Analysis of Github Pull Requests

Canon Ellis
canone@smu.edu

Follow this and additional works at: https://scholar.smu.edu/engineering_compsci_etds

Part of the Data Science Commons, Software Engineering Commons, and the Theory and Algorithms Commons

Recommended Citation
https://scholar.smu.edu/engineering_compsci_etds/16

This Thesis is brought to you for free and open access by the Computer Science and Engineering at SMU Scholar. It has been accepted for inclusion in Computer Science and Engineering Theses and Dissertations by an authorized administrator of SMU Scholar. For more information, please visit http://digitalrepository.smu.edu.
ANALYSIS OF GITHUB PULL REQUESTS

Approved by:

___________________________________
Dr. LiGuo Huang
Professor of Software Eng.

_______________________________________
Dr. Jeff Tian
Professor of Software Eng.

_______________________________________
Dr. Suku Nair
Professor of Electrical Eng.
ANALYSIS OF GITHUB PULL REQUESTS

A Thesis Presented to the Graduate Faculty of the
Bobby B. Lyle School of Engineering
Southern Methodist University
in
Partial Fulfillment of the Requirements
for the degree
of
Master of Science
With a Major in Software Engineering
by
Daniel Canon Ellis
December 19, 2020
ACKNOWLEDGMENTS

Foremost, I would like to express my sincere gratitude to my advisor Dr. LiGuo Huang for the continuous support of my master thesis study and research, for her patience, motivation, enthusiasm, and immense knowledge. Her guidance helped me in time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my master study.

Besides my advisor, I would like to thank the rest of my thesis committee: Dr. Jeff Tian and Dr. Suku Nair, for their encouragement, insightful comments, and hard questions.
The popularity of the software repository site GitHub has created a rise in the Pull Based Development Models' use. An essential portion of pull-based development is the creation of Pull Requests. Pull Requests often have to be reviewed by an individual to be approved and accepted into the Master branch of a software repository. The reviewing process can often be time-consuming and introduce a relatively high level of lost development time. This paper examines thousands of pull requests to understand the most valuable metadata of pull requests. We then introduce metrics in comparing the metadata of pull requests to understand what makes an effective pull request. Breaking pull requests into specific metadata pieces and evaluating what each piece brings to the whole allows us to review pull requests more efficiently. A pull request is successful if and only if it merges with the Master Branch. The Master Branch is the main branch of the code repository and is the production codebase. Using data analysis tools, we can determine which parts of a pull request are critical in its merge time. As well as the formation for a framework and creation of a data structure to track and manage development resources in the Pull Based Development Model.
# TABLE OF CONTENTS

LIST OF FIGURES .................................................................................................................. viii

LIST OF TABLES .................................................................................................................... xii

CHAPTER 1 ............................................................................................................................... 1

1.1 Introduction ....................................................................................................................... 1

1.2 Motivation ........................................................................................................................ 1

1.3 Organization of Chapters ............................................................................................... 2

CHAPTER 2 ............................................................................................................................... 3

2.1 Pull Based Development ................................................................................................. 3

2.2 Pull Request ..................................................................................................................... 5

2.3 Related Work .................................................................................................................. 6

CHAPTER 3 ............................................................................................................................... 9

3.1 Empirical Method .......................................................................................................... 9

3.2 Data Collection ............................................................................................................. 10

3.3 Data Cleaning ............................................................................................................... 13

CHAPTER 4 ............................................................................................................................... 15

4.1 Data Analysis ................................................................................................................. 15

4.2 Results ............................................................................................................................ 22

4.3 Graphs ............................................................................................................................ 22

4.4 Titles ............................................................................................................................... 58
LIST OF FIGURES

Figure 1.................................................................................................................................4
Figure 2........................................................................................................................................11
Figure 3.........................................................................................................................................12
Figure 4.........................................................................................................................................14
Figure 5: Github Title Example...................................................................................................15
Figure 6: Github Description..........................................................................................................17
Figure 7: Angular Average Merge Time .......................................................................................23
Figure 8: Angular Files Changed and Total Merged.................................................................23
Figure 9: Angular Total Pull Requests and Total Merged ...........................................................24
Figure 10: Angular Number of Labels .........................................................................................24
Figure 11: Angular Total Pull Requests and Total Descriptions ..............................................25
Figure 12: Angular Total Pull Requests and #Files Changed ....................................................25
Figure 13: Angular Total Merged and Total Labels .................................................................26
Figure 14: Bitcoin Average Merge Time ....................................................................................26
Figure 15: Bitcoin Files Changed and Total Merged ...............................................................27
Figure 16: Bitcoin Total Pull Requests and Total Merged .........................................................27
Figure 17: Bitcoin Number of Labels .........................................................................................28
Figure 18: Bitcoin Total Pull Requests and Total Descriptions ..............................................28
Figure 19: Bitcoin Total Pull Requests and #Files Changed.....................................................29
Figure 20: Bitcoin Total Merged and Total Labels .................................................................29
Figure 44: Keras Total Pull Requests and Total Merged ...........................................................41
Figure 45: Keras Number of Labels ..........................................................................................42
Figure 46: Keras Total Pull Requests and Total Descriptions ..................................................42
Figure 47: Keras Total Pull Requests and #Files Changed .......................................................43
Figure 48: Keras Total Merged and Total Labels ......................................................................43
Figure 49: NigthScout Average Merge Time .............................................................................44
Figure 50: NigthScout Files Changed and Total Merged ..........................................................44
Figure 51: NigthScout Total Pull Requests and Total Merged ...................................................45
Figure 52: NigthScout Number of Labels ..................................................................................45
Figure 53: NigthScout Total Pull Requests and Total Descriptions ..........................................46
Figure 54: NigthScout Total Pull Requests and #Files Changed ................................................46
Figure 55: NigthScout Total Merged and Total Labels ..............................................................47
Figure 56: Tensorflow Average Merge Time .............................................................................47
Figure 57: Tensorflow Files Changed and Total Merged ..........................................................48
Figure 58: Tensorflow Total Pull Requests and Total Merged ...................................................48
Figure 59: Tensorflow Number Of Labels ...............................................................................49
Figure 60: Tensorflow Total Pull Requests and Total Descriptions .........................................49
Figure 61: Tensorflow Total Pull Requests and #Files Changed ...............................................50
Figure 62: TensorFlow Total Merged and Total Labels .............................................................50
Figure 63: TensorFlow Total Pull Requests and Total Descriptions .........................................51
Figure 64: The Algorithms Average Merge Time .....................................................................51
Figure 65: TheAlgorithms Total Pull Requests and Total Merged .............................................52
Figure 66: TheAlgorithms Number of Labels .........................................................................52
Figure 67: The Algorithms Total Pull Requests and Total Descriptions ............................................53
Figure 68: The Algorithms Total Pull requests and Number Files Changed ......................................53
Figure 69: The Algorithms Total Merged and Total Labels .................................................................54
Figure 70: Vinta Average Merge Time ..................................................................................................54
Figure 71: Vinta Files Changed and Total Merged ...............................................................................55
Figure 72: Vinta Total Pull Requests and Total Merged .......................................................................55
Figure 73: Vinta Number of Labels ......................................................................................................56
Figure 74: Vinta Total Pull Requests and Total Descriptions ...............................................................56
Figure 75: Vinta Total Pull Requests and #Files Changed .....................................................................57
Figure 76: Vinta Total Merged and Total Labels ..................................................................................57
Figure 77: Short Pull Request Title .......................................................................................................58
Figure 78: Verbose Pull Request ...........................................................................................................58
Figure 79: Pull Request Implementation Issue .......................................................................................64
Figure 80: FeatureNode Data ...............................................................................................................65
# LIST OF TABLES

Table 1 .................................................................................................................. 10

Table 2 .................................................................................................................. 12

Table 3: Github Labels ......................................................................................... 16

Table 4: Angular Reference Chart ....................................................................... 18

Table 5: Bitcoin Reference Chart ......................................................................... 18

Table 6: Bootstrap Reference Chart .................................................................... 18

Table 7: Django Reference Chart ......................................................................... 19

Table 8: FacebookReact Reference Chart .............................................................. 19

Table 9: Keras Reference Chart ........................................................................... 19

Table 10: NightScout Reference Chart ................................................................. 20

Table 11: Tensorflow Reference Chart ................................................................. 20

Table 12: TheAlgorithms Reference Chart ........................................................... 20

Table 13: Vinta Reference Chart ......................................................................... 21
Finally, I would like to thank my Family. My parents Jorge and Amy, have supported me entirely in writing this master's thesis. I would also like to thank my older brother Major and my younger sister Aidan for their constant encouragement and pushing me to excel.
CHAPTER 1

1.1 Introduction

In the realm of Software Engineering, Pull Based Development is a type of project development used by Software Development practitioners. The pull-based development model allows individual developers to pull a project into their repository and then request their branch to be merged with the previous master repository. The Pull Based Development Models' efficiency comes from the ability for multiple developers to work on the same project and then have their work merged with the master branch. Before a pull can be pushed to the master branch and fully merged, the developer must first create a pull request, including a title, labels, and a description. These three pieces of information allow the master branch owner to understand what occurred in the development process. Pull Requests are supposed to allow for a streamlined development process. However, for larger companies, a single repository can have hundreds of pull requests per month. Each pull request can change from a couple of lines to thousands. The number of pull requests in conjunction with the amount of LOC(Lines Of Code) can cause a significant amount of developmental overhead.

1.2 Motivation

The motivation behind this research is to discover what aspects of the pull requests are most important to a developer and affect the acceptance of a pull request. The popularity and accessibility of pull-based development make it an ideal choice to examine the important aspects of it. This paper examines the Pull Requests of ten of the most actively used GitHub repositories
and extracts the important metadata to run a series of analysis to statistically evaluate what makes an efficient and generally accepted pull request.

1.3 Organization of Chapters

Chapter 2 explains the Pull Based Development Life Cycle and Pull Requests; the experienced software engineer only needs to skim this chapter. Chapter 3 contains the data collection and processing methodology used to make analytical statements. Chapter 4 contains the experiments and results of the analysis that has been applied to the datasets. This chapter is broken into a few different sections to explain the research onto the Pull Requests' different aspects. Chapter 5 is the final chapter and contains the threats to validity, future work, and conclusion of this research.
CHAPTER 2

2.1 Pull Based Development

Pull based development, also referred to as the Fork and Pull Model, is the software engineering practice of having multiple branches of a single repository, the central repository being the master branch. Whenever a change needs to be made to the repository for a bug fix or a new version, a developer will fork the repository master branch and begin developing that separate new branch. After the developer has finished their changes, the developer will issue a pull request, asking for their developmental branch to be merged into the master branch. When a pull request is issued, the quality assurance engineer or developer in charge will review the pull request, open it up, and examine it for specific information regarding what it is that pull request fixes. During this review, the reviewer will check for merge conflicts, a situation in which code has been overwritten or changed into a way that will no longer be compatible with the master branch's previous codebase. If the reviewer decides that the pull request is not valid or does not add any meaningful changes for the current development stage, the reviewer can deny the pull request. Still, suppose the pull request reviewer believes that there are no issues with the pull request and that the pull request can be successfully merged with the master branch codebase. In that case, the reviewer will accept the pull request, and both branches, the developer branch and the master branch, will be merged. The diagram below outlines a visual representation of the Pull Based Development life cycle.
The diagram shows the potential lifecycle of a single pull request. The current popularity of the Pull Based Development Model does allow research into an important opportunity for research. In 2015 GitHub reported 400,000 monthly pull requests opened on the site[1]. That level of data is important to breakdown and research how efficient repositories can handle large numbers of pull requests. The next section will explain the pull request parts and what makes it useful for developers.

Figure 1: Pull Based Development Life Cycle
2.2 Pull Request

While all the individual technical parts of the Pull Based Development Model are essential for this paper, most empirical research is focused on studying pull requests in real-world practice. A pull request's goal is to allow a development team to make changes to a codebase with a reduced impact of flaws introduced during development. A pull request allows for a higher version control level during the development cycle using a developer/reviewer schema. Version control, tracking updates in a program by issuing version numbers, is an important task when managing and building medium to large-scale projects. After a developer completes the work on the development branch, they create a pull request. The pull request requires a few fields of information that must be completed. The first item that a developer fills out is the pull request Title. The Title is a limited high-level description of what was accomplished.

The second item is selecting the tags. Tags are a selection of predefined terms that GitHub has chosen that allow for quickly labeling pull requests. Repository owners often decide whether or not Labels will be used in the development of the project. Labels and tags are explained, along with their importance, in more detail in Chapter 4.

The third and final phase of creating a pull request is the description. The description is used as a detailed low-level explanation of what the pull request covers. It is similar to a developmental report that lists specifics of what was accomplished during that phase of development. Once a pull request has been submitted, a separate developer will review the pull request checking for merge conflicts and the correctness of the coding schema. Depending on the pull request, the reviewer can either accept or deny the pull request. Pull requests not only allow developers to work on projects but also allow outsourcing to the project development.
community. Many repositories are open source and will accept pull requests from outside developers.

2.3 Related Work

Due to the variation of works in the subject of Pull Requests, the following section will be divided into subsections to explain the distribution of related work more comprehensively. Simultaneously, the related work on this topic is sparse; the few documents related to this work cover different topics within pull-based development.

2.3.1 Data Centric

Data-centric research focuses solely on the meta-data of repositories and pull-requests. While there might be a small amount of statistical analysis done, it is mostly about creating data dumps that other researchers can then work with its creating tools for others to perform analysis with after the fact.

In the paper “A Dataset for Pull-Based Development Research” [5] by Georgios Gousios and Andy Zaidman, the authors set out to create a large-scale data repository for pull request research. This data set included 900 projects and 350,000 pull requests. The authors also make use of Machine Learning to build a statistics environment in the R programming language. The difference between this work and the research discussed in this paper are the implications.

2.3.2 Development Centric

Development centric research focuses on the Pull-Based Development Life Cycle and pull-requests. This Development centric research tends to include research about the user
interactions in Pull Based Development. Development Centric research often focuses on the different individuals who work with pull requests specifically: the integrator and the contributor.

In the paper, "Work Practices and Challenges in Pull-Based Development: The Integrator's Perspective" [3] by G. Gousios, A. Zaidman, M. Storey, and A. V. Deursen, the authors conduct a qualitative study into the work habits of 749 integrators. Integrators are the individuals who review and merge any contribution to a codebase. The paper's insights show that integrators often struggle to keep the projects' quality maintained while also integrating new contributions.

In the paper "Work Practices and Challenges in Pull-Based Development: The Contributor's Perspective" by authors G. Gousios, M. Storey, and A. Bacchelli, research into the Contributors' point of view of Pull-Based Development. The authors surveyed 645 active contributors in the OSS project domain. The authors analyzed the contributor's efforts in maintaining active repositories and making changes. The authors discovered that while most contributors attempt to limit the potential duplicate work, there is little to no communication between contributors. The authors also found poor communication between contributors and integrators with minimal transparency. Gousios and Bacchelli's conclusion is to suggest ways that allow for better collaboration between integrators and contributors.

2.3.3 Differences:

The difference between this paper and data centric paper is that this paper will use interviews and analysis tools to understand what makes efficient and successful pull requests. The paper by authors Gousios and Zaidmen creates a data set for future work but doesn’t delve into the pull requests to pull extract information other than meta-data. In both the papers by these
authors use interviews and qualitative analysis to understand the different users of pull requests. The authors relied on interviews and surveys to gather data from real world practitioners. In this paper interviews are used to address what developers encounter in real world pull based development then use quantitative research methodology to find the data that backs up the developers. A second round of interviews then addresses the issues and how they can be solved with the implementation of a new Pull Based Development Tracking Methodology.
CHAPTER 3

3.1 Empirical Method

This research analyzes software development methods used by development teams and interviews with real-world practitioners. The motivation for looking at the movement of a repository monthly had been discussed at first with team leads and project managers. Most of these individuals use third-party tools to manage development teams, but most third-party tools do not have a complete analytical view of pull requests' monthly workings. The second set of interviews is conducted with real-world developers and programmers. These programmers and developers are the individuals who are working closely with the project, and also forking the main repository and creating the pull request. The developer interviews shined a light on creating the metadata in the pull request and the importance of different pieces of a pull request. Some of the important pieces were discovered through the interview process are that certain aspects of pull requests like Descriptions and Labels are utilized differently between repositories. As will be shown in later chapters but some developmental teams don’t make use of labels.

After interviewing the developers and programmers, the next phase required a detailed set of steps to ensure the reproducibility of the results. This phase required breaking down the interview into the critical aspects to form a list of critical data created and used by real-world practitioners. This list of aspects is the main source of requirements required for real-world practitioners to use GitHub. The list of requirements outlined the data that needed to be extracted from the repositories.
The final phase of the research required statistical analysis to find potential correlations and visualizing the data. The graphs and correlation charts serve to understand the parts deemed essential by developers and managers, work together to complete pull-requests. The data analysis and results of this part of the empirical method are in the following chapter, Chapter 4.

3.2 Data Collection

For the data collection process used in this research, a python script was created to get the metadata of pull requests for twelve months from ten of the most forked repositories on GitHub. The python script uses python 3.8 and PyGithub to collect GitHub related content. Creating a new dataset was preferred due to the data set's size and the relevant information were easier to control and maintain than any premade datasets found in other papers. The dataset was created from the topmost active repositories on GitHub. This meant that each repository had the highest number of forks and pull requests. Table 2 shows the list of the repositories used.

<table>
<thead>
<tr>
<th>Repository Name:</th>
</tr>
</thead>
<tbody>
<tr>
<td>angular/angular</td>
</tr>
<tr>
<td>bitcoin/bitcoin</td>
</tr>
<tr>
<td>twbs/bootstrap</td>
</tr>
<tr>
<td>django/django</td>
</tr>
<tr>
<td>facebook/react</td>
</tr>
<tr>
<td>keras-team/keras</td>
</tr>
<tr>
<td>tensorflow/tensorflow</td>
</tr>
<tr>
<td>nightscout/cgm-remote-monitor</td>
</tr>
<tr>
<td>TheAlgorithms/Python</td>
</tr>
<tr>
<td>vinta/awesome-python</td>
</tr>
</tbody>
</table>

Table 1: Repository Names
Other datasets had been tested for this research before the custom dataset route was taken. The custom datasets proved to be bulky and overcluttered. The extra information negatively affected the processing time and efficiency, so a custom data set was created. The script pulled certain values from the GitHub metadata to create a view of each pull request. The data gathered for this phase was data that was deemed useful for the overall observation of the pull request.

Pull Request:
Title: Update README.md
Pull Request id: 32157449
labels: []
labels # : 0
Number of Changed Files 1
Created: 2015-03-28 02:02:41
Closed: 2015-03-28 02:05:16
Time to Close: 0:02:35
Is Merged: True
Merged: 2015-03-28 02:05:15
Comments Url: https://api.github.com/repos/keras-team/keras/issues/1/comments
body: Typo fix.
base: PullRequestPart{sha="238d16974fe0856b2e7e5ae13da09f129f6baf0b"}

Figure 2
The metadata was decided on by interviewing real-world practitioners of software engineering and asking what data is most valuable for this research. The table below shows the metadata gathered from the pull requests.

<table>
<thead>
<tr>
<th>Title</th>
<th>id</th>
<th>Labels</th>
<th>ofLabels</th>
<th>#ofFilesChanged</th>
<th>Created</th>
<th>Closed</th>
<th>TimeToClose</th>
<th>isMerged</th>
<th>TimeToMerge</th>
<th>Body</th>
</tr>
</thead>
</table>

Table 2
The following data is stored in a CSV file that contains all the entries for a specific year. Each of the repositories is stored in separate CSVs to make the manual review of the pull requests more efficient and human-readable. During the data collection, the two pieces of metadata, Created and Closed, and calculated TimeToClose. TimeToClose gives the amount of time it took for a reviewer to close that specific pull request. The format is Y: M:D, H: M:S (Years:Months: Days, Hours:Minutes: Seconds) from time opened to time closed. This is a valuable data point in the research and will be discussed in detail in future sections.

3.3 Data Cleaning

The Data Cleaning stage required manually combing through the data to find any irregularities, potential error data that can throw off future analysis because of NaN (Not A Number), or improperly encoded strings. This phase was important due to the inherent nature of pulling data from a site. The manual examination of the data revealed some incorrectly formatted date: time values that could have affected the data analysis phase results. To fix the incorrect date: time errors, the pull request id was used to search and manually input the correct opened and closed time in the CSV file.

Another key component of the data cleaning was to make sure all the data for the repositories are organized monthly. While this wasn't a 100% necessary part because the python library Pandas can filter a CSV by months, this would be essential to prove the validity, and to
have a well organized CSV would be invaluable. The following table shows what the data in the CSV file looks like after it has been cleaned.

![Table showing data](image)

Figure 4

The data has been verified as acceptable and error-free. Once it has reached this phase, the data can now move on to the next phase, which is data analysis. The following chapters cover the data analysis phase of the research.
CHAPTER 4

4.1 Data Analysis

The data analysis was done in three separate phases. The first phase was to break the pull requests into groups of months. The main reasons for this research are to take a cross-section of all the data from a monthly perspective and understand the activity that takes place.

During the first phase, certain monthly averages needed to be obtained and created: Number of Pull Requests, Total Time to Close, Average Time to Close, Lines of Code Changed. These monthly averages were saved in a CSV along with other data from the following sections.

The second phase of the analysis required the observation of large specifics of the pull request. For this phase, the pull request was broken into three major components: Titles, Labels, and Descriptions. The Titles part of the pull request is the Title or what the developer has named that specific pull request. Often, the Title is a very brief summary of what the pull request accomplished. Figure 5 is an example of pull request title.

![Github Title Example](image)

Figure 5: Github Title Example
The labels of a pull request are a preset list of identifiers that are expected to increase review time by creating a form of filtering. The labels of the pull requests on the different repositories showed a varying level of frequency and use. In the next chapter, there will be a more in-depth explanation of the data gathered from the labels and what that data shows in terms of pull request acceptance. From interviews with industry professionals is has been identified the labels are usually used, when they are used, as a tool to know which department or development group needs to review that pull request. The following table shows a table of the GitHub labels.

<table>
<thead>
<tr>
<th>Label</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bug</td>
<td>Indicates an unexpected problem or unintended behavior</td>
</tr>
<tr>
<td>Documentation</td>
<td>Indicates a need for improvement or additions to documentation</td>
</tr>
<tr>
<td>Duplicate</td>
<td>Indicates similar issues or pull requests</td>
</tr>
<tr>
<td>Enhancement</td>
<td>Indicates new feature requests</td>
</tr>
<tr>
<td>Good First Issue</td>
<td>Indicates a good issue for first-time contributors</td>
</tr>
<tr>
<td>Help Wanted</td>
<td>Indicates that a maintainer wants help on an issue or pull request</td>
</tr>
<tr>
<td>Invalid</td>
<td>Indicates that an issue or pull request is no longer relevant</td>
</tr>
<tr>
<td>Question</td>
<td>Indicates that an issue or pull request needs more information</td>
</tr>
<tr>
<td>won’t fix</td>
<td>Indicates that work won’t continue on an issue or pull request</td>
</tr>
</tbody>
</table>

Table 3: Github Labels

The final part of this data analysis phase was to break down the GitHub pull request descriptions. This was a more complicated part than the previous labels and titles because of the amount of data used in a description. The character count and the use of a description are more varied than the Title. Often the length of the description is related to the lines of code changed. A
developer will outline and write pseudo-reports explaining the changes that have been made and the justification for those changes. The more quality content of the description, the fewer changelogs need to be reviewed to understand what changed in that specific pull request. Figure 6 shows an example of a pull request description.

![Figure 6: Github Description](image)

The data for each repo is stored in a reference chart that would serve as an easily viewable and understandable analytical tool. The chart shows numerical information for all the previous sections that can be used to view the historical data. A reference chart was made for each repository and is in the CSV file format. The following quick reference charts are also used to decipher any correlation between meta-data artifacts.
Table 4: Angular Reference Chart

<table>
<thead>
<tr>
<th>Month</th>
<th>TotalPullRequest</th>
<th>TotalTimeToClose</th>
<th>AvgTimeToClose</th>
<th>NumberFilesChanged</th>
<th>#Descriptions</th>
<th>TotalDiffLength</th>
<th>AvgDiffLength</th>
<th>TotalMerged</th>
<th>MergeRatio</th>
<th>TotalLabels</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/1/2011</td>
<td>36</td>
<td>218.3</td>
<td>6.12</td>
<td>12.0</td>
<td>24</td>
<td>20</td>
<td>660</td>
<td>33</td>
<td>17</td>
<td>90.95</td>
</tr>
<tr>
<td>2/1/2011</td>
<td>12</td>
<td>51.32</td>
<td>7.61</td>
<td>25</td>
<td>11</td>
<td>147</td>
<td>46.4</td>
<td>6</td>
<td>50</td>
<td>1</td>
</tr>
<tr>
<td>3/1/2011</td>
<td>39</td>
<td>451.06</td>
<td>11.57</td>
<td>176</td>
<td>39</td>
<td>1834</td>
<td>47.03</td>
<td>15</td>
<td>38.46</td>
<td>2</td>
</tr>
<tr>
<td>4/1/2011</td>
<td>34</td>
<td>1395.06</td>
<td>41.03</td>
<td>370</td>
<td>32</td>
<td>1853</td>
<td>57.91</td>
<td>18</td>
<td>52.94</td>
<td>7</td>
</tr>
<tr>
<td>5/1/2011</td>
<td>63</td>
<td>1972.18</td>
<td>31.35</td>
<td>271</td>
<td>54</td>
<td>3159</td>
<td>59.07</td>
<td>36</td>
<td>57.14</td>
<td>12</td>
</tr>
<tr>
<td>6/1/2011</td>
<td>45</td>
<td>1019.72</td>
<td>22.66</td>
<td>227</td>
<td>40</td>
<td>2222</td>
<td>55.35</td>
<td>24</td>
<td>53.33</td>
<td>9</td>
</tr>
<tr>
<td>7/1/2011</td>
<td>54</td>
<td>1547.33</td>
<td>28.03</td>
<td>186</td>
<td>48</td>
<td>2414</td>
<td>44.6</td>
<td>37</td>
<td>68.32</td>
<td>10</td>
</tr>
<tr>
<td>8/1/2011</td>
<td>23</td>
<td>1274.65</td>
<td>55.42</td>
<td>115</td>
<td>22</td>
<td>1076</td>
<td>48.91</td>
<td>17</td>
<td>73.91</td>
<td>9</td>
</tr>
<tr>
<td>9/1/2011</td>
<td>48</td>
<td>203.03</td>
<td>4.23</td>
<td>367</td>
<td>36</td>
<td>2379</td>
<td>66.08</td>
<td>40</td>
<td>85.33</td>
<td>27</td>
</tr>
<tr>
<td>10/1/2011</td>
<td>45</td>
<td>1467.57</td>
<td>32.61</td>
<td>179</td>
<td>40</td>
<td>1675</td>
<td>41.88</td>
<td>22</td>
<td>48.89</td>
<td>30</td>
</tr>
<tr>
<td>11/1/2011</td>
<td>29</td>
<td>200.24</td>
<td>7.11</td>
<td>181</td>
<td>24</td>
<td>1452</td>
<td>66.5</td>
<td>26</td>
<td>89.06</td>
<td>10</td>
</tr>
<tr>
<td>12/1/2011</td>
<td>39</td>
<td>562.87</td>
<td>14.43</td>
<td>217</td>
<td>31</td>
<td>1346</td>
<td>43.42</td>
<td>30</td>
<td>76.92</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 5: Bitcoin Reference Chart

<table>
<thead>
<tr>
<th>Month</th>
<th>TotalPullRequest</th>
<th>TotalTimeToClose</th>
<th>AvgTimeToClose</th>
<th>NumberFilesChanged</th>
<th>#Descriptions</th>
<th>TotalDiffLength</th>
<th>AvgDiffLength</th>
<th>TotalMerged</th>
<th>MergeRatio</th>
<th>TotalLabels</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/1/2012</td>
<td>48</td>
<td>40.05</td>
<td>0.83</td>
<td>2917</td>
<td>43</td>
<td>1291</td>
<td>30.02</td>
<td>5</td>
<td>10.42</td>
<td>3</td>
</tr>
<tr>
<td>2/1/2012</td>
<td>193</td>
<td>1998.53</td>
<td>10.0</td>
<td>104</td>
<td>148</td>
<td>4644</td>
<td>31.38</td>
<td>7</td>
<td>3.93</td>
<td>11</td>
</tr>
<tr>
<td>3/1/2012</td>
<td>96</td>
<td>555.8</td>
<td>6.26</td>
<td>342</td>
<td>76</td>
<td>1509</td>
<td>31.7</td>
<td>1</td>
<td>1.11</td>
<td>21</td>
</tr>
<tr>
<td>4/1/2012</td>
<td>73</td>
<td>698.48</td>
<td>9.58</td>
<td>880</td>
<td>65</td>
<td>2849</td>
<td>44.45</td>
<td>1</td>
<td>1.37</td>
<td>28</td>
</tr>
<tr>
<td>5/1/2012</td>
<td>54</td>
<td>802.05</td>
<td>14.87</td>
<td>260</td>
<td>51</td>
<td>2020</td>
<td>40.2</td>
<td>0</td>
<td>0</td>
<td>28</td>
</tr>
<tr>
<td>6/1/2012</td>
<td>47</td>
<td>390.92</td>
<td>8.12</td>
<td>501</td>
<td>39</td>
<td>1002</td>
<td>25.89</td>
<td>0</td>
<td>0</td>
<td>21</td>
</tr>
<tr>
<td>7/1/2012</td>
<td>34</td>
<td>128.21</td>
<td>3.77</td>
<td>635</td>
<td>31</td>
<td>946</td>
<td>10.52</td>
<td>0</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>8/1/2012</td>
<td>34</td>
<td>578.82</td>
<td>8.86</td>
<td>1578</td>
<td>78</td>
<td>2166</td>
<td>27.77</td>
<td>1</td>
<td>1.19</td>
<td>25</td>
</tr>
<tr>
<td>9/1/2012</td>
<td>63</td>
<td>1018.84</td>
<td>16.17</td>
<td>913</td>
<td>55</td>
<td>1857</td>
<td>33.76</td>
<td>1</td>
<td>1.59</td>
<td>32</td>
</tr>
<tr>
<td>10/1/2012</td>
<td>37</td>
<td>380.01</td>
<td>10.27</td>
<td>504</td>
<td>34</td>
<td>1125</td>
<td>33.09</td>
<td>0</td>
<td>0</td>
<td>14</td>
</tr>
<tr>
<td>11/1/2012</td>
<td>37</td>
<td>275.81</td>
<td>7.44</td>
<td>820</td>
<td>34</td>
<td>1006</td>
<td>29.59</td>
<td>0</td>
<td>0</td>
<td>16</td>
</tr>
<tr>
<td>12/1/2012</td>
<td>33</td>
<td>451.99</td>
<td>14.61</td>
<td>96</td>
<td>33</td>
<td>6345</td>
<td>192.88</td>
<td>3</td>
<td>9.99</td>
<td>18</td>
</tr>
</tbody>
</table>

Table 6: Bootstrap Reference Chart
Table 7: Django Reference Chart

Table 8: FacebookReact Reference Chart

Table 9: Keras Reference Chart
Table 10: NightScout Reference Chart

<table>
<thead>
<tr>
<th>Month</th>
<th>TotalPullRequest</th>
<th>TotalTimeToClose</th>
<th>AvgTimeToClose</th>
<th>NumberOfFilesChanged</th>
<th>#Descriptions</th>
<th>TotalDescLength</th>
<th>AvgDescLength</th>
<th>TotalMerged</th>
<th>MergeRatio</th>
<th>TotalLabels</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/1/2016</td>
<td>34</td>
<td>3.11</td>
<td>3.11</td>
<td>151</td>
<td>8</td>
<td>15</td>
<td>1.58</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2/1/2016</td>
<td>23</td>
<td>17.29</td>
<td>0.75</td>
<td>442</td>
<td>11</td>
<td>40</td>
<td>3.04</td>
<td>3</td>
<td>13.04</td>
<td>0</td>
</tr>
<tr>
<td>3/1/2016</td>
<td>16</td>
<td>11.19</td>
<td>0.7</td>
<td>124</td>
<td>5</td>
<td>16</td>
<td>3.2</td>
<td>1</td>
<td>6.25</td>
<td>0</td>
</tr>
<tr>
<td>4/1/2016</td>
<td>26</td>
<td>6.77</td>
<td>0.26</td>
<td>794</td>
<td>8</td>
<td>53</td>
<td>6.62</td>
<td>3</td>
<td>11.54</td>
<td>0</td>
</tr>
<tr>
<td>5/1/2016</td>
<td>28</td>
<td>12.46</td>
<td>0.45</td>
<td>1228</td>
<td>12</td>
<td>31</td>
<td>2.38</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6/1/2016</td>
<td>18</td>
<td>15.9</td>
<td>1.99</td>
<td>1635</td>
<td>4</td>
<td>7</td>
<td>1.75</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7/1/2016</td>
<td>13</td>
<td>12.98</td>
<td>1</td>
<td>314</td>
<td>9</td>
<td>39</td>
<td>4.33</td>
<td>1</td>
<td>7.69</td>
<td>0</td>
</tr>
<tr>
<td>8/1/2016</td>
<td>12</td>
<td>17.7</td>
<td>1.48</td>
<td>979</td>
<td>3</td>
<td>5</td>
<td>1.67</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>9/1/2016</td>
<td>20</td>
<td>80.00</td>
<td>1.6</td>
<td>1949</td>
<td>18</td>
<td>29</td>
<td>2.23</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>10/1/2016</td>
<td>87</td>
<td>25.69</td>
<td>0.3</td>
<td>1032</td>
<td>35</td>
<td>201</td>
<td>5.74</td>
<td>3</td>
<td>3.43</td>
<td>0</td>
</tr>
<tr>
<td>11/1/2016</td>
<td>41</td>
<td>25.93</td>
<td>0.63</td>
<td>1160</td>
<td>11</td>
<td>82</td>
<td>7.45</td>
<td>1</td>
<td>2.44</td>
<td>0</td>
</tr>
<tr>
<td>12/1/2016</td>
<td>52</td>
<td>45.28</td>
<td>0.87</td>
<td>512</td>
<td>17</td>
<td>86</td>
<td>5.06</td>
<td>2</td>
<td>3.85</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 11: Tensorflow Reference Chart

<table>
<thead>
<tr>
<th>Month</th>
<th>TotalPullRequest</th>
<th>TotalTimeToClose</th>
<th>AvgTimeToClose</th>
<th>NumberOfFilesChanged</th>
<th>#Descriptions</th>
<th>TotalDescLength</th>
<th>AvgDescLength</th>
<th>TotalMerged</th>
<th>MergeRatio</th>
<th>TotalLabels</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/1/2016</td>
<td>117</td>
<td>799.75</td>
<td>6.64</td>
<td>1152</td>
<td>88</td>
<td>2944</td>
<td>33.45</td>
<td>41</td>
<td>35.04</td>
<td>116</td>
</tr>
<tr>
<td>2/1/2016</td>
<td>152</td>
<td>729.25</td>
<td>4.8</td>
<td>817</td>
<td>114</td>
<td>3489</td>
<td>30.61</td>
<td>77</td>
<td>17.76</td>
<td>150</td>
</tr>
<tr>
<td>3/1/2016</td>
<td>182</td>
<td>1024.48</td>
<td>5.63</td>
<td>1493</td>
<td>128</td>
<td>3890</td>
<td>36.39</td>
<td>116</td>
<td>63.74</td>
<td>183</td>
</tr>
<tr>
<td>4/1/2016</td>
<td>173</td>
<td>797.63</td>
<td>4.61</td>
<td>5170</td>
<td>122</td>
<td>4597</td>
<td>37.58</td>
<td>128</td>
<td>73.99</td>
<td>174</td>
</tr>
<tr>
<td>5/1/2016</td>
<td>140</td>
<td>1041.53</td>
<td>7.44</td>
<td>66059</td>
<td>101</td>
<td>3339</td>
<td>33.06</td>
<td>99</td>
<td>76.71</td>
<td>149</td>
</tr>
<tr>
<td>6/1/2016</td>
<td>192</td>
<td>764.84</td>
<td>3.96</td>
<td>6160</td>
<td>122</td>
<td>3812</td>
<td>31.25</td>
<td>129</td>
<td>67.19</td>
<td>217</td>
</tr>
<tr>
<td>7/1/2016</td>
<td>140</td>
<td>596.18</td>
<td>4.26</td>
<td>2779</td>
<td>97</td>
<td>3092</td>
<td>35.27</td>
<td>101</td>
<td>72.14</td>
<td>174</td>
</tr>
<tr>
<td>8/1/2016</td>
<td>157</td>
<td>1192.12</td>
<td>7.59</td>
<td>5778</td>
<td>105</td>
<td>4032</td>
<td>37.33</td>
<td>116</td>
<td>73.89</td>
<td>213</td>
</tr>
<tr>
<td>9/1/2016</td>
<td>202</td>
<td>1089.29</td>
<td>5.44</td>
<td>9095</td>
<td>141</td>
<td>4628</td>
<td>32.82</td>
<td>142</td>
<td>70.3</td>
<td>274</td>
</tr>
<tr>
<td>10/1/2016</td>
<td>238</td>
<td>1227.18</td>
<td>3.45</td>
<td>11250</td>
<td>102</td>
<td>4647</td>
<td>28.09</td>
<td>140</td>
<td>62.34</td>
<td>306</td>
</tr>
<tr>
<td>11/1/2016</td>
<td>227</td>
<td>1327.66</td>
<td>5.85</td>
<td>6962</td>
<td>164</td>
<td>6160</td>
<td>57.56</td>
<td>163</td>
<td>71.81</td>
<td>275</td>
</tr>
<tr>
<td>12/1/2016</td>
<td>197</td>
<td>1058.88</td>
<td>5.37</td>
<td>8118</td>
<td>134</td>
<td>4579</td>
<td>34.17</td>
<td>126</td>
<td>63.96</td>
<td>248</td>
</tr>
</tbody>
</table>

Table 12: TheAlgorithms Reference Chart

<table>
<thead>
<tr>
<th>Month</th>
<th>TotalPullRequest</th>
<th>TotalTimeToClose</th>
<th>AvgTimeToClose</th>
<th>NumberOfFilesChanged</th>
<th>#Descriptions</th>
<th>TotalDescLength</th>
<th>AvgDescLength</th>
<th>TotalMerged</th>
<th>MergeRatio</th>
<th>TotalLabels</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/1/2018</td>
<td>11</td>
<td>5.92</td>
<td>0.54</td>
<td>19</td>
<td>9</td>
<td>382</td>
<td>42.44</td>
<td>11</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>2/1/2018</td>
<td>0</td>
<td>3.24</td>
<td>0.54</td>
<td>9</td>
<td>5</td>
<td>35</td>
<td>7</td>
<td>5</td>
<td>83.33</td>
<td>0</td>
</tr>
<tr>
<td>3/1/2018</td>
<td>25</td>
<td>8.18</td>
<td>0.33</td>
<td>50</td>
<td>10</td>
<td>158</td>
<td>13.8</td>
<td>25</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>4/1/2018</td>
<td>12</td>
<td>5.73</td>
<td>0.98</td>
<td>16</td>
<td>9</td>
<td>242</td>
<td>36.89</td>
<td>10</td>
<td>83.33</td>
<td>2</td>
</tr>
<tr>
<td>5/1/2018</td>
<td>7</td>
<td>496.93</td>
<td>99.27</td>
<td>11</td>
<td>0</td>
<td>219</td>
<td>36.5</td>
<td>5</td>
<td>71.43</td>
<td>5</td>
</tr>
<tr>
<td>6/1/2018</td>
<td>12</td>
<td>1976.77</td>
<td>184.73</td>
<td>15</td>
<td>10</td>
<td>167</td>
<td>18.7</td>
<td>5</td>
<td>41.67</td>
<td>2</td>
</tr>
<tr>
<td>7/1/2018</td>
<td>7</td>
<td>490.83</td>
<td>70.12</td>
<td>9</td>
<td>6</td>
<td>159</td>
<td>26.5</td>
<td>5</td>
<td>71.43</td>
<td>2</td>
</tr>
<tr>
<td>8/1/2018</td>
<td>2</td>
<td>875.56</td>
<td>437.78</td>
<td>2</td>
<td>2</td>
<td>22</td>
<td>11</td>
<td>1</td>
<td>50</td>
<td>0</td>
</tr>
<tr>
<td>9/1/2018</td>
<td>22</td>
<td>3790.88</td>
<td>171.86</td>
<td>119</td>
<td>19</td>
<td>515</td>
<td>27.11</td>
<td>8</td>
<td>36.36</td>
<td>2</td>
</tr>
<tr>
<td>10/1/2018</td>
<td>200</td>
<td>32315.6</td>
<td>161.6</td>
<td>533</td>
<td>104</td>
<td>1791</td>
<td>17.22</td>
<td>70</td>
<td>35</td>
<td>7</td>
</tr>
<tr>
<td>11/1/2018</td>
<td>24</td>
<td>1512.48</td>
<td>83.02</td>
<td>59</td>
<td>22</td>
<td>491</td>
<td>22.32</td>
<td>17</td>
<td>70.83</td>
<td>0</td>
</tr>
<tr>
<td>12/1/2018</td>
<td>47</td>
<td>6673.34</td>
<td>141.99</td>
<td>63</td>
<td>38</td>
<td>798</td>
<td>21</td>
<td>29</td>
<td>61.7</td>
<td>2</td>
</tr>
</tbody>
</table>
Table 13: Vinta Reference Chart

<table>
<thead>
<tr>
<th>Month</th>
<th>TotalPullRequest</th>
<th>TotalTimeToClose</th>
<th>AvgTimeToClose</th>
<th>NumberOfFilesChanged</th>
<th>NumberDescriptions</th>
<th>TotalDesLength</th>
<th>AvgDesLength</th>
<th>TotalMerged</th>
<th>MergeRatio</th>
<th>TotalLabels</th>
</tr>
</thead>
<tbody>
<tr>
<td>3/1/2015 0:00</td>
<td>14</td>
<td>5525.5</td>
<td>394.68</td>
<td>24</td>
<td>7</td>
<td>74</td>
<td>10.57</td>
<td>3</td>
<td>21.43</td>
<td>0</td>
</tr>
<tr>
<td>2/2/2015 0:00</td>
<td>5</td>
<td>2503.27</td>
<td>500.65</td>
<td>5</td>
<td>4</td>
<td>179</td>
<td>44.75</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3/1/2015 0:00</td>
<td>25</td>
<td>9790.31</td>
<td>388.01</td>
<td>25</td>
<td>14</td>
<td>359</td>
<td>25.64</td>
<td>9</td>
<td>36</td>
<td>0</td>
</tr>
<tr>
<td>4/1/2015 0:00</td>
<td>25</td>
<td>9725.06</td>
<td>389</td>
<td>25</td>
<td>14</td>
<td>161</td>
<td>11.5</td>
<td>7</td>
<td>28</td>
<td>0</td>
</tr>
<tr>
<td>5/1/2015 0:00</td>
<td>20</td>
<td>6729.54</td>
<td>336.48</td>
<td>20</td>
<td>9</td>
<td>157</td>
<td>17.44</td>
<td>6</td>
<td>30</td>
<td>0</td>
</tr>
<tr>
<td>6/1/2015 0:00</td>
<td>16</td>
<td>5248.07</td>
<td>328</td>
<td>16</td>
<td>7</td>
<td>92</td>
<td>13.14</td>
<td>7</td>
<td>43.75</td>
<td>0</td>
</tr>
<tr>
<td>7/1/2015 0:00</td>
<td>18</td>
<td>1907.84</td>
<td>217.1</td>
<td>13</td>
<td>15</td>
<td>158</td>
<td>10.38</td>
<td>8</td>
<td>44.44</td>
<td>0</td>
</tr>
<tr>
<td>8/1/2015 0:00</td>
<td>15</td>
<td>3575.01</td>
<td>226.4</td>
<td>15</td>
<td>6</td>
<td>72</td>
<td>12</td>
<td>8</td>
<td>53.33</td>
<td>0</td>
</tr>
<tr>
<td>9/1/2015 0:00</td>
<td>17</td>
<td>3659.16</td>
<td>225.84</td>
<td>17</td>
<td>12</td>
<td>264</td>
<td>17.7</td>
<td>7</td>
<td>31.18</td>
<td>0</td>
</tr>
<tr>
<td>10/1/2015 0:00</td>
<td>16</td>
<td>3070.86</td>
<td>279.57</td>
<td>16</td>
<td>10</td>
<td>1091</td>
<td>36.37</td>
<td>8</td>
<td>22.22</td>
<td>0</td>
</tr>
<tr>
<td>11/1/2015 0:00</td>
<td>36</td>
<td>3030.99</td>
<td>189.44</td>
<td>36</td>
<td>13</td>
<td>774</td>
<td>83.34</td>
<td>7</td>
<td>43.75</td>
<td>0</td>
</tr>
<tr>
<td>12/1/2015 0:00</td>
<td>36</td>
<td>3565.17</td>
<td>187.64</td>
<td>36</td>
<td>15</td>
<td>373</td>
<td>24.87</td>
<td>6</td>
<td>31.58</td>
<td>0</td>
</tr>
</tbody>
</table>
4.2 Results

With the use of the Reference chart for each of the ten repositories, it is now possible to look closely at the monthly progression of pull requests. The following section contains the results of both the statistical and inferential analysis of the monthly breakdown of pull requests. While this data mostly shows pull requests' characteristics, some repository behavior can be inferred from the data as well. The main components of a pull request, Title, labels, description, are discussed in their sections. The final section discusses the merge rate and merge time.

4.3 Graphs

The following section is dedicated to all the graphs made from the reference chart in 4.1. The content of the graphs is supplied by the reference charts with information from the interview conducted at the beginning of the project. Each graph contains one to two data sources to show correlations or differences in the scale of the pull request contents. The graphs are organized by repository and in the following order: Average Merge Time, Files Changed and Total Merged, Total Pull Requests and Total Merged, Number of Labels, Total Pull Requests and Total Descriptions, Total Pull Requests and Number of Files Changed, and finally, Total Merged and Total Labels. Each graph is made using the MatPlotLib Library for Python 3.8.
Figure 7: Angular Average Merge Time

Figure 8: Angular Files Changed and Total Merged
Figure 9: Angular Total Pull Requests and Total Merged

Figure 10: Angular Number of Labels
Figure 11: Angular Total Pull Requests and Total Descriptions

Figure 12: Angular Total Pull Requests and #Files Changed
Figure 13: Angular Total Merged and Total Labels

Figure 14: Bitcoin Average Merge Time
Figure 15: Bitcoin Files Changed and Total Merged

Figure 16: Bitcoin Total Pull Requests and Total Merged
Figure 17: Bitcoin Number of Labels

Figure 18: Bitcoin Total Pull Requests and Total Descriptions
Figure 19: Bitcoin Total Pull Requests and #Files Changed

Figure 20: Bitcoin Total Merged and Total Labels
Figure 21: Bootstrap Merge Time

Figure 22: Bootstrap Files Changed and Total Merged
Figure 23: Bootstrap Total Pull Requests and Total Merged

Figure 24: Bootstrap Number of Labels
Figure 25: Bootstrap Total Pull Requests and Total Descriptions

Figure 26: Bootstrap Total Pull Requests and #Files Changed
Figure 27: Bootstrap Total Merged and Total Labels

Figure 28: Django Merge Time
Figure 29: Django Files Changed and Total Merged

Figure 30: Django Total Pull Requests and Total Merged
Figure 31: Django Number of Labels

Figure 32: Django Total Pull Requests and Total Descriptions
Figure 33: Django Total Pull Requests and #Files Changed

Figure 34: Django Total Merged and Total Labels
Figure 35: FacebookReact Average Merge Time

Figure 36: FacebookReact Files Changed and Total Merged
Figure 37: FacebookReact Total Pull Requests and Total Merged

Figure 38: FacebookReact Number of Labels
Figure 39: FacebookReact Total Pull Requests and Total Descriptions

Figure 40: FacebookReact Total Pull Requests and #Files Changed
Figure 41: FacebookReact Total Merged and Total Labels

Figure 42: Keras Average Merge Time
Figure 43: Keras Files Changed and Total Merged

Figure 44: Keras Total Pull Requests and Total Merged
Figure 45: Keras Number of Labels

Figure 46: Keras Total Pull Requests and Total Descriptions
Figure 47: Keras Total Pull Requests and #Files Changed

Figure 48: Keras Total Merged and Total Labels
Figure 49: NighthScout Average Merge Time

Figure 50: NighthScout Files Changed and Total Merged
Figure 51: NightScout Total Pull Requests and Total Merged

Figure 52: NightScout Number of Labels
Figure 53: NightScout Total Pull Requests and Total Descriptions

Figure 54: NightScout Total Pull Requests and #Files Changed
Figure 55: NightScout Total Merged and Total Labels

YearDataNightScoutPR: Total Merged and Total Labels

Figure 56: Tensorflow Average Merge Time

YearDataTensorflowPR: AverageMergeTime
Figure 57: Tensorflow Files Changed and Total Merged

Figure 58: Tensorflow Total Pull Requests and Total Merged
Figure 59: Tensorflow Number Of Labels

![Bar chart showing the number of labels for each month.](image1)

Figure 60: Tensorflow Total Pull Requests and Total Descriptions

![Bar chart showing the total pull requests and total descriptions for each month.](image2)
Figure 61: Tensorflow Total Pull Requests and #Files Changed

Figure 62: Tensorflow Total Merged and Total Labels
Figure 63: TensorFlow Total Pull Requests and Total Descriptions

Figure 64: The Algorithms Average Merge Time
Figure 65: TheAlgorithms Total Pull Requests and Total Merged

Figure 66: TheAlgorithms Number of Labels
Figure 67: TheAlgorithms Total Pull Requests and Total Descriptions

Figure 68: TheAlgorithms Total Pull requests and Number Files Changed
Figure 69: The Algorithms Total Merged and Total Labels

Figure 70: Vinta Average Merge Time
Figure 71: Vinta Files Changed and Total Merged

Figure 72: Vinta Total Pull Requests and Total Merged
Figure 73: Vinta Number of Labels

Figure 74: Vinta Total Pull Requests and Total Descriptions
Figure 75: Vinta Total Pull Requests and #Files Changed

Figure 76: Vinta Total Merged and Total Labels
4.4 Titles

The data collection phase speculated that Titles might play an important role in pull request merge time. However, after continued interviews with real-world practitioners and data early in the data collection phase, it was decided that Titles have little if any meaningful effect on merge rates or the pull requests as a whole. The reason for this is less about Titles' effectiveness and more about how pull requests are used. According to the several different individuals interviewed, when a pull request is created, unless the Title specifically has certain keywords, i.e., "TEST PULL REQUEST DO NOT MERGE," the pull request is still thoroughly reviewed and checked before it is merged or declined. The reason for this action, at least according to the majority of the developers, is that the Title is not capable of explaining what a pull request is accomplishing. The Title can give a negligible amount of information and still requires a look at the files changed, description, and pull request labels. For these reasons, the titles have been deemed a non-factor for the success of pull requests.

**Title: Fixed failing function**

Figure 77: Short Pull Request Title

**Title: Adjusted directories and log files added functions for processing results through supported compilers**

Figure 78: Verbose Pull Request
Both the figures above show examples of issues titles either being short with not enough information or having too much information. A Title should give an idea of what feature was changed but not try to explain the feature changed.

4.5 Labels

The results of the data analysis showed that the use of labels, at least in the top ten most forked dataset, was about 50% usage. This meant that 50% of the repos used in this study didn’t make any use of labels at any point in their development cycle. This percentage is not indicative of every repository but is an important statistical observation for the validity of this study. One observation that can be made from comparing the repositories that use labels and the repositories that do not is the difference in how quickly pull requests are closed. Looking at the five repositories that use Labels, *Bitcoin, Bootstrap, FacebookReact, TensorFlow*, and *TheAlgorithms*, had faster close times even when the number of pull requests increased. Labels are predetermined by GitHub and currently don’t allow for custom labels, so they are simply tools for categorization. The assumption that can be made is that Labels allow a development team the ability to organize pull requests into sets. Each set can then be reviewed by a reviewer within that development category. If the reviewer finishes their set of pull requests they can move to review another set. The act of categorization creates a more simplistic way for the right developer to review the pull request. Without labels there is potentially lag time created when a developer checks a pull request and then forwards it to the correct developer for the review. Often times different development teams may work on the same repository but not be involved with each others work. A reviewer would need to be part of the development team that is aware of the work being done by that team. A larger scale review of the use of labels would need to be
constructed to formulate a realistic scale for the usefulness of the label feature on GitHub. However, with the lack of required format and since the content of each pull request is up to the developer creating it there would need to be enforcement of a repository to only accept pull requests with labels. Its up the repository owners to create a Label Requirement for pull requests if that is how they want to handle pull requests.

4.6 Descriptions

The description results required a decent amount of inference to understand as compared to the Labels. The length and content of the description is dependent on the developer who worked on that pull request. The large level of description variance is why it takes more time to review a pull request that only has a description. If a description does not explain the changes made in the pull request in enough detail the reviewer will have to manually go through each of the files changed to understand what has been changed. The results of the data analysis showed that the longer the description the longer the merge time. However, according to interviews with real world practitioners, the more detailed a description the higher chance for a successful merge. A poorly written description adds merge time due to the difficulties of deciphering and understanding what the contributor attempted in the pull request. For organized development teams this issue can be avoided by having a standard for writing pull request descriptions. The potential delay comes into place when a repository allows for open pull requests from the development community.

4.7 Merge Rate

A key part of this research is understanding what effects the merge time and merge of pull requests. If a developer spends a large portion of time on a pull request that cannot be
merged, that developer has just created a development lag in that project's software development life cycle. Another issue that arises is a developer spending time on a new feature or bug fix to open a pull request that is either never reviewed or reviewed at a point that makes the changes non-viable at the current version. For this reason, during the data analysis phase, a merge ratio was calculated for that specific month. For this research, the merge ratio factor pull requests opened during a month but closed during another month. The merge ratio shows how successful a development team was in making progress in pull request closure. In the reference charts for the repositories, four columns are important for this section. The columns that need to be looked at are Total Pull Requests, Avg. Time To Close, Total Merged, and Merge Ratio. To understand this information, first look at Total Pull Requests and Total Merged. These two columns show how many pull requests had been opened and merged that month. Secondly, review the merge ratio column. This column contains the percentage of pull requests successfully closed. Finally, looking at the Avg. Time To Close if the month's average is under 30.00 days, it suggests that most pull requests had been completed within the 30-31 days of that month. Any number larger than 30 suggests that many pull requests had not been closed by the end of the month. After viewing all these columns, it is possible to understand how successful a pull request had been in a certain month. A Team Lead or Project Manager would be interested in understanding why the Avg. Time To Close is greater than thirty. A repository owner would be concerned about the merge ratio because it would indicate potential bottlenecks in the reviewing and integrating process.
CHAPTER 5

5.1 Threats to Validity

While this research was conducted to limit potential threats to validity, there are still a few discussed in this section. The introduction of threats comes from the use of this work to benefit future work. The below list are all the threats recognized during the research:

- Size of Development Group
- Popularity of Repository
- No Uniform Pull Request Format

The development group's size dedicated to each repository is a potential threat due to how it changes merge speed. For example, repository A has five individuals working as pull requests reviewers, and repository B has twenty individuals working as reviewers; there will be a noticeable difference in the repository's merge time. If both repositories receive 30 pull requests each week, repository A and B's workload will look dramatically different.

The popularity of a repository is a threat that can have similarities to the previous threat. The more popular the repository or how active a repository actively affects the number of pull requests the repository receives: the more pull requests, the more time required to review. A quick look at the reference charts in Chapter 3 will show that repositories often receive high numbers of pull requests, and not all of them result in merges. Any pull request that does not pass review reduces the effectiveness of the reviewers.
The final threat to validity comes from the varying development cultures that exist on GitHub. As can be seen in Chapter 3, not every repository uses all the tools that GitHub Offers. An example of this is the sporadic use of Labels between repositories. This threat was also well known from the beginning of the research. During the interview phase of the study, it was discovered that not every development team used Github labels. The reason is that some development teams are small and the individuals developing are also the ones reviewing the pull requests. Some development teams make use of pair programming, and one of the developers in the pair will review the others pull requests. With this threat recognized, it became a part of the research to see if the use of labels added any benefits.

5.3 Contribution

The findings of this paper are to show that even though the Pull-Based Development model common place and well accepted there are still ways to improve how it is utilized in the real-world practice of software development. The interview phase of the empirical method showed how many developers encounter certain issues with the open-source structure of pull requests. The value of open sourcing projects to developers in the online community can speed up development time and limit cost however as shown in this paper there are a few downsides. Majority of the potential issues outlined in this paper are due to lack of transparency within the developer and reviewer of pull requests. The issue with pull request that following section will attempt to overcome is the ability to successfully track fork and pull requests. The following figure shows potential technical lag that can be caused when using private development repository or public outsourced repository website such as GitHub.
The figure above introduces the main issue prevalent in pull-based development. Development Team A is spending development time on a feature that because it hasn’t been reviewed yet is unknown whether it is a practical solution. Due to commitments in the development cycle of the project Team A must begin work on new features. If the previous feature pull request from Team A gets rejected or fails to merge during the future sprints Team A will be pulled away from their current work to work on the previous sprint work. The problem that emerges from Team A having to resume work on a previous feature is now there will be technical lag that could cause the project to fall behind. One of the main issues that cause the technical lag in this situation is the lack of transparency in tracking pull requests.
The rest of this section will describe a process and tool for tracking the implementation and pull request phase of development using the pull-based development. The following is a description and list of features that this specific process would require to successfully and efficiently manage pull requests. The essential parts of the process are the Feature Node and the Implementation List.

```
FeatureNode:
{
    FeatureName: string
    SprintNumber: int
    LOCChanged: int
    NumberOfDevelopers: int
    Labels: list[]
    Description: string
    Title: string
    ForkDate: Date:Time
    CloseDate: Date:Time
    MergeStatus: Boolean
}
```

Figure 80: FeatureNode Data

The Implementation Node contains the FeatureName, SprintNumber, ForkDateandTime, LOCChanged, NumberOfDevelopers, Labels, Description, Title, CloseTime, MergeStatus. Each of these data points will be packed into a data structure known as a Node. A Node is a basic unit for a data structure that can be used to store data and connect to other nodes. The previous image shows the potential data layout of this node. The node is the important part for tracking individual features and the development information assigned to these features.

The next crucial part of this framework is the Implementation List. The Implementation List contains each Feature Node that is involved in that project. The goal of the implementation list is to create a data structure that can generate statistics and track the implementation of the project. The implementation list is simply a container for all the Feature Nodes. Its called the
Implementation list because after all the feature nodes are added it is a complete implementation of the project.

5.2 Future Work

The data gathered and research conducted in this paper will be used in future work to create an Automated Pull Request Summary Generator. The first part of this research was the data gathering and interview stage. There needed to be empirical evidence that suggested pull-based development can have technical development lag. Meaning, during development, certain features or requirements fall behind schedule and the progress of other deliverables. These behind schedule tasks are either abandoned after some time or continuously developed. The future work will be to design tools to help design tools for tracking and eliminating behind schedule or outdated tasks. The Automated Pull Request Summary Generator will involve interviews with industry practitioners to create a valuable software engineering tool. The tool will take advantage of Pull Based Development on GitHub by looking at the forking of a repository and all the pull requests that come from that forked branch.

5.4 Conclusion

The research presented in this paper demonstrates that there are a few key aspects of Pull Based Development that cause development lag. The key feature about this development lag is that it scales with company size. In an industry with varying sizes of development teams and companies lost time in development can cause financial loss. The main body of this work justifies the need for further resource tracking in the Pull Based Development Model usage scenarios. The contribution of this paper is not just the interviews and graphs but also the justification for a new type of Fork and Pull tracking data structure that can be used as the
steppingstone for future work branching from this paper. The final part of this research is the future work which will be the creation of a machine learning driven automated sprint summary tool for the Pull Based Development Model practitioners.
APPENDIX
BIBLIOGRAPHY


