(will be inserted by the editor)

Ecosystem-wide influences on pull request decisions: insights from NPM

Willem Meijer \cdot Mirela Riveni \cdot Ayushi Rastogi

Received: date / Accepted: date

Abstract The pull-based development model facilitates global collaboration within open-source software projects. Most research on the pull request decision-making process explored factors within projects, not the broader software ecosystem they comprise. We uncover ecosystem-wide factors that influence pull request acceptance decisions. We collected a dataset of approximately 1.8 million pull requests and 2.1 million issues from 20,052 GitHub projects within the NPM ecosystem. Of these, 98% depend on another project in the dataset, enabling studying collaboration across dependent projects. We employed social network analysis to create a collaboration network in the ecosystem, and mixed effects logistic regression and random forest techniques to measure the impact and predictive strength of the tested features. We find that gaining experience within the software ecosystem through active participation in issue-tracking

This manuscript is a revision of the first author's master's thesis (Meijer, 2023), supervised and graded by the second and third authors. This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

Willem Meijer University of Groningen / Linköping University Groningen, The Netherlands / Linköping, Sweden E-mail: willem.meijer@liu.se

Mirela Riveni University of Groningen Groningen, The Netherlands E-mail: m.riveni@rug.nl 0000-0002-4991-3455

(D) 0000-0001-8482-3917

Ayushi Rastogi University of Groningen Groningen, The Netherlands F-mail: a rastogi@rug.nl

(D) 0000-0002-0939-6887

systems, submitting pull requests, and collaborating with pull request integrators and experienced developers benefits all open-source contributors, especially project newcomers. The results show that combining ecosystem-wide factors with features studied in previous work to predict the outcome of pull requests reached an overall F1 score of 0.92.

Keywords open-source software ecosystem \cdot social coding platform \cdot software package \cdot software dependency \cdot project newcomer \cdot collaborative software engineering

1 Introduction

Software ecosystems are crucial in modern open-source software as very few projects exist in complete isolation because they depend on other projects (Decan et al., 2019). Open-source software ecosystems refer to "a network of open-source software communities working on a common technology" (Franco-Bedoya et al., 2017; Hanssen and Dybå, 2012). The NPM package library is an example of this, containing over two million JavaScript projects.

Since projects in an ecosystem share a common technology, it is possible to transfer knowledge acquired in one project to another. This has been reported to affect project downloads (Fershtman and Gandal, 2011; Méndez-Durón and García, 2009; Peng et al., 2013) and increase project longevity (Qiu et al., 2019; Valiev et al., 2018; Wang, 2012; Wattanakriengkrai et al., 2023). Consequently, understanding how knowledge acquired in the broader ecosystem affects its comprising projects is paramount, as this can improve collaboration between projects, benefiting the ecosystem as a whole. Because many open-source software projects employ the pull-based development model (Gousios et al., 2014, 2015), separating decision-making from software development through "pull requests," we study how ecosystem-wide contributions affect pull request decisions in projects part of that ecosystem.

Software engineering research significantly contributed to our understanding of software development using the pull-based development model, exploring how decisions are made (Zhang et al., 2022a) and the time required to reach that decision (Zhang et al., 2022b). Many studies on pull request decisions addressed factors of pull requests themselves, like the number of commits a pull request has (Gousios et al., 2014; Khadke et al., 2012; Kononenko et al., 2018; Soares et al., 2015a,b; Yu et al., 2016; Zampetti et al., 2019), or intraproject factors (i.e., factors measured within a project), like the number of previously accepted pull requests of a developer (Gousios et al., 2014; Khadke et al., 2012; Rastogi et al., 2018; Zampetti et al., 2019; Zhang et al., 2022a). While several studies included factors that transcend project borders, studying follower networks (Celińska, 2018; Iyer et al., 2021; Soto et al., 2017; Tsay et al., 2014; Yu et al., 2016; Zhang et al., 2022a), Dev and Mockus (2020) were the first to consider the effect of ecosystem-wide factors on pull request acceptance in open-source software. We study the different types of ecosystem-wide experience: contributions in dependent projects and developer collaboration. Further, we zoom in on one special group of developers: project newcomers, because, although they might be new to a project, they might have experience in the broader ecosystem that improves their onboarding process. Using this analysis, we suggest how project maintainers and contributors can operationalize these results to their benefit.

2 Research Questions

Our work is inspired by and builds on the work of Dey and Mockus (2020), who suggested that the experience acquired by making code-related contributions in an open-source software ecosystem increases the chance of pull request acceptance in another project in the ecosystem. Although software communities are commonly defined as a group of coders, various studies have shown the prominence of non-coding collaborators (Cánovas Izquierdo and Cabot, 2021; Geiger et al., 2021; Trinkenreich et al., 2020, 2022), like bug reporters or community managers, who participate in issue-tracker discussions (Panichella et al., 2014). Intuition suggests that non-coding contributions positively relate to decision outcomes, as they are used to discuss domain-relevant information, like architectural knowledge (Soliman et al., 2021) or requirements (Pérez-Verdejo et al., 2021), and often complement information stored in pull requests (Alshara et al., 2023). However, no formal study has been done to test the impact of non-coding contributions on pull request decisions. Therefore, beyond replicating Dey and Mockus' (2020) findings, we include non-coding contributions in our analysis to verify this intuition. Thus, our first research question is:

RQ₁ Do coding and non-coding contributions in the ecosystem affect pull request decisions?

A unique aspect of software ecosystems is that projects have technical dependencies; such that a project reuses the functionalities contained in another (Decan et al., 2019). An example of this is a web application built using the React framework; reusing the framework's functionality and extending it wherever necessary. When software does this, it has a dependency on the other project. Dependencies are defined in two directions: downstream and upstream. To illustrate, for React the web application is a downstream dependency. An upstream dependency refers to the inverse: React is an upstream dependency of the web application.

Research has shown that congruence between a project's dependencies and the developers working on the projects can positively affect project longevity (Valiev et al., 2018; Wattanakriengkrai et al., 2023), that developers tend to join upstream/downstream projects (Geiger et al., 2021; Müller and Rosenkranz, 2023), and sometimes receive priority when requesting help (Geiger et al., 2021). We believe that whether two projects are close in an ecosystem could affect pull request decisions because the knowledge required to implement a package could help to improve and extend it (Maeprasart et al., 2023; Palyart

et al., 2018; Rehman et al., 2022; Shah, 2006; Subramanian et al., 2022; Valiev et al., 2018), and experience building a package could help implement it (Bogart et al., 2016, 2021).

The work of Dey and Mockus (2020) claimed that "if any of the projects the pull request creator previously contributed to depend on the repository to which the pull request is being created, it is more likely to be accepted." Although their work considered "projects that depend on the repository" (i.e., downstream dependencies), the benefit of upstream collaborators has been indicated (Bogart et al., 2016, 2021). Because this has not been studied yet, we include it in our work. To contextualize upstream and downstream contributions, we include non-dependency contributions as well. Non-dependent projects are projects the pull request creator previously contributed to that do not have an upstream or downstream dependency on the project where the pull request is submitted. Including non-dependent projects is crucial because it enables us to assess the additional benefits gained from experience in technically similar projects. Studying this is a direct continuation of RQ₁, providing new insights into the added benefits of technical dependencies by zooming in on the different ecosystem-wide contribution levels. Therefore, our second research question is:

RQ₂ Do contributions made in upstream and downstream projects in the ecosystem affect pull request decisions?

Prior studies have shown that the intra-project "social connectedness" can positively affect the pull request acceptance (Carillo et al., 2017; Gharehyazie et al., 2013, 2015), project longevity (Qiu et al., 2019), general productivity (Zöller et al., 2020), at the risk of increased bug rates (Chen et al., 2024). Similarly, the "social distance" between pull request submitters and their integrators influences the decision outcome (Iyer et al., 2021; Soto et al., 2017; Tsay et al., 2014; Yu et al., 2016; Zhang et al., 2022a), indicated by whether the submitter followed the integrator before submitting the pull request. Although these studies explain the importance of social connections within projects, and to some extent outside projects, they give no insights into the relevance of social connections in software ecosystems.

We argue that the social distance among contributors could also influence pull request decision-making at the ecosystem level. Compared to the follower network (Celińska, 2018; Iyer et al., 2021; Soto et al., 2017; Tsay et al., 2014; Yu et al., 2016; Zhang et al., 2022a), we include two stronger indicators of professional relationships: direct collaboration and ecosystem-wide community standing. This is measured by constructing a social network using developers' development activities, enabling us to study the impact of professional connections at an ecosystem scale. Our third research question is:

RQ₃ Do direct collaboration and community standing in the ecosystem affect pull request decisions?

Finally, studying software ecosystems yields the opportunity to study "experienced project newcomers;" developers who are new to a project, but have

previously made contributions inside the wider ecosystem. Newcomers are paramount to open-source development as joining and leaving a project is easy (Forte and Lampe, 2013) and they can extend project longevity (Qiu et al., 2019). Regardless, changes proposed by newcomers are more commonly refused (Kovalenko and Bacchelli, 2018; Lee and Carver, 2017; Soares et al., 2015a,b), and large projects can be reluctant to accept newcomers (Palyart et al., 2018). Various reasons have been identified why newcomers decide to discontinue contributing even though they were motivated to contribute at the start (Geiger et al., 2021; Rastogi et al., 2015; Rehman et al., 2022; Steinmacher et al., 2015, 2018, 2019a,b; Wermke et al., 2022), like lack of knowledge or poor communication behaviors. This leaves the question of whether newcomers who have had the opportunity to hone these skills experience a different onboarding process. Additionally, although intra-project socialization behavior was shown to improve onboarding (Carillo et al., 2017; Gharehyazie et al., 2013, 2015) and that people tend to join projects and ecosystems they are socially connected to (Casalnuovo et al., 2015; Hahn et al., 2008; de Souza et al., 2016), no study has been conducted on the effect of ecosystem-wide socialization on pull request acceptance decisions submitted by newcomers. Rastogi and Gousios (2021) hypothesized that changes proposed by experienced newcomers are more likely to be accepted, however, do not provide any empirical proof for this. Therefore, we address RQ₁₋₃ specifically for newcomers, operationalizing Rastogi and Gousios' hypothesis by asking:

RQ₄ Do contributions and collaborations in the ecosystem affect pull request decisions for **project newcomers**?

3 Related Work

Our work is motivated by a recent literature study (Zhang et al., 2022a) identifying 94 factors influencing pull request decisions. Most of these are intraproject factors (i.e., factors measured inside a project) like the number of pull requests that someone submitted in the project (Gousios et al., 2014; Khadke et al., 2012; Rastogi et al., 2018; Zampetti et al., 2019; Zhang et al., 2022a). Zhang et al. ranked these factors based on their importance and found that they changed based on context. For example, the pull request integrator's experience is less important for self-integrated pull requests. Table 1 overviews the variables included in this study, which are the principal factors reported by Zhang et al. (2022a) for general pull request decision research. Although many decision factors are technical, relevant human factors were found as well, like interacting with core developers (Yu et al., 2016), geographical location (Rastogi, 2016; Rastogi et al., 2018) and gender (Terrell et al., 2017).

Gharehyazie et al. (2013, 2015) studied the impact of intra-project socialization in mailing lists in Apache projects. They find that socialization — in particular two-way communication — is an important positive indicator of whether a contributor will become a "committer" (a type of membership in Apache projects).

Table 1: Overview of control variables suggested by Zhang et al. (2022a) that are used in this study

Name	Description			
PR commit count	The number of commits in a pull request (Gousios et al., 2014; Khadke et al., 2012; Kononenko et al., 2018; Soares et al., 2015a,b; Yu et al., 2016;			
	Zampetti et al., 2019). This has differing impacts in the literature.			
PR age in minutes	The time between the pull request was opened and closed (Legay et al., 2019; Soares et al., 2015b; Zampetti et al., 2019). This has a negative impact.			
Integrator experience	The number of pull requests the integrator reviewed in the project (Baysal et al., 2016). This has a positive impact.			
PR is self- integrated	This is true if the pull request is submitted and integrated by the same person (Zhang et al., 2022a). This has a negative impact.			
PR has comments	Whether the pull request has comments (Golzadeh et al., 2019; Soares et al., 2015a). This has a negative impact.			
$\begin{array}{ccc} PR & has \\ com. & ext. \\ contr. \end{array}$	Whether someone different than the submitter, integrator, or a project contributor commented on the pull request (Golzadeh et al., 2019). This has a positive impact.			
PR text contains "#"	Whether the pull request's title or description contains a reference to another issue or pull request (Gousios et al., 2014; Yu et al., 2016). This has a positive impact.			
Is project newcomer	Whether the pull request submitter has successfully contributed to the project before (Kovalenko and Bacchelli, 2018; Lee and Carver, 2017; Soares et al., 2015a,b). This has a negative impact. This feature was not part of the principal features reported by Zhang et al. (2022a) but was			
	added as it motivates RQ_4 .			

Some studies explored the effect of project-transcending factors (Celińska, 2018; Iyer et al., 2021; Soto et al., 2017; Tsay et al., 2014; Yu et al., 2016; Zhang et al., 2022a). Most of these (Iyer et al., 2021; Soto et al., 2017; Tsay et al., 2014; Yu et al., 2016; Zhang et al., 2022a) evaluated the GitHub's follower feature, taking this as a proxy for social connectedness on the platform. They found that being followed by others and following others (Soto et al., 2017; Yu et al., 2016; Zhang et al., 2022a), and following your integrator (Iyer et al., 2021; Tsay et al., 2014; Zhang et al., 2022a) positively impacts pull request acceptance. Finally, Celińska (2018) found a positive relationship between follower and collaboration networks on the number of pull requests and forks a project receives.

The works of Dey and Mockus (2020), Cheng et al. (2017), and Jergensen et al. (2011) lie closest to our work. Dey and Mockus studied the effect of ecosystem experience on pull request acceptance, using three ecosystem factors: the submitter's pull request track record (pull request count and acceptance rate), their general experience (the number of commits and projects worked on), and whether they worked on a downstream project. They showed that all factors, except pull request count, positively affect pull request acceptance. However, this study does not account for the difference between intra-project and ecosystem-wide experience as these metrics were aggregated in their analysis. Because previous work suggested the impact of intra-project

experience (Gousios et al., 2014; Khadke et al., 2012; Rastogi et al., 2018; Zampetti et al., 2019; Zhang et al., 2022a), thus confounding the results.

Cheng et al. (2017) and Jergensen et al. (2011) both studied the impact of ecosystem experience on becoming a member of GNOME projects. Here, Cheng et al. (2017) showed that having more contributions and projects worked on positively affects becoming a core member, indicated by the number of commits they make and the number of people they collaborate with inside a project. In contrast, Jergensen et al. (2011) showed that developers with ecosystem experience, indicated by the number of releases they were active on in other projects, do not change a system's core functionality more often, which they argue is preserved for core project members, whereas developers with intra-project experience do. Although these studies highlight the impact of ecosystem-wide contributions on the types and number of code changes, these findings cannot be translated to the pull-based development model.

We further differentiate ourselves by addressing coding and non-coding contributions, upstream and downstream dependencies, and ecosystem-wide collaborations. We specifically test their impact on pull requests submitted by project newcomers.

4 Methodology

4.1 Data Collection

Open-source software projects commonly use a combination of pull requests and issue-tracking systems (Alshara et al., 2023) to make coding and non-coding contributions, respectively. Consequently, three data sources are required to answer the research questions: pull requests, issues, and a list of project dependencies from a software ecosystem. Although various projects use media like mailing lists (Panichella et al., 2014), these are excluded as they are not integrated into platforms like GitHub. The dataset created by Katz

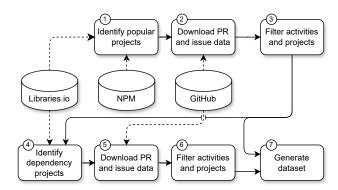


Fig. 1: Visualization of the data collection process. For more details, refer to Figure 1 in the original master's thesis (Meijer, 2023)

(2020) (also known as libaries.io) is the starting point of this research, containing project archive data of 32 software package ecosystems (like NPM, Maven, and Go). It contains information like package names, repository descriptions, and dependencies. Although Katz's dataset only contains data up to 2020, it is valuable as creating a similar dataset is very time-consuming. We zoom into the NPM ecosystem, a JavaScript package library. We chose NPM for three reasons: 1) because Dey and Mockus (2020) used it as well for which our results can be compared, 2) because it is the largest ecosystem in the dataset, and 3) because NPM packages have explicit dependencies on each other. Since this dataset does not contain pull requests and issues, we collected these separately using GrimoireLab (Dueñas et al., 2021). Figure 1 visualizes the data collection process elaborated in the following sections.

4.1.1 Project Sampling

Katz (2020)'s dataset contains 1.2 million NPM packages. Because this is too many to process completely, we sampled their dataset. This is done in two stages; collecting development activities from 1) popular projects, and 2) upstream/downstream projects of the popular projects. Respectively, these ensure that our dataset contains many development activities and projects involved in a dependency. To simplify this process, a simple set of inclusion criteria was defined: 1) projects must have a GitHub repository, 2) they cannot be a fork of another project, and 3) the project must have at least five valid pull requests to ensure that the project uses pull requests systematically (Section 4.1.2 elaborates on this).

Popular projects were sampled using one additional criterion: "the project must have at least 10,000 downloads in the past 16 months." This definition aligns with Dey and Mockus (2020) and is a proxy for project activity, assuming that popular projects have more development activities. To ensure the analysis is not biased toward upstream or downstream projects, we took a stratified sample of pull requests. Because a preliminary analysis showed that downstream projects have approximately half the number of pull requests compared to upstream projects, the sample of downstream projects was approximately doubled.

4.1.2 Activity Filtering

Development activities (i.e., pull requests and issues) were removed using the following criteria:

- They cannot be submitted by a deleted account. When users delete their account, their development activities are assigned to a "ghost" user, who has many activities assigned to it.
- The development activity is closed. This ensures that the discussion in the activity has come to fruition.
- The development activity has no missing data. Some development activities miss data like user details.

- They cannot be submitted by a bot. We removed bots because Dey and Mockus (2020) attributed some of the quirks in their results to bots.

4.1.3 Bot Removal

Many actions in GitHub can be automated using bots (Rombaut et al., 2023). An example is "Dependabot" which can be used to manage dependency updates. When unaccounted for, bots can affect results in developer-oriented research (Dey and Mockus, 2020). However, Identifying bots is not a trivial task. GitHub metadata was used as a first line of defense, as creators can mark their accounts as bots. However, this method does not work perfectly because various bots use user accounts. Although bot detection is an active field study (Chidambaram and Mazrae, 2022; Dey et al., 2020b; Golzadeh et al., 2020b, 2021), none of these classifiers could be directly applied to this study due to stringent data requirements and the substantial amount of manual effort they require. Fortunately, the classifications of Dey et al. (2020a,b) and Golzadeh et al. (2020a, 2021) are publicly available, and all bots contained in their datasets were removed. Finally, to ensure no major bots are missed, all users with over 400 closed pull requests (slightly less than 500 users) have been manually checked, excluding four additional bots from the dataset.

4.2 Contribution and Collaboration Metrics

4.2.1 Developer Contributions

Two answer RQ_{1,2,4}, developer contributions are calculated. Various studies have previously taken the number of pull requests submitted (Gousios et al., 2014; Khadke et al., 2012; Rastogi et al., 2018; Zampetti et al., 2019; Zhang et al., 2022a) or the number of code commits (Gousios et al., 2014; Khadke et al., 2012; Kononenko et al., 2018; Soares et al., 2015a,b; Yu et al., 2016; Zampetti et al., 2019; Zhang et al., 2022a), not considering the contributions done in alternative sources, like issue tracking systems or the comment section pull requests. To limit the scope of this study, we do not include code commits. However, issues have been included because non-coding contributions are prominent in open-source communities (Cánovas Izquierdo and Cabot, 2021; Trinkenreich et al., 2020) but have not been formally studied yet. Table 2a overviews the included contribution types.

Given a software ecosystem, developer contributions can be measured at two levels: intra-project and ecosystem-wide. Distinguishing between these two is important, as intra-project contributions have shown to be an important factor of pull request acceptance (Gousios et al., 2014; Khadke et al., 2012; Rastogi et al., 2018; Zampetti et al., 2019; Zhang et al., 2022a) and would confound the results. To prevent confounding, we define ecosystem-wide contributions as "all contributions made in the ecosystem, excluding intra-project contributions." More formally, given a focal project X_i and a contributor c,

ecosystem-wide contributions are the sum of all contributions made by c in ecosystem projects X_j minus the contributions made by c in X_i . To account for the fact that software changes rapidly over time, each variable is calculated using a snapshot of 90 days before the pull request is closed.

4.2.2 Dependency Contributions

Modern software commonly depends on other projects by reusing their functionalities (Decan et al., 2019). Dependencies are defined in two directions: downstream and upstream. A downstream dependency refers to a dependency from another project (a downstream project) to the focal project and an upstream dependency to the inverse. Consequently, ecosystem-wide contributions can be divided into upstream, downstream, and non-dependency contributions. Non-dependency contributions represent the contributions made to projects that are neither upstream nor downstream and make it possible to test the added benefit of dependencies. Calculating these contributions is equal to calculating ecosystem contributions; the only difference being that they account for the (non-)existence of dependencies. Concretely, given a focal project X_i , downstream contributions are the sum of all contributions in ecosystem projects X_j such that there exists a dependency from X_j on X_i , upstream contributions such that there exists a dependency from X_i on X_j , and non-dependency contributions such that neither exists.

The benefit of cascading dependencies (Geiger et al., 2021) has been reported in the past (Valiev et al., 2018). These are transitive dependencies from project X_k on X_i such that X_k depends on an intermediary project X_j which depends on X_i . In other words: X_k depends on X_j depends on X_i , thus X_k transitively depends on X_i . However, we do not include it in our analysis because it is not obvious how it differs from regular dependencies due to limited evidence in the literature and the extra complexity it adds to our method.

4.2.3 Ecosystem-wide Collaboration and Community Standing

Some studies already transcended project boundaries by studying the social connections between developers, suggesting that both general connectedness (Soto et al., 2017; Yu et al., 2016; Zhang et al., 2022a) and direct connectedness with your pull request integrator (Iyer et al., 2021; Tsay et al., 2014; Zhang et al., 2022a) can affect request acceptance. These studies measure social connectedness using GitHub's follower feature. Although this provides insights into the social component of software engineering, it can hardly represent professional connectedness because following someone is accessible to anyone for any reason, not just professional reasons. Therefore, this study includes a new measure of connectedness: through collaboration in the ecosystem.

Collaboration refers to the extent to which a contributor has worked with others, gaining experience and establishing trust with other developers. We consider connectedness in two manners: direct collaboration and ecosystemwide community standing. Direct collaboration refers to the number of times

Table 2: Overview of contribution and collaboration types

(a) Contribution types $(RQ_{1,2,4})$

Name	Description
PR merge ratio	The fraction of accepted pull requests.
$PRs\ submitted$	The total number of submitted pull requests.
PR comments	The number of comments to a pull request.
Issues submitted	The number of submitted issues.
Issue comments	The number of comments to an issue.

(b) Collaboration types (RQ_{3,4})

Name	Description
PR review	One collaborator reviewed the pull request submitted by the other.
PR comment	One collaborator commented on the pull request submitted by the other.
PR discussion	The collaborators participated in the pull request discussion.
Issue comment	One collaborator commented on the issue submitted by the other.
Issue discussion	The collaborators participated in the issue discussion.

two developers have directly cooperated. For example, by discussing something in an issue. Similarly, ecosystem-wide community standing is measured to account for the experience of the people they have collaborated with, as cooperating with many or well-experienced developers might be more valuable than working by yourself. We considered the five types of collaboration shown in Table 2b.

We calculated direct collaboration and community standing using two metrics taken from social network analysis (Newman, 2018; Schreiber and Zylka, 2020): link strength and node centrality. Link strength refers to the number of collaborations connecting two contributors; i.e. direct collaboration. In turn, node centrality describes the general importance of a node in the network, which we use to measure community standing. Various well-known metrics for node centrality exist, like HITS, Eigenvector centrality, or PageRank (Newman, 2018). Although these metrics are well-applicable to smaller datasets, their high computation time makes them impractical when applied with this study's intended level of granularity: applied to a 90-day window preceding each pull request; i.e., a sliding window. Therefore, we use an alternative: second-order degree centrality.

The intuition behind second-order degree centrality is that a contributor's centrality depends on the activity of the people they have collaborated with in the ecosystem (i.e., their neighbors). This metric grows when the collaborator either collaborated with many people who made few contributions, collaborated with few people who made many contributions, or somewhere in-between. This metric is inspired by previous works studying influential people in social networks (Bródka et al., 2012; Chen et al., 2012) and trust in social networks (Stickgold et al., 2013). A variant was also considered by Jergensen et al. (2011) to identify core developers in OSS projects, however, it was excluded due to multicollinearity with other network variables. We ex-

clude direct collaborations between developers as this has already been measured using link strength. In addition, we exclude any contributions of the neighbors that happened after the focal developer and their neighbor collaborated, as not accounting for time chronology would inflate the metric (Holme and Saramäki, 2019). Finally, similarly to ecosystem-wide contributions, we exclude neighbors' contributions to the focal project to prevent confounding by intra-project contributions.

Since we include multiple collaboration types, link strength and node centrality can be calculated for each. To reduce data analysis complexity, instead of calculating link strength and second-order degree centrality separately for each collaboration type, we aggregated the results using a weighted sum where the weights are inverse-proportional to the total number of collaborations (Kivelä et al., 2014). For example, because participating in the discussion in an issue-tracking system is more common than integrating someone's pull request, its weight is lower. During preliminary analysis, experimentation was done using the analytical hierarchy process (Casola et al., 2009) to fine-tune these weights. This is a systematic approach to manually assign weights to edges based on, for example, domain knowledge. Among others, we experimented based on the anticipated amount of information exchanged as, for example, the argument can be made that more information is exchanged in a comment on a pull request (giving feedback or providing context) than integrating it (clicking the accept/reject button). However, this step was omitted in the final methodology because no notable effect was observed in the results.

4.3 Data Analysis

We used a combination of mixed-effects logistic regression (logit) and a random forest classifier to study the impact and predictive strength of the measured variables on pull request acceptance, respectively. Mixed-effects logit describes the linear relationship between variables, by calculating coefficients while respecting random effects across projects. This is important as some projects have higher pull request acceptance rates than others (Palyart et al., 2018). The control variables listed in Table 1 have been included to place the new variables in context. These variables are suggested for pull request research by Zhang et al. (2022a), listing them as the principal factors for pull request decisions. We control for these variables by modeling them alongside the tested variables. Although Zhang et al. suggests including "whether continuous integration (CI) is used," we did not include it as the diversity of CI platforms makes it difficult to collect this data. We do not expect this to affect our results as Zhang et al. did not report that it correlates with any variable similar to the ones introduced in this work. Similarly, they suggest including "whether the pull request submitter is a core member," which is excluded because it is multicollinear with "intra-project pull request merge ratio." Further, although "integrator experience" is included in the random forest models, it is excluded from the regression models as it is multicollinear with intra-project experience.

Table 3: Overview of the coefficients (coef.), standard error ($std.\ err.$) calculated by the mixed effects logistic regression (logit) models, containing control variables, and contribution (con.) and collaboration (coll.) variables. The sign of the coefficient indicates the relationship type (negative vs. positive). $^{\diamond}$ Add-one log-transformed variable; $^*p < .001$

(a) Models that used all data points, used to answer RQ_{1-3}

	Ecosystem Model (RQ_1)	Dependency Model (RQ_2)	Collaborative Model (RQ_3)
Control Variables	Coef. (std. err.)	Coef. (std. err.)	Coef. (std. err.)
Is project newcomer	129*(.001)	129*(.001)	n/a
PR is self-integrated	262*(.001)	261*(.001)	n/a
PR has comments	127*(.001)	127*(.001)	118*(.001)
PR contains "#"	.037*(.001)	.037*(.001)	.040*(.001)
PR has com. ext. contr.	.030*(.001)	.030*(.001)	006*(.001)
$PR\ lifetime^{\diamond}$	482*(.002)	478*(.002)	435*(.002)
$PR \ commit \ count^{\diamond}$.189*(.005)	.186*(.005)	.124*(.005)
Contribution Variables	Coef. (std. err.)	Coef. (std. err.)	Coef. (std. err.)
Intra-project issues [⋄]	.272*(.003)	.274*(.003)	.202*(.003)
$Ecosystem PRs^{\diamond}$.252*(.002)	n/a	n/a
Non-dependency PRs^{\diamond}	n/a	.215*(.002)	n/a
$Downstream \ PRs^{\diamond}$	n/a	.128*(.005)	n/a
$Upstream\ PRs^{\diamond}$	n/a	.097*(.006)	n/a
Eco. community standing $^{\diamond}$	n/a	n/a	$.167^*(.004)$
$Direct\ collaboration^{\diamondsuit}$	n/a	n/a	.322*(.004)

(b) Models distinguishing between project newcomer and non-newcomer data points, used to answer RQ4

	Ecosystem Model		Dependency Model		Collaborative Model	
	Newcomer	Non-newcomer	Newcomer	Non-newcomer	Newcomer	Non-newcomer
Control Variables	Coef. (std. err.)	Coef. (std. err.)				
PR is self-integrated	562*(.002)	125*(.001)	563*(.002)	125*(.001)	n/a	n/a
PR has comments	156*(.002)	116*(.001)	156*(.002)	117*(.001)	043*(.002)	096*(.001)
$PR\ contains\ "\#"$.044*(.001)	.032*(.001)	.044*(.001)	.032*(.001)	.048*(.002)	.032*(.001)
PR has com. ext. contr.	.025*(.002)	$.047^*(.001)$.024*(.002)	$.047^*(.001)$	066*(.002)	.028*(.001)
$PR\ lifetime^{\diamond}$	503*(.003)	494*(.002)	500*(.003)	490*(.002)	202*(.003)	458*(.002)
$PR\ commit\ count^{\diamond}$.216*(.010)	.282*(.005)	.213*(.010)	.279*(.005)	140*(.012)	.204*(.005)
Contribution Variables	Coef. (std. err.)	Coef. (std. err.)				
Intra-project issues [⋄]	.213*(.013)	.169*(.003)	.214*(.013)	.170*(.003)	.186*(.015)	.109*(.003)
$Ecosystem\ PRs^{\diamond}$.376*(.004)	.164*(.002)	n/a	n/a	n/a	n/a
Non-dependency PRs^{\diamond}	n/a	n/a	.324*(.005)	.133*(.002)	n/a	n/a
$Downstream\ PRs^{\diamond}$	n/a	n/a	.295*(.016)	.094*(.004)	n/a	n/a
$Upstream\ PRs^{\diamond}$	n/a	n/a	.318*(.013)	.071*(.006)	n/a	n/a
Eco. community standing \diamond	n/a	n/a	n/a	n/a	.335*(.008)	.071*(.004)
$Direct\ collaboration^{\diamond}$	n/a	n/a	n/a	n/a	.517*(.012)	.240*(.004)

This is not a problem in random forest models. Finally, although Zhang et al. do not suggest "whether the pull request submitter is a project newcomer," we include it because it motivates asking RQ₄. This variable is calculated by tracing whether a pull request submitted by the user was accepted in the project before.

To accurately answer the different research questions, we created multiple logit models, simplifying the interpretation process: an ecosystem model (RQ_1), a dependency model (RQ_2), and a collaboration model (RQ_3). In addition, to answer RQ_4 , we created two additional models for each, distinguishing between the pull requests submitted by project newcomers and nonnewcomers.

The preliminary analysis showed that the top 2% of the projects were responsible for 49% of the pull requests, skewing the results significantly in favor of these projects. Therefore, a cap of 694 pull requests is put on the pull requests of these projects, matching the number of pull requests of the largest project that is not part of the top 2%. The pull requests in the top 2% projects were sampled randomly.

Mixed effects logit makes three assumptions: linearity between the variables and the log-odds ratio, absence of multicollinearity, and no strong outliers, as each harms the validity of the model results. To improve the log-linearity of the models, most numerical variables were transformed using add-one log transformation $x' = \ln(x+1)$ because these variables follow a long-tail distribution. We used add-one log transform instead of normal log transform because all transformed features contain zeroes (for which the logarithm cannot be calculated). The transformed features are marked in Table 3 This tempers the impact of developers with a very high number of contributions. Further, all numerical data was normalized using min-max normalization, simplifying the comparison between variables as it transforms their range to the same scale. Multicollinearity is tested using the variance inflation factor (VIF) and the Spearman correlation coefficient ρ between pairs of features. Variables that strongly correlated with another ($|\rho| \geq 0.5$) and are highly multicollinear ($VIF \geq 5$) were removed. Outliers were removed using Cook's distance (Cook, 2000), filtering data points that affect regression coefficients disproportionally, using a cut-off threshold of 4/(n-k-1), where n is the number of observations, and k is the number of predictors. This removed between 0.1% and 2.6% data points across models.

Finally, to identify the predictive strength of each feature and feature group, random forest is used (Breiman, 2001; Pedregosa et al., 2011). To measure the importance of individual factors, the mean decrease in Gini is calculated, which estimates the amount of information lost when a predictor is removed (Breiman, 2001; Pedregosa et al., 2011). Then, to evaluate the predictive strength of variable groups, an *inverse ablation* study was performed (i.e., separate models were trained using subsets of features), and evaluated using F1 scores.

Table 4: The mean and standard deviation of the F1 scores calculated with the random forest models, trained through inverse ablation using variable groups, calculated using 5-fold cross-validation. The models are trained using all data and the (non-)newcomer subsets. The baseline is the probability of observing a merged pull request

Variable Group	All Data	Newcomer	Non-newcomer
Baseline	.787	.658	.857
Control	.877 (.0004)	.796 (.0009)	.921 (.0003)
Intra-project	.881 (.0003)	.793 (.0006)	.925 (.0007)
All intra-project variables	.910 (.0002)	.807 (.0009)	.957 (.0003)
Ecosystem (RQ_1)	.877 (.0002)	.782 (.0011)	.920 (.0003)
Non-dep. (RQ_2)	.877 (.0001)	.783 (.0010)	.920 (.0005)
Downstream (RQ_2)	.880 (.0005)	.793 (.0011)	.923 (.0006)
$Upstream (RQ_2)$.880 (.0007)	.793 (.0014)	.923 (.0005)
Collaboration (RQ_3)	.864 (.0006)	.759 (.0011)	.911 (.0002)
All ecosystem variables	.874 (.0003)	.773 (.0013)	.920 (.0006)
All variables	.921 (.0002)	.830 (.0005)	.959 (.0004)

5 Results and Discussion

The collected dataset consists of 1.8 million pull requests and 2.1 million issues spread across 20,052 projects, submitted by 190,898 unique users, across 9 years. Of these, 72% of the projects were collected using the popularity criterion and 28% with the dependency criterion. The latter was almost equally split into upstream and downstream projects. Of these projects, 19,609 (98%) have an upstream dependency on another project in the dataset, and 8,688 (43%) have a downstream dependency, which could suggest a core-periphery structure (e.g., Barabási and Albert (1999)) in the dependency network. As described in Section 4.3, this dataset was sampled according to the number

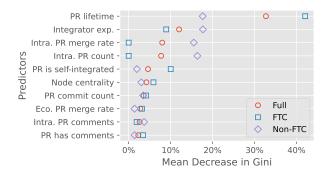


Fig. 2: Feature importance plot showing the mean decrease in Gini calculated using random forest trained using the full dataset and the (non-)newcomer subsets

of pull requests per project to improve internal validity, for which 1.2 million data points (66%) were used for inference.

Table 3 presents the results of the mixed effects logistic regression models. During analysis, it became apparent that all contribution types listed in Table 2a are strongly positively correlated. This means that for ecosystem-wide, upstream, downstream, non-dependency, and intra-project contributions, all contribution variables shown in Table 2a had a Spearman coefficient greater than 0.5. Therefore, the logit models only include one variable for each group. The results show that each measured variable significantly and positively correlates with pull request acceptance. For example, looking at " $Ecosystem\ PRs$ " in Table 3a, a pull request submitted by someone who submitted many pull requests in the ecosystem is 25.2% more likely to be accepted (as indicated by its coefficient 0.252).

To identify the importance of the used features, Figure 2 overviews the importance of the top 10 features of the random forest models. It stands out that the majority of information (56%) is acquired from the control variables: PR lifetime (32%) integrator experience (12%), intra-project experience (8%) and PR is self-integrated (4%). This is unsurprising as previous work (Zhang et al., 2022a) listed them as principal factors of pull request decisions. Conversely, the absence of the other control variables stands out as it suggests some of our features yield more information, like node centrality (4%) and pull requests submitted in the ecosystem (3%).

To compare the importance of the different contribution levels (intraproject, ecosystem, etc.) in the random forest models, we performed a pairwise comparison using the feature importance scores of their comprising contribution types (pull requests submitted, etc.). Here, we observe whether the feature importance of all contribution types of a contribution level is higher than those in another, suggesting that the contribution level is of higher importance. We found that intra-project contributions are more important than upstream contributions, downstream contributions are more important than upstream and downstream contributions. Interestingly, only the last of these three applied to project newcomers, suggesting that downstream and intra-project contributions are relatively less important when predicting the outcome of their pull requests.

Table 4 provides the broadest view of the analysis, showing the results of the reverse ablation study using variable groups. These results include a baseline for each model, equaling the probability of observing a merged pull request in the dataset, as pull requests are more commonly accepted than refused. Although any model with an F1 score over 0.5 (intuitively, 50% correct classifications) would have some predictive power, only models outperforming this baseline score better than a probabilistic guesser. Looking at the variable groups in Table 4, we see that each group performs almost equally, reaching F1 scores of approximately 0.88, outperforming the baseline, with almost no

¹ Refer to the replication package for an overview of all 35 variables.

deviation across data folds used for cross-validation. Models trained using only collaboration metrics perform slightly worse, scoring 1.3-1.6% lower.

Interestingly, although the model including all intra-project variables improves these separate models between 1.3% and 3.6%, creating a model that uses the newly included ecosystem variables did not improve classification performance compared to the models trained with their comprising sub-groups. Regardless, combining all variables improves classification by 0.2% to 2.3%, especially for pull requests submitted by project newcomers. This suggests that classifying pull requests submitted by this group benefits from ecosystem-wide information.

5.1 Answering Questions

5.1.1 Ecosystem-wide Contributions

The results of this study (shown in the ecosystem model in Table 3a) showed that a developer's ecosystem-wide contributions positively affect their pull request acceptance. Although this notion might seem intuitive, it transcends previous claims as most previous studies emphasized the impact of intra-project technical experience (Baysal et al., 2012; Gousios et al., 2014; Iyer et al., 2021; Khadke et al., 2012; Lee and Carver, 2017; Pinto et al., 2018; Rastogi et al., 2018; Soares et al., 2015b; Tsay et al., 2014; Zampetti et al., 2019). Some studies considered the impact of ecosystem-wide experience, addressing its impact on pull request acceptance (Dey and Mockus, 2020), becoming a core contributor (Cheng et al., 2017), and developers' joining behavior (Jergensen et al., 2011). Our study transcends these studies twofold.

Firstly, this is done by considering non-coding contributions, namely participation in issue-tracking systems. This is relevant as recent studies indicated the prominence of non-coding contributions in OSS projects (Cánovas Izquierdo and Cabot, 2021; Geiger et al., 2021; Trinkenreich et al., 2020, 2022). Previous work already hinted towards the relevance of issue-tracking systems, as they are commonly used to discuss architectural knowledge (Soliman et al., 2021) and requirements (Pérez-Verdejo et al., 2021), and commonly complement pull requests (Alshara et al., 2023). However, the relevance of non-coding contributions has not been empirically explored yet. We identify a positive relationship between the number of coding and non-coding contributions. Previous work suggested the importance of this factor concerning attaining "committer" status in Apache projects (Gharehyazie et al., 2013, 2015). Our study translates this result to pull-based development, finding that non-coding contributions matter when joining the project as a non-core contributor.

Secondly, we separately addressed ecosystem-wide developer contributions in pull request decision-making, which Dey and Mockus (2020) attempted to do as well. We decided to separate intra-project contributions from ecosystem-wide contributions, eliminating the possibility of intra-project contributions confounding the results (Gousios et al., 2014; Khadke et al., 2012; Rastogi

et al., 2018; Zampetti et al., 2019; Zhang et al., 2022a). We successfully replicated the results by Dey and Mockus (2020), and nuance their claims by showing that, intra-project contributions have a 2% larger impact on pull request decisions than ecosystem-wide contributions in the general case.

Answering $\mathbf{RQ_1}$: Our results suggest that pull request acceptance decisions depend on more than merely intra-project factors and code-oriented contributions. At an intra-project and ecosystem level, non-coding participation in issue-tracking systems and code reviews positively affects pull request decisions.

5.1.2 Contributions in Dependent Projects

Although understanding the impact of general ecosystem contributions is important, ecosystems are special because projects often depend on each other (Decan et al., 2019). Dependencies are defined in two directions: upstream and downstream, such that a downstream project implements the functionality of an upstream project. This means that projects have some technical relevance to each other, as experience implementing a package could help improve it (Maeprasart et al., 2023; Palyart et al., 2018; Rehman et al., 2022; Shah, 2006; Subramanian et al., 2022) and building a package could help implement it (Bogart et al., 2016, 2021). The results in the dependency model in Table 3a suggest that contributions in upstream and downstream projects positively affect pull request decision outcomes. This is consistent with previous findings (Dey and Mockus, 2020) identifying the positive impact of downstream contributions.

The comparison of upstream and downstream contributions suggests that downstream contributions have a 3% larger impact than upstream contributions. This could be partially explained through the different roles that upstream and downstream developers have in dependency networks. Although downstream developers are paramount when fixing issues and building new features (Geiger et al., 2021; Palyart et al., 2018; Shah, 2006; Valiev et al., 2018), and in some cases even receive priority with problems (Geiger et al., 2021), limited evidence has been found regarding the benefits of upstream developers. Although Bogart et al. (2016, 2021) find that upstream developers occasionally help downstream projects integrate breaking API changes, they explicitly conclude this is uncommon in the NPM ecosystem.

To contextualize the relevance of dependency contributions, a comparison has been made with non-dependency ecosystem contributions. These are contributions made in neither upstream nor downstream projects. The results suggest that the impact of non-dependency contributions is 11.8% and 8.7% more important than upstream and downstream contributions, respectively. Although dependency contributions are relevant, the added specific knowledge of a technically related package does not outweigh the relevance of the general contributions of the ecosystem's common technology. Alternatively, this could mean that the tasks picked up by this group are more complex.

Previous work vouched for the benefits of dependencies that are congruent with the developers working on the involved projects, finding that it can positively affect the longevity of projects (Valiev et al., 2018; Wattanakriengkrai et al., 2023) and that contributors tend to join upstream and downstream projects (Müller and Rosenkranz, 2023). Regardless, dependency networks are not always congruent with the developers working on projects (Syeed et al., 2014). Our results complement these studies in two fashions: 1) by showing its positive impact on a developer's technical success, as indicated by their pull request acceptance, and 2) that, collaborating with dependent projects has clear benefits, it only partially explains developer success as contributions made in non-dependent projects have a stronger impact on pull request decisions.

Answering RQ₂: Our results suggest that contributions in downstream and upstream projects positively affect pull request decisions. In general, downstream contributions are more important than upstream contributions. This highlights the importance of dependency-specific knowledge in cross-project collaboration. However, contributions made in upstream/downstream projects are less impactful than contributions made in non-dependent projects.

5.1.3 Direct Collaboration and Ecosystem-wide Standing

Open-source software development is commonly described as a socio-technical process (Forte and Lampe, 2013) where people are judged on their technical merit (Steinmacher et al., 2015; Wermke et al., 2022), but might be trusted earlier because of their social connections (Iyer et al., 2021; Soto et al., 2017; Tsay et al., 2014; Yu et al., 2016; Zhang et al., 2022a). Prior research showed that social connections can positively affect project longevity (Fang et al., 2023; Hahn et al., 2006; Wang, 2012), the number of pull requests and forks a project receives (Celińska, 2018), and project downloads (Fershtman and Gandal, 2011; Méndez-Durón and García, 2009; Peng et al., 2013). Further, people tend to join open-source projects (Casalnuovo et al., 2015; Hahn et al., 2008) and closed-source ecosystems (de Souza et al., 2016) they are socially connected to.

Although these studies give insights into the likelihood of joining and successfully joining a project, they limitedly explain the impact of social connectedness while submitting a pull request. Several studies have addressed this at an intra-project level, identifying a relationship with prolonged project participation (Qiu et al., 2019), pull request acceptance (Gharehyazie et al., 2013, 2015), task performance (Carillo et al., 2017), and in some cases productivity (Zöller et al., 2020), at the risk of higher bug proneness (Chen et al., 2024). Although this gives a good overview of the impact of social connectedness inside a project, it limitedly explains the impact at an ecosystem scale.

Studies that transcend a project attempted to fill this gap using GitHub's follower feature, finding that following your pull request integrator (Soto et al., 2017; Yu et al., 2016; Zhang et al., 2022a), and being well-connected in general

(Iyer et al., 2021; Tsay et al., 2014; Zhang et al., 2022a) are positively related to pull request acceptance. Although these findings highlight the impact of social network features in social coding platforms like GitHub, they can only limitedly represent the professional network. This is because the follower feature can be used by anyone for any reason, without much regard for their professional relationship. We constructed a professional social network, extracting connections between collaborators using the pull requests and issues they were involved in, ensuring two connected developers have collaborated in the past.

We measure participation in the ecosystem in two fashions, direct collaboration and community standing, to identify the impact of direct proximity to your pull request integrator and your general connectedness in the ecosystem on pull request decisions, respectively. The results in the collaboration model in Table 3a suggest that both of these factors positively affect pull request acceptance. It stands out that the impact of direct collaboration (i.e., link strength) has a 12% larger influence on pull request decisions than intra-project contributions. This underlines the importance of professional connectedness to contribution success and can be explained through higher pre-established trust between them. Regardless of whether the developer might still need to learn the project's specifics, this prior connection might be sufficient to vouch for their competencies. Alternatively, because the developers have collaborated already, they might be more familiar with each other's manner of working (e.g., coding style, or manner of explaining things), for which it could be easier to achieve a fruitful discussion and consensus on implementation details. This feature was unimportant in the predictive model, explaining only 0.6% of the information. This is not surprising because, at an ecosystem level, collaboration between a pull request submitter and a pull request integrator is relatively rare, as this occurred in only 10% of the cases. Testing for this subset of data significantly increased its importance, becoming the seventh most important feature, just below ecosystem-wide community standing.

We also find that ecosystem-wide community standing has a positive effect on pull request decisions, however, 3.5% less than intra-project contributions and 12% less than direct collaboration. This can be explained because the metric is less tangible — as it measures if someone has collaborated with many or experienced developers. This result can be explained similarly to Wang (2012), who studied the external network of a project; i.e., the number of developers a project is connected to because one of its contributors also collaborates in other projects. Wang reasons that having an increased number of connections increases the likelihood that a developer interacts with and consequently acquires high-quality resources or deeper knowledge of the system. Because a developer collaborated with many or experienced others in the ecosystem, they might have learned the specific knowledge necessary to produce a high-quality pull request, which is more likely to be accepted.

Answering RQ₃: Our results suggest that working directly with your pull request integrator and collaborating with many or well-experienced developers in the open-source software ecosystem positively affects pull request acceptance.

This underlines the importance of professional connections at an ecosystem level. Direct collaboration with your integrator is more important than intraproject contributions, highlighting the importance of social factors in software engineering, regardless of potential project-specific knowledge gaps.

5.1.4 Project Newcomers

Project newcomers are the foundation of OSS projects, as the threshold of joining and leaving is low (Forte and Lampe, 2013), and having more newcomers can positively affect project longevity (Qiu et al., 2019). Regardless of their fundamental role in OSS projects, various studies have suggested that large projects are more reluctant to let new people in (Palyart et al., 2018), and that pull requests submitted by newcomers are more frequently refused (Kovalenko and Bacchelli, 2018; Lee and Carver, 2017; Soares et al., 2015a,b) — a finding that was supported in this study, as their pull requests were 12.9% more commonly refused (see Table 3a).

Looking deeper into the underlying reason, various socio-technical barriers have been identified that prevent newcomers from successfully onboarding in open-source projects (Geiger et al., 2021; Steinmacher et al., 2015, 2019a,b). An example of this is the perceived lack of expertise, which was also identified in the works of Wermke et al. (2022) and Rehman et al. (2022), listing it as a means to gain trust (through meritocracy) and a hurdle that prevents them from successfully onboarding, respectively. This barrier is not only present in open-source projects, as Rastogi et al. (2015) identified the same phenomenon in closed-source projects.

Various socialization factors are listed as a barrier for newcomers (Steinmacher et al., 2015, 2019a,b), like low-quality discourse or low responsiveness. The work of Gharehyazie et al. (2013, 2015) and Carillo et al. (2017) zoomed in on this topic, both studying the impact of intra-project socialization on onboarding in open-source projects. Gharehyazie et al. (2013, 2015) identified a positive relationship between socialization in the project's mailing list and becoming a committer, and Carillo et al. (2017) that it positively relates to newcomers' task performance.

Each of these studies vouches for the importance of developer experience and collaboration at an intra-project level, showing positive effects on new-comer onboarding. This leaves a gap in current literature that explores their impact at an ecosystem scale. Our work operationalizes a hypothesis by Rastogi and Gousios (2021), proposing that "unit changes from new contributors originating in the same ecosystem are more likely to be accepted."

Our results in Table 3b suggest that each ecosystem-wide contribution and collaboration factors positively affect the outcomes of newcomers' pull requests decisions. It stands out that the measured impact of all variables exceeds that of intra-project contributions, through non-coding contributions like posting an issue, between 8% and 33%. This is substantially different from non-newcomers and the two groups combined, as we observe the inverse such that

intra-project contributions yield between 0.5% and 13% greater impact than the ecosystem-wide contribution and collaboration metrics. The only exception is direct collaboration between the pull request submitter and integrator, which has a 13% greater impact than intra-project contributions. However, this is notably smaller than the 33% increase measured for newcomers. The predictive models for newcomers' pull requests also depend more strongly on these variables.

Expanding on the results of RQ_3 , a substantial difference can be observed in the impact of the newcomers' ecosystem-wide community standing and their prior direct collaborations with their pull request integrator, increasing the likelihood of pull request acceptance with 15% and 33%, respectively. Interestingly, whereas the impact of community standing is 3% lower than intraproject experience for non-newcomers, direct collaboration with their pull request integrator remains a stronger indicator of pull request acceptance, outweighing intra-project contributions by 14%. This highlights the importance of ecosystem-wide collaboration for non-newcomers.

We further observe that all three ecosystem contribution types (i.e., upstream, downstream, and non-dependency contributions) affect pull request decisions 8–11% more than intra-project contributions. Additionally, upstream contributions exceed downstream contributions by 2.3%, which is the opposite for non-newcomers. An explanation for this phenomenon is that experience acquired through upstream contributions translates better to downstream projects compared to the inverse. For example, implementing an updated API in a downstream project largely requires knowledge of the API and local knowledge of where it is implemented, whereas adding a new feature for an upstream project requires in-depth knowledge of that project. This could suggest that having contributed to a project that the focal project is aware of (a project that they have integrated), might yield an increased level of trust initially. However, as the developer continues to contribute, picking up more difficult tasks, its impact decreases. Similarly, changes proposed to upstream projects might be more complicated.

Interestingly, although the impact of ecosystem-wide variables shown in Table 3b is high, predicting the outcome of pull requests submitted by newcomers remains difficult, as Table 4 only reports an F1 score of 0.83. This score is noticeably lower than the classification performance of non-newcomers, having an F1 score of 0.96. However, including ecosystem-wide factors increased the classification performance by 2.3% and is noticeably higher than the impact for non-newcomers, increasing only 0.2%. We conclude that ecosystem-wide collaboration and contribution factors are a meaningful addition to models predicting the outcome of pull requests submitted by newcomers regardless of the generally lower prediction performance, compared to pull requests submitted by non-newcomers.

Our results support the hypothesis presented by Rastogi and Gousios (2021) showing that ecosystem-wide experience positively affects the success of newcomers in software projects, as the results suggest the positive impact of 1) prior domain knowledge and 2) a connection built with the development

team before proposing a change. Therefore, although participating in intraproject activities might help shed light on implementation considerations that otherwise might have been overlooked, explain the community development guidelines, or act as a means to establish trust with other members, similar socio-technical factors exist at an ecosystem level, such that the competencies acquired in one project naturally translate themselves to another in the same ecosystem. This is especially true for newcomers as they have limited knowledge of the focal project, inherently depending on everything outside it.

Answering RQ₄: Our results suggest that ecosystem-wide contributions in upstream, downstream and non-dependency projects positively affect newcomers' pull request decisions and are a stronger indicator of pull request acceptance than non-coding participation inside the project. Further, we see that direct collaboration with your pull request integrator in the ecosystem and a newcomer's ecosystem-wide community standing strongly and positively affects their pull request decisions. This underlines the existence and importance of ecosystem-wide socio-technical factors in open-source software development, which can translate themselves across projects in the same ecosystem.

6 Implications

The study is relevant to research and practice as the ecosystem perspective is becoming more critical in software engineering. This is due to the increased emphasis on software supply chains and the intricate dependencies between the many systems of today. It is paramount for organizations and individuals to understand how people engage across the entire ecosystem, not only in projects.

Our results show that upstream/downstream/non-dependency ecosystem-wide experience and collaboration are important factors in pull request decisions, especially for newcomers. We highlight the importance of non-coding contributions, like discussions in issue-tracking systems. Although these address notably different goals in software projects than coding activities, they strengthen skills developers need to collaborate in the broader ecosystem. Organizations can leverage these results to identify the group of external developers who can contribute to their projects. The NPM community can learn from other OSS ecosystems, as previous work has suggested that collaboration between projects is less common in the NPM ecosystem (Bogart et al., 2016, 2021). They could strengthen cross-project cooperation and reciprocity, allowing projects to grow as an ecosystem rather than separate entities in a shared environment.

Contributors (including project newcomers) can use these socio-technical relationships to navigate the ecosystem. For example, it might be beneficial to join a project in the ecosystem's periphery and move to more central projects over time. Our results suggest that pull requests integrated by a professional acquaintance are substantially more likely accepted. This suggests a valuable

strategy to navigate an ecosystem: to join the projects your professional peers already partake in. Combined with recommended intra-project joining behaviors (e.g., Steinmacher et al. (2019b), Gharehyazie et al. (2013, 2015) and Carillo et al. (2017)), like participating in project discussions before proposing a change and submitting a pull request, could further speed up the project onboarding process. This could greatly benefit people having difficulty joining the community, whereas they would improve its productivity, diversity, inclusion, and overall well-being.

The ecosystem perspective affects research discussing barriers that new-comers experience when joining projects (Geiger et al., 2021; Rastogi et al., 2015; Rehman et al., 2022; Steinmacher et al., 2015, 2018, 2019a,b; Wermke et al., 2022) because these barriers are observably lowered for experienced new-comers, as indicated by the results of our work. Future research should address the extent to which developers migrate across open-source ecosystems, identify the best practices and common downfalls of this process, and explore the benefits and drawbacks of these migrations on open-source projects. For example, continued collaboration between developers inside a project (i.e., when they have worked together a lot in that project) has been linked to bug proneness by Chen et al. (2024). This could translate to an ecosystem level, such that continued collaboration in one project introduces risks in another project they work on together. Beyond yielding insights into how ecosystem-wide collaboration changes software projects as single entities (Rastogi and Gousios, 2021), it also teaches us how software ecosystems change as a whole.

7 Threats to Validity

Internal Validity

The core collection of the projects studied in this work was selected based on their popularity in the NPM ecosystem. Although this set was complemented by dependency projects, all of which are less popular, it does not ensure that the general representation of NPM projects improved. Additionally, this method introduced a stark skewness in the distribution of pull requests across projects, such that 2% of the projects contained 49% of the pull requests. By randomly sampling these pull requests, we reduced the impact of this inequality.

People commonly use multiple aliases in platforms like GitHub (Wiese et al., 2016), which can substantially affect the structure of social networks. Although various solutions exist to merge aliases (Amreen et al., 2020; Vasilescu et al., 2015), none were applied due to stringent data requirements or the large amount of manual labor required to accurately apply these at an ecosystem scale. Furthermore, although this study intended to only include activities performed by humans, filtering out anything performed by bots using previous work (Dey et al., 2020a,b; Golzadeh et al., 2020a, 2021) as well as manual

analysis of users with over 400 pull requests, we cannot assure all bots were removed.

The dataset created by Katz (2020) is the foundation of this study because of the large amount of archival data it contains. Our study inherits the flaws in this dataset, one of which is the accuracy of the identified dependencies. The data represents a snapshot without regard for changes happening over time. Therefore, although the listed dependencies were correct at some point, they might not accurately represent the project's real dependencies over time (Decan et al., 2019). Although sometimes projects do not list their dependencies, this did not impact our results as this was only the case in 2% of the projects in our dataset.

Construct Validity

We employed two social network analysis metrics to measure direct collaboration and ecosystem-wide community standing: link strength and node centrality. Although the latter can be measured in many ways, we applied second-order degree centrality because this has much lower computational complexity than commonplace metrics, like Eigenvector centrality (Newman, 2018). Although our metric is coarser, it is representative as it captures the impact of a collaborator's neighbors, which is essential when determining the node's centrality (Newman, 2018). Further, this study identified a relationship between ecosystem-wide variables and pull request decisions. To ensure our results reflect reality, several control variables recommended by Zhang et al. (2022a) were included, allowing us to contextualize our results. We omitted one recommended variable regarding continuous integration because it is difficult to collect this data. However, after studying the results of Zhang et al. (2022a) in detail, we have no reason to believe this affects our results.

External Validity

This work addresses NPM, for which our conclusions might not represent other ecosystems. Regardless, we think these results could translate to other packaging ecosystems (e.g., PyPI or Maven), as they are similar to NPM. Further, although Valiev et al. (2018) identified the relevance of cascading dependencies (Geiger et al., 2021). We did not consider this because little is known about it in the literature and the complexity it would add to our analysis. It could be explicitly addressed in future work. Consequently, our results are limited to only direct dependencies between projects.

8 Conclusion

Our study objective was to explore the impact of ecosystem-wide experience and collaboration on pull request acceptance in NPM. It is crucial to understand this relationship for two reasons. One, software projects are not developed in isolation (Decan et al., 2019). Therefore, to understand software

development better, we must understand the software ecosystem it comprises. Two, developers and projects seek meaningful ways to engage in open-source software development to attract new contributors. Our study explores four topics: 1) general coding and non-coding ecosystem-wide experience, 2) experience in upstream/downstream projects, 3) direct collaboration and community standing in ecosystems, and 4) the impact of ecosystem-wide experience and collaboration on pull requests submitted by project newcomers (i.e., contributors who have not submitted a pull request in the past that was accepted into the project). We investigate these relationships on 1.8 million pull requests and 2.1 million issues from 20,052 projects of the NPM ecosystem.

The results show that ecosystem-wide experience and collaboration positively affect pull request acceptance decisions. In general, ecosystem-wide metrics do not exceed the importance of developers' intra-project experience, except for prior collaboration in the ecosystem between the pull request integrator and submitter. The opposite is true for project newcomers, as the impact of all ecosystem-wide variables exceeded that of intra-project contributions like participating in discussions in the issue-tracking system. The results show that ecosystem-wide experience and collaboration factors complement previously known factors when predicting the outcomes of pull requests. Our metrics increased their classification performance between 0.2% and 2.3%, yielding an overall F1 score of 0.83 for pull requests submitted by project newcomers, 0.96 for non-newcomers, and an overall F1 score of 0.92.

Conflict of Interest

The authors have no competing interests to declare that are relevant to the content of this article.

Data Availability Statement

The replication package of this paper can be found on Zenodo (Meijer et al., 2024) (or click the following link: zenodo.org/records/13286685).

Acknowledgements We thank Dániel Varró for his insightful feedback and suggestions. His expertise and constructive comments have been greatly appreciated and were crucial in shaping the final version of this paper.

References

Alshara Z, Shatnawi A, Eyal-Salman H, Seriai AD, Shatnawi M (2023) PIlink: A ground-truth dataset of links between pull-requests and issues in GitHub. IEEE Access 11:697–710, DOI https://doi.org/10.1109/ACCESS. 2022.3232982

- Amreen S, Mockus A, Zaretzki R, Bogart C, Zhang Y (2020) ALFAA: Active learning fingerprint based anti-aliasing for correcting developer identity errors in version control systems. Empir Softw Eng 25(2):1136–1167, DOI https://doi.org/10.1007/s10664-019-09786-7
- Barabási AL, Albert R (1999) Emergence of scaling in random networks. Science 286(5439):509–512, DOI https://doi.org/10.1126/science.286.5439.509
- Baysal O, Kononenko O, Holmes R, Godfrey MW (2012) The secret life of patches: A firefox case study. In: Ninteenth Working Conference on Reverse Engineering, pp 447–455, DOI https://doi.org/10.1109/WCRE.2012.54
- Baysal O, Kononenko O, Holmes R, Godfrey MW (2016) Investigating technical and non-technical factors influencing modern code review. Empir Softw Eng 21(3):932–959, DOI https://doi.org/10.1007/s10664-015-9366-8
- Bogart C, Kästner C, Herbsleb J, Thung F (2016) How to break an API: Cost negotiation and community values in Three Software Ecosystems. In: Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering, pp 109–120, DOI https://doi.org/10.1145/2950290.2950325
- Bogart C, Kästner C, Herbsleb J, Thung F (2021) When and how to make breaking changes: Policies and practices in 18 open source software ecosystems. Trans Softw Eng Methodol 30(4):42:1–42:56, DOI https://doi.org/10.1145/3447245
- Breiman L (2001) Random forests. Machine Learning 45(1):5–32, DOI https://doi.org/10.1023/A:1010933404324
- Bródka P, Kazienko P, Musiał K, Skibicki K (2012) Analysis of neighbourhoods in multi-layered dynamic social networks. Int J Comput Intell Syst 5(3):582–596, DOI https://doi.org/10.1080/18756891.2012.696922
- Carillo K, Huff S, Chawner B (2017) What makes a good contributor? understanding contributor behavior within large free/open source software projects A socialization perspective. J Strateg Inf Syst 26(4):322–359, DOI https://doi.org/10.1016/j.jsis.2017.03.001
- Casalnuovo C, Vasilescu B, Devanbu P, Filkov V (2015) Developer onboarding in GitHub: The role of prior social links and language experience. In: Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering, pp 817–828, DOI https://doi.org/10.1145/2786805.2786854
- Casola V, Fasolino A, Mazzocca N, Tramontana P (2009) An AHP-based framework for quality and security evaluation. In: International Conference on Computational Science and Engineering, vol 3, pp 405–411, DOI https://doi.org/10.1109/CSE.2009.391
- Celińska D (2018) Coding together in a social network: Collaboration among GitHub users. In: Proceedings of the 9th International Conference on Social Media and Society, pp 31–40, DOI https://doi.org/10.1145/3217804. 3217895
- Chen D, Lü L, Shang MS, Zhang YC, Zhou T (2012) Identifying influential nodes in complex networks. Physica 391(4):1777-1787, DOI https://doi.org/10.1016/j.physa.2011.09.017

- Chen HM, Kazman R, Catolino G, Manca M, Tamburri DA, Van Den Heuvel WJ (2024) An empirical study of social debt in open-source projects: Social drivers and the "known devil" community smell. In: Proceedings of the 57th Hawaii International Conference on System Sciences, URL https://hdl.handle.net/10125/107255
- Cheng C, Li B, Li ZY, Zhao YQ, Liao FL (2017) Developer role evolution in open source software ecosystem: An explanatory study on GNOME. J Comput Sci Technol 32(2):396–414, DOI https://doi.org/10.1007/s11390-017-1728-9
- Chidambaram N, Mazrae PR (2022) Bot detection in GitHub repositories. In: Proceedings of the 19th International Conference on Mining Software Repositories, pp 726–728, DOI https://doi.org/10.1145/3524842.3528520
- Cook RD (2000) Detection of influential observation in linear regression. Technometrics 42(1):65–68, DOI https://doi.org/10.1080/00401706.2000. 10485981
- Cánovas Izquierdo JL, Cabot J (2021) On the analysis of non-coding roles in open source development. Empir Softw Eng 27(1):18, DOI https://doi.org/10.1007/s10664-021-10061-x
- Decan A, Mens T, Grosjean P (2019) An empirical comparison of dependency network evolution in seven software packaging ecosystems. Empir Softw Eng 24(1):381–416, DOI https://doi.org/10.1007/s10664-017-9589-y
- Dey T, Mockus A (2020) Effect of technical and social factors on pull request quality for the NPM ecosystem. In: Proceedings of the 14th ACM / IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM), pp 1–11, DOI https://doi.org/10.1145/3382494.3410685
- Dey T, Mousavi S, Ponce E, Fry T, Vasilescu B, Filippova A, Mockus A (2020a) A dataset of bot commits. DOI https://doi.org/10.5281/zenodo.3694401
- Dey T, Mousavi S, Ponce E, Fry T, Vasilescu B, Filippova A, Mockus A (2020b) Detecting and characterizing bots that commit code. In: Proceedings of the 17th International Conference on Mining Software Repositories, pp 209–219, DOI https://doi.org/10.1145/3379597.3387478
- Dueñas S, Cosentino V, Gonzalez-Barahona JM, Felix AdCS, Izquierdo-Cortazar D, Cañas-Díaz L, García-Plaza AP (2021) GrimoireLab: A toolset for software development analytics. Peer J Comput Sci 7:e601, DOI https://doi.org/10.7717/peerj-cs.601
- Fang H, Herbsleb J, Vasilescu B (2023) Matching skills, past collaboration, and limited competition: Modeling when open-source projects attract contributors. In: Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, pp 42–54, DOI https://doi.org/10.1145/3611643.3616282
- Fershtman C, Gandal N (2011) Direct and indirect knowledge spillovers: The "social network" of open-source projects. RAND J Econ 42(1):70–91, DOI https://doi.org/10.1111/j.1756-2171.2010.00126.x
- Forte A, Lampe C (2013) Defining, understanding, and supporting open collaboration: Lessons from the literature. Am Behav Sci 57(5):535–547, DOI https://doi.org/10.1177/0002764212469362

- Franco-Bedoya O, Ameller D, Costal D, Franch X (2017) Open source software ecosystems: A systematic mapping. Inf Softw Technol 91:160–185, DOI https://doi.org/10.1016/j.infsof.2017.07.007
- Geiger RS, Howard D, Irani L (2021) The labor of maintaining and scaling free and open-source software projects. Proceedings of the ACM on Human-Computer Interaction 5:175:1–175:28, DOI https://doi.org/10.1145/3449249
- Gharehyazie M, Posnett D, Filkov V (2013) Social activities rival patch submission for prediction of developer initiation in OSS projects. In: 2013 IEEE International Conference on Software Maintenance, pp 340–349, DOI https://doi.org/10.1109/ICSM.2013.45
- Gharehyazie M, Posnett D, Vasilescu B, Filkov V (2015) Developer initiation and social interactions in OSS: A case study of the apache software foundation. Empir Softw Eng 20(5):1318–1353, DOI https://doi.org/10.1007/s10664-014-9332-x
- Golzadeh M, Decan A, Mens T (2019) On the Effect of Discussions on Pull Request Decisions. In: Proceedings of the 18th Belgium-Netherlands Software Evolution Workshop, URL https://ceur-ws.org/Vol-2605/16.pdf
- Golzadeh M, Decan A, Legay D, Mens T (2020a) A ground-truth dataset to identify bots in GitHub. DOI https://doi.org/10.5281/zenodo.4000388
- Golzadeh M, Legay D, Decan A, Mens T (2020b) Bot or not? Detecting bots in GitHub pull request activity based on comment similarity. In: Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops, pp 31–35, DOI https://doi.org/10.1145/3387940.3391503
- Golzadeh M, Decan A, Legay D, Mens T (2021) A ground-truth dataset and classification model for detecting bots in GitHub issue and PR comments. J Syst Softw 175:110911, DOI https://doi.org/10.1016/j.jss.2021.110911
- Gousios G, Pinzger M, Deursen Av (2014) An exploratory study of the pull-based software development model. In: Proceedings of the 36th International Conference on Software Engineering, pp 345–355, DOI https://doi.org/10.1145/2568225.2568260
- Gousios G, Zaidman A, Storey MA, Deursen Av (2015) Work practices and challenges in pull-based development: The integrator's perspective. In: 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering, vol 1, pp 358–368, DOI https://doi.org/10.1109/ICSE.2015.55
- Hahn J, Moon JY, Zhang C (2006) Impact of social ties on open source project team formation. In: Open Source Syst, pp 307–317, DOI https://doi.org/10.1007/0-387-34226-5 31
- Hahn J, Moon JY, Zhang C (2008) Emergence of new project teams from open source software developer networks: Impact of prior collaboration ties. Inf Syst Res 19(3):369–391, DOI https://doi.org/10.1287/isre.1080.0192
- Hanssen GK, Dybå T (2012) Theoretical foundations of software ecosystems. In: Proceedings of the Fourth International Workshop on Software Ecosystems, URL https://ceur-ws.org/Vol-879/paper1.pdf
- Holme P, Saramäki J (2019) A map of approaches to temporal networks. In: Temporal Network Theory, Springer International Publishing, pp 1–24, DOI

- $https://doi.org/10.1007/978\text{-}3\text{-}030\text{-}23495\text{-}9_1$
- Iyer RN, Yun SA, Nagappan M, Hoey J (2021) Effects of personality traits on pull request acceptance. Trans Soft Eng 47(11):2632–2643, DOI https://doi.org/10.1109/TSE.2019.2960357
- Jergensen C, Sarma A, Wagstrom P (2011) The onion patch: Migration in open source ecosystems. In: Proceedings of the 19th ACM SIGSOFT symposium and the 13th European conference on Foundations of software engineering, pp 70–80, DOI https://doi.org/10.1145/2025113.2025127
- Katz J (2020) Libraries.io open source repository and dependency metadata. DOI https://doi.org/10.5281/zenodo.3626071
- Khadke N. Teh MH, (2012)Predicting ceptance of GitHub pull requests. Stanford Univer-URL https://cs229.stanford.edu/proj2012/ sity, Khadke Teh Shen-Predicting Acceptance Of Git HubPull Requests.pdf
- Kivelä M, Arenas A, Barthelemy M, Gleeson JP, Moreno Y, Porter MA (2014) Multilayer networks. J Complex Netw 2(3):203–271, DOI https://doi.org/10.1093/comnet/cnu016
- Kononenko O, Rose T, Baysal O, Godfrey M, Theisen D, de Water B (2018) Studying pull request merges: A case study of shopify's active merchant. In: Proceedings of the 40th International Conference on Software Engineering: Software Engineering in Practice, pp 124–133, DOI https://doi.org/10.1145/3183519.3183542
- Kovalenko V, Bacchelli A (2018) Code review for newcomers: Is it different? In: Proceedings of the 11th International Workshop on Cooperative and Human Aspects of Software Engineering, pp 29–32, DOI https://doi.org/10.1145/3195836.3195842
- Lee A, Carver JC (2017) Are one-time contributors different? A comparison to core and periphery developers in FLOSS repositories. In: 2017 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM), pp 1–10, DOI https://doi.org/10.1109/ESEM.2017.7
- Legay D, Decan A, Mens T (2019) On the impact of pull request decisions on future contributions. In: Belgium-Netherlands Software Evolution Workshop, URL https://ceur-ws.org/Vol-2361/short12.pdf
- Maeprasart V, Wattanakriengkrai S, Kula RG, Treude C, Matsumoto K (2023) Understanding the role of external pull requests in the NPM ecosystem. Empir Softw Eng 28(4):84, DOI https://doi.org/10.1007/s10664-023-10315-w
- Meijer W (2023) The influence of ecosystem-wide experience and collaboration on pull request acceptance in open-source software ecosystems. Master's thesis, University of Groningen, URL https://fse.studenttheses.ub.rug.nl/31331/
- Meijer W, Riveni M, Rastogi A (2024) Replication package (obfuscated): Ecosystem-wide influences on pull request decisions. URL https://tinyurl.com/5449javb
- Méndez-Durón R, García CE (2009) Returns from social capital in open source software networks. J Evol Econ 19(2):277–295, DOI https://doi.org/10.1007/s00191-008-0125-5

- Müller M, Rosenkranz C (2023) The role of dependency networks in developer participation decisions in open source software ecosystems: An application of stochastic actor-oriented models. In: Proceedings of the 56th Hawaii International Conference on System Sciences, pp 689–698, URL https://hdl.handle.net/10125/102715
- Newman M (2018) Measures and metrics. In: Networks, Oxford University Press, pp 158–217, DOI https://doi.org/10.1093/oso/9780198805090.003. 0007
- Palyart M, Murphy GC, Masrani V (2018) A study of social interactions in open source component use. Trans Soft Eng 44(12):1132–1145, DOI https://doi.org/10.1109/TSE.2017.2756043
- Panichella S, Bavota G, Penta MD, Canfora G, Antoniol G (2014) How developers' collaborations identified from different sources tell us about code changes. In: 2014 IEEE International Conference on Software Maintenance and Evolution, pp 251–260, DOI https://doi.org/10.1109/ICSME.2014.47
- Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay É (2011) Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research 12(85):2825—2830, URL http://jmlr.org/papers/v12/pedregosa11a.html
- Peng G, Wan Y, Woodlock P (2013) Network ties and the success of open source software development. J Strateg Inf Syst 22(4):269–281, DOI https://doi.org/10.1016/j.jsis.2013.05.001
- Pérez-Verdejo JM, Sánchez-García AJ, Ocharán-Hernández JO, Mezura-Montes E, Cortés-Verdín K (2021) Requirements and GitHub issues: An automated approach for quality requirements classification. Program Comput Softw 47(8):704–721, DOI https://doi.org/10.1134/S0361768821080193
- Pinto G, Dias LF, Steinmacher I (2018) Who gets a patch accepted first? Comparing the contributions of employees and volunteers. In: Proceedings of the 11th International Workshop on Cooperative and Human Aspects of Software Engineering, pp 110–113, DOI https://doi.org/10.1145/3195836. 3195858
- Qiu HS, Nolte A, Brown A, Serebrenik A, Vasilescu B (2019) Going farther together: The impact of social capital on sustained participation in open source. In: 2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE), pp 688–699, DOI https://doi.org/10.1109/ICSE. 2019.00078
- Rastogi A (2016) Do biases related to geographical location influence work-related decisions in GitHub? In: 2016 IEEE/ACM 38th International Conference on Software Engineering Companion (ICSE-C), pp 665–667, DOI https://doi.org/10.1145/2889160.2891035
- Rastogi A, Gousios G (2021) How does software change? DOI https://doi.org/ $10.48550/\mathrm{arXiv.}2106.01885$
- Rastogi A, Thummalapenta S, Zimmermann T, Nagappan N, Czerwonka J (2015) Ramp-up journey of new hires: Tug of war of aids and impediments. In: 2015 ACM/IEEE International Symposium on Empirical Software Engi-

- neering and Measurement (ESEM), pp 1–10, DOI https://doi.org/10.1109/ESEM.2015.7321212
- Rastogi A, Nagappan N, Gousios G, van der Hoek A (2018) Relationship between geographical location and evaluation of developer contributions in Github. In: Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, pp 1–8, DOI https://doi.org/10.1145/3239235.3240504
- Rehman I, Wang D, Kula RG, Ishio T, Matsumoto K (2022) Newcomer OSS-candidates: Characterizing contributions of novice developers to GitHub. Empir Softw Eng 27(5):109, DOI https://doi.org/10.1007/s10664-022-10163-0
- Rombaut B, Cogo FR, Adams B, Hassan AE (2023) There's no such thing as a free lunch: Lessons learned from exploring the overhead introduced by the Greenkeeper dependency bot in NPM. Trans Softw Eng Methodol 32(1):11:1–11:40, DOI https://doi.org/10.1145/3522587
- Schreiber RR, Zylka MP (2020) Social network analysis in software development projects: A systematic literature review. Int J Softw Eng Knowl Eng 30(03):321–362, DOI https://doi.org/10.1142/S021819402050014X
- Shah SK (2006) Motivation, governance, and the viability of hybrid forms in open source software development. Manag Sci 52(7):1000–1014, DOI https://doi.org/10.1287/mnsc.1060.0553
- Soares DM, De Lima Junior ML, Murta L, Plastino A (2015a) Rejection factors of pull requests filed by core team developers in software projects with high acceptance rates. In: 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA), pp 960–965, DOI https://doi.org/10.1109/ICMLA.2015.41
- Soares DM, de Lima Júnior ML, Murta L, Plastino A (2015b) Acceptance factors of pull requests in open-source projects. In: Proceedings of the 30th Annual ACM Symposium on Applied Computing, pp 1541–1546, DOI https://doi.org/10.1145/2695664.2695856
- Soliman M, Galster M, Avgeriou P (2021) An exploratory study on architectural knowledge in issue tracking systems. In: Software Architecture, pp 117–133, DOI https://doi.org/10.1007/978-3-030-86044-8 8
- Soto M, Coker Z, Le Goues C (2017) Analyzing the impact of social attributes on commit integration success. In: 2017 IEEE/ACM 14th International Conference on Mining Software Repositories (MSR), pp 483–486, DOI https://doi.org/10.1109/MSR.2017.34
- de Souza CR, Figueira Filho F, Miranda M, Ferreira RP, Treude C, Singer L (2016) The social side of software platform ecosystems. In: Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems, pp 3204–3214, DOI https://doi.org/10.1145/2858036.2858431
- Steinmacher I, Conte T, Gerosa MA, Redmiles D (2015) Social barriers faced by newcomers placing their first contribution in open source software projects. In: Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing, pp 1379–1392, DOI https://doi.org/10.1145/2675133.2675215

- Steinmacher I, Pinto G, Wiese IS, Gerosa MA (2018) Almost there: A study on quasi-contributors in open source software projects. In: Proceedings of the 40th International Conference on Software Engineering, pp 256–266, DOI https://doi.org/10.1145/3180155.3180208
- Steinmacher I, Gerosa M, Conte TU, Redmiles DF (2019a) Overcoming social barriers when contributing to open source software projects. Comput Support Cooperative Work 28(1):247–290, DOI https://doi.org/10.1007/s10606-018-9335-z
- Steinmacher I, Treude C, Gerosa MA (2019b) Let me in: Guidelines for the successful onboarding of newcomers to open source projects. IEEE Software 36(4):41–49, DOI https://doi.org/10.1109/MS.2018.110162131
- Stickgold E, Lofdahl C, Farry M (2013) Trust metrics and results for social media analysis. In: Social Computing, Behavioral-Cultural Modeling and Prediction, pp 458–465, DOI https://doi.org/10.1007/978-3-642-37210-0_50
- Subramanian VN, Rehman I, Nagappan M, Kula RG (2022) Analyzing first contributions on GitHub: What do newcomers do? IEEE Software 39(1):93–101, DOI https://doi.org/10.1109/MS.2020.3041241
- Syeed MMM, Hansen KM, Hammouda I, Manikas K (2014) Socio-technical congruence in the Ruby ecosystem. In: Proceedings of The International Symposium on Open Collaboration, pp 1–9, DOI https://doi.org/10.1145/2641580.2641586
- Terrell J, Kofink A, Middleton J, Rainear C, Murphy-Hill E, Parnin C, Stallings J (2017) Gender differences and bias in open source: Pull request acceptance of women versus men. PeerJ Comput Sci 3:e111, DOI https://doi.org/10.7717/peerj-cs.111
- Trinkenreich B, Guizani M, Wiese I, Sarma A, Steinmacher I (2020) Hidden figures: roles and pathways of successful OSS contributors. Proceedings of the ACM on Human-Computer Interaction 4(CSCW2):180:1–180:22, DOI https://doi.org/10.1145/3415251
- Trinkenreich B, Guizani M, Wiese I, Conte T, Gerosa M, Sarma A, Steinmacher I (2022) Pots of gold at the end of the rainbow: What is success for open source contributors? Trans Soft Eng 48(10):3940–3953, DOI https://doi.org/10.1109/TSE.2021.3108032
- Tsay J, Dabbish L, Herbsleb J (2014) Influence of social and technical factors for evaluating contribution in GitHub. In: Proceedings of the 36th International Conference on Software Engineering, pp 356–366, DOI https://doi.org/10.1145/2568225.2568315
- Valiev M, Vasilescu B, Herbsleb J (2018) Ecosystem-level determinants of sustained activity in open-source projects: A case study of the PyPI ecosystem. In: Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, pp 644–655, DOI https://doi.org/10.1145/3236024.3236062
- Vasilescu B, Serebrenik A, Filkov V (2015) A data set for social diversity studies of GitHub teams. In: 2015 IEEE/ACM 12th Working Conference on Mining Software Repositories, pp 514–517, DOI https://doi.org/10.1109/

MSR.2015.77

- Wang J (2012) Survival factors for free open source software projects: A multi-stage perspective. Eur Manag J 30(4):352–371, DOI https://doi.org/10.1016/j.emj.2012.03.001
- Wattanakriengkrai S, Wang D, Kula RG, Treude C, Thongtanunam P, Ishio T, Matsumoto K (2023) Giving back: Contributions congruent to library dependency changes in a software ecosystem. Trans Soft Eng 49(4):2566–2579, DOI https://doi.org/10.1109/TSE.2022.3225197
- Wermke D, Wöhler N, Klemmer JH, Fourné M, Acar Y, Fahl S (2022) Committed to trust: A qualitative study on security & trust in open source software projects. In: 2022 IEEE Symposium on Security and Privacy (SP), pp 1880–1896, DOI https://doi.org/10.1109/SP46214.2022.9833686
- Wiese IS, Da Silva JT, Steinmacher I, Treude C, Gerosa MA (2016) Who is who in the mailing list? Comparing six disambiguation heuristics to identify multiple addresses of a participant. In: 2016 IEEE International Conference on Software Maintenance and Evolution (ICSME), pp 345–355, DOI https://doi.org/10.1109/ICSME.2016.13
- Yu Y, Yin G, Wang T, Yang C, Wang H (2016) Determinants of pull-based development in the context of continuous integration. Sci China Inf Sci 59(8):080104, DOI https://doi.org/10.1007/s11432-016-5595-8
- Zampetti F, Bavota G, Canfora G, Penta MD (2019) A study on the interplay between pull request review and continuous integration builds. In: 2019 IEEE 26th International Conference on Software Analysis, Evolution and Reengineering (SANER), pp 38–48, DOI https://doi.org/10.1109/SANER. 2019.8667996
- Zhang X, Yu Y, Gousios G, Rastogi A (2022a) Pull request decisions explained: An empirical overview. Trans Soft Eng pp 1–26, DOI https://doi.org/10.1109/TSE.2022.3165056
- Zhang X, Yu Y, Wang T, Rastogi A, Wang H (2022b) Pull request latency explained: An empirical overview. Empir Softw Eng 27(6):126, DOI https://doi.org/10.1007/s10664-022-10143-4
- Zöller N, Morgan JH, Schröder T (2020) A topology of groups: What GitHub can tell us about online collaboration. Technol Forecast Soc Change 161:120291, DOI https://doi.org/10.1016/j.techfore.2020.120291