DATA 542 Project - AI Coding Agents Analysis - Milestone 2

XUANRUI QIU, The University of British Columbia, Canada
JINGTAO YANG, The University of British Columbia, Canada

This is the mid-project report for milestone 2. This summarizes our progress on analyzing AI-generated Pull Requests using the AIDev dataset. We have completed work for two research questions, focusing on patch characteristics and semantic consistency between PR descriptions and code changes. We describe the data cleaning and wrangling steps, present preliminary statistical findings, provide visualizations, and outline the remaining work for Milestone 3.

Project GitHub Repo: https://github.com/andrewyang0620/DATA_542_Project_Group_1.git

1 Introduction

In our study, we analyzed the AIDev dataset uploaded by hao-li, which contains PR-level and file-level details of AI-generated contributions across several open-source repositories. This dataset allows us to compare different AI coding agents (Copilot, Codex, Devin, Cursor, Claude Code) in terms of patch size, structure, and semantic consistency.

In milestone 2, we have completed progress on two of our research questions. Specifically, we have finished the full data cleaning and preprocessing steps, constructed patch-level and semantic-consistency features, and conducted statistical analyses for RQ1 and RQ2. These results allow us to characterize differences in patch scale across AI agents and to evaluate how semantic alignment influences merge outcomes. The remainder of this report presents our methodology, data wrangling procedures, preliminary findings, and the remaining work planned for milestone 3.

2 Research Questions

Here are our research questions (refined from milestone 1) :

- **RQ1: Patch Characteristics.** How do Agentic-PRs change code, and what are the different characteristics (e.g., additions, deletions, files touched)?
- **RQ2: Semantic Consistency.** How consistent are PR descriptions with the actual code changes?
- **RQ3: File-Type Modification Patterns Across AI Agents.** How does the distribution of modified file types (e.g., production code vs. test files) characterize the structural composition of patches produced by different AI agents? (Not yet complete for milestone 2)

3 Methodology

3.1 Data Extraction and Joining

We utilize three tables from the AIDev dataset: all_pull_request, pull_request, and pr_commit_details. We select the required columns for analysis. The selected ones are: **pull request metadata** (PR identifier, title, body, state), **agent labels and status** (agent type and merged_at timestamp), and **commit-level diffs** (filename, additions, deletions, and patch text).

The PR-level tables are joined on id, and commit-level details are joined with pull_request.id = pr_commit_details.pr_id.

3.2 Data Cleaning and Wrangling

We apply a multi-step preprocessing pipeline, with data wrangling techniques:

Authors' Contact Information: Xuanrui Qiu, The University of British Columbia, Canada; Jingtao Yang, The University of British Columbia, Canada.

- **Missing values:** PR bodies with null or empty are replaced with ‘No description’, and a binary variable has_description is constructed.
- **Invalid values:** Rows with missing filenames or missing additions or deletions are removed from pr_commit_details.
- **Consistency:** Only commit-detail rows whose pr_id appears in the main PR table are retained. Duplicate PR identifiers are dropped.
- **Patch metric construction:** For each PR, we compute the sum of additions, sum of deletions, and the number of unique files touched.
- **Outliers:** For each patch metric, we apply an IQR-based rule ($Q3 + 3 \times IQR$) to flag extreme values. Outliers are retained for now. However, they are notified and printed out

3.3 Methodology for RQ1 - Patch Characteristics

We analyze three metrics: total additions, total deletions, and unique files changed. We aggregate file-level differents for every PR to get a PR-level view of patch size. We merging patch metrics with each of the agent labels (Copilot, Devin, OpenAI_Codex, Cursor, Claude_Code). We plot the distribution of each metric across agents using boxplots for visualization to identify patterns in patch size and file change breadth. Because metrics are highly non-normal, we apply the non-parametric Kruskal–Wallis H-test (p < 0.05) to assess whether agents differ significantly in patch magnitude.

3.4 Methodology for RQ2 - Semantic Consistency

We check whether the description of a PR (title + body) aligns with the code changes included in the patch. For each PR, we create a natural-language description by combining the title and body together, and a unified code representation by combining all patch associated with the PR. we calculate the cosine similarity between the TF-IDF vectors of the PR description (title + body) and the connected code patch. We use the Mann-Whitney U test to assess if higher consistency correlates with merge success.

3.5 Methodology for RQ3 - File Type Distribution

We plan to analyze the types of files modified by each AI agent to understand their structural patch profiles. We will apply simple heuristic rules to classify files into production code, test files, and configuration, and compute proportions. Then, we will compare the distribution of patch types across agents and apply a chi-square test to evaluate whether file-type distributions differ significantly among the 5 AI agents.

4 Results (From Compiling the Code)

4.1 Research Question 1

We analyzed patch size characteristics across 33,105 pull requests, summarizing total additions, deletions, and the number of unique files touched per PR. The Kruskal–Wallis H-test indicates statistically significant differences across agents for all three metrics. Specifically, additions show $H = 1608.00$ with p near zero; deletions show $H = 2555.29$ with p near zero; and unique files touched show $H = 686.68$ with $p = 2.67 \times 10^{-147}$. The distribution of these metrics for each agent is visualized in Figure 1. The total time used for running and getting the results is 27.93 seconds.

4.2 Research Question 2

For RQ2, we analyzed semantic similarity between PR descriptions and their corresponding patches using TF–IDF cosine similarity. After merging pull requests with their aggregated patch content,

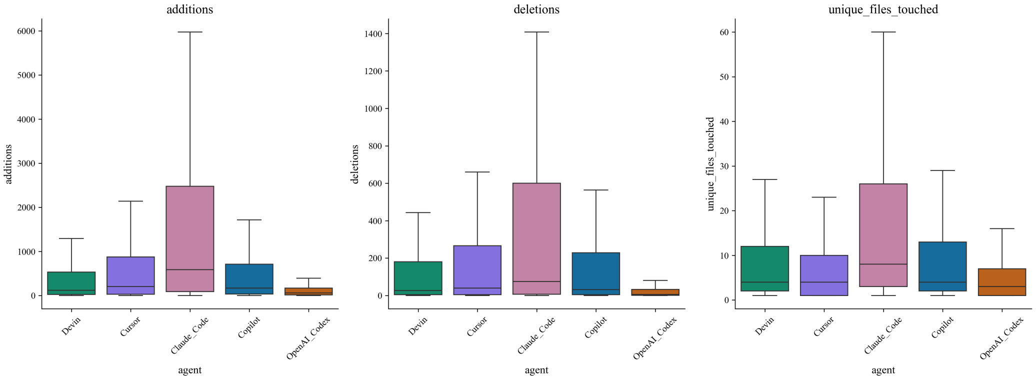

Fig. 1. Distribution of additions, deletions, and unique files touched across AI agents.

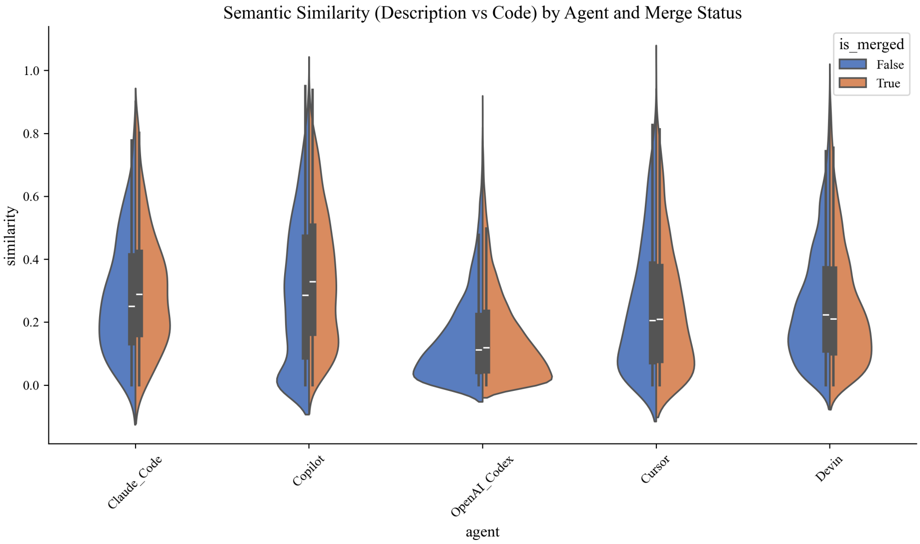

Fig. 2. Semantic similarity between PR description and code patch, grouped by merge status.

we conducted Mann–Whitney U tests to compare similarity distributions between merged and unmerged PRs for each agent.

The results show that Copilot has a highly significant difference between merged and unmerged PRs ($p = 2.27 \times 10^{-17}$), while OpenAI_Codex also has a weaker but significant difference ($p = 0.0235$). For Claude_Code, Cursor, and Devin, the differences are not statistically significant, with $p = 0.1318$, $p = 0.9048$, and $p = 0.1374$. The distribution of similarity scores for different agents and merge statuses is shown in Figure 2. The total running time is 235.70 seconds.

5 Result Interpretation and Discussion

5.1 Research Question 1

The results from RQ1 shows strong differences in patch magnitude for all the five AI agents. Copilot is more likely to produces small patches, which aligns with its character as an assistant-style tool designed for addition types of edits. Devin and OpenAI Codex generate relatively larger patches, which span many more files. This indicates that these two agents operate in a larger scale, which has a more system-level mode. Cursor and Claude Code's output fall between these two , producing moderate patch sizes.

These findings shows that agentic systems do not simply scale up assistant-style behavior. They behave qualitatively differently, frequently modifying multiple modules at once. This finding is directly relevant to RQ1.

For our current assessment, a limitation might be that the size of the patch does not reflect the quality of the patch. Large patches may reflect unnecessary over-editing.

5.2 Research Question 2

The results from RQ2 shows a more significant difference on how description–patch similarity relates to merge outcomes for different AI agents. Copilot shows a strong positive association between context similarity and merge success, showing that the textual explanations from reviews is very important when evaluating small, localized edits. OpenAI Codex also shows this effect, only not as significant. However, for Devin, Cursor, and Claude Code, similarity has no significant relationship with merge outcomes. This suggests that for agents producing large, multi-file patches, reviewers may not rely on the textual description as a primary decision standard.

These findings highlights a review dynamic difference: assistant tools are evaluated through textual clarity, while agent-style systems are evaluated through functional validation.

Limitations include the limited reliance on TF–IDF similarity, which TF-IDF captures surface-level lexical overlap but may underestimate semantic alignment for complex patches or for descriptions written at a higher abstraction level.

6 Future Work

For the next step, we will complete RQ3 by analyzing file-type distributions across agents, following the RQ3 methodology mentioned previously in the report. We will then summarize our findings to answer the research questions in a more integrated manner, implement further discussions and conduct conclusions.