Factors Driving Developer Influence and Efficiency in the AI Community

ZIHUAN LUI, University of British Columbia, Canada
AVALVIR KAUR SEKHON, University of British Columbia, Canada

We analyze the AIDev dataset to understand the factors driving developer influence and efficiency in the AI development community. Focusing on user activity, repository metadata, and issue tracking, we investigate the relationship between contribution volume and follower count, predictors of popularity, and factors affecting issue resolution time. Our results indicate that contribution volume has a negligible relationship with influence ($r \approx 0$), while specific tech stack choices (e.g., Java, TypeScript) are strong predictors of popularity. Furthermore, contrary to initial assumptions, we find that engagement levels significantly impact issue resolution time ($p < 0.001$), with engaged issues being resolved faster than those with no activity.

CCS Concepts: • **Human-centered computing** → **Collaborative and social computing**; • **Software and its engineering** → *Software maintenance tools.*

Additional Key Words and Phrases: AIDev, software engineering, developer influence, issue resolution, GitHub analysis

1 Introduction

The rapid growth of Artificial Intelligence (AI) development has fostered a massive community of developers on platforms like GitHub. Understanding what drives influence and efficiency in this specific domain is crucial for both individual developers seeking to grow their impact and organizations aiming to optimize their workflows. This project analyzes the AIDev dataset, focusing on user activity, repository metadata, and issue tracking dabbish2012social. The primary objective is to identify the factors that drive developer influence (popularity) and efficiency (issue resolution) within this specific domain.

2 Research Questions

In this study, we investigate the following three research questions (RQs):

- **RQ1: Contribution Volume vs. Influence.** To what extent does repository creation frequency correlate with follower count, and does this vary by programming language?
- **RQ2: Predictors of Popularity.** Which developer features (account tenure, language, repository count) are the strongest predictors of a user's follower count?
- **RQ3: Factors Affecting Issue Resolution Time.** Does higher engagement (comment volume) affect the time-to-resolution (TTR) for issues?

3 Methodology

This section details the data wrangling and statistical methodology used to address the RQs.

3.1 Data Preprocessing and Wrangling

We utilized Python (Pandas) to clean and merge the AIDev dataset. Key preprocessing steps included:

(1) **Complex Merging:** We performed an inner join between the Users and Repositories tables by extracting the username from the repository `full_name` string (splitting "owner/repo") to match the `login` column.

Authors' Contact Information: Zihuan Lui, University of British Columbia, Kelowna, Canada; Avalvir Kaur Sekhon, University of British Columbia, Kelowna, Canada.

(2) **Issue-Comment Linkage:** We merged the Issues table with aggregated data from the Comments table using `id` and `pr_id` as foreign keys. Crucially, we imputed missing comment counts with 0 to accurately reflect issues with no engagement.
(3) **Time Standardization:** All timestamp columns (`created_at`, `closed_at`) were converted to UTC. We calculated Time-to-Resolution (`TTR_hours`) by subtracting issue creation time from close time, filtering out data errors (negative durations or durations < 0.01 hours) kalliamvakou2016depth.
(4) **Handling Missing Data:** We imputed missing values for `main_language` with "Unknown" and applied One-Hot Encoding to categorical language variables for regression analysis.

3.2 Analysis Approach

- **RQ1:** We aggregated user repository counts and correlated them with follower counts. We applied Spearman's rank correlation to handle the non-normal distribution of follower counts, segmented by the top 5 languages.
- **RQ2:** We constructed a Linear Regression model using features such as `account_age`, `repo_count`, and encoded `language`. We interpreted coefficients to determine feature importance.
- **RQ3:** We categorized issues based on engagement (Has Comments vs. No Comments) using the median split (Median = 0). A Two-sample T-test (Welch's t-test) was used to determine if there is a statistically significant difference in TTR between these groups.

4 Results

The following subsections present the statistical results and visualizations of our analysis.

4.1 RQ1: Contribution Volume vs. Influence

Our analysis using Spearman's rank correlation reveals that the volume of contributions (repository count) has a negligible relationship with user influence (followers) across all major languages. The correlation coefficients are consistently near zero.

The specific coefficients were as follows:

- **Python:** 0.0252
- **Java:** 0.0340
- **TypeScript:** 0.0331
- **JavaScript:** -0.0111
- **HTML:** -0.0152

4.2 RQ2: Predictors of Popularity

The Linear Regression model identified distinct drivers for follower counts.

- **Positive Drivers:** The strongest predictors were specific languages: **Java (+4.27)** and **TypeScript (+3.21)**. Repository count had a moderate positive impact (+3.72).
- **Negative Drivers:** Users primarily associated with PHP (-13.7) and HTML (-6.3) tended to have fewer followers.
- **Neutral Factors:** Account tenure (`account_age_days`) had a coefficient near zero (+0.014).

4.3 RQ3: Factors Affecting Issue Resolution Time

We analyzed the impact of engagement on resolution time. The median comment count for the dataset was 0. Comparing issues with comments (High Engagement) versus those without (Low Engagement), the Two-sample T-test yielded a T-statistic of -18.02 and a p-value of 3.63×10^{-69}.

Since $p < 0.05$, we reject the null hypothesis. Issues with engagement (comments) have a significantly **lower** time-to-resolution (Mean $\approx$ 22 hours) compared to unengaged issues (Mean $\approx$ 2675 hours).

5 Interpretation of Results

5.1 Quantity vs. Quality

The findings from RQ1 suggest that developers cannot simply increase their influence by creating a large volume of repositories. The lack of correlation implies that the community values the quality or utility of a project over sheer quantity. "Empty" contributions do not yield social capital in the AI ecosystem.

5.2 Tech Stack Influence

RQ2 results indicate that tech stack choice outweighs account longevity. The strong positive coefficients for Java and TypeScript suggest these ecosystems currently offer higher visibility. Conversely, the neutral impact of account age suggests that newer developers can gain influence quickly if they contribute to the right ecosystems.

5.3 Efficiency Dynamics

The results from RQ3 provide a compelling insight: engagement accelerates resolution. Issues that attract community discussion are resolved significantly faster than those that remain silent. This suggests that “noise” (comments) in the AI community often represents active collaboration or clarification that aids the maintainer, rather than obstruction or debate that delays the fix. The extremely high TTR for low-engagement issues likely reflects "stale" issues that are ignored by maintainers.

6 Threats to Validity

While our results are statistically significant, several limitations exist:

1. **Language Imputation:** A significant portion of repositories had missing language data, which we imputed as "Unknown." This may obscure relationships for less popular languages.
2. **Causality vs. Correlation:** In RQ3, we cannot confirm directionality. Do comments cause faster resolution, or do easily resolvable issues naturally attract more comments?
3. **Timezone Bias:** While we converted timestamps to UTC, we did not account for developer working hours, which might introduce noise into TTR calculations on a granular level.

7 Conclusion

This study provides empirical evidence on how developers gain influence and resolve issues in the AI community. We conclude that tech stack selection is a more potent driver of popularity than mere longevity or repository volume. Furthermore, we find that community engagement is not a bottleneck but a catalyst for efficiency, drastically reducing issue resolution times.

8 Team Contributions

Our project combined collaborative planning with individual technical responsibilities. During the initial phase, both members participated in defining the research questions, selecting data tables, and designing the data cleaning logic (including Python/Pandas merging strategies).

- **Avalvir Kaur Sekhon:** Responsible for the statistical analysis of developer influence (RQ1 & RQ2).

- RQ1 Analysis: Aggregated repository creation frequency, segmented data by language, and applied Spearman's rank correlation to determine the relationship between contribution volume and influence.
- RQ2 Analysis: Constructed the Linear Regression model using features like account tenure and encoded language variables to identify predictors of popularity.

- **Zihuan Liu:** Responsible for the efficiency analysis (RQ3) and overall documentation.
  - RQ3 Analysis: Categorized issues by engagement level and performed the Two-sample T-test to validate factors affecting resolution time.
  - Report Writing & Formatting: Synthesized analysis results into the full report and managed document formatting to ensure compliance with ACM standards.

9 Project Resources

- **GitHub Repository:** https://github.com/RoxyLiu66/data-wrangling-group-8.git
- **Dataset Source:** AIDev Dataset

10 GenAI Usage Statement

We utilized Generative AI (ChatGPT) to assist in debugging Python syntax for the data merging functions and to refine the wording of the methodology section. All code logic and statistical interpretations were verified manually by the team.

Acknowledgments

We would like to thank the course instructors for their guidance on DATA 542.