# When Bots Write Code: Measuring How Much Help They Need

PHAN TRUONG PHUOC NGUYEN, University of British Columbia, Canada
SAGE YANG, University of British Columbia, Canada

AI coding agents are increasingly used to generate pull requests (PRs) in real GitHub repositories. These "agentic" PRs can automate routine development work, yet it is unclear which kinds of changes are delegated to agents and when humans still need to step in. In the overall project we define three research questions (RQ1–RQ3) about automation extent, work types, and bot-level differences. This milestone report focuses on RQ2: how work type and PR size/complexity vary across different levels of human interaction. We build on the human-interaction labels prepared for RQ1 and analyze task types and size metrics for each PR. We find that fully automated PRs tend to be smaller and more localized, while larger or more complex PRs are more likely to involve human review or direct commits.

## 1 Introduction

With the rapid rise of AI coding agents, systems capable of writing, reviewing, and modifying code are increasingly integrated into real-world software development. For example, autonomous agents such as OpenAI Codex have produced hundreds of thousands of pull requests across public repositories [2, 3]. In an ideal workflow, a developer would simply request a feature or report a bug, and an autonomous agent would implement the change, review its own work, and merge the pull request (PR) without human oversight.

However, in practice, human intervention remains essential. Current agents often lack full codebase context, leading to incomplete or error-prone changes. Developers frequently step in to provide feedback, request modifications, or fix issues directly before merging.

To study these dynamics, we use **AIDev**, a large-scale dataset of agent-generated pull requests (Agentic-PRs) from real GitHub repositories. AIDev contains **932,791 PRs** produced by five major agents—OpenAI Codex, Devin, GitHub Copilot, Cursor, and Claude Code—across more than 100,000 repositories and 70,000 developers. The dataset also provides an enriched subset of **33,596 PRs** containing review comments, commit diffs, event timelines, and linked issues. We focus our analyses on this enriched subset to better characterize the nature of human–agent collaboration.

Our goal is to examine **how autonomous current coding agents truly are**, and to what extent human developers remain part of the PR lifecycle. We investigate the following research questions:

- **RQ1 — Automation:** To what extent are PRs fully automated (i.e., all commits authored by bots with no human comments or reviews), and how often do they require human involvement?
- **RQ2 — Work Type and Complexity:** What types of tasks and complexity levels are associated with fully automated PRs versus those that involve human feedback or intervention?
- **RQ3 — Differences Across Agents:** Which coding agents produce PRs that most frequently require human review, feedback, or corrective commits?

## 2 Dataset

We use the AIDev dataset, which provides a large collection of agent-generated pull requests from real GitHub repositories. The dataset contains several linked tables, including `pull_request`, `pr_comments`, `pr_reviews`, `pr_commits`, and `pr_commit_details`—each keyed by `pr_id`. These

Authors' Contact Information: PHAN TRUONG PHUOC NGUYEN, University of British Columbia, Canada, nguy3492@student.ubc.ca; SAGE YANG, University of British Columbia, Canada, ---.

tables capture metadata, discussion threads, review events, commit authorship, and file-level code changes. A detailed description of all tables is available in the dataset documentation.[1]

## 3 Methods

We analyze PR-level automation behavior by merging AIDev tables using `pr_id` as the primary linking key. Our methodology is structured around the three research questions.

### 3.1 RQ1: Automation Extent

To quantify human participation and automation within pull requests, we combine information from three AIDev tables: `pr_commits`, `pr_reviews`, and `pr_comments`. Human involvement is detected when a PR contains at least one human-authored commit, one human review, or one human-authored comment.

*Identifying bots.* The review and comment tables contain a `user_type` column that indicates whether an action was performed by a bot. Since `pr_commits` does not include such metadata, we construct a bot list by aggregating all users labeled as bots in the review and comment tables, and we additionally classify accounts that follow GitHub's standard `[bot]` suffix convention as bots.

*Filtering administrative / non-code commits.* Timeline events often include administrative updates unrelated to code changes. To avoid overcounting such commits, we exclude commit identifiers associated with non-code event types such as `closed`, `merged`, `reopened`, `auto_merge_disabled`, `auto_merge_enabled`, `labeled`, `unlabeled`, `milestoned`, `demilestoned`, `assigned`, `unassigned`, `subscribed`, `unsubscribed`, `mentioned`, `referenced`, `user_blocked`, `locked`, and `unlocked`.

After removing bot-authored and non-code-related commits, we construct indicators of human participation at the PR level and assign each pull request an automation level:

- **Level 0:** No human involvement (bot-only PR).
- **Level 1:** Human comments or reviews are present, but no human-authored commits.
- **Level 2:** At least one human-authored commit.

We report the distribution of these three automation levels overall and by project in the full RQ1 analysis (outside the scope of this milestone).

### 3.2 RQ2: Work Type and Complexity

#### 3.2.1 Dataset and Scope

For RQ2, a PR is our unit of analysis. We work with the enriched subset of AIDev for which we have interaction-level labels from RQ1. We exclude merge commits when computing commit statistics, since these commits are created by the platform and do not reflect new work by agents or humans.

#### 3.2.2 Work-Type Labels

Each PR is assigned a work-type label derived from the conventional-commit style titles used by the agents. We group these into several high-level categories:

- `feat` (feature work),
- `fix` (bug fixes),
- `docs` (documentation),
- `refactor`,
- `test`,
- and an `other` bucket for less frequent categories.

[1] See: `https://huggingface.co/datasets/hao-li/AIDev/blob/main/data_table.md`.

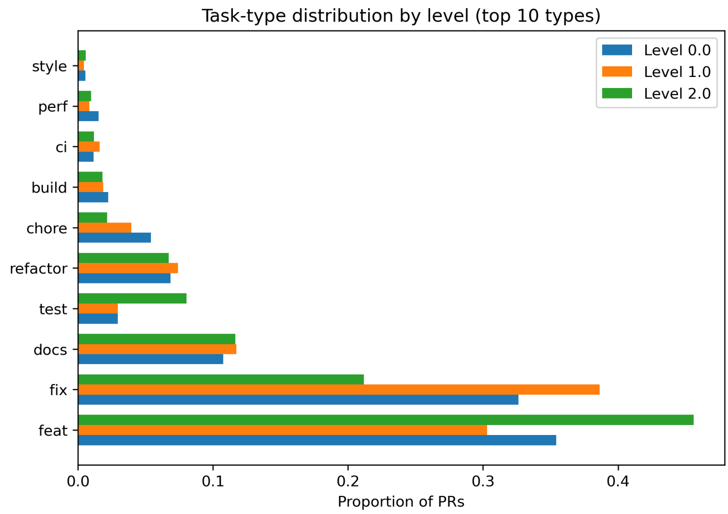


Fig. 1. Task-type distribution by human-interaction level for the most frequent work types.

We then compute, for each interaction level, the distribution of these work types across PRs.

#### 3.2.3 PR Size and Complexity Measures

To approximate PR size and complexity, we aggregate commit-level metadata to the PR level and compute three metrics:

- **Number of commits** (`n_commits`): the count of non-merge commits associated with the PR.
- **Number of files touched** (`n_files`): the number of distinct files that are modified, added, or deleted.
- **Lines changed** (`lines_changed`): the total number of lines added plus lines deleted, or the provided changes field when available.

For each interaction level we summarize these metrics using medians and interquartile ranges and visualize their distributions with boxplots.

### 3.3 RQ3: Differences Across Agents (Planned)

For RQ3, which is beyond the scope of this milestone, we plan to aggregate the interaction levels and size metrics by bot in order to compare how frequently different agents produce fully automated versus human-involved PRs. We only outline this analysis here.

## 4 Results

### 4.1 Work Type and Complexity

#### 4.1.1 Work Types Across Human-Interaction Levels

Figure 1 compares the task-type distributions for the three interaction levels. Across all levels, feature and bug-fix work dominate, but their relative proportions change with the level of human involvement.

Fully automated PRs (level 0) are mainly `feat` and `fix`, which together make up roughly two thirds of PRs. Documentation and refactor work appear but are less frequent. For PRs with human review only (level 1), `fix` becomes more common, while `feat` drops slightly; documentation

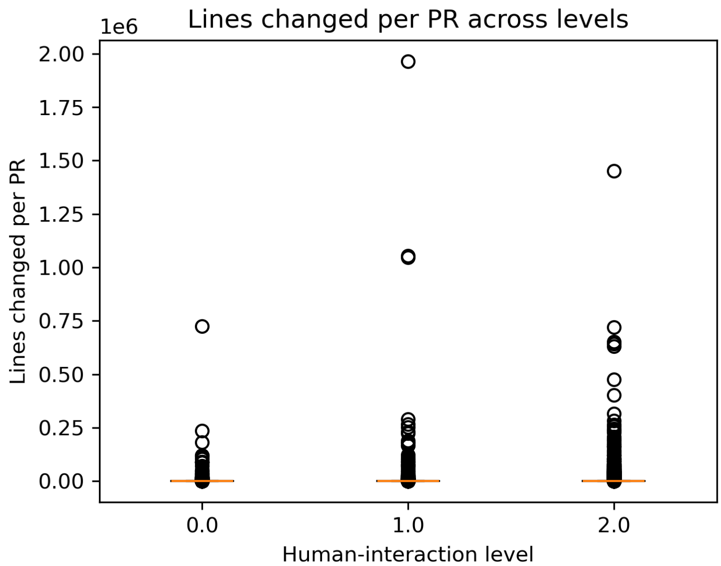


Fig. 2. Lines changed per PR across human-interaction levels.

and refactor work remain important. When humans add commits (level 2), `feat` becomes clearly dominant, whereas `fix`, `docs`, `test`, and `refactor` are present but less frequent. Overall, deeper human involvement is more strongly associated with feature development, while "review-only" involvement is relatively more skewed toward fixes and documentation.

### 4.1.2 Lines Changed per PR

Figure 2 shows the distribution of total lines changed (`lines_changed`) for each interaction level. Fully automated PRs are consistently smaller: they change tens of lines of code on median, with most PRs falling within a relatively narrow range. In contrast, PRs with human involvement tend to be larger and have a wider spread.

PRs with human review only (level 1) show the largest median number of changed lines, suggesting that reviewers are called in when the diff becomes more substantial. Level-2 PRs, where humans add commits, are also larger than fully automated PRs, but their median size is slightly lower than level 1, possibly because some level-2 PRs bundle a targeted human fix into an otherwise agent-generated PR.

### 4.1.3 Number of Files Touched

Figure 3 compares the number of files touched per PR. Fully automated PRs typically modify a small number of files (around two at the median), indicating that agents are mainly used for localized changes. Both level-1 and level-2 PRs tend to touch more files, which is consistent with the idea that multi-file or cross-cutting changes are more likely to require human oversight or direct edits.

### 4.1.4 Number of Commits per PR

Figure 4 shows the number of commits per PR. Fully automated PRs have a median of about two commits, reflecting the way agents often split their work into a small sequence of updates. Level-1

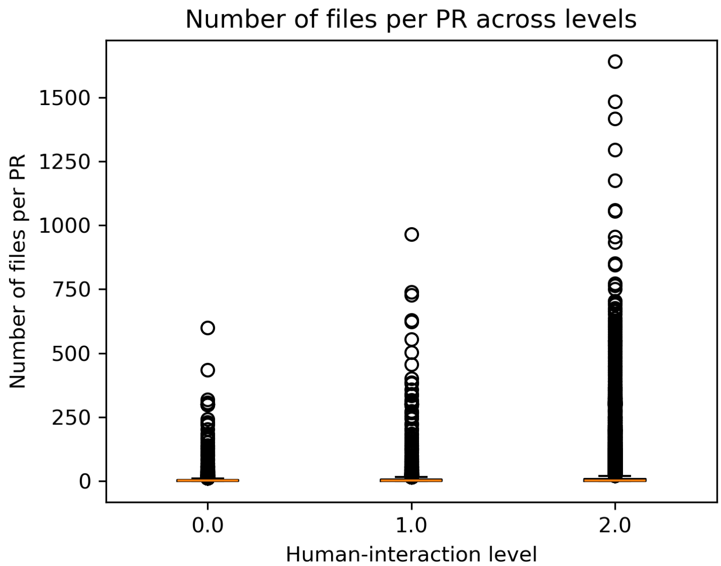


Fig. 3. Number of files touched per PR across human-interaction levels.

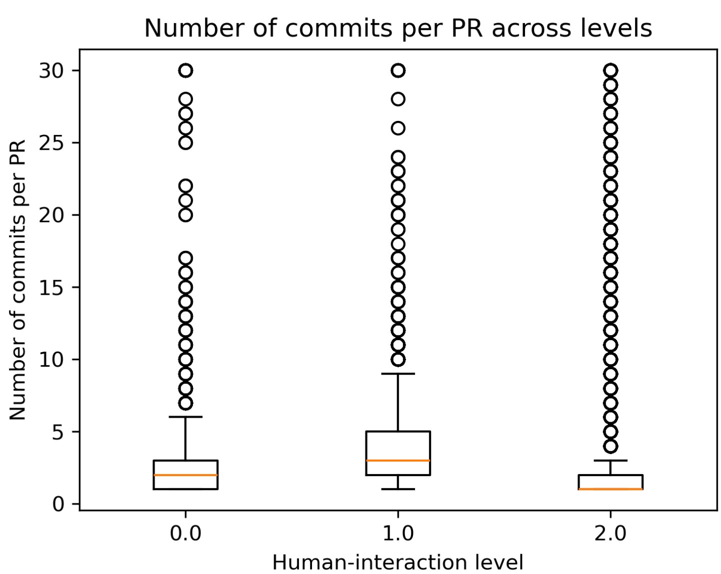


Fig. 4. Number of commits per PR across human-interaction levels.

PRs (with human review only) also tend to have multiple commits, which may represent iterative agent updates in response to review feedback.

Interestingly, level-2 PRs (with human commits) often have fewer commits overall, with a median around one. This pattern suggests that some human-involved PRs are used to apply a focused fix or adjustment on top of an agent-generated change, rather than going through many small iterations.

Across all three interaction levels, feature and bug-fix work dominate, but feature work is especially common in PRs where humans add commits. Fully automated PRs tend to be smaller and more localized: they change fewer lines, touch fewer files, and have modest numbers of commits. Larger or more complex PRs—those that change more lines or span more files—are increasingly likely to involve human review or direct human edits. These patterns suggest that current agentic systems are primarily used for well-scoped changes, while humans remain heavily involved in broader or riskier modifications.

## 5 Discussion

### 5.1 RQ1: Extent of Human Participation

After assigning automation levels, we obtain the following distribution:

| | |
|---|---|
| **Level 0** (bot-only) | 3,269 |
| **Level 1** (human comments/reviews) | 3,635 |
| **Level 2** (human commits) | 26,692 |

These results suggest that fully automated workflows remain rare: only about 10% of PRs have no human interaction, another 10% require human feedback without human-authored commits, and over 75% still require at least one human-authored code contribution.

This pattern is similar to findings from prior small-scale studies examining direct developer interactions with AI coding assistants. While tools such as Copilot and LLM-based generators often provide useful starting points and substantially reduce initial coding or search efforts, developers still spend significant time interpreting, validating, and debugging the generated code. Consequently, human oversight remains essential even when AI systems accelerate early-stage development [1, 4].

### 5.2 RQ2: Work Type and Complexity

Our RQ2 analysis indicates that teams are comfortable letting agents handle small, localized changes end-to-end, but they still rely on human oversight for larger or more complex work. Even when agents generate PRs for important work types such as features and fixes, humans frequently review or directly modify the proposed changes.

Several limitations apply to these findings. First, we rely on interaction-level labels and size metrics derived from metadata; these are proxies and may not fully capture semantic complexity. Second, work-type labels are based on commit titles and may misclassify some PRs. Third, we only study PRs that are part of the AIDev dataset; projects that opt into this dataset may use agents differently from other repositories. Finally, our analysis is cross-sectional and does not model how usage patterns evolve over time.

Despite these limitations, the consistent differences in work types and size distributions across interaction levels provide initial evidence that automation is concentrated on smaller changes, while humans remain central for larger and more complex PRs.

## Team Roles and Contributions

**Phan Truong Phuoc Nguyen.** Designed and implemented the RQ1 pipeline to label PRs with human-interaction levels; prepared the interaction-level dataset used as input for RQ2.
**Sage Yang.** Led the RQ2 analysis on work types and PR size. Implemented the aggregation of task types and size metrics, produced the RQ2 visualizations, and drafted the RQ2 methodology, results, and discussion sections. Organized the structure of this milestone report.

## Acknowledgments

We thank the AIDev dataset authors for collecting and releasing the data used in this study.

## References


[1] H. Barke, J. Alber, and S. Baltes. 2023. Grounded Copilot: How Programmers Interact with Code-Generating Models. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. ACM.

[2] Hao Li. 2023. AIDev: Agentic Pull Requests Dataset. https://huggingface.co/datasets/hao-li/AIDev. Accessed 2025-01-01.

[3] Hao Li. 2023. The Rise of AI Teammates in Software Engineering (SE 3.0): How Autonomous Coding Agents Are Reshaping Software Engineering. arXiv preprint arXiv:2307.XXXX (2023).

[4] D. Vaithilingam and R. Miller. 2023. Expectation vs. Experience: Evaluating the Usability of Code Generation Tools Powered by Large Language Models. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. ACM.

[5] Hao Li, Haoxiang Zhang, and Ahmed E. Hassan. 2025. AIDev: Studying AI Coding Agents on GitHub. Preprint and dataset.