LGTM! Characteristics of Auto-Merged LLM-based Agentic PRs

Ruben Branco*
rmbranco@fc.ul.pt
University of Lisbon
Lisboa, Portugal

Paulo Canelas*†
pacsantos@cmu.edu
Carnegie Mellon University
Pittsburgh, USA

Catarina Gamboa*†
cgamboa@cmu.edu
Carnegie Mellon University
Pittsburgh, USA

Alcides Fonseca
amfonseca@fc.ul.pt
University of Lisbon
Lisboa, Portugal

Abstract

AI tools are generating code faster than humans can properly review it, leading repositories to skip review and auto-merge agentic Pull Requests (PR) directly. In this study, we analyze the characteristics of auto-merged agentic PRs and compare them to human-authored ones. We examine code characteristics, repository ecosystems, and agentic tools across the AIDev dataset, spanning diverse software engineering tasks. We find that auto-merged PRs are smaller and more focused, and that repositories tend to either auto-merge all or none agentic PRs, with more mature repositories favoring the latter. Compared to human-authored auto-merges, maintainers auto-merge agentic PRs more often but show caution toward PRs that delete existing code. Among agents, OpenAI Codex and Claude Code receive the highest auto-merge rates. These findings can inform agentic tool design and repository's auto-merge decisions.

CCS Concepts

• **General and reference** → *Empirical studies*; • **Computing methodologies** → **Artificial intelligence**; • **Information systems** → *Open source software*.

Keywords

Auto-Merge, Pull Requests, Agents, Empirical Study

ACM Reference Format:
Ruben Branco, Paulo Canelas, Catarina Gamboa, and Alcides Fonseca. 2026. LGTM! Characteristics of Auto-Merged LLM-based Agentic PRs. In *23rd International Conference on Mining Software Repositories (MSR '26), April 13–14, 2026, Rio de Janeiro, Brazil*. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3793302.3793621

1 Introduction

The introduction of LLM-based agentic tools has accelerated software development by automating tasks like test creation [15], and refactoring [20]. These time-consuming tasks are now automated by AI agents, increasing overall developer productivity [5, 23, 24].

Nevertheless, the fast software development pace now requires developers to manually review more pull requests (PRs) to maintain code quality [10]. When reviewing, developers must quickly assess the basic quality criteria of the PR [11], shifting the burdensome task of writing code toward the manual effort of reasoning about and accepting PRs. As software development increases with the use of AI-driven tools, the task of analyzing and accepting PRs becomes even more challenging [16], leading to stale pull requests [18].

Auto-merge, the process of automatically merging pull requests without human intervention [13], can quicken the development workflow. However, trusting automated changes requires safeguards to increase the likelihood of acceptance [30]: adherence to best practices in the pull request (e.g., good descriptions [29], linting) and in the repository ecosystem (e.g., tests, CI).

Prior work studied the role and limitations of agents in open-source development. Soares et al. [25] found that smaller PRs with fewer commits and files are more likely to be merged quickly. Moataz et al. [6] reported that developers use ChatGPT for complex, review-heavy PRs with high code churn and longer lifecycles. Watanabe et al. [27] observed an 83.8% acceptance rate for Claude Code's PRs focused on refactoring and testing, though rejections still occur when lacking project-specific context.

However, despite ongoing efforts to study agentic code contributions [27], no systematic analysis has compared auto-merged agentic-generated pull requests across models, task types, or against human-authored baselines. **Understanding agentic auto-merges and how they differ from human-authored ones can inform both agent tool design and repository auto-merge decisions.**

In this work, we quantitatively study auto-merged agentic pull requests to understand their characteristics and how they differ from human PRs. We analyze the AIDev dataset [17], examining pull request characteristics across different tasks, their ecosystems, and in comparison with their human-authored counterparts.

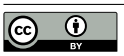
Our findings show that agentic PR acceptance is bimodally distributed, typically either fully accepted or rejected. Auto-merged Agentic PRs are notably smaller and more focused, and less common in more mature, well-governed projects. Compared to human PRs, agentic PRs are auto-merged more frequently, but maintainers are more cautious when agents remove existing code; among tools, OpenAI Codex [22] and Claude Code [1] show the highest trust.

2 Methodology

We define *Auto-Merge* as merging a PR without reviews or review comments, and without comments from anyone other than the author or bots (e.g., Dependabot). This definition accommodates agentic workflows where agents commit iteratively (e.g., Devin) or act under user credentials (e.g., Codex), while excluding PRs with human interaction. To understand the characteristics of auto-merged agentic PRs, we propose the following research questions.

RQ1 What properties of agentic-pull requests and repositories relate to auto-merges?
RQ2 What characteristics distinguish human and agent authored auto-merge PRs?

*All authors contributed equally to this work.
†Also with LASIGE, University of Lisbon.



MSR '26, Rio de Janeiro, Brazil

ACM ISBN 979-8-4007-2474-9/2026/04
https://doi.org/10.1145/3793302.3793621

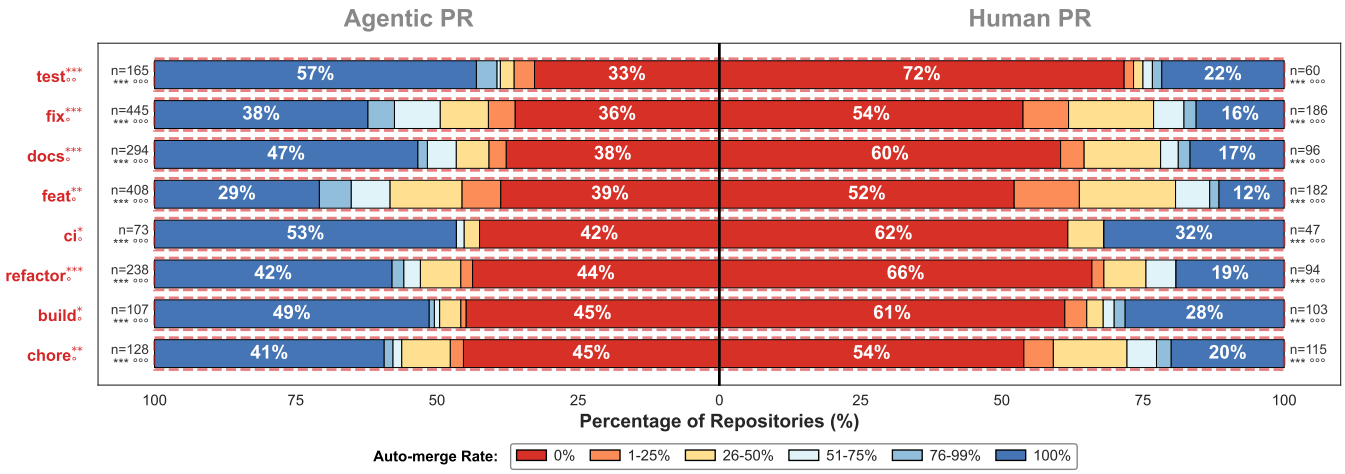


Figure 1: Distribution of auto-merge rates per feature across repositories for AI and Human Pull Requests. Highlighted bars across AI-Human PRs represent significance between Agentic and Human PRs, where $^{*}p < 0.05$, $^{**}p < 0.01$, and $^{***}p < 0.001$. Statistical significance (*) presented in the repository (n) label shows impact within Agentic PRs and within Human PRs. For statistical significant results, we also present the effect size: small (°), medium (°°), and high (°°°).

To ground our analysis, we use AIDev [17], a dataset with approximately 33,000 PRs from 2,800 popular repositories (≥ 100 stars). These PRs are generated by five widely used AI Coding Agents: Claude Code [1], Codex [22], Copilot [12], Cursor [2], and Devin [8]. The dataset also includes a section of human-authored PRs from the same repositories for comparison. AIDev provides metadata at multiple levels, including repository, pull request, issues, and user information. To identify defining characteristics, we analyze agentic and human auto-merged pull requests along three dimensions:

PR Dimension We analyze PR size and complexity through code metrics (additions, deletions, total lines changed, and files changed) and examine auto-merge rates across task types (e.g., feature implementations, bug fixes, documentation).

Repository Dimension We examine how repository characteristics affect auto-merge rates, including maturity (stars, forks), the presence of CI workflows, and community health indicators such as codes of conduct, contribution guidelines, licenses, and README files [26].

Agent Dimension We split auto-merged PRs by agent to reveal whether different tools —with distinct workflows and reasoning approaches— exhibit different auto-merge patterns.

To conduct our analysis, we perform a three-step approach.

First, we perform a preliminary analysis of AIDev and extend the dataset of human-authored pull requests with comments, file-level changes, and PR interactions. For both human and agentic PRs, we also collect CI jobs and repository health metrics. To mitigate bias from repositories that dominate the dataset (mochilang/mochi and MontrealAI/AGI-Alpha-Agent-v0 represent 37.2% of PRs), we compute auto-merge rates per repository and treat each repository as a single observation. We also filter out repositories with fewer than five merged PRs (below the 75th percentile), as they do not provide sufficient evidence of auto-merge patterns.

Second, we perform a pull-request–level analysis of the dataset. We analyze the distribution of PR properties (e.g., lines changed) across three categories: AI-generated auto-merged PRs, AI-generated PRs with interaction, and human-authored PRs. We verify statistical differences using the Mann-Whitney U Test [19]. We further compare AI and human-authored PRs across AIDev task categories: `build`, `chore`, `ci`, `docs`, `feat`, `fix`, `refactor`, and `test`. For each category, we test statistical differences in PR characteristics at two thresholds: 0% (no auto-merge) and 100% (fully auto-merged). When comparing groups (Agentic vs. human-authored PRs), we apply the Bonferroni correction [3] to account for multiple comparisons and control the familywise error rate.

Finally, we perform a repository-level analysis, applying the Mann-Whitney U test to stars, forks, health score, and CI runs.

3 Analysis and Findings

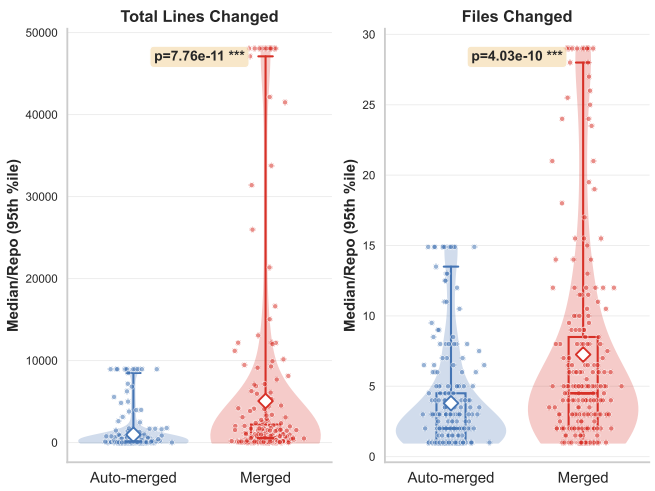
In this section, we present the results of our empirical analysis based on the research questions defined in Section 2. Our findings showcase the differences in auto-merges between agents and human pull requests, and the impact of the project and task type. The analysis and mined data are publicly available in our artifact [4].

3.1 Agentic PRs merged and auto-merged

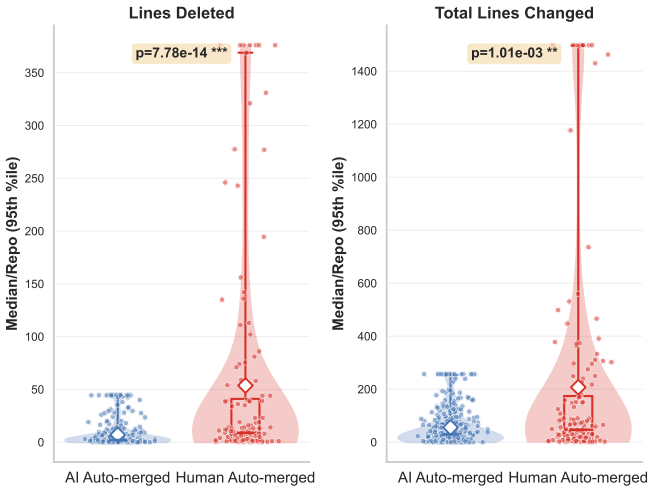
When comparing all agentic PRs that were merged and auto-merged, we have identified four main findings as presented below.

Agentic Auto-merge is Bimodally Distributed. Auto-merge resembles a bimodal distribution: repositories either use auto-merge or avoid it, leaving little middle ground. This can be visualized in the left section of Figure 1, where the majority of repositories either has a 0% or 100% auto-merge rate, with little density in between.

Auto-merged PRs are Significantly Smaller. Figure 2a compares PR size characteristics between auto-merged PRs and their counterparts. Auto-merged PRs are significantly smaller in lines changed, files changed, additions, and deletions. This suggests auto-merge works well for incremental, focused changes.



(a) Comparison of Total Lines and Files changed between Agentic PRs auto-merged and merged with human interactions.



(b) Comparison of Total Lines changed and lines deleted between Agentic and Human auto-merged PRs.

Figure 2: Code metrics of auto-merged PRs compared at a repository-level (with ≥5 PRs per repository). The violin plots include a box plot within and a diamond symbolizing the average value. Significance: $^{*}p < 0.05$, $^{**}p < 0.01$, and $^{***}p < 0.001$.

Table 1: Comparison of repository and CI information between AI auto-merges and no auto-merges.

Metric	Auto-merge	No auto-merge	P-Value
Stars	440.0	983.0	5.84e-04 ***
Forks	81.0	264.0	5.89e-10 ***
Health Score	50.0	75.0	9.75e-12 ***
M_d Check Runs	4.0	10.0	1.55e-13 ***

Repository Maturity Matters. We find that repositories with 0% auto-merges rate tend to be more mature, well-governed projects with robust processes, that require more human oversight (Table 1). More mature repositories with more CI checks and agent-guiding files have less auto-merge rates with regards to smaller, less formal environments, where requirements are looser.

Tests are Highly Trusted, Features are Less. Figure 1 shows that agentic PRs have significantly higher auto-merge rates across all task types ($p < 0.001$), though some vary. Tests and CI achieve 100% auto-merge rates in over half of repositories, with Build close behind at 49%. In contrast, feature implementations (`feat`) reach 100% auto-merge in only 29% of repositories as the lowest among task types. This difference may reflect risk as customer-facing features carry higher stakes, while developer-facing tasks like tests, CI, and builds are lower-risk and easier to trust.

RQ 1. Agentic Auto-merged PRs Characteristics

Auto-merge follows a bimodal distribution and is more common for smaller PRs, in less mature repositories, and for lower-risk developer-facing tasks, such as tests, CI, and builds.

3.2 Agentic and Human auto-merged PRs

For our second research question, we compare agentic and human-authored auto-merged PRs from the same repositories. Most repositories use a single agent (91.7%) and a small percentage use multiple agents (8.3%), with only one repository using all five agents. Below we present our four main takeaways from this comparison.

AI-dominated Automerging. Figure 1 shows that for all tasks, repositories with 100% auto-merge rate by agents are significantly higher when compared to their human counterpart, all of them with statistical significance. This suggests that repository maintainers trust AI-produced code sufficiently for its auto-merge rate to be superior to that of humans across all task types.

Auto-merged PRs Show Different Deletion Patterns. Human auto-merged PRs have more lines changed and more deletions compared to agentic PRs as visible in Figure 2b. Besides agentic PRs being smaller, they reveal a fundamental behavioral difference since agentic PRs with deletions are less auto-merged when compared to human PRs with deletions. This suggests that users are more cautious of agents removing existing code.

Not All Agents are the Same. Although human-authored auto-merged PRs follow a bimodal distribution similar to agent-authored ones at the task-type level (Figure 1), this pattern appears to shift when studied at the agent level (Figure 3). For most agents, the bimodal pattern fades, with in-between rates increasing compared to the aggregate values and yielding more balanced, single-mode distributions. Some agents, such as OpenAI Codex and Claude Code, still exhibit high auto-merge rates in our sample, though Claude Code has limited adoption time and a smaller number of observations ($n = 12$). The bimodality observed in Section 3.1 may partly reflect the contrast between agents with higher auto-merge rates and those with lower rates across repositories, though sample sizes vary considerably across agents. These patterns may

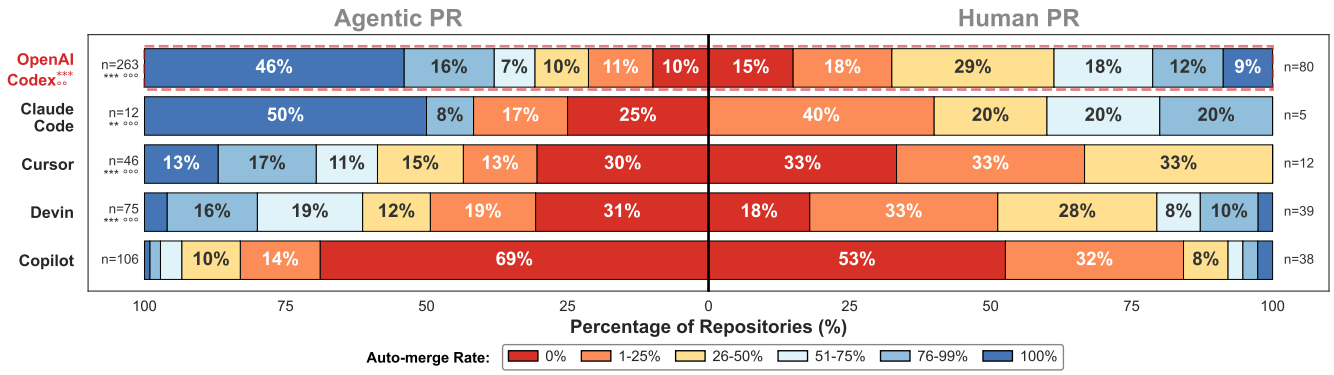


Figure 3: Distribution of auto-merge rates per agent and humans across repositories. Highlighted bar(s) represent significance w.r.t. model to human PR, where $^{*}p < 0.05$, $^{**}p < 0.01$, and $^{***}p < 0.001$. Significance within agents is added to the repo (n) label. Significant results effect size are also presented with small (°), medium (°°), and high (°°°).

also reflect tool design philosophies. AI assistants and editors such as Copilot and Cursor are designed to improve the developer's workflow, while AI agents such as OpenAI Codex and Claude Code operate more independently [9]. Devin, which is marketed as a "junior engineer" [7], still requires more reviews.

Only OpenAI Codex Significantly Differs from Humans. Figure 3 shows that OpenAI Codex is the only agent whose auto-merge rates differ significantly from human PRs in the same repositories ($p < 0.001$). While Claude Code achieves the highest proportion of repositories with 50% auto-merge, its small sample size ($n = 12$) limits its statistical comparison. The remaining agents show no significant difference from human auto-merge patterns.

RQ 2. Agentic Vs Human Auto-merged Pull Requests

Agentic auto-merged PRs are smaller and achieve higher auto-merge rates than human PRs across all task types. However, maintainers show caution toward agentic PRs that delete existing code. Auto-merge patterns also vary by agent: autonomous agents (Codex, Claude Code) achieve higher rates than assistant-style tools (Copilot, Cursor).

4 Threats to Validity

Internal Validity. Excluding repositories with fewer than five PRs may introduce survivorship bias by omitting early adopters who later abandoned agent usage. Although we aggregate at the repository level to reduce the impact of high-volume projects, repositories following similar workflows may still introduce correlated patterns. Furthermore, repository health metrics reflect file presence rather than content quality, which may not fully capture project maturity.

External Validity. Our findings are based on AIDev, a large but single-source dataset of agentic pull requests. Auto-merge behaviors may vary in proprietary codebases with stricter review policies, or with agents not represented in AIDev. Agent representation is also uneven: for instance, Claude Code was released in late February 2025, resulting in less time for adoption.

Construct Validity. We define auto-merge as PRs merged without external review comments, a strict definition that may exclude PRs with brief approvals (e.g., "LGTM"). Alternative definitions could yield different characteristics, while potentially capturing a broader range of out-of-scope semi-automated or lightly reviewed merges.

5 Related Work

Prior work has studied the technical and social factors that affect pull request (PR) outcomes [25, 30], showing that attributes such as change size, contributor identity, and reviewer assignment significantly impact merge decisions. In automated pull-based workflows, studies found that bot-authored PRs are delayed or rejected more frequently than human-authored ones, despite being of similar quality [28]. Dependabot, for instance, sees high acceptance but is actively managed to reduce fatigue and misalignment [14]. More recent work explores LLM-assisted PRs, showing that developers use ChatGPT for complex, high-effort tasks [6], often treating its output as a draft to refine before merging [21]. Watanabe et al. [27] studied Claude Code generated PRs and found that autonomous systems succeed on low-risk tasks but still require human feedback. Beyond a human or single-agent focus, we extend prior work by analyzing AI auto-merged PRs across multiple agents and tasks.

6 Conclusion

In this paper, we conducted a quantitative analysis of auto-merged agentic pull requests across agents, repositories, and task types. Our findings show that auto-merged PRs are smaller, more focused than other agentic merged PRs, and occur more in less mature repositories that are more lenient in their requirements. Additionally, auto-merge decisions follow a bimodal distribution, where repositories primarily either auto-merge all or none of the agentic PRs. When looking at agent-level, the distribution is more balanced, revealing that OpenAI Codex and Claude Code achieve higher auto-merge rates, while the others lack in that aspect. Moreover, we find that maintainers trust AI produced code enough to auto-merge it often, but are cautious when agents remove existing code. These findings can help developers navigate an increasingly agentic software engineering ecosystem, and inform the design of agentic frameworks that produce more targeted and trustworthy code.

Acknowledgments

This work is funded by national funds through FCT – Fundação para a Ciência e a Tecnologia, I.P., under the project CMU-Portugal, ref. SFRH/BD/151469/2021 and PRT/BD/154254/2021, a PhD Scholarship (RB) https://doi.org/10.54499/2022.10727.BD, and under the LASIGE Research Unit, ref. UID/00408/2025, DOI https://doi.org/10.54499/UID/00408/2025. This work was also funded by the European Union - NextGenerationEU through the PRR - Recovery and Resilience Plan under the LASIGE Research Unit, ref. UID/PRR/00408/2025, DOI https://doi.org/10.54499/UID/PRR/00408/2025 and ref. UID/PRR2/00408/2025.

References

[1] Anthropic. 2025. Claude Code. https://code.claude.com/docs/en/overview.
[2] Anysphere. 2023. Cursor. https://cursor.com/.
[3] Richard A Armstrong. 2014. When to use the Bonferroni correction. *Ophthalmic and physiological optics* 34, 5 (2014), 502–508.
[4] Ruben Branco, Paulo Canelas, Catarina Gamboa, and Alcides Fonseca. 2026. *Artifact for "LGTM! Characteristics of Auto-Merged LLM-based Agentic PRs".* doi:10.5281/zenodo.18341679
[5] Leah Brown. 2024. New research reveals AI coding assistants boost developer productivity by 26%: What it leaders need to know. https://itrevolution.com/articles/new-research-reveals-ai-coding-assistants-boost-developer-productivity-by-26-what-it-leaders-need-to-know/
[6] Moataz Chouchen, Narjes Bessghaier, Mahi Begoug, Ali Ouni, Eman Alomar, and Mohamed Wiem Mkaouer. 2024. How Do Software Developers Use ChatGPT? An Exploratory Study on GitHub Pull Requests. In *International Conference on Mining Software Repositories*. 212–216. doi:10.1145/3643991.3645084
[7] Cognition AI. 2025. *When to Use Devin.* https://docs.devin.ai/essential-guidelines/when-to-use-devin
[8] Cognition Labs. 2024. Devin. https://devin.ai/.
[9] Tech Croc. 2025. https://dev.to/tech_croc_f32fbb6ea8ed4/claude-code-vs-cursor-vs-copilot-why-the-future-of-coding-lives-in-the-terminal-3n8m
[10] Zheyuan Kevin Cui, Mert Demirer, Sonia Jaffe, Leon Musolff, Sida Peng, and Tobias Salz. 2025. The effects of generative AI on high-skilled work: Evidence from three field experiments with software developers. *Available at SSRN 4945566* (2025).
[11] Hugo Dias. 2018. https://hugooodias.medium.com/the-anatomy-of-a-perfect-pull-request-567382bb6067
[12] GitHub. 2021. GitHub Copilot. https://github.com/features/copilot.
[13] GitHub. 2026. *Automatically Merging a Pull Request.* https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/incorporating-changes-from-a-pull-request/automatically-merging-a-pull-request
[14] Runzhi He, Hao He, Yuxia Zhang, and Minghui Zhou. 2023. Automating Dependency Updates in Practice: An Exploratory Study on GitHub Dependabot. *Transactions on Software Engineering* 49, 8 (2023), 4004–4022. doi:10.1109/TSE.2023.3278129
[15] Kush Jain and Claire Le Goues. 2025. TestForge: Feedback-Driven, Agentic Test Suite Generation. *CoRR* abs/2503.14713 (2025). doi:10.48550/ARXIV.2503.14713
[16] SayedHassan Khatoonabadi, Diego Elias Costa, Rabe Abdalkareem, and Emad Shihab. 2023. On Wasted Contributions: Understanding the Dynamics of Contributor-Abandoned Pull Requests-A Mixed-Methods Study of 10 Large Open-Source Projects. *ACM Transactions on Software Engineering and Methodology* 32, 1 (2023), 15:1–15:39. doi:10.1145/3530785
[17] Hao Li, Haoxiang Zhang, and Ahmed E. Hassan. 2025. The Rise of AI Teammates in Software Engineering (SE) 3.0: How Autonomous Coding Agents Are Reshaping Software Engineering. arXiv:2507.15003 [cs.SE] https://arxiv.org/abs/2507.15003
[18] Zhixing Li, Yue Yu, Tao Wang, Gang Yin, ShanShan Li, and Huaimin Wang. 2022. Are You Still Working on This? An Empirical Study on Pull Request Abandonment. *Transactions on Software Engineering* 48, 6 (2022), 2173–2188. doi:10.1109/TSE.2021.3053403
[19] Patrick E. McKnight and Julius Najab. 2010. *Mann-Whitney U Test.* John Wiley & Sons, Ltd, 1–1. arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/9780470479216.corpsy0524 doi:10.1002/9780470479216.corpsy0524
[20] Ansong Ni, Daniel Ramos, Aidan Z.H. Yang, Inês Lynce, Vasco Manquinho, Ruben Martins, and Claire Le Goues. 2021. SOAR: A Synthesis Approach for Data Science API Refactoring. In *Proceedings of the 43rd International Conference on Software Engineering* (Madrid, Spain) *(ICSE '21)*. IEEE Press, 112–124. doi:10.1109/ICSE43902.2021.00023
[21] Daniel Ogenrwot and John Businge. 2025. PatchTrack: A Comprehensive Analysis of ChatGPT's Influence on Pull Request Outcomes. *CoRR* abs/2505.07700 (2025). doi:10.48550/ARXIV.2505.07700
[22] OpenAI. 2025. OpenAI Codex. https://chatgpt.com/features/codex/.
[23] Elise Paradis, Kate Grey, Quinn Madison, Daye Nam, Andrew Macvean, Vahid Meimand, Nan Zhang, Benjamin Ferrari-Church, and Satish Chandra. 2025. How Much Does AI Impact Development Speed? an Enterprise-Based Randomized Controlled Trial. In *International Conference on Software Engineering: Software Engineering in Practice*. 618–629. doi:10.1109/ICSE-SEIP66354.2025.00060
[24] Sida Peng, Eirini Kalliamvakou, Peter Cihon, and Mert Demirer. 2023. The Impact of AI on Developer Productivity: Evidence from GitHub Copilot. *CoRR* abs/2302.06590 (2023). doi:10.48550/ARXIV.2302.06590
[25] Daricélio Moreira Soares, Manoel Limeira de Lima Júnior, Leonardo Murta, and Alexandre Plastino. 2015. Acceptance factors of pull requests in open-source projects. In *Symposium on Applied Computing*. 1541–1546. doi:10.1145/2695664.2695856
[26] TryAPIs. 2025. *GitHub API | Get Community Profile Metrics.* https://tryapis.com/github/api/repos-get-community-profile-metrics
[27] Miku Watanabe, Hao Li, Yutaro Kashiwa, Brittany Reid, Hajimu Iida, and Ahmed E. Hassan. 2025. On the Use of Agentic Coding: An Empirical Study of Pull Requests on GitHub. *CoRR* abs/2509.14745 (2025). doi:10.48550/ARXIV.2509.14745
[28] Marvin Wyrich, Raoul Ghit, Tobias Haller, and Christian Müller. 2021. Bots Don't Mind Waiting, Do They? Comparing the Interaction With Automatically and Manually Created Pull Requests. In *International Workshop on Bots in Software Engineering*. 6–10. doi:10.1109/BOTSE52550.2021.00009
[29] Tao Xiao, Hideaki Hata, Christoph Treude, and Kenichi Matsumoto. 2024. Generative AI for Pull Request Descriptions: Adoption, Impact, and Developer Interventions. *Proceedings of the ACM on Software Engineering* 1, FSE (2024), 1043–1065. doi:10.1145/3643773
[30] Xunhui Zhang, Yue Yu, Georgios Gousios, and Ayushi Rastogi. 2023. Pull Request Decisions Explained: An Empirical Overview. *Transactions on Software Engineering* 49, 2 (2023), 849–871. doi:10.1109/TSE.2022.3165056