




Identifying Testing Behaviour in Open Source Projects: A Case Analysis for Apache Spark

Asli Sari¹^a, Ayse Tosun¹^b and Gülfem Işıklar Alptekin²^c

¹Faculty of Computer and Informatics Engineering, Istanbul Technical University, Turkey

²Department of Computer Engineering, Galatasaray University, Turkey

Keywords: Open Source Software, Software Testing, Human Factors, Pareto Analysis, Hierarchical Clustering, Apache Spark.

Abstract: Open Source Software (OSS) projects have the potential to achieve high software quality through community collaboration. However, the collaborative nature of OSS development presents unique challenges, particularly in maintaining software quality through testing practices. The lack of formal testing roles and structures underscores the importance of understanding testing patterns to enhance project quality. To address this need, our study investigates key aspects of testing contributions within Apache Spark project. The study aims to identify the top testing contributors responsible for the majority of test-related commits, as well as their engagement levels and evolving testing focus over time. Additionally, it examines how these contributors' activities vary across different time periods and explores their distinct engagement patterns within the community. Our findings reveal that only 9.8% of contributors handle the majority of test-related commits, exceeding the traditional 80/20 Pareto principle. Additionally, hierarchical clustering of these contributors over three years identified three activity levels: Highly-Active, Moderately-Active, and Lowly-Active. Each cluster exhibits unique patterns of testing focus and engagement across different time periods. These insights emphasize the critical role of a small core group in managing the project's testing workload and underscore the need for strategies to broaden participation in testing activities.


1 INTRODUCTION


Open Source Software (OSS) offers a valuable opportunity for individuals and organizations to explore, share, and modify software freely. This freedom encourages developers from diverse profiles and various experience levels to collaborate with other developers, driven by their shared interests, reputations, and professional needs. While the collaborative nature of OSS creates efficient and productive software development (Hao et al., 2008), it also presents challenges in maintaining software quality, as the diversity among contributors can complicate the consistency of OSS project standards (Abdou et al., 2012).


Testing is an essential aspect of ensuring the high quality of OSS products (Napoleão et al., 2020). The collaborative nature of OSS, where developers contribute voluntarily, introduces challenges in imple-

menting effective testing practices. For instance, OSS can be developed by volunteers who lack experience in quality engineering, or it can be released prematurely, without adequate testing or addressing critical bug reports (Abdou et al., 2012). There are several studies that have explored how to systematically model quality assurance processes for OSS projects. Notably, researchers such as Abdou *et al.* and Seigneur *et al.* have proposed conceptual frameworks for the OSS testing process (Abdou et al., 2012), (Seigneur, 2007). However, while these frameworks offer structured methodologies for testing, they do not explicitly define the specific roles and responsibilities of testers within OSS projects. This lack of clarity challenges the identification of contributors involved in testing activities and makes it difficult to assess how testing responsibilities and contributions are distributed among participants in OSS development.

In traditional software product teams, testing responsibilities are distributed among members according to their relevant skill sets, with specific roles assigned for different types of testing. For example, the

^a <https://orcid.org/0009-0009-8348-5747>

^b <https://orcid.org/0000-0003-1859-7872>

^c <https://orcid.org/0000-0003-0146-1581>

test leader is critical to the conduct of downstream testing, such as integration and system testing (Desikan and Ramesh, 2006). A test engineer is expected to demonstrate competence in handling categorized defects and develop high-quality documentation (Desikan and Ramesh, 2006). However, there is currently a lack of empirical studies regarding such explicit assignment roles in OSS. In other words, testers and developers are not specifically designated as testers or developers; instead, participants can shift between testing and development teams as needed. Developers typically engage a common series of actions when working on the software source, for instance, developing and testing the code within their local copy of the source (Mockus et al., 2002). This absence of formal role designation highlights the significance of characterizing the testing behavior of contributors from diverse profiles and backgrounds. Therefore, examining how contributors engage in testing and identifying those who are highly involved in test-related tasks provides a better structure for testing practices in OSS projects.

To address this gap, we focus on Apache Spark, an open-source data analytics platform designed for distributed data-sharing. Numerous research papers have explored various aspects of Apache projects, including investigations of merged pull requests to characterize refactoring activities (Coelho et al., 2021) and analyses of the relationship between commits and their associated source files (Li et al., 2021). However, a notable gap exists in understanding the testing behaviors of contributors. We selected Apache Spark for its use of Apache Maven, which provides a standardized directory layout that separates test code from source code, facilitating the distinction between software development and testing activities. Additionally, the project maintains well-documented connections between modified files and their associated commits, providing a reliable basis for analysis. The diverse community of contributors offers an excellent opportunity to examine testing patterns. This study aims to explore testing patterns and identify potential testers within the Apache Spark project. To guide our investigation, we propose two research questions.

RQ1: How are testing contributions distributed among contributors in Apache Spark project?

We conducted a Pareto analysis of testing activities to pinpoint the key contributors who are responsible for the majority of testing activities within the Apache Spark project. This analysis enabled us to identify and designate this core group of contributors as *top testing contributors* for our study.

RQ2: How do the testing activities of contributors change over time?

Given the long-term nature of the Apache Spark project, which has spanned over a decade, we recognize that the testing behaviors of contributors may change over time. Consequently, some contributors we currently identify as top testing contributors may not be actively involved in recent activities. Additionally, those acknowledged as top testing contributors in our RQ1 may not demonstrate consistent testing involvement throughout the project's lifecycle. To address these temporal variations, we conducted an investigation into their activity levels and how the testing activities of top contributors have evolved over time. Our focus was on a limited but recent three-year period (inspired by (Cheng and Guo, 2019)) to capture both activity levels and contemporary testing activities. We clustered the 159 top testing contributors based on their total commits during this time-frame and examined their activity levels and temporal testing contributions over time, analyzing the test commit ratio as the count of test commits relative to total commits.

The structure of the paper is as follows: Section 2 reviews related work on contributor behaviors in OSS projects, the challenges of OSS testing, and the identification of test contributors in software projects. Section 3 outlines our research methodology for analyzing commits to identify test-related activities, the levels of activity levels concerning project involvement, and examining testing focus over various time periods. Section 4 presents our findings. Section 5 discusses the implications of our results for identifying the role of testers. Section 6 addresses potential threats to validity. Finally, Section 7 concludes the paper, highlighting key insights and proposing directions for future research.

2 RELATED WORK

There are numerous participants involved in OSS projects, each exhibiting varying levels of engagement, yet their specific roles related to software engineering remain undefined. This section reviews existing research pertinent to our objectives, highlights individuals who significantly impact test-related activities, and identifies the most active contributors in testing. Section 2.1 explores methods for identifying contributor roles within OSS projects. Section 2.2 addresses the testing challenges encountered in these OSS initiatives. Additionally, Section 2.3 focuses on identifying test contributors in the software projects.

2.1 Contributor Roles and Behaviors in OSS

Open source is a collaborative endeavor that brings together various contributors with diverse backgrounds and profiles that extend beyond the clearly defined roles typical of traditional software development. Several studies have highlighted this collaborative nature and offered frameworks for categorizing contributors into over-arching roles. For instance, core members play a significant role in the OSS project with authority to accept or reject (Ducheneaut, 2005), while peripheral contributors sequentially achieve more central roles as they gain experience and visibility (Trinkenreich et al., 2020). Additionally, casual contributors participate in the project with minimal knowledge and engagement (Pinto et al., 2016). However, such overarching categorization in OSS is complex and not always straightforward, particularly when the diverse backgrounds of the contributors develop the OSS project. This complexity highlights the need to expand our horizons beyond merely the technical aspects of contribution structures, paving the way for a more nuanced classification of roles that surpasses these general categories.

Researchers focus on quantitative approaches to analyzing contributor behaviors within OSS projects through repository mining. For example, parsing messages in email archives and calculating the similarities among email messages are utilized to determine OSS participants' coordination and communication activities and the developer's social patterns among the peers in the OSS project (Bird et al., 2006). Another study examined commit sizes regarding the number of source lines of code added and removed to identify the code contribution behaviors of OSS participants (Arafat and Riehle, 2009). The commit messages from OSS participants were analyzed to categorize their maintenance activities within the project (Levin and Yehudai, 2017). While these studies highlight the exploration of the OSS contributor behaviors and categorize them through various metrics, there remains a research gap in identifying contributors who focus on testing activities.

2.2 Testing Challenges in OSS Projects

Testing activities in OSS projects pose unique challenges when compared to traditional software development. OSS projects as a whole hardly ever follow traditional software engineering paradigms (Morasca et al., 2011), which can restrict the applicability of well-established testing approaches in the open-source domain. Zhao and Elbaum conducted a sur-

vey to investigate the testing activities in OSS projects (Zhao and Elbaum, 2003). They found a significant gap in testing familiarity by revealing that only 5% of the respondents employed testing tools. Another survey indicated that testing activities are less emphasized in OSS compared to traditional software development (Tosi and Tahir, 2010). Specifically, over 40% of OSS products lack testing activities, and 67% of the projects do not have any test planning documentation. These findings indicate that testing activities are relatively underutilized in OSS. To tackle these challenges, several studies have proposed solutions such as testing guidelines and checklists, which outline a set of issues and recommend appropriate testing approaches (Morasca et al., 2011). However, a notable limitation of these proposed solutions is that they primarily emphasize testing methodologies, often overlooking the crucial aspect of identifying the contributors involved in testing activities. Without a clear understanding of who participates in testing within OSS projects, it becomes increasingly challenging to assess their overall impact or to implement meaningful improvements in testing practices.

2.3 Identifying Test Contributors in Software Projects

While existing studies have explored testing activities in OSS projects, there remains limited research on the testing profile, i.e., quantitative evaluation of OSS contributors' testing efforts. This gap significantly impacts our ability to identify the relevance of contributors' roles in relation to testing activities and their overall impact on projects. Therefore, there is a need for more systematic approaches to identify contributors engaged in testing. Establishing a structured evaluation of contributors' behaviors, particularly in testing, has become a cornerstone for not only recognizing their involvement but also for enhancing the effectiveness of test practices.

Some studies have explored the identification of contributors within software projects. For instance, Zhang *et al.* employed an online survey to define the specifications for testers in the software project (Zhang et al., 2020). Similarly, Beller *et al.* utilized the survey to determine whether and how individuals engage in software testing (Beller et al., 2015). While these studies offer valuable qualitative insights into testing involvement, they primarily depend on self-reported data, lacking a rigorous empirical analysis of actual testing contributions. This highlights a significant gap in the existing literature, emphasizing the need for more comprehensive research that quantitatively evaluates testing activities and system-

atically identifies contributors to these efforts. Our study aims to address this gap by analyzing commit data from open-source software (OSS) project, providing an empirical evaluation of contributors' behaviors in testing activities.

3 METHODOLOGY

This section outlines our systematic approach for identifying and analyzing the testing focuses of contributors within the Apache Spark project. We conduct both Pareto analyses on commits and a time-based examination of contribution patterns to gain insights into the participants involved in testing activities and how their behaviors evolve over time. The following subsections detail our data collection methodology and the statistical analysis techniques applied in this study.

3.1 Data Collection and Preparation

To explore contributor behaviors related specifically to testing, we conducted a detailed case study on the commit records of the Apache Spark project. The primary motivation for this study is to examine testing-related contributions within the real-world context of OSS development, using Apache Spark as a representative example. Each commit represents a contributor's interaction with the project repository, involving local changes submitted to the remote repository, such as adding, modifying, or removing lines of code or files (Chelkowski et al., 2016). To identify test-related files, we utilized Apache Spark's Maven standardized directory layout and conducted a manual review of the repository file organization on its GitHub page. We specifically targeted executable test files within commits, as these represent direct indicators of a contributor's testing activities. Our ultimate goal is to identify top testing contributors and their activity levels within the project, as well as the testing focus over time in Apache Spark, by assessing contributors' test commit activities.

To achieve this, we focus on the total number of commits and the subset that included executable test files as our primary units of analysis. Data collection was conducted using Github, enabling us to gather comprehensive data for each contributor to the Apache Spark project. The specific data items collected for each contributor are summarized in Table 1.

Table 1: Data items collected per contributor.

#	Name	Description
D1	CmtAuNa	Commit author name
D2	CmtAuMl	Commit author e-mail
D3	CmtNum	Number of total commits
D4	CmtTesNum	Number of commits including executable test files

3.2 Pareto Analysis on Apache Spark

We have effectively applied the Pareto 80/20 Rule, which asserts that a small number of causes (20%) are responsible for a large percentage (80%) of the effects to pinpoint top testing contributors within the Apache Spark project. This empirical approach enables thorough various aspects of human activity in the software engineering domain (Lipovetsky, 2009). For example, Geldenhuys investigated user activity in OSS projects by analyzing contribution patterns and concluded that about 20% of the participants in an OSS project contribute 80% of the work, highlighting the significant influence of a small core group (Geldenhuys, 2010). Another study shows that the exact ratio of the Pareto Principle varies, but it is applicable: a relatively small group drives the majority of work to commit activity, mailing activity, and bug activity (Goeminne and Mens, 2011). While these studies have shown variations in contribution distributions, revealing that not all metrics align with the 80/20 ratio, a corresponds to this rule approximately. These studies affirm the efficacy of the Pareto principle in analyzing various contribution patterns and identifying key contributors in OSS projects.

Notably, there remains an unexplored issue regarding the application of the Pareto Principle in the context of testing contributions. Given that Pareto Law has successfully identified key contributors in OSS projects via several metrics, it presents a promising approach to how testing efforts are distributed and to pinpoint potential testers for OSS projects effectively. To tackle this research gap, during the initial phase of our analysis, we identify contributors most actively involved in testing activities by examining the number of commits made to executable test files. We employed the Pareto analysis approach based on the methodology outlined by Yamashita *et al.* (Yamashita et al., 2015) study. While they focus on core developers in selected projects, we adapted this method to specifically address the testing workload within the Apache Spark project. By leveraging the Pareto principle and analyzing the distribution of test commit count (CmtTesNum), we confidently identify the contributors who are responsible for the majority of testing contributions. Our systematic analysis involved the following steps:

Step 1: Clone the Apache Spark project repository from Github.

Step 2: Extract commit author information, including author names (CmtAuNa) and e-mail addresses (CmtAuMI).

Step 3: Identify executable test files by examining the repository's directory structure, focusing on standardized test directories such as 'src/test/"/>.

Step 4: Calculate the number of commits containing executable test files (CmtTesNum) for each contributor.

Step 5: Filter out contributors whose commits do not include any executable test files.

Step 6: Apply the Pareto principle to the distribution of test commit counts (CmtTesNum) to identify key contributors to testing activities.

Through this systematic analysis of test commit counts, we can identify the key contributors driving the majority of testing activities in the Apache Spark project, i.e., top testing contributors. This foundational work enables an insightful time-based analysis of their testing engagement patterns.

3.3 Time-Based Analysis of Activities

Contributors often demonstrate a range of engagement levels over time, underscoring the importance of time-based analysis. Such an approach allows for the classification of contributors' activities into specific groups based on their participation levels. For instance, Lee and Carver (Lee and Carver, 2017) have investigated OSS contributors over time according to the number of commits. Core contributors, for instance, are those who have been engaged with the project for a long time and have contributed many patches to the project, while the peripheral contributors tend to have shorter involvement compared to the core contributors (Lee and Carver, 2017). Within this latter group, One-Time Contributors (OTCs) have made only a single code contribution to the project repository. Another time-based analysis focuses on the average time between commits and the duration of involvement, enabling the identification of active, occasional, and rare developers (Di Bella et al., 2013). A more recent study further categorizes the OSS developers within the active roles and supporting roles by analyzing code contribution metrics from a three-year period (Cheng and Guo, 2019). Such a combined approach is particularly promising for exploring not only who the key testers are but also how their activities develop or diminish over time. Building on this framework, the next phase of our analysis investigates the recent engagement patterns of the top testing contributors identified through the Pareto Analysis.

This approach aims to determine whether these top contributors remain consistently active or show signs of declining participation in activities over time. To achieve this, we adopt the time-based analysis framework proposed by Cheng and Guo (Cheng and Guo, 2019), which focuses on a three-year timeframe divided into quarterly intervals. Their study aligns with our objectives of examining activities within the OSS projects by analyzing code contribution metrics, such as the number of commits and modified files. However, we specifically focus on identifying top testing contributors and evaluating their activity level in terms of project involvement and testing focus over time within Apache Spark.

We evaluate top testing contributor's overall engagement by analyzing their total commit counts (CmtNum) as a proxy for their activity levels. Analyzing this metric from 2021 to 2023 allows us to effectively measure their recent participation trends. We identified the contributors by their names (CmtAuNa) over the three years and aggregated their commit counts (CmtNum). To categorize top testing contributors based on activity levels, we applied a hierarchical clustering approach to their total commit counts. This clustering method allows us to categorize contributors into three activity strata: Highly-Active, Moderately-Active, and Lowly-Active. Using dendrograms as visual aids, we provide an intuitive graphical representation of clusters, facilitating easier interpretation of contributor activity patterns (Bruce et al., 2020).

After identifying the top testing contributors through Pareto analysis (Section 3.2) and categorizing their recent activity levels over the three-year period (Section 3.3), we further explore their sustained testing focus by examining test commit counts. By analyzing test commit counts across quarterly intervals from 2021 to 2023, we further examine the continuity and variability of testing focus among the Highly-Active, Moderately-Active, and Lowly-Active groups. This analysis provides valuable insights into the long-term testing focus of key contributors within the Apache Spark community and highlights how these testing focuses evolve over time.

4 RESULTS

This section presents our findings, directly addressing the research questions outlined in Section 1. We provide a detailed analysis of testing contributions, along with a time-based evaluation of recent contributors, to thoroughly investigate their contribution patterns over time.

4.1 Distribution of Testing Contributions (RQ1)

We examined the GitHub commit data to investigate the Apache Spark testing workload. The primary objective of RQ1 is to identify top testing contributors who are predominantly involved in testing activities using Pareto Analysis. To this end, we counted the number of commits containing executable test files for each contributor.

Our dataset encompasses 41,291 total commits made by 2,883 contributors. Among these, 20,468 commits (49.57% of the total) include modifications to executable test files, derived through a detailed examination of the Apache Spark project directory. Table 2 presents an overview of our dataset.

Table 2: Overview of the Apache Spark Dataset.

Metrics	Count
Total number of commits	41,291
Commits with test modifications	20,468
Total number of contributors	2,883

4.1.1 Pareto Analysis Results

Our analysis of test-related commits reveals notable patterns in the distribution of testing workloads among contributors. Out of 2,883 contributors, 1,625 (56.36%) have made at least one test-related commit. A Pareto Analysis of these contributors indicates an even more core distribution than the traditional 80/20 principle: 80% of all test-related commits were made by just 159 developers, accounting for only 9.8% of contributors in this category.

This analysis implies that a window of flexibility exists, allowing projects with core contributors falling within the range of $20\% \pm 10\%$ to adhere to the Pareto principle. Figure 1 visually depicts this concentrated distribution pattern.

These findings provide key insights for RQ1, demonstrating that testing efforts in Apache Spark are heavily reliant on a small group of highly dedicated contributors. This emphasizes the crucial role of core contributors play in managing the testing workload.

4.1.2 Core Contributors in Testing

An in-depth examination of core contributors through the lens of the Pareto principle reveals notable patterns in their test-related commit activities. Table 3 outlines the statistics for the top 10 contributors, whose testing workload constitutes 30.46% of the total testing workload, with their test-related commit count (CmtTesNum) reaching 16,367.

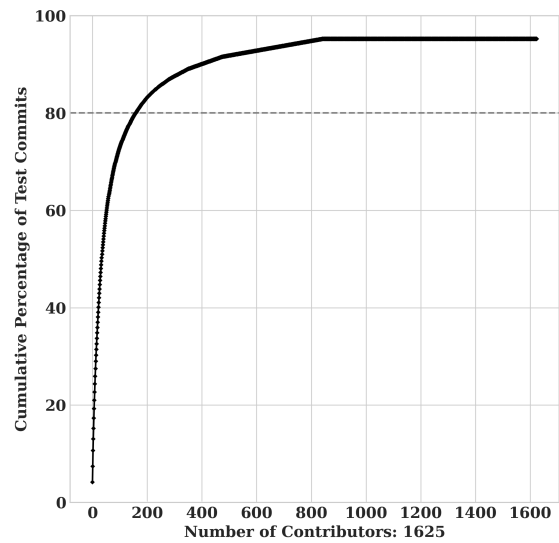


Figure 1: Test commits Pareto analysis.

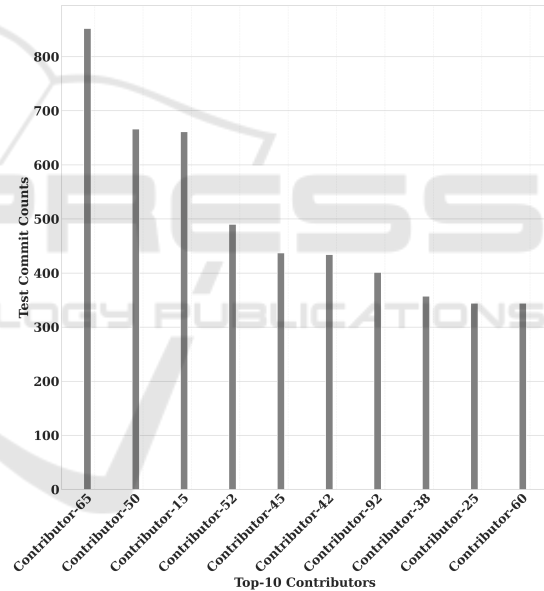


Figure 2: Test contribution analysis of Top 10 contributors.

Among these core contributors, we can draw significant insights regarding their testing patterns. As illustrated in Figure 2, Contributor-65 stands out as an outlier with 852 test-related commits (CmtTesNum), which shows a strong commitment to testing activities, which account for 71.96% of their total commits. To further investigate the testing contributions of Contributor-65, we analyzed their test-related commits and the modified files associated with them. For example, Contributor-65 indicated involvement in writing tests for the project and enhancing test infrastructure in [SPARK-XXXXX][CORE][TESTS],

Table 3: Statistics of Top 10 testing contributors.

Contributor	CmtNum	CmtTesNum	% of CmtTesNum	CmtTesNum/CmtNum Ratio
Contributor-65	1,184	852	5.20%	71.96%
Contributor-50	1,482	666	4.07%	44.94%
Contributor-15	1,584	661	4.04%	41.7%
Contributor-52	825	490	2.99%	59.39%
Contributor-45	577	437	2.67%	75.74%
Contributor-42	676	434	2.65%	64.20%
Contributor-92	1,592	401	2.45%	25.19%
Contributor-38	578	357	2.18%	61.76%
Contributor-25	874	344	2.10%	39.36%
Contributor-60	564	344	2.10%	60.99%

along with modifying test files in this commit. This evidence underscores their direct involvement in testing activities. The existence of multiple similar commits emphasizes that Contributor-65 not only focuses on core development but also takes an active role in writing tests and improving test infrastructure, thus solidifying their role as a testing contributor. Contributor-45 holds the highest ratio of test commits to total commits at 75.74%, with 437 test commit counts. An analysis of Contributor-45's commits reveals their efforts in contributing to new test suites for the Apache Spark project, enhancing test coverage, and eliminating duplicate tests, as evidenced by their [SPARK-XXXXX][SQL][TESTS] commit. Notably, this commit comprises only test-related code modifications. These findings confirm that Contributor-45 plays a significant role in enhancing the effectiveness of testing within the Apache Spark project. According to Table 3, the proportion of test commits (CmtTesNum) to total commits (CmtNum) varies by 25.19% among the top contributors, with Contributor-92 and Contributor-45 ranking high in testing activities. These findings affirm that testing responsibilities within Apache Spark are predominantly handled by a small group of contributors despite variations in both the number and proportion of their testing activities.

4.2 Time-Based Analysis in Testing Activities (RQ2)

To explore the contribution patterns of core contributors identified through the Pareto Analysis, we conducted a time-based analysis to determine whether these top testers have maintained consistent activity levels or exhibited shifts in their testing focus. By examining quarterly periods over the past three years, we categorized contributors' engagement activity patterns, capturing fluctuations and trends in their testing contributions.

4.2.1 Clustering on Contributor Commit Activities

To further investigate the activity patterns of top testing contributors, we applied hierarchical clustering to explore their project engagement activity levels based on total commit counts over the last three years. The results of this analysis are visualized in Figure 3, which uses a dendrogram to depict these activity patterns.

Although the silhouette measure suggests the highest coefficient (0.84) for two clusters, this results in a notably imbalanced distribution, with only four active top testing contributors (Contributor-15, Contributor-25, Contributor-52, and Contributor-88) compared to 87 non-active contributors. In order to provide a more nuanced understanding of contribution patterns among the recently active contributors, we examined the largest vertical distance between successive horizontal cuts in the dendrogram, which represents the merging of dissimilar clusters. The height of these merges identified three distinct activity patterns, offering a clearer categorization of contributor behaviors, as shown in Figure 3. The clusters were classified into three groups based on total commit counts: Highly-Active Contributors (4 members), Moderately-Active Contributors (23 members), and Lowly-Active Contributors (64 members).

The first group, Highly-Active Contributors, demonstrates exceptional activity, with total commit counts ranging from 637 to 726 over the last three years. Members of this group, Contributor-52 (637), Contributor-25 (642), Contributor-15 (654), and Contributor-88 (726), are responsible for 2,659 commits, representing 31.27% of total commits. This analysis indicates a significant portion of commit activity among these four contributors, highlighting the strong dependency on a small core group within the Apache Spark project.

The second group, Moderately-Active Contributors, exhibits more balanced activity patterns, with

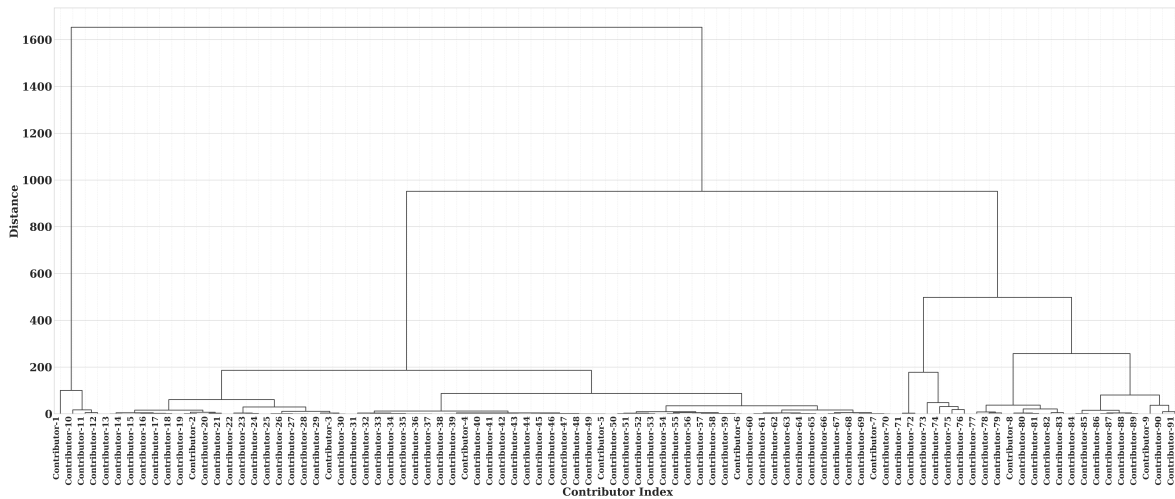


Figure 3: Dendrogram visualization of hierarchical clustering results regarding contributors' total commit counts.

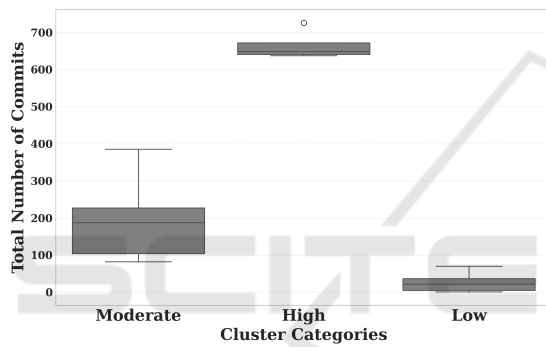


Figure 4: Distribution of total commit counts across contributor clusters.

commit counts varying from 82 to 385. Key contributors in this group are Contributor-83 (304 commits), Contributor-84 (382 commits), and Contributor-45 (385 commits).

The final group, Lowly-Active Contributors, contributed the least, representing 18% of total commits, with fewer than 70 commits per member.

This hierarchical clustering points to a highly skewed distribution of contribution activity, where a small number of Highly-Active contributors (4 members) and a larger number of Moderately-Active Contributors (23 members) dominate the commit activities in Apache Spark over the last three years, as shown in Figure 4.

4.2.2 Time-Based Activity Patterns

Our time-based analysis of quarterly test commit patterns from 2021 to 2023 deduces several outcomes of how contributors within each cluster engage in testing focus over time. In order to achieve this, we analyzed the test commit ratio, which is defined as the num-

ber of test commits divided by the total commit count for each period within each activity cluster identified in the previous section, Section 4.2.1. This analysis reveals the proportion of work related to testing across different activity levels. As shown in Figure 5, the members of each cluster show distinct patterns in testing-focused activities across these three-month intervals over the last three years.

Highly-Active Contributors exhibit a notable focus on testing when compared to other clusters. Beginning with a significant peak in test commit ratio of approximately 0.22 in early 2021 (2021-Q1), this group demonstrates consistent testing involvement throughout the observed timeframe, despite experiencing some fluctuations. There was a dramatic decline in 2021-Q3, where the test commit ratio dropped to around 0.07. Nevertheless, this group has maintained the highest test commit ratios relative to the other clusters during all periods reviewed. Another peak occurred in 2023-Q2, reaching about 0.22 in the test commit ratio. This trend indicates this relatively small group sustains substantial involvement in testing, consistently outperforming both Moderately-Active and Lowly-Active Contributors in terms of test commit ratios.

Moderately-Active Contributors show a relatively stable in their involvement with test commits. According to Figure 5, they show a slight increase during 2021-Q2, in which their test commit ratio is approximately 0.14. Their test commit ratio then gradually decreases until 2021-Q4, before peaking at around 0.15 in 2022-Q2. Overall, their testing engagement demonstrates a general downward trend, with the ratio fluctuating between approximately 0.06 and 0.10 throughout 2023. Although their test commit ratios are lower than those of Highly-Active con-

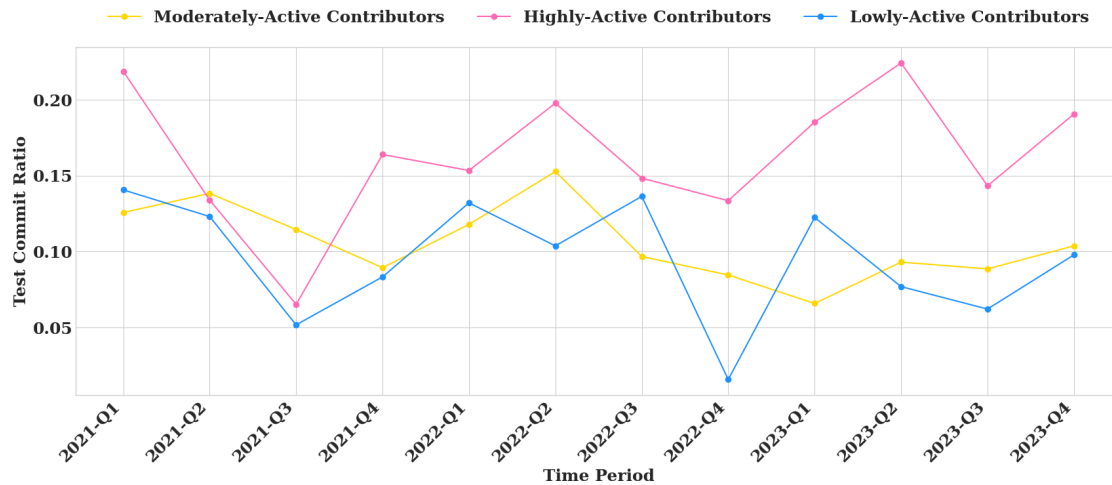


Figure 5: Test commit ratio over time by cluster.

tributors during the examined periods, they maintain a more consistent level of testing involvement.

Lowly-Active Contributors display considerable fluctuations in their test commit ratios, beginning at approximately 0.14 in 2021-Q1. Their testing involvement is comparable to those of Moderately-Active Contributors. Interestingly, the test commit ratio for Lowly-Active Contributors test is nearly equivalent to the around 0.08 observed for Moderately-Active Contributors in 2021-Q4. They experience the most drastic decreases compared to the Moderately-Active Contributors cluster, highlighted by a sharp drop of around 0.02 in 2022-Q4, followed by a remarkable increase of around 0.14 in 2023-Q1. By the final quarter of 2023, their test commit ratio once again approaches that of the Moderately-Active Contributors, stabilizing around 0.10. Consequently, Lowly-Active Contributors maintain periodic testing involvement and demonstrate less consistency in testing activities compared to other clusters.

These time-based analyses underscore the evolution of distinct testing activities within the Apache Spark project, revealing that each cluster exhibits unique patterns over time. **Highly-Active Contributors** consistently maintain the highest test commit ratios, indicating their significant focus in testing activities during the observed time periods. **Moderately-Active Contributors** and **Lowly-Active Contributors** demonstrate test commit ratios that are closer to each other during certain periods, such as 2021-Q4 and 2022-Q1. However, **Moderate-Active Contributors** show more stable test commit ratios overall. These findings provide valuable insights into testing focus over time and shed light on whether such involvement is consistently sustained across different

contributor groups or if it heavily depends on specific activity clusters.

5 DISCUSSION

In this section, we revisit our two research questions. Using the results described in Section 4 along with previous results from the literature.

RQ1: How Are the Contributions to Testing Distributed Among Contributors in the Apache Spark Project?

Our study clearly demonstrates a significant concentration in the Apache Spark project, where 9.8% of contributors are responsible for 80% of test commits. This distribution pattern emphasizes an atypical focus on testing activities than the traditional Pareto principle observed in prior literature, which typically shows that 20% of contributors account for 80% of the work (Geldenhuys, 2010), (Lipovetsky, 2009). Through quantitative analysis of test-related commits, we found that this small but dedicated group of contributors participated in testing activities, indicating the presence of informal testers within the Apache Spark community. This phenomenon could allow for the assignment of testing roles to OSS contributors, similar to the tester roles in traditional software. While this outcome presents intriguing possibilities for defining specific testing roles for open-source software (OSS) contributors, it also raises significant concerns regarding the effectiveness of testing efforts. Relying on a limited group of testing contributors may result in an imbalanced distribution of testing responsibilities. Moreover, when testing efforts are concentrated among a small group, it can lead to insufficient

testing across various software modules, thereby diminishing the effectiveness of the tests. This focused concentration may also create uneven workflows in testing activities, resulting in gaps in test coverage, where certain components of the software receive thorough testing while others are neglected. Our findings reveal a noteworthy disparity in testing efforts within Apache Spark, which is dependent on a small number of contributors, thus underscoring the challenge of uneven test coverage throughout the project. In contrast, OSS projects like Mozilla have embraced a more inclusive and collaborative approach to testing (Mockus et al., 2002). Mozilla actively involves a sizeable group of contributors who are dedicated to testing activities, which helps to distribute testing efforts more equitably across the entire project and reduces the likelihood of components being either overlooked or excessively tested. For instance, a substantial group within Mozilla maintains the test cases and test plans, ensuring broader participation in testing activities (Mockus et al., 2002). The differences in the testing strategies between Apache Spark and Mozilla emphasize a critical concern: substantial gaps in testing participation can diminish test effectiveness, leading to inconsistent coverage. Encouraging broader collaboration among participants is crucial for ensuring the quality and reliability of open-source software projects (Wang et al., 2024). Therefore, it is essential to encourage wider participation in testing activities to ensure that all aspects of the software undergo rigorous and comprehensive testing, enhancing test effectiveness.

RQ2: How do the Testing Activities of Contributors Change Over Time?

Our time-based analysis of contributor activity within the Apache Spark project between 2021 and 2023 reveals significant differences in the engagement levels of top testing contributors. It is evident that not all contributors maintained a consistent involvement; while some have shown active participation, others exhibit moderate or low involvement. This dynamic participation among the top testing contributors is categorized into Highly-Active, Moderately-Active, and Lowly-Active clusters as seen in Figure 4. This analysis indicates that top testing contributors do not always sustain equal levels of activity; rather, their levels of contribution change over time.

With these activity-based clusters established, we assessed each cluster's test commit ratio to examine their testing focus throughout the periods. This approach enables us to discern whether their testing focus remains stable or experiences fluctuations. Highly-Active contributors exhibit the highest testing

focus during 2021-Q1 and 2023-Q4, showing their strong commitment to testing. As a pivotal group within the community, these contributors bear the primary responsibility for the testing within the Apache Spark OSS project. This strong emphasis on testing not only underscores their dedication but also aligns with existing research that identifies core contributors as key players who drive substantial contributions and propel the project forward (Lee and Carver, 2017). Moderately-Active and Lowly-Active demonstrate similar test focuses, albeit with some temporal variability. The observed patterns highlight that the testing focus of top testing contributors not only differs by activity level but also exhibits noticeable temporal trends.

6 THREATS TO VALIDITY

Several factors discussed in this section may pose challenges to the validity of our proposed approach. The selection of a three-year time period could affect our assessment of contributor patterns. We addressed this by choosing this specific period to maintain recency while having sufficient data for meaningful analysis. Additionally, our use of specific thresholds in Pareto Analysis (80/20 principle) and hierarchical clustering parameters could influence how we identify and categorize testing contributors and their activity levels. To address these challenges, we validated our Pareto threshold against established OSS studies and carefully interpreted the dendrogram to determine optimal cluster numbers in our hierarchical analysis, ensuring our classifications accurately reflect contributor activity levels. In our research, analyzing the testing behaviors of OSS contributors through commits that contain test files may not fully capture their testing responsibilities. Future research could expand our methodology by examining additional activity-based metrics related to testing, such as the number of test cases developed or executed. It would also be beneficial to conduct a study across multiple projects. Our research centered exclusively on the Apache Spark project. To enhance the external validity of our findings, we intentionally selected this project, which has been the subject of numerous studies, features a diverse set of contributors, and has a well-documented commit history. While our findings are drawn from Apache Spark, they may not be generalizable to all OSS projects. Nevertheless, the methodological framework we conducted can indeed be adapted for use with other OSS projects. Future research could further validate and expand upon our findings across a variety of OSS contexts. In this vein,

we have presented initial results and conceptual ideas.

7 CONCLUSION

Our aim in this paper is to explore how to identify key contributors responsible for the majority of testing in open-source projects, specifically through the analysis of test commits within the Apache Spark project. By focusing on contributors who bear testing responsibilities (**RQ1**), we observe that only 9.8% of contributors are responsible for the majority of test-related commits. This distribution emphasizes that a concentrated group of individuals is responsible for testing activities, a more skewed distribution than the traditional 80/20 Pareto Principle. Our further analysis uncovered varying engagement levels among the top testing contributors by examining their total commits over the past three years. We categorized them as Highly-Active Contributors, Moderately-Active Contributors, and Lowly-Active Contributors (**RQ2**). Additionally, we analyzed the testing focuses of each activity level across quarterly periods. It becomes apparent that Highly-Active Contributors who are deeply engaged with the Apache Spark project also exhibit the highest testing focus across other clusters, positioning them as the backbone of the project's testing practices. Our analysis highlights significant implications for OSS projects, particularly in fostering broader participation in testing activities among OSS contributors and strategizing to achieve a more balanced involvement across the community.

REFERENCES

- Abdou, T., Grogono, P., and Kamthan, P. (2012). A conceptual framework for open source software test process. In *2012 IEEE 36th Annual Computer Software and Applications Conference Workshops*, pages 458–463. IEEE.
- Arafat, O. and Riehle, D. (2009). The commit size distribution of open source software. In *2009 42nd Hawaii International Conference on System Sciences*, pages 1–8. IEEE.
- Beller, M., Gousios, G., Panichella, A., and Zaidman, A. (2015). When, how, and why developers (do not) test in their ideo. In *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering*, pages 179–190.
- Bird, C., Gourley, A., Devanbu, P., Gertz, M., and Swaminathan, A. (2006). Mining email social networks. In *Proceedings of the 2006 international workshop on Mining software repositories*, pages 137–143.
- Bruce, P., Bruce, A., and Gedeck, P. (2020). *Practical statistics for data scientists: 50+ essential concepts using R and Python*. O'Reilly Media.
- Chełkowski, T., Gloor, P., and Jemielniak, D. (2016). Inequalities in open source software development: Analysis of contributor's commits in apache software foundation projects. *PLoS One*, 11(4):e0152976.
- Cheng, J. and Guo, J. L. (2019). Activity-based analysis of open source software contributors: Roles and dynamics. In *2019 IEEE/ACM 12th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE)*, pages 11–18. IEEE.
- Coelho, F., Tsantalis, N., Massoni, T., and Alves, E. L. (2021). An empirical study on refactoring-inducing pull requests. In *Proceedings of the 15th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, pages 1–12.
- Desikan, S. and Ramesh, G. (2006). *Software testing: principles and practice*. Pearson Education India.
- Di Bella, E., Sillitti, A., and Succi, G. (2013). A multivariate classification of open source developers. *Information Sciences*, 221:72–83.
- Ducheneaut, N. (2005). Socialization in an open source software community: A socio-technical analysis. *Computer Supported Cooperative Work (CSCW)*, 14:323–368.
- Geldenhuys, J. (2010). Finding the core developers. In *2010 36th EUROMICRO Conference on Software Engineering and Advanced Applications*, pages 447–450. IEEE.
- Goeminne, M. and Mens, T. (2011). Evidence for the pareto principle in open source software activity. In *The Joint Proceedings of the 1st International workshop on Model Driven Software Maintenance and 5th International Workshop on Software Quality and Maintainability*, pages 74–82. Citeseer.
- Hao, X., Zhengang, Z., Chunpei, L., and Zhuo, D. (2008). The study on innovation mechanism of open source software community. In *2008 4th International Conference on Wireless Communications, Networking and Mobile Computing*.
- Lee, A. and Carver, J. C. (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)*, pages 1–10. IEEE.
- Levin, S. and Yehudai, A. (2017). Boosting automatic commit classification into maintenance activities by utilizing source code changes. In *Proceedings of the 13th International Conference on Predictive Models and Data Analytics in Software Engineering*, pages 97–106.
- Li, Z., Qi, X., Yu, Q., Liang, P., Mo, R., and Yang, C. (2021). Multi-programming-language commits in oss: An empirical study on apache projects. In *2021 IEEE/ACM 29th International Conference on Program Comprehension (ICPC)*, pages 219–229. IEEE.
- Lipovetsky, S. (2009). Pareto 80/20 law: derivation via random partitioning. *International Journal of Mathematical Education in Science and Technology*, 40(2):271–277.

- Mockus, A., Fielding, R. T., and Herbsleb, J. D. (2002). Two case studies of open source software development: Apache and mozilla. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 11(3):309–346.
- Morasca, S., Taibi, D., and Tosi, D. (2011). Oss-tmm: guidelines for improving the testing process of open source software. *International Journal of Open Source Software and Processes (IJOSSP)*, 3(2):1–22.
- Napoleão, B. M., Petrillo, F., and Hallé, S. (2020). Open source software development process: a systematic review. In *2020 IEEE 24th International Enterprise Distributed Object Computing Conference (EDOC)*, pages 135–144. IEEE.
- Pinto, G., Steinmacher, I., and Gerosa, M. A. (2016). More common than you think: An in-depth study of casual contributors. In *2016 IEEE 23rd international conference on software analysis, evolution, and reengineering (SANER)*, volume 1, pages 112–123. IEEE.
- Seigneur, J.-M. (2007). Trustworthiness of collaborative open source software quality assessment. In *2007 Third International Conference on Security and Privacy in Communications Networks and the Workshops-SecureComm 2007*, pages 20–26. IEEE.
- Tosi, D. and Tahir, A. (2010). How developers test their open source software products-a survey of well-known oss projects. In *International Conference on Software and Data Technologies*, volume 2, pages 22–31. SCITEPRESS.
- Trinkenreich, B., Guizani, M., Wiese, I., Sarma, A., and Steinmacher, I. (2020). Hidden figures: Roles and pathways of successful oss contributors. *Proceedings of the ACM on human-computer interaction*, 4(CSCW2):1–22.
- Wang, Q., Wang, J., Li, M., Wang, Y., and Liu, Z. (2024). A roadmap for software testing in open-collaborative and ai-powered era. *ACM Transactions on Software Engineering and Methodology*.
- Yamashita, K., McIntosh, S., Kamei, Y., Hassan, A. E., and Ubayashi, N. (2015). Revisiting the applicability of the pareto principle to core development teams in open source software projects. In *Proceedings of the 14th international workshop on principles of software evolution*, pages 46–55.
- Zhang, X., Stafford, T. F., Hu, T., and Dai, H. (2020). Measuring task conflict and person conflict in software testing. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 29(4):1–19.
- Zhao, L. and Elbaum, S. (2003). Quality assurance under the open source development model. *Journal of Systems and Software*, 66(1):65–75.