NetSciX2026: International School and Conference on Network Science
February 17^{th}-20^{th}, 2026 - Auckland, New Zealand

Using Temporal Collaboration Networks to Understand Activity Cascades in OSS Communities

collaboration networks, open source communities, computational social science

Extended Abstract

Open source software (OSS) development represents one of the most successful forms of large-scale collaborative knowledge in the digital age. Understanding the collective social dynamics that enables thousands of developers to coordinate their actions is fundamental to comprehend how complex software emerges from decentralized collaboration. While existing works focused on individual developer motivation [1] and project governance structures [2], we still lack an understanding of the microscopic social mechanisms that drive interactions in large OSS communities. Co-editing relationships—where one developer modifies code originally authored by another—represent an important type of interactions in software development. These relationships capture not just technical but also social dependencies, including mentorship, code review, collaborative problem-solving, and knowledge transfer. Using data from the public git repositories of major OSS communities, we can model these interactions as temporal collaboration networks. This enables us to analyze how the resulting social structure and dynamics of developer teams influence OSS projects [3]. Going beyond previous works, we use the resulting temporal collaboration networks to study the emergence of *activity cascades*, a phenomenon where the action of developers trigger downstream actions by collaborators. In software development, when developer A edits code originally written by developer B, this action may prompt B to respond with a subsequent commit more quickly than expected. Such "responses" can be driven by various mechanisms: defensive coding to protect one's work, collaborative enhancement of code, mentor-mentee interactions, or increased awareness of project activity. These patterns yield insights into how distributed teams self-organize, how information spreads in collaboration networks, and how individual actions influence collective behavior.

Addressing this issue, we perform an empirical investigation of activity cascades in 51 OSS communities. To establish the statistical significance of observed cascades, we implement a null model that randomly permutes timestamps of co-editing links while preserving the involved developers. This tests whether cascades depend on the temporal ordering of co-edits. For each community, we compare observed cascade rates against distributions generated from 100 realizations of the null model. We calculate p-values by determining the fraction of shuffled instances where observed cascade counts equal or exceed the expected counts.

We use this approach to address two research questions: **RQ1:** Do activity cascades exist in OSS communities? Our empirical analysis reveals statistically significant cascade effects in 28 out of 51 communities (54.9%) when comparing observed cascade counts against the null model. Communities demonstrating the largest cascade count include large-scale repositories such as 'woocommerce' (1741 observed cascades vs. 337 in null model, p = 0.010), 'birt' (2261 vs. 852, p = 0.010), and 'readability' (301 vs. 8, p = 0.010). The results show that observed cascades are not merely random fluctuations but represent genuine social patterns. Apart from significance, we use Cohen's d to quantify effect size, ranging from small ($d < 0.5$) to extremely large ($d > 20$) for many communities. This large effect size shows that activity cascades are an important phenomenon, which has not been previously described.

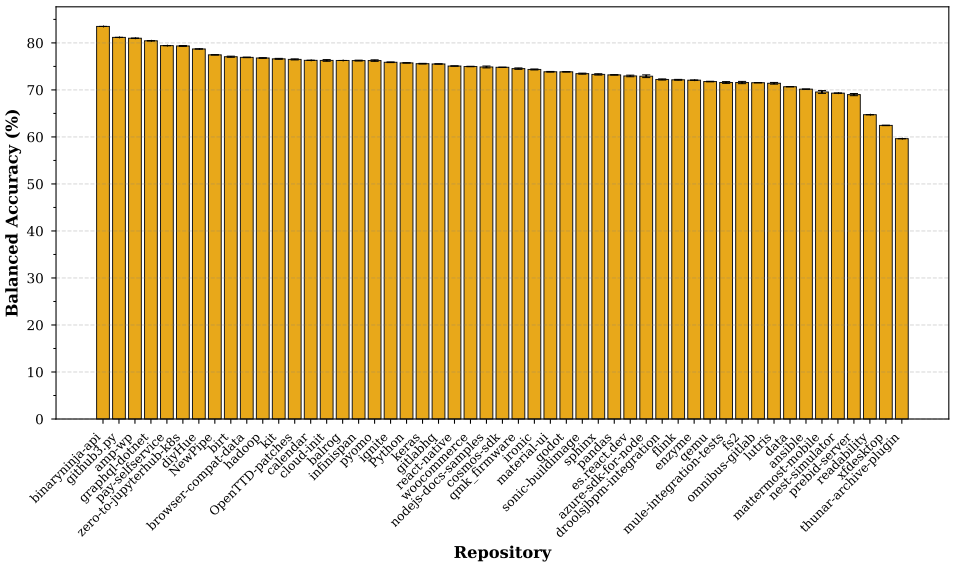

Figure 1: Balanced accuracy for prediction whether developers leave a project across 51 OSS communities. Error bars represent standard deviations across five independent experiments.

RQ2: Can insights into activity cascades help to predict which developers will leave a community? To answer this question, we use a logistic regression model. For each of the 51 OSS communities, we create non-overlapping temporal windows of 12 months. Each time window serves as an independent training instance, where we extract features from the developer collaborations within that year and predict which developers will be inactive in future time windows. A set of *network features* captures positions of developers in the collaboration network. We include node degree, betweenness and closeness centrality, and the distance to and collaboration strength with the project's original founder. Informed by our cascade analysis, we further use *neighbor inactivity features* that capture temporal activity patterns of a developer's collaborators, including minimum, maximum, and mean inactivity time. If a developer's collaborators exhibit long inactivity periods, this may increase their risk to leave the community. Conversely, active neighbors may encourage developers to remain active. Addressing RQ2, the combination of network and cascade-inspired features achieves balanced accuracies ranging from 59.1% to 85.5%, considerably outperforming a random baseline. We further use logistic regression coefficients to quantify feature importance, which reveals maximum inactivity time of neighbors as the most important predictor. This finding affirmatively answers RQ2.

Our work shows that co-editing networks can help us to understand activity cascades in OSS communities. It further provides perspectives for further applications of (temporal) network analysis to gain insights into the collective dynamics of OSS communities. We note that we do not see immediate ethical implications of our work. We exclusively use publicly available data and do not release information that could be used to identify individuals. In our talk, we will however highlight risks from possible misuse of our prediction model.

References

[1] Guido Hertel, Sven Niedner, and Stefanie Herrmann. "Motivation of software developers in Open Source projects: an Internet-based survey of contributors to the Linux kernel". In: *Research Policy* 32.7 (2003), pp. 1159–1177.

[2] Maha Shaikh and Ola Henfridsson. "Governing open source software through coordination processes". In: *Information and Organization* 27.2 (2017), pp. 116–135.

[3] Christoph Gote, Ingo Scholtes, and Frank Schweitzer. "git2net-mining time-stamped co-editing networks from large git repositories". In: *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*. IEEE. 2019, pp. 433–444.