSE)** 22.4, pp. 2050–2094, 2017.
[2] T. Zimmermann, A. Zeller, P. Weissgerber, and S. Diehl. “Mining version histories to guide software changes.” In: **Transactions on Software Engineering (TSE)** 31.6, pp. 429–445, 2005.
[3] T. Mens. “A state-of-the-art survey on software merging.” In: **Transactions on Software Engineering (TSE)** 28.5, pp. 449–462. issn: 1939-3520, 2002.
[4] G. Ghiotto, L. Murta, M. Barros, and A. Hoek. “On the Nature of Merge Conflicts: A Study of 2,731 Open Source Java Projects Hosted by GitHub.” In: **Transactions on Software Engineering (TSE)** 46.8, pp. 892–915. issn: 1939-3520, 2020.
[5] G. Vale, H. Costa, and S. Apel. “Predicting Merge Conflicts Considering Social and Technical Assets”. In: **Empirical Software Engineering (EMSE)**, 29, 24, pp. 1-26, 2024.