

AgentLogs: A Dataset for Opening the Black Box of Cloud Agents on GitHub

Jonan Richards*, Kosei Horikawa[†], Youmei Fan[†], Yutaro Kashiwa[†] and Mairieli Wessel*

*Radboud University, the Netherlands, [†]Nara Institute of Science and Technology, Japan

Abstract—Coding agents now complete development tasks in real software projects and submit the results as pull requests. A pull request shows the outcome of an agent’s execution but not the process behind it, such as which files the agent read, which commands it ran, or how many tokens it consumed. This is recorded only in the agent’s log, and no existing dataset provides such logs of real-world use, including failed agent executions, across a wide range of repositories. We introduce **AgentLogs**, a large-scale dataset of activity on GitHub’s cloud agent platform on public repositories, collected without requiring opt-in from developers. **AgentLogs** comprises 363,624 tasks and 635,886 sessions, each session an execution of the agent, in 40,565 popular public repositories, with 75,631,136 log entries that record each session step by step, including prompts, intermediate reasoning, tool calls (e.g., file edits, git operations, and GitHub interactions), and token usage, together with the linked pull requests, timelines, workflow runs, and users. Sessions are included regardless of their outcome. **AgentLogs** enables research on agent behavior and failure modes, efficiency and cost, task formulation, human-agent collaboration, and repository adoption. A demonstration video can be found at: <https://youtu.be/ZnFpjuK9Azg>

Index Terms—Coding agents, GitHub Copilot, cloud agents, agent logs, mining software repositories, dataset

I. INTRODUCTION

Coding agents are now deployed on real software projects, where they complete development tasks and submit the results as pull requests [1]. A developer assigns an issue or writes a short request, and the agent proceeds autonomously until it opens or updates a pull request. Each session, one execution of the agent, produces two records. The pull request holds the result: a diff, a description, and any discussion that follows. It does not show which files the agent read before editing, which commands it executed, whether it ran the tests and how it responded when they failed, or how many tokens it consumed before completing or abandoning the task. All of this is recorded only in the agent’s log. Questions such as why a session resulted in a closed pull request, or what a merged one cost, must therefore be answered from the log rather than from the pull request [2].

Existing datasets do not provide logs of agents in real-world use. AIDev collects the pull requests that agents open on public GitHub repositories, together with reviews and comments, but not the logs that produced them [1]. TraceLab collects logs of developers’ local coding agents but removes prompts, tool inputs, and agent output before release [3]. Logs of agent executions on benchmarks [4]–[7] are complete, but they are recorded on repositories and tasks chosen and executed by the benchmark authors rather than on tasks that developers

assigned to an agent. SWE-chat [8] is closest to our work, as it records full logs from developers’ coding agent use, but it requires developers to install a logging tool and commit its checkpoints, which limits coverage to a few hundred repositories and biases the sample toward successful outcomes.

AgentLogs provides such logs at scale. A developer gives a cloud agent on GitHub’s cloud agent platform a task by assigning an issue, mentioning the agent in a pull request comment, or writing a prompt. The platform runs the Copilot cloud agent and third-party agents such as Claude and Codex, although nearly all collected sessions are the Copilot cloud agent (Section V-D). Each session is one execution of the agent on that task, in a fresh sandbox (Section II). On public repositories, the platform exposes every task, session, and log on the Agents tab, which allowed us to collect them without requiring any opt-in from developers. The dataset comprises 363,624 tasks and 635,886 sessions on 40,565 popular repositories, with 75,631,136 log entries. A session log contains the prompt, the agent’s reasoning text, every tool call with its arguments and result, and the token usage of each step. The 28,487,431 tool calls cover file views and edits, shell commands, git operations, and comments posted on GitHub. Sessions are included regardless of outcome, including those that ended in an error or whose pull request was never merged. Each session is linked to its pull request and the pull request’s timeline events, to the GitHub Actions workflow run that executed it, and to the users involved (Table I).

With **AgentLogs**, a researcher can inspect any individual session to see the developer’s request, the agent’s actions step by step, and the resulting outcome, or aggregate over hundreds of thousands of sessions. This supports studies of agent behavior and failure modes, of efficiency and cost from per-step token usage, of how developers formulate tasks and respond to results, and of which repositories adopt GitHub’s cloud agent platform. We expect the dataset to serve empirical software engineering researchers who study a development process in which agents now participate, as well as researchers and builders of coding agents who seek to understand how agents are used, how they operate, and where they fail.

II. GITHUB’S CLOUD AGENT PLATFORM

GitHub’s cloud agent platform runs agents asynchronously in an ephemeral development environment, where they search and edit files, run shell commands such as tests and git, and use GitHub, for example, to open a pull request or reply to a

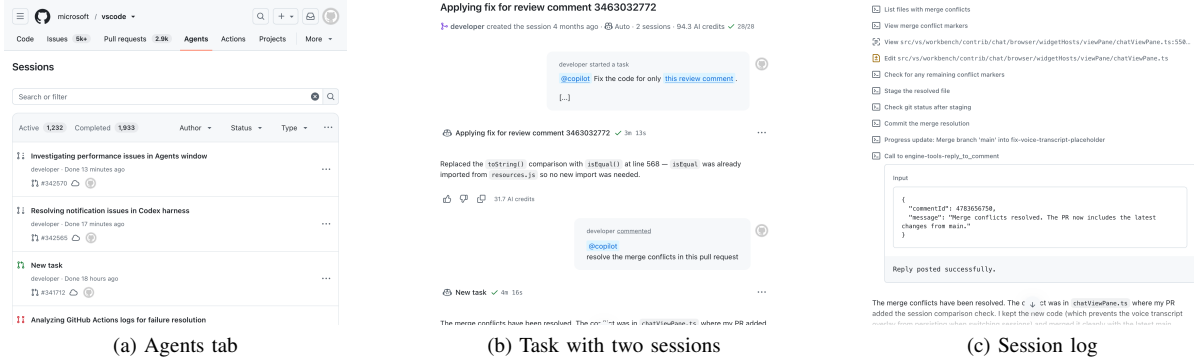


Fig. 1. A Copilot cloud agent task on microsoft/vscode. Usernames and avatars are redacted. (a) The Agents tab lists recent tasks. (b) A task with two sessions: a review-comment fix, then a request to resolve the resulting merge conflicts. (c) Part of the log of the second session.

comment.¹ Developers can run the Copilot cloud agent or a third-party agent such as Claude or Codex (Section V-D).

Developers can assign a cloud agent a *task*, which typically works on a single branch and a single pull request. That assignment initiates its first *session*, and later requests from the developer add further sessions on the same task (Figure 1b), where each session is one execution of the agent in a fresh sandbox environment. These sessions can be initiated in several ways (Section V-A), e.g., from the Agents tab (Figure 1a), by mentioning the agent, for example @copilot, on a pull request, or from an integration such as Slack or Jira. The *log* of a session is the transcript of its execution, shown in an abridged form in the GitHub web interface (Figure 1c). AgentLogs stores each log entry, including prompts, reasoning traces, tool calls and arguments (Section V-C), and token usage.

III. RELATED WORK

AIDev collects pull requests that local and remote agents such as Codex, Copilot, and Claude Code opened on public GitHub repositories, with their comments and reviews [1]. AIDev has been used to study agents’ real-world outcomes [9] and how developers interact with agents [10], but cannot show how the agent worked to go from prompt to outcome. DevGPT links ChatGPT conversations that developers shared on GitHub and Hacker News to the artifacts they discuss [11], but these are chats rather than agent executions with tool calls.

Benchmark datasets record that process as logs of agent executions on fixed tasks. Some take tasks from existing GitHub issues [4], [5], others synthesize them [6], [7]. Such logs have been used to study how an agent works through a known task [2], but do not reflect how developers use agents, because the benchmark authors chose the repository and task, and started the agent themselves.

In other studies of developers using coding agents, a conversation is often called a session, analogous to a task in AgentLogs. A turn or prompt in that conversation is what we refer to as a session. TraceLab released 4,265 conversations with about 39,200 turns from 43 developers, but

removed prompts, tool input, and agent output [3]. Microsoft researchers characterized logs of 13.5 million conversations with 95.1 million turns in production, but reported only aggregates, without prompts, tool input, or agent output, and did not release their logs [12]. Closest to AgentLogs is SWE-chat, which includes logs for nearly 18,000 conversations with more than 229,000 user prompts in 707 repositories, together with line-level authorship of the committed code [8]. It only covers developers who installed Entire, a local logging tool, and committed its checkpoints.

AgentLogs records public activity on GitHub’s cloud agent platform without additional opt-in or a committed checkpoint, reducing bias toward successful sessions. Compared to SWE-chat, it collected agent logs from $50.4\times$ more repositories, with $20.2\times$ more tasks than SWE-chat’s sessions and $2.8\times$ more sessions than its user prompts.

IV. THE AGENTLOGS DATASET

A. Data collection

We collected sessions on GitHub’s cloud agent platform on public repositories. These sessions execute on GitHub’s infrastructure, and each task, prompt, and log is already visible on the repository’s Agents tab. Locally run sessions and private repositories are not collected.

The seed is SEART GitHub Search Engine (GHS) [13], a dataset of 1,997,424 public repositories with at least 10 stars, accessed on September 28, 2026. For each repository we listed agent tasks, including archived tasks, with the GitHub REST API. The documented REST task omits fields such as session trigger and workflow run, so tasks and sessions were fetched from an undocumented endpoint of the GitHub Copilot API.² The GitHub Acceptable Use Policies explicitly allow any method of obtaining publicly available data.³ Session logs come from the same Copilot API as a server-sent event stream. More than 99.99% of events parsed as JSON; the rest were kept as raw strings. We also collected the sessions’ workflow runs, the pull requests of tasks and sessions with their timeline

¹<https://docs.github.com/en/copilot/responsible-use/agents>

²<https://api.github.com/copilot/v1/sessions>

³<https://docs.github.com/tos>

TABLE I
SUMMARY OF THE AGENTLOGS DATASET

	# Records	Size	Table	Content
Repositories – 2.18% with agent tasks	1,861,300 40,565	462.7 MB	<code>repositories</code>	Public GitHub repositories with at least 10 stars (incl. name, size, language, stars, forks, labels, topics).
Agent tasks	363,624	83.7 MB	<code>agent_tasks</code>	Agent assignments on a repository (incl. creator, name, state, sessions, branches, pull requests).
Agent sessions	635,886	243.1 MB	<code>agent_sessions</code>	Executions of an agent on a task (incl. user, triggering event, model, prompt, usage, workflow run, pull request).
Log entries	75,631,136	63.0 GB	<code>agent_session_logs</code>	Session log events: chat-completion chunks (incl. reasoning, tool calls, token usage) and user and tool messages.
Pull requests	317,878	483.9 MB	<code>pull_requests</code>	Pull requests worked on in tasks and sessions (incl. title, author, state, branches, change size).
Pull request timeline events	9,070,939	327.4 MB	<code>pull_request_timelines</code>	Timeline events for a pull request (incl. event type, actor, label, commit SHA, message).
Workflow runs	554,496	83.3 MB	<code>workflow_runs</code>	GitHub Actions workflow runs that executed an agent session (incl. status, conclusion, triggering user).
Users	857,428	211.5 MB	<code>users</code>	Users, organizations, and bots related to the other tables (incl. GitHub ID, login slug, creation date).
Total	89,292,687	64.9 GB		

events, and all linked users. As data minimization, user records kept only the login, GitHub id, and account creation date, not profile fields such as name, email, or bio.

Repositories no longer available through the REST task endpoint, e.g., after deletion, renaming, or being made private, are left out. Repositories with no tasks are kept. Tasks, pull requests, workflow runs, timeline events, and users that the API does not return are left out as well, although other tables may still refer to them. A session whose log is not found is kept, with no log entries.

B. Internal structure

AgentLogs is a set of Parquet tables (Table I), sharded into chunks of ~80–120MB so each table can be downloaded and scanned in parallel. Records store the IDs of related rows.

Repository rows are the SEART GHS seed data, with links to their tasks. Tasks and sessions largely match the documented REST task resource. Pull requests, workflow runs, timeline events, and users follow their GitHub REST schema. Log entries are a tool result, user message, or an OpenAI-format chat-completion chunk with assistant text, reasoning, tool calls (Section V-C), and token usage, plus fields such as billing, warnings, and tool telemetry. The schema documentation (Section IV-C) maps each column to its source field.

C. How to access

The AgentLogs dataset is available on Hugging Face.⁴ The data collection pipeline, schema documentation, a data sample (4 tasks, 4 sessions, 458 log entries), and example notebooks that query the dataset with DuckDB, Polars, or a low-memory typed streaming reader are available on GitHub.⁵ Python type definitions for every table are available as the PyPI package `risenlab-agentlogs`. Code is licensed under MIT and the dataset under CC BY 4.0. The released pipeline

allows re-collection, and we plan to publish updated snapshots on a best-effort basis.

V. PRELIMINARY ANALYSIS

A. Initiating a session

Almost all sessions (99.9%) record their trigger in the `event` field. Sessions can be started on a pull request, for example by mentioning `@copilot` in a comment (29.5% of sessions) or a review (12.8%), by requesting a Copilot code review (5.6%), or by requesting fixes for review feedback (1.6%). Developers can also enter a prompt, for example in the GitHub web interface (15.9%), the mobile app (2.5%), an IDE (0.6%), or the CLI (0.2%). Developers can assign an issue (18.0%), including on a tracker outside GitHub. Some sessions are initiated through the GitHub API, Copilot chat, or an integration such as Slack, Teams, Sentry, or an MCP tool (10.8%), others are triggered by a failed GitHub Actions workflow run (1.9%), or by a security, quality, or coverage alert (0.3%). The remaining sessions (0.1%) have a rare trigger, such as a scheduled workflow run.

B. Sessions over time

Figure 2 shows sessions created per day, from the earliest session on May 11, 2025, to the cutoff on September 29, 2026. The spike is `review-request` sessions: from November 24 through December 8, 35,080 of 54,299 sessions have that event type, and the type is rare outside that period. Copilot code review remains a feature on GitHub, suggesting that GitHub only briefly recorded review requests as cloud agent sessions. Disregarding this spike, sessions per day rose to a peak of about 3,000 in early April 2026 and then declined to about 840 per day by September.

C. Tool calls

One log entry can contain several tool calls. AgentLogs contains 28,487,431 tool calls, for an average of 45.6 calls per

⁴<https://huggingface.co/datasets/risenlab/agentlogs>

⁵<https://github.com/risenlab/agentlogs>

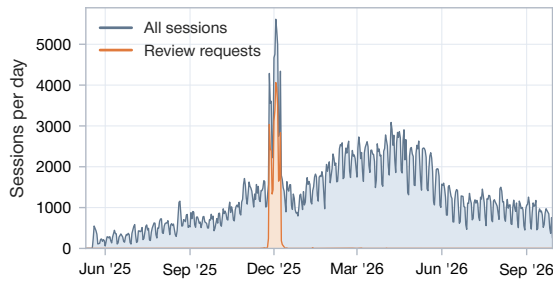


Fig. 2. Agent sessions per day. Blue shows all sessions and orange shows the share of sessions with a `review-request` event type.

session with a non-empty log. File operations such as `view`, `edit`, and `grep` make up most of the tool calls (39.9%), together with shell interaction (e.g. `bash` and `read_bash`; 37.4%). Also common are tool calls related to setup and progress (14.2%), operations on GitHub such as commenting and reading issues (3.8%), and code review and scanning tools (2.1%). The rest include subagent delegation, planning, memory, and browser interaction.

D. Agents and models

The workflow-run user, not the task, records which agent ran. Of 554,496 workflow runs, 98.8% are `copilot-swe-agent`, 0.8% `anthropic-code-agent`, and 0.2% `openai-code-agent`; every other account, including third-party agents such as Bright Security, is at most 0.1%.

Each session records the model that ran. Claude accounts for 56.4% of sessions and GPT for 17.5%, with the most common versions being `claude-sonnet-4.6` (25.9% of sessions), `claude-sonnet-4.5` (17.4%), and `gpt-5.3-codex` (9.5%). MAI, Gemini, Kimi, Grok, and other models each account for 0.1% of sessions or less. The model is unrecorded for 25.3% of sessions.

VI. DISCUSSION

The preliminary analysis shows the scale and variety of the activity in AgentLogs from May 2025 through September 2026, with sessions triggered by pull requests, issues, prompts, integrations, and other events, and 28,487,431 tool calls spanning file operations, shell commands, GitHub interactions, code review, and subagents.

However, AgentLogs has several limitations. Although third-party agents can run on GitHub’s cloud agent platform, nearly all sessions are the Copilot cloud agent, so comparisons across sessions are concentrated on one agent. The model is unrecorded for 25.3% of sessions, reducing the number of sessions useful for comparisons that split by model.

Session context, such as a custom Markdown instruction file, the triggering text, or a pull request comment thread, is often not stored on the session record. Commits are not collected, and timeline events are not tied to sessions or log entries. Session context, commits, and timeline events often

appear in the log instead, as a prompt to the agent or as a tool call, but may be hard to extract reliably.

Still, no other released dataset includes the prompts, reasoning, and tool calls of developer-initiated sessions, including sessions that fail, across public repositories at this scale. AgentLogs uniquely enables researchers to answer questions about agent behavior, such as which patterns agents follow and how those patterns differ between successful and failed sessions, about collaboration in the GitHub ecosystem, such as how developers initiate sessions and how triggers and repository context relate to the agent’s execution and outputs, and about repository adoption, such as what characterizes the 2.18% of popular repositories that adopt the platform, compared to those that do not.

REFERENCES

- [1] H. Li, H. Zhang, and A. E. Hassan, “AIDev: Studying AI Coding Agents on GitHub,” in *Proceedings of the 23rd International Conference on Mining Software Repositories*, ser. MSR ’26. New York, NY, USA: Association for Computing Machinery, Jul. 2026, pp. 1029–1033.
- [2] I. Bouzenia and M. Pradel, “Understanding Software Engineering Agents: A Study of Thought-Action-Result Trajectories,” in *2025 40th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, Nov. 2025, pp. 2846–2857.
- [3] K. Zhu, M. Jacob, C. Ma, Y. Pan, S. Wang, A. Krishnamurthy, and B. Kasikci, “TraceLab: Characterizing Coding Agent Workloads for LLM Serving,” <https://arxiv.org/abs/2606.30560v2>, Jun. 2026.
- [4] J. Pan, X. Wang, G. Neubig, N. Jaitly, H. Ji, A. Suhr, and Y. Zhang, “Training Software Engineering Agents and Verifiers with SWE-Gym,” arXiv:2412.21139, Jun. 2025.
- [5] I. Badertdinov, A. Golubev, M. Nekrashevich, A. Shevtsov, S. Karasik, A. Andriushchenko, M. Trofimova, D. Litvinseva, and B. Yangel, “SWE-rebench: An Automated Pipeline for Task Collection and Decontaminated Evaluation of Software Engineering Agents,” *Advances in Neural Information Processing Systems*, vol. 38, Apr. 2026.
- [6] J. Yang, K. Lieret, C. Jimenez, A. Wettig, K. Khandpur, Y. Zhang, B. Hui, O. Press, L. Schmidt, and D. Yang, “SWE-smith: Scaling Data for Software Engineering Agents,” *Advances in Neural Information Processing Systems*, vol. 38, Apr. 2026.
- [7] N. Jain, J. Singh, M. Shetty, L. Zheng, K. Sen, and I. Stoica, “R2E-Gym: Procedural Environments and Hybrid Verifiers for Scaling Open-Weights SWE Agents,” arXiv:2504.07164, Apr. 2025.
- [8] J. Baumann, V. Padmakumar, X. Li, J. Yang, D. Yang, and S. Koyejo, “SWE-chat: Coding Agent Interactions From Real Users in the Wild,” Oct. 2026.
- [9] R. Ehsani, S. Pathak, S. Rawal, A. Al Mujahid, M. M. Imran, and P. Chatterjee, “Where Do AI Coding Agents Fail? An Empirical Study of Failed Agentic Pull Requests in GitHub,” in *Proceedings of the 23rd International Conference on Mining Software Repositories*. New York, NY, USA: Association for Computing Machinery, Jul. 2026, pp. 807–811.
- [10] P. Rooijendijk, C. Treude, and M. Wessel, “Who Said CVE? How Vulnerability Identifiers Are Mentioned by Humans, Bots, and Agents in Pull Requests,” in *Proceedings of the 23rd International Conference on Mining Software Repositories*, ser. MSR ’26. New York, NY, USA: Association for Computing Machinery, Jul. 2026, pp. 994–998.
- [11] T. Xiao, C. Treude, H. Hata, and K. Matsumoto, “DevGPT: Studying developer-ChatGPT conversations,” in *Proceedings of the 21st International Conference on Mining Software Repositories*, ser. MSR ’24. New York, NY, USA: Association for Computing Machinery, 2024, pp. 227–230.
- [12] B. Liu, H. Qiu, Í. Goiri, R. Fonseca, R. Bianchini, and E. Choukse, “Agentic Coding in the Wild: Characterizing GitHub Copilot Traces at Production Scale,” <https://arxiv.org/abs/2608.00101v1>, Jul. 2026.
- [13] O. Dabic, E. Aghajani, and G. Bavota, “Sampling projects in Github for MSR studies,” in *2021 IEEE/ACM 18th International Conference on Mining Software Repositories (MSR)*. IEEE, 2021, pp. 560–564.