Using Github Profiles in Software Developer Recruitment and Hiring

*Master’s Thesis*

ADITYA REDDY NAGARAM
Using Github Profiles in Software Developer Recruitment and Hiring

THESIS

submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER SCIENCE

by

ADITYA REDDY NAGARAM
born in Jagtial, India

TU Delft

Software Engineering Research Group
Department of Software Technology
Faculty EEMCS, Delft University of Technology
Delft, the Netherlands
www.ewi.tudelft.nl
Using Github Profiles in Software Developer Recruitment and Hiring

Abstract

Social coding platforms can provide initial understanding about the skills exhibited by the developers on these platforms. In contexts where candidates social profile information is useful for recruiting software developers, the information regarding the developers on these platforms can be leveraged by the recruiters with some software knowledge. However, recruiters have to put many efforts in inferring about a developer skill on social coding platforms.

In this thesis, we investigate on providing relevant information regarding software developer capabilities on a social coding platform to the recruiters. We used GitHub as our social coding platform for this purpose. We explored regarding, the attributes to use for indicating the skills exhibited by a developer on GitHub. We also investigated GitHub as a resource containing some potential software developer candidates by recommending GitHub developer profile, solely based on skill set requirements of job advertisements.

Our results indicate that the generated developer skill profiles have a valid set of attributes when combined, to indicate the regarding three skills exhibited by a software developer on GitHub. However, the generated profile was slightly preferred by the technical recruiters because of the profile’s complexity in understanding and incompleteness. In the investigation of recommending developer profiles to suit job advertisement requirements our recommendation strategy could only achieve a precision of 0.39 on average and an NDPM ranking accuracy value of 0.43 on average.
Thesis Committee:

Chair: Prof. Dr. Arie Van Deursen, Faculty EEMCS, TU Delft
University supervisor: Dr. Claudia Hauff, Faculty EEMCS, TU Delft
Committee Member: Dr. Georgios Gousios, Digital security group, Radbound Universiteit
Over the past two years, my journey as a master’s student of TU Delft have been a great experience. Firstly, I would like to express my sincere thanks to Claudia Hauff, who have guided me in every phase of work and helped me in the successful completion of my thesis work. I admire her patience to correct me in all my shallow works during my thesis work.

I would like to extend my sincere thanks to Georgios Gousios, who have guided me in the initial part of thesis work. Furthermore, his feedback in finalising the thesis report was useful and have also accepted to be part of my thesis committee. I would also like to thank Arie van Deursen for his support while finalising the thesis report.

Finally, I would like to thank my brother’s family in the Netherlands and my friends who have always supported me in all the ways possible during my studies in Delft. I would particularly like to thank all my friends and other respondents who have extended their support during evaluation of my thesis. Last, but not least, I am grateful for the support and love my family have given me all through my life to achieve my dreams.

ADITYA REDDY NAGARAM
Delft, the Netherlands
August 13, 2015
# Contents

<table>
<thead>
<tr>
<th>Preface</th>
<th>iii</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contents</td>
<td>v</td>
</tr>
<tr>
<td>List of Figures</td>
<td>ix</td>
</tr>
<tr>
<td>List of Tables</td>
<td>xi</td>
</tr>
</tbody>
</table>

## 1 Introduction

1.1 Problem Description .................................................. 2
1.2 Research Goal and Questions ................................. 2
1.3 Thesis Workflow ...................................................... 3
1.4 Thesis Structure .................................................... 4

## 2 Related Work

2.1 Social coding platform GitHub ........................................ 5
  2.1.1 GitHub work flow .............................................. 5
  2.1.2 Peer Developer Profiling ................................... 6
  2.1.3 Recruiter Developer Profiling ......................... 7
  2.1.4 Attributes used in literature to indicate popularity of projects on GitHub ........................................ 8
  2.1.5 Conclusions on investigated GitHub related work .......... 9
2.2 Recommendation Systems ............................................. 10
  2.2.1 Recommendation system task ................................ 10
  2.2.2 Recommendation Algorithm Techniques ....................... 11
  2.2.3 Recommendation Systems For Facilitating Recruitment .... 13
  2.2.4 Conclusions on investigated recommendation systems related work ........................................ 14

## 3 Approach

3.1 Software developer profile ........................................... 15
  3.1.1 Approach ....................................................... 15
# CONTENTS

3.1.2 Collaboration skills exhibited by GitHub user .......................... 16  
3.1.3 Project Management Ability ............................................... 17  
3.1.4 Programming language skill ............................................. 19  
3.2 Recommending developer profiles to a job advertisement ............... 21  
3.2.1 Approach ................................................................. 21  
3.2.2 Entity types differences between the keyword sets of a job advertise- 

tisement and developer profile ........................................... 21  
3.3 Evaluation Approach .......................................................... 23  
3.3.1 Evaluating developer profile ............................................ 23  
3.3.2 Evaluating a recommendation task ................................. 23  

4 Implementation 25  
4.1 Workflow overview for developer profile and recommendations ........... 25  
4.2 Software Developer Profile Generator ..................................... 27  
4.2.1 GHTorrent GitHub user’s data ........................................ 27  
4.2.2 Generating collaboration skill ........................................... 29  
4.2.3 Generating project management skill ................................ 30  
4.2.4 Generating level of programming language skill profile ............... 31  
4.3 Recommending Profiles ....................................................... 33  
4.3.1 Schema used for recommendations .................................... 33  
4.3.2 Job advertisement data indexing ...................................... 33  
4.3.3 Representing software developer profile to accomplish recommen- 

dations ............................................................... 36  
4.3.4 Recommendation strategy ................................................. 41  

5 Evaluation 47  
5.1 Evaluating Generated Developer Profile ................................... 47  
5.1.1 Assessment aspects of developer profile ............................ 47  
5.1.2 Questions for the developer profile assessment .................... 48  
5.1.3 Questionnaire and Interview setup for evaluation ................. 52  
5.2 Evaluating Recommendations Of Profiles For The Job Advertisements 54  
5.2.1 Metrics Selection for Evaluating the recommendations .......... 55  
5.2.2 Experiment setting for evaluation of recommendations ............ 56  

6 Results 61  
6.1 Software Developer Profile Assessment Results ........................ 61  
6.1.1 ASP1: Developer profile and its attributes validity ............... 62  
6.1.2 Developer Profile Assessment in the Recruitment Process .......... 82  
6.1.3 Conclusions on the Proposed Developer Profile ................... 89  
6.2 Accuracy of the recommendation task ..................................... 90  
6.2.1 Profile Recommendation Precision .................................. 90  
6.2.2 Ranking Accuracy ...................................................... 92  
6.2.3 Draw Backs of our recommendation Strategy ...................... 93
## 7 Conclusions and Future Work

### 7.1 Contributions

- GitHub Developer Skill Profiles

### 7.2 Discussion

- Recommending developer profiles to job advertisements

### 7.3 Future work

## Bibliography

## A Implementation

## B Evaluation
List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Thesis Work-flow</td>
<td>3</td>
</tr>
<tr>
<td>2.1</td>
<td>Basic Workflow on GitHub for a Repository</td>
<td>6</td>
</tr>
<tr>
<td>3.1</td>
<td>Graph on GitHub developer profile indicating the amount of public activity the developer</td>
<td>16</td>
</tr>
<tr>
<td>3.2</td>
<td>Samples of software related job advertisement and a developer profile</td>
<td>22</td>
</tr>
<tr>
<td>4.1</td>
<td>Workflow for generating profile and recommendations</td>
<td>26</td>
</tr>
<tr>
<td>6.1</td>
<td>Steps in a Traditional recruitment process where respondents preferred to use the developer profile, obtained from the interview</td>
<td>83</td>
</tr>
<tr>
<td>6.2</td>
<td>Steps in a Traditional recruitment process where respondents preferred to use the developer profile, obtained from the questionnaire</td>
<td>84</td>
</tr>
<tr>
<td>6.3</td>
<td>Precision Values for the recommended developer profiles to job advertisements</td>
<td>91</td>
</tr>
<tr>
<td>A.1</td>
<td>Schema used for indexing documents</td>
<td>103</td>
</tr>
<tr>
<td>B.1</td>
<td>Attributes descriptions and questions asked For the collaboration skills</td>
<td>105</td>
</tr>
<tr>
<td>B.2</td>
<td>Attributes descriptions and questions asked For the project management skills</td>
<td>106</td>
</tr>
<tr>
<td>B.3</td>
<td>Attributes descriptions and questions asked For the programming language skills</td>
<td>107</td>
</tr>
<tr>
<td>B.4</td>
<td>Infographic used for the collaboration skills</td>
<td>108</td>
</tr>
<tr>
<td>B.5</td>
<td>Infographic used for the project management skills</td>
<td>109</td>
</tr>
<tr>
<td>B.6</td>
<td>Infographic used to indicate the programming language skills</td>
<td>110</td>
</tr>
<tr>
<td>B.7</td>
<td>Instructions used to describe the task of the crowdflower evaluation job</td>
<td>111</td>
</tr>
<tr>
<td>B.8</td>
<td>Instructions used to describe the task of the crowdflower evaluation job</td>
<td>112</td>
</tr>
<tr>
<td>B.9</td>
<td>Instructions used to describe the task of the crowdflower evaluation job</td>
<td>113</td>
</tr>
<tr>
<td>B.10</td>
<td>Job advertisement presented in the task of crowdflower evaluation job</td>
<td>113</td>
</tr>
<tr>
<td>B.11</td>
<td>Developer Profile presented in the task of crowdflower evaluation job</td>
<td>114</td>
</tr>
</tbody>
</table>
### List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Project attributes usage in the literature for indicating project popularity or success</td>
<td>9</td>
</tr>
<tr>
<td>2.2</td>
<td>Ways in which recommendation techniques are combined</td>
<td>13</td>
</tr>
<tr>
<td>2.3</td>
<td>Few job recommendation systems in literature</td>
<td>14</td>
</tr>
<tr>
<td>3.1</td>
<td>Attributes to indicate collaboration skills exhibited by a developer on GitHub</td>
<td>17</td>
</tr>
<tr>
<td>3.2</td>
<td>Attributes to indicate project management skills</td>
<td>18</td>
</tr>
<tr>
<td>3.3</td>
<td>Attributes to indicate the quantity of code contributions</td>
<td>20</td>
</tr>
<tr>
<td>4.1</td>
<td>Basic properties of the developer profile set used for recommendation</td>
<td>28</td>
</tr>
<tr>
<td>4.2</td>
<td>Top 10 programming language distribution in the generated profile data set</td>
<td>28</td>
</tr>
<tr>
<td>4.3</td>
<td>Basic Schema Details for Indexing</td>
<td>33</td>
</tr>
<tr>
<td>4.4</td>
<td>Properties of the job advertisement data set used in recommendation task</td>
<td>34</td>
</tr>
<tr>
<td>4.5</td>
<td>Top 10 programming languages distribution in the job advertisement data set</td>
<td>35</td>
</tr>
<tr>
<td>4.6</td>
<td>Initial Score For Each Commits Category</td>
<td>39</td>
</tr>
<tr>
<td>4.7</td>
<td>Changes Made to the initial Score According to Commits Category</td>
<td>39</td>
</tr>
<tr>
<td>4.8</td>
<td>Recomputing the Percentage of Commits According to Commits Category</td>
<td>40</td>
</tr>
<tr>
<td>4.9</td>
<td>Combining initial score with recomputed percentage of commits</td>
<td>40</td>
</tr>
<tr>
<td>4.10</td>
<td></td>
<td>43</td>
</tr>
<tr>
<td>6.1</td>
<td>Median and variance values for each attribute on relevance to indicate collaboration skill based on recruiters responses from the interview</td>
<td>63</td>
</tr>
<tr>
<td>6.2</td>
<td>Opinion on set of attributes together would better indicate collaboration skill based on recruiters responses from the interview</td>
<td>64</td>
</tr>
<tr>
<td>6.3</td>
<td>Median and variance values for each attribute on relevance to indicate collaboration skill based on recruiters responses from questionnaire</td>
<td>65</td>
</tr>
<tr>
<td>6.4</td>
<td>Opinion on set of attributes together would better indicate collaboration skill based on recruiters responses from the questionnaire</td>
<td>65</td>
</tr>
<tr>
<td>6.5</td>
<td>Median and variance values for each attribute on relevance to indicate collaboration skill based on developers responses from questionnaire</td>
<td>66</td>
</tr>
<tr>
<td>6.6</td>
<td>Opinion on set of attributes together would better indicate collaboration skill based on developers responses from the questionnaire</td>
<td>67</td>
</tr>
</tbody>
</table>
### List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.7</td>
<td>Median and variance values for each attribute, on relevance to indicate project management skill based on recruiter’s responses from the interview</td>
<td>69</td>
</tr>
<tr>
<td>6.8</td>
<td>Opinion on set of attributes together would better indicate project management skill based on recruiters responses from the interview</td>
<td>70</td>
</tr>
<tr>
<td>6.9</td>
<td>Median and variance values for each attribute, on relevance to indicate project management skill based on recruiter’s responses from the questionnaire</td>
<td>71</td>
</tr>
<tr>
<td>6.10</td>
<td>Opinion on set of attributes together would better indicate project management skill based on recruiters responses from the questionnaire</td>
<td>72</td>
</tr>
<tr>
<td>6.11</td>
<td>Median and variance values for each attribute on, relevance to indicate project management skill based on developer’s responses from the questionnaire</td>
<td>73</td>
</tr>
<tr>
<td>6.12</td>
<td>Opinion on set of attributes together would better indicate project management skill based on recruiters responses from the questionnaire</td>
<td>74</td>
</tr>
<tr>
<td>6.13</td>
<td>Median and variance values for each attribute on, relevance to indicate programming language skill based on recruiters responses from the interview</td>
<td>76</td>
</tr>
<tr>
<td>6.14</td>
<td>Opinion on set of attributes together with commit category would better indicate programming language skill based on recruiters responses from the interview</td>
<td>77</td>
</tr>
<tr>
<td>6.15</td>
<td>Median and variance values for each attribute on, relevance to indicate programming language skill based on recruiters responses from the questionnaire</td>
<td>78</td>
</tr>
<tr>
<td>6.16</td>
<td>Opinion on set of attributes together with commit category would better indicate programming language skill based on recruiters responses from the questionnaire</td>
<td>79</td>
</tr>
<tr>
<td>6.17</td>
<td>Median and variance values for each attribute on, relevance to indicate programming language skill based on developers responses from the questionnaire</td>
<td>80</td>
</tr>
<tr>
<td>6.18</td>
<td>Opinion on set of attributes together with commit category would better indicate programming language skill based on developers responses from the questionnaire</td>
<td>81</td>
</tr>
<tr>
<td>6.19</td>
<td>Median and variance values for each skill profile on ease of understanding based on recruiters responses from the interview and questionnaire</td>
<td>85</td>
</tr>
<tr>
<td>6.20</td>
<td>Usage preference median values for each skill profile from respondents with recruiting experience from the interview and questionnaire</td>
<td>87</td>
</tr>
<tr>
<td>6.21</td>
<td>Precision Metric Statistical Values</td>
<td>91</td>
</tr>
<tr>
<td>6.22</td>
<td>Normalised Distance-Based Performance Measure Statistical Values</td>
<td>92</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

Social media platforms such as Facebook, LinkedIn and Twitter enable millions of users across the world to interact and share content. The content shared on these platforms reflects the interests and views of their users. The information inferred from this shared content is useful for organisations in making key decisions and provisioning better services. For instance, this information can be used by enterprises for provisioning personalised web services to their customers. Similarly, this increasing influence of social media triggered a new advancement in the field of candidate recruitment, called as social recruiting. In such a kind of recruitment, recruiters use available information regarding potential candidates on social media platforms in their recruitment process. This information provides an opportunity for the recruiters to make more informed decisions while recruiting candidates. A recent survey[1] conducted by Jobvite, a recruiting platform, indicates that 93% of recruiters review a candidate’s social profile before making a hiring decision on the applicants, indicating the growing popularity of social recruiting. Furthermore, social networking profiles of the candidates does have an impact on the recruitment decisions [5].

In the context of social recruiting of software developers, information regarding potential candidates on social coding platforms could be of a great significance to recruiters. Social coding platforms allow developers located remotely to collaborate in writing faster and better code. There are various social coding platforms such as GitHub, BitBucket, and Google Code, which help developers to collaborate during software development. These platforms also accommodate community-based projects from large enterprises such as Google, Microsoft and Amazon where the quality of code contributions are of high standards. Among social coding platforms, GitHub[2] is the most prominent one, which claims to be world’s largest code hosting site and has over 8 million users[3]. Moreover, GitHub provides access to developers information such as their contributions and code commits on public projects. Insights such as the quality of contributions made by a developer on GitHub could provide an initial understanding of a potential candidate’s coding capabilities [25]. Therefore, GitHub platform can be viewed as a resource for recruiters, containing a pool of talented

[3]https://github.com/about
Considering the rising popularity of social recruiting and the kind of potential GitHub platform offers, GitHub can be leveraged for software developer recruitment. However, GitHub does not present information specifically for the recruiters to infer about skills exhibited by a developer. Therefore, recruiters interested in knowing about the skills exhibited by a developer have to manually browse the corresponding developer’s GitHub profile. Marlow et al. [25] indicated that recruiters have to invest a lot of effort and time to assess certain relevant skills exhibited by developers on GitHub. For instance, to assess the quality of contributions by a developer, recruiters have to browse through the repositories that a developer has contributed to and check for the corresponding code commits by that developer. Furthermore, if the contributed repositories have more than one programming language, recruiters have to browse further through each commit for assessing the programming languages in which the developer has coded in. Finally, if the recruiters wanted to find relevant developers for their job positions then the task of the recruiter becomes much more complicated because finding appropriate candidates on GitHub would involve assessing skills of a large set of developers. Therefore, in this thesis, we focus on leveraging GitHub user information that addresses these shortcomings for recruiting software developers.

1.1 Problem Description

During our preliminary research, we found that online trace information of developers on GitHub has potential to indicate developer skills, which can be leveraged in context of social recruiting of software developers. The main reason behind such a choice is GitHub’s growing network of software developers and their contributions to highly influential open source software projects. However, recruiters still have to manually analyse and verify skills of the developers on GitHub for their recruitment needs. Thus, to make a relevant advancement along facilitation of GitHub user information for social recruiting of software developers we identify the following two issues from recruiter’s perspective in scope of this thesis:

1. Automatised extraction of relevant skill profile of a software developer on GitHub is still a challenge.

2. A recommendation strategy that minimises the efforts of finding relevant developers on GitHub that suit recruiter’s skill set requirements is still a challenge.

1.2 Research Goal and Questions

The goal of this thesis is to explore extraction of relevant skill profile of a developer on GitHub and recommend developer profiles according to recruiter’s skills set requirements while minimising their efforts in social recruiting.
From the research objective of this thesis, the following two research questions are formulated:

**RQ1:** What are the appropriate set of attributes for indicating relevant skills of a developer on GitHub to facilitate social recruitment?

**RQ2:** What recommendation strategy is appropriate for recommending developer profiles to suit recruiter’s skill set requirements?

### 1.3 Thesis Workflow

The work flow for our thesis is provided in the figure 1.1 and each step of our work flow is described as follows:

![Figure 1.1: Thesis Work-flow](image-url)
1. **INTRODUCTION**

1. Step 1: Based on our initial research on challenges in using online trace information of developers on social coding platforms for software developer recruitment, we defined our research objective.

2. Step 2: We explored the research work that focused on inferring skills from the social coding platforms. This helped us in understanding the kind of signals that are being used for inferring developer skills to formulate developer skill profile. Furthermore, we explored the research on recommendation techniques and job recommendation systems, which helped us to formulate our recommendation strategy and select an appropriate source for inferring recruiters skill set requirements.

3. Step 3: Based on the insights we gained from the related research work, we proposed a developer profile to indicate the skills of a developer on GitHub. Similarly, we formulated a recommendation strategy for accomplishing our research objective.

4. Step 4: After the formulation of the developer profile and the recommendation strategy, we implemented our solution using appropriate set of tools and techniques.

5. Step 5: We then evaluated our proposed developer profile and the recommendation strategy using appropriate metrics and experimental setup.

6. Step 6: Finally, we validated our research findings based on the results we obtained from our evaluation procedure.

1.4 **Thesis Structure**

The rest of the thesis report is organised as follows. In Chapter 2, we mention the related work on social coding platforms and existing recommendation systems. In Chapter 3, we describe the set of attributes used to indicate the skills exhibited by a GitHub user along with the recommendation strategy for realising our research objective. In Chapter 4, implementation details regarding extraction of a developer skill profile based on identified set of attributes along with the recommendation of developer skill profiles to suit a job advertisement based on its skill set requirements. In Chapter 5, we describe the experimental setup and metrics for evaluating our implementation approach and strategy. In Chapter 6, we mention and discuss the results to validate our research objective. Finally, Chapter 7 concludes this thesis along with our contributions and recommendations for future work.
Chapter 2

Related Work

In this chapter, we mention how previous research perceived social coding platforms to infer about skills exhibited by a developer and what kind of recommendation techniques are being used to accomplish recommendation tasks. Furthermore, we discuss few recommendation systems that aimed to accomplish job recommendation tasks. Before presenting the previous work in the respective fields, we introduce some basic concepts required to follow the details in further chapters quickly.

2.1 Social coding platform GitHub

In this section, we first briefly mention, how social coding platforms facilitate collaborative development by using GitHub workflow. We then discuss regarding social coding research in three directions that are relevant to address our research questions. In section 2.1.2, we describe how peers tried to infer about the skills of a developer on social coding platforms. In section 2.1.3, we describe how the recruiters tried to infer skills of a developer on these platforms. In section 2.1.4, we discuss regarding the kind of attributes considered on these platforms for inferring about the popularity of projects. Finally in section 2.1.5, we summarise the related work on the social coding sites and draw few conclusions that would help us in accomplishing our research objective.

2.1.1 GitHub workflow

GitHub, as mentioned previously in the introduction chapter, is a prominent social coding platform. Two essential aspects were considered for enabling social coding on GitHub. Firstly, in a collaborative environment there are many developers who might want to contribute a few lines of code to a project. Secondly, a project owner should be able to manage the incoming code contributions efficiently.

Figure 2.1 provides basic workflow scenario for a project on GitHub to enable social coding. GitHub provides a concept called forking through which any developer can get a personal copy of the project and make appropriate changes to that personal copy. For a developer to be able to propose changes, GitHub has another concept called pull request which enables a developer to propose changes.

5
2. RELATED WORK

A project owner (or) collaborators of the project have rights to manage the received pull requests appropriately. Having received pull requests from developers, a collaborator can handle the pull requests in either of the following ways:

- Directly merge or reject a pull request
- Collaborators can discuss regarding that pull request and take a collective decision, in which even the developer who made the pull request can participate.
- A collaborator or Collaborators could propose further changes on pull request for merging a pull request.

Furthermore, GitHub allows developers to contribute to a particular project by providing their opinion on the pull requests received for that project. The other ways in which a developer could provide contributions are by pointing out the issues with the project or solve the issues reported by another GitHub developer.

2.1.2 Peer Developer Profiling

The research explored regarding social coding platforms in diverse directions. The initial research on social coding platform investigated about basic properties such as geographical
distribution [33] of GitHub developers. The research on social coding platforms further advanced towards understanding in-depth properties of the social coding ecosystem. For instance, Kochhar et al. [20] investigated the correlation between test cases in a project with various project development characteristics by using GitHub. In the context of our thesis problem, we looked at the research that focused on inferring skills of a developer on social coding platform.

Dabbish et al. [11] studied the value of transparency in a collaborative development environment by considering GitHub as their collaborative development environment. Dabbish et al. discussed regarding developers looking for contributors to their project based on the visible signals about developers. Dabbish et al. indicated that cues such as, recent and volume of activity were being used to infer about the interest and level of commitment by the developer. For a recruiter, it could be interesting to know regarding the characteristics of a potential candidate, in this case knowing about the level of commitment of a candidate.

Marlow et al. [26] while referring to Rigby et al. [29] indicated that contributions from inexperienced or unknown contributors that required changes were not accepted into the project. Marlow et al. also mentioned some cues that were used to judge coding ability of the contributors. For example, signals such as Amount of activity, frequency of commits, the number of projects owned vs. forked, languages used were considered to infer about general coding ability of a developer. Knowing about the coding ability of a potential candidate would be useful for a recruiter to take an informative decision.

Bosu et al. [6] performed a survey on peer impressions in an open source development environment. Since social coding platforms are host to many open source projects, their investigation on peer impressions would be useful for our thesis problem. One of the research questions investigated was, what makes the participants of an open source project to lose trust on a contributor. Their results on the question found factors related to coding, professional and, interpersonal could make peers loose trust on a peer. Another hypothesis discussed was regarding evaluating the productivity or competency of a developer. To infer about this, peers used the contributions of a participant to a project as an important factor. For a recruiter, it would be useful to know if a potential candidate is trust worthy or not and also about the candidate productivity.

Although we could identify some research work that investigated regarding the signals used to infer about developer characteristics and skills, the research was not done specifically from a recruiter’s perspective. The discussed signals from the peers perspective can be useful if the recruiters are also considering the same signals for inferring the respective characteristics and skills of a candidate. Thus, we looked further for the research work that investigated on signals being used specifically from the recruiters perspective.

### 2.1.3 Recruiter Developer Profiling

Marlow et al. [25] discussed few signals that recruiters considered reliable to use for knowing about the developer skills. Marlow et al. used semi-structured interview approach to collect the responses and asked recruiters regarding their recent usage of GitHub in hiring process. The responses from the recruiters were analysed to know about the signals recruiters were using to infer about the skills exhibited by the developers on GitHub. Marlow
et al. provided four reliable signals namely, active open source involvement, contributions to the high-status projects, project ownership and, side projects to infer about developer skills. Furthermore, recruiters who took part in the felt that verifying some of the reliable signals took much effort. Following are the major signals that were discussed in [25]:

- **Active Open Source Involvement** Recruiters inferred the signal of active open source involvement as having understanding about shared open source values.

- **Contributions accepted to high-status project** Recruiters viewed the acceptance of contributions to a high-status project as developers quality of work.

- **Project Ownership** If a developer had project ownership and managing contributions then the developer was considered to have some project management skills.

- **Side Projects** If a developer had side projects then recruiters inferred that the developer had a passion for coding.

- **Number of watchers or forks of a project** This signal indicates a project’s popularity but the recruiters were sceptical about using this signal and considered the signal unreliable.

Singer et al. [31] mentioned some sites that extracts aggregated developer profiles. For instance, MasterBranch[^1] it uses developer data from different sources such as StackOverflow[^2], GitHub, BitBucket, GoogleCode and computed DevScore for a developer along with the same skills list of a developer, but how a DevScore was computed for a user was not known precisely. Singer et al. interviewed few recruiters to understand their perspective on the aggregated developer profile. Recruiters started by filtering developers by the skills list and then further looked at the achievements on skills mentioned in the profile. Recruiters with software knowledge were assessing the technical skills based on the endorsements received from other developers that were similar to what Marlow et al. [25] had indicated about code being accepted by the project community. Singer et al. [31] indicated that recruiters without the software knowledge struggled to interpret the signals provided in the aggregated profiles.

We found research that investigated specifically recruiter’s perspective of looking at social coding platforms by considering Github as an example for social coding platform. In the research the attribute number of forks was not considered reliable. Thus, further investigation of research was required to find an attribute that could indicate high-status of a project.

### 2.1.4 Attributes used in literature to indicate popularity of projects on GitHub

Thung et al. [34] explored the influential projects and influential developers in GitHub. Thung et al. measured influential projects and influential developers using page rank. As-
2.1 Social coding platform GitHub

2.1.1 Social coding platform GitHub

Bissyande et al. [4] studied the correlations between programming languages and characteristics of projects such as lines of code, project success. Bissyande et al. used the number of forks and number of watchers of a project for determining the success of the project.

Chen et al. [10] proposed a model to predict the number of forks a project would likely have. The model used attributes of both users and repositories to predict the number of forks that the project could have. Although they did not directly indicate of using project popularity they used attributes such as the number of contributors, number of forks, the number of commits in their model for predicting the number of forks.

Bissyande et al. [3] investigated co-relation between the reported issues of the project with other aspects of a GitHub project. [3] again used the number of forks and the number of users watching a project as an indicator of the project success.

### Usage in Research

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Finding influential projects</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Project success</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Predicting number</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>forks for a repository</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>co-relation between issues</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>reported and project success</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

Table 2.1: Project attributes usage in the literature for indicating project popularity or success

Table 2.1 provides attributes that were considered to indicate popularity or success of a project in literature. Out of them, the number of forks and the number of watchers were used widely, but those attributes were considered unreliable from the recruiter’s perspective.

### Conclusions on investigated GitHub related work

We looked at the research work that focused on investigating social coding platforms usage to infer developer characteristics and skills. We found a few useful signals that can be used for realising our research objective and the following conclusions can be drawn based upon investigated related work:

- GitHub is widely being used by the research community while studying about the developer skills on social coding platforms.
2. Related Work

- There was relatively less research on understanding recruiters' perspective of developer skill profiling on social coding platforms when compared to the research on peer developer profiling.

- Marlow et al. focused on studying recruiters' perspective of using GitHub to know about the developer skills and their research work would be useful in answering our RQ1.

- Recruiters without software background were finding it difficult to use developers' information on social coding platforms and infer their skills. On the other hand, recruiters with software background are facing the problem of not having a tool that can extract developer skills specifically from a recruiter's perspective.

2.2 Recommendation Systems

In this section, we first briefly mention some basic information about recommendation systems and its relevance in the context of our research objective. In section 2.2.2, we describe few recommendation techniques that could be useful to answer the second research question. In section 2.2.3, we discuss few job recommendation systems that facilitated candidate recruitment. Finally in section 2.1.5, we summarise the related work on the recommendation systems and draw few conclusions which are

2.2.1 Recommendation system task

Ricci et al. [28] described recommender systems as software tools and techniques providing suggestions for items to be of use for a user. The basic elements (or) terminology that are used in the recommendation systems are as following:

- Items: Things that are being recommended.
- Users: To whom the recommendations are made.
- Ratings: The ratings are used to express an opinion.
- Transactions: Interactions between a user and the recommendation system could be called as transactions. The recommendation generation algorithm might again use the transactions for making recommendations to the user.

In our research objective, we have mentioned regarding suggesting a list of developer profiles based on the skill set requirements. We defined this aspect of our research objective in terms of a recommendation problem. Thus, items that would be suggested are the GitHub developer profiles, and they are based upon the skill set requirements. Ricci et al. [28] discussed regarding eleven popular tasks that a Recommendation system can assist in implementing. In the described eleven tasks, some were the basic tasks of the recommender system, and some were opportunistic ways of using recommender system. Our recommendation task indicated in the research objective could be defined as the task of "find some,
but good items” i.e., finding some developer profiles that are best suited for mentioned skill set requirements.

### 2.2.2 Recommendation Algorithm Techniques

In this section, we describe few algorithms and techniques that were discussed in the literature for implementing the recommendation tasks.

#### 2.2.2.1 Content-Based recommendation technique

Lops et al. [23] described the basic process of content-based recommendation technique as, matching the attributes of a user profile in which preferences and interests are stored with the attributes containing content of items. Lops et al. describe three steps involved in content-based recommendation approach that are as follows:

1. **Item Representation**: By content analysis techniques the items are represented in a Keyword-based Vector Space Model. Lops et al. [23] discusses commonly used term weighting scheme TF-IDF (Term Frequency-Inverse Document Frequency) to provide weights for the keywords. However, the keywords representation have some problems such as POLYSEMY (the presence of multiple meanings for one word) and SYNONYMY (multiple words with the same meaning). Lops et al. also mentioned about the Semantic analysis by using Encyclopedic Knowledge Sources for representing items.

2. **Learning about User Preferences**: Lops et al. describes few machine learning techniques like Naive Bayes, Nearest Neighbour, Relevance feedback that are used to indicate the items the user might like and the items the user might not like.

3. **Determining items to recommend**: In recommending the items according to the user preferences, different kinds of similarity measurement metrics such as cosine similarity metric are used. The Cosine Similarity function for items $a$ and $b$ can be computed as mentioned in the equation 4.11.

$$\text{Cosine Similarity}_{V(a),V(b)} = \frac{V(a) \cdot V(b)}{|V(a)||V(b)|}$$  \hspace{1cm} (2.1)

Where $V(a) \cdot V(b)$ is the dot product of the weighted vectors, and $|V(a)|$ and $|V(b)|$ are their Euclidean norms.

#### 2.2.2.2 Collaborative Filtering Recommendation

Ekstrand et al. [13] described collaborative filtering recommendation as making predictions and recommendations based on the ratings or behaviour of other users in the recommendation system. The intuition was that similar users prefer similar items and recommendations made based on this intuition was called user-user collaborative filtering recommendation. However, user-user collaborative filtering has some problems, for instance, some items may
not be rated by similar users. If the user preferences were updated, then the similarity between users had to be recomputed. Thus to deal with these issues item-item collaborative filtering was proposed. In item-item based recommendation item similarity was used to compute the user preference towards unrated products. Another technique in collaborative filtering approach is using matrix factorization technique to predict how much a user would like an unrated item. The intuition behind using matrix factorization is, there would be some features of an item across which the user that rates an item. For instance, two users would give high ratings to a particular movie if they liked the actors/actresses in the movie, or if the movie belongs to a genre preferred by both users.

2.2.2.3 Demographic Recommendation Technique

Demographic recommendation technique recommends items based on the demographic profile of the user. The main intuition was that different kinds of recommendations were generated for different demographic details of the user. For example suggesting items based on user age or gender of the user. These kinds of suggestions are made based on the marketing strategies.

2.2.2.4 Context-based Recommendation Technique

Just like demographic recommendation technique considers demographic details in recommendation technique the context-based recommendation technique considers the contextual information of the user for making recommendations. For example, to recommend a movie to a person who wants to see it in a movie theatre on a Saturday night, the recommendation system would use only the available ratings of the movies seen in the movie theatres on the weekends.

2.2.2.5 Hybrid Recommendation Technique

Hybrid recommendation technique combines two or more recommendation techniques to gain better performance with fewer of the drawbacks of any individual one [8]. Content-based and collaborative based recommendations are combined in hybrid recommendation technique. A few ways in which the recommendation techniques are combined are provided in the table [2.2] [8] and a brief description of those ways are briefly described as follows:

- Weighted: In this method, the score of a recommended item is computed from the results of all of the considered recommendation techniques present.

- Switched: In this method, the system uses some criterion based on the items or the user preferences and switch between recommendation techniques.

- Mixed: In this method, the recommendations from more than one recommendation technique are presented together when a large number of recommendations are made simultaneously.
2.2. Recommendation Systems

- Feature combination: In this method, one recommendation technique is treated as additional feature data associated with each item and another recommendation technique is augmented by the data set for recommendations.

- Cascade: In this method, one recommendation technique is employed first to produce a coarse ranking of candidates and a second technique refines the recommendation from among the candidate set.

- Feature Augmentation: In this method, one technique is employed to produce some relevant feature information about items which is then incorporated into the processing of the next recommendation technique.

- Meta-level: In this method, a model generated by one recommendation technique is provided as an input to another recommendation technique.

<table>
<thead>
<tr>
<th>Hybridization method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weighted</td>
<td>The scores (or votes) of several recommendation techniques are combined together to produce a single recommendation</td>
</tr>
<tr>
<td>Switching</td>
<td>The system switches between recommendation techniques depending on the current situation.</td>
</tr>
<tr>
<td>Mixed</td>
<td>Recommendations from several different recommenders are presented at the same time</td>
</tr>
<tr>
<td>Feature combination</td>
<td>Features from different recommendation data sources are thrown together into a single recommendation algorithm.</td>
</tr>
<tr>
<td>Cascade</td>
<td>One recommender refines the recommendations given by another</td>
</tr>
<tr>
<td>Feature Agumentation</td>
<td>Output from one technique is used as an input feature to another</td>
</tr>
<tr>
<td>Meta-level</td>
<td>The model learned by one recommender is used as input to another</td>
</tr>
</tbody>
</table>

Table 2.2: Ways in which recommendation techniques are combined

2.2.3 Recommendation Systems For Facilitating Recruitment

In literature we investigated few job recommendation systems such as Casper described in Bradley et al. [7], Bilateral people-job recommender described in Malinowski et al. [24], Proactive described in Lee et al. [21] and, Absolventen.at indicated in Siting et al. [32]. Except in Bilateral people-job recommender system job advertisements were recommended to job seekers based on the user preferences and activity. On the other hand Bilateral people-job recommender system recommended jobs to job seekers based on the preferences of job advertisement along with job advertisement preferences. As a step towards social recruiting Diaby et al. [12] recommended job advertisements to Facebook and LinkedIn users by using the above mentioned social networks profile data. Diaby et al. [12] used content-based recommendation technique and showed that using basic similarity measures together with the user interacted data could improve the suggestions. The table 2.3 summarises information regarding the job advertisement recommendation systems we investigated.
2. Related Work

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Candidates information inferred using</td>
<td>User activity recorded on recommendation system.</td>
<td>Candidates Resume / CV and user activity on recommendation system.</td>
<td>Details provided by candidates.</td>
<td>Candidate Resume / CV and user activity recorded on recommendation system.</td>
</tr>
<tr>
<td>Recommendation Technique</td>
<td>Collaborative filtering</td>
<td>Hybrid</td>
<td>Hybrid</td>
<td>Hybrid</td>
</tr>
</tbody>
</table>

Table 2.3: Few job recommendation systems in literature

2.2.4 Conclusions on investigated recommendation systems related work

For answering the recommendation aspect of RQ2, we looked at recommendation techniques and a few recommendation systems that facilitate recruitment. The following conclusions can be drawn based upon the investigated related work:

- All the job recommendation systems we investigated considered job advertisements as recruiters requirements.

- Most of the job recommendation systems we investigated considered using user activity and CV for recommending job advertisements, but relatively less research investigated using social coding platforms. Further, the job recommendation systems we investigated had their recommendation task as recommending job advertisements to candidates, but many of them did not consider recommending candidates profiles to job advertisements.

- Moreover to best of our knowledge there were no recommendation strategies that focused on recommending GitHub developer profiles based on skill set requirements of recruiters.

- Much recommendation techniques were available which we can be leveraged to answer recommendation task aspect of our RQ2.
Chapter 3

Approach

In this chapter, we describe the software developer skill profiles that aim to answer our RQ1. Finally, we describe the recommendation task and the approach we took to accomplish the recommendation task that aim to answer our RQ2.

3.1 Software developer profile

In this section, we first describe the approach we took for considering the signals, and then we describe the attributes used and how those attributes were used to indicate the skills exhibited by the developer on GitHub.

3.1.1 Approach

As indicated previously in Related Work Chapter, we will use Marlow et al. [25] work to answer our RQ1. Marlow et al. classified the signals used to infer developer skills across two categories. The first category was based on the recruiter’s difficulty in verifying the signals and the second category was based on the reliability aspect of the signals. Thus, our approach to answer the RQ1 was to convert some of the hard to verify and reliable signals into a set of attributes that could indicate the skills exhibited by developers on GitHub.

We choose to indicate few hard to verify and reliable signals because GitHub already provides few signals that recruiters can use for knowing regarding a developer. For instance, Marlow et al. mentioned that recruiters wanted to know how actively a developer was contributing on GitHub. The recruiters can infer this information by using the developers contributions graph that is publicly available. A sample of GitHub developer profile indicating the user activity is shown in the figure 3.1. From the figure 3.1 it can be inferred that the developer was quite frequently making contributions over past one year. Moreover, the amount of contributions that were made the developer on a particular day is indicated using five different shades of green colour. The darker the green colour is, the developer made the more contributions on that particular day. On the other hand, there were few hard to verify signals such as the quality of contributions made by the developer that requires automatic extraction. Thus, we have selected three reliable signals used to indicate three skills exhibited by a developer.
3. APPROACH

Figure 3.1: Graph on GitHub developer profile indicating the amount of public activity the developer

In the following sections, we describe the attributes we used to indicate the skills exhibited by a developer on GitHub. The attributes we have mentioned are based on our understanding of the signals described by [25]. Furthermore, in some instances Marlow et al. have directly described the attributes that would indicate the exhibited skill, which we have used in our proposed developer skill profiles. For instance, recruiters inferred open source values of a developer by considering, if the developer is contributing back to the forked projects or not which was used in our collaboration skill profile.

3.1.2 Collaboration skills exhibited by GitHub user

Collaboration skills indicate the developer ability to contribute by working with two or more people to achieve common goals and moreover GitHub main aim is to facilitate collaboration in software development. We inferred the collaboration skills of a developer based on GitHub user contributions to different projects and a few other project properties. GitHub defined contribution\(^1\) for a repository as:

“Whenever you commit to a project’s default branch or the GH-pages branch, open an issue, or propose a Pull Request, we’ll count that as a contribution.”

3.1.2.1 Attributes to indicate the level of collaboration skills

The table[3.1] provides the attributes and the range of values that are considered to indicate the collaboration skills exhibited by a developer on GitHub.

Each attribute mentioned in Table 3.1 are described below along with why the mentioned attributes were selected.

1. **Number of projects forked by the user**: A forked project is a personal copy of another user’s project, which allows a developer to experiment freely and contribute

\(^1\)https://github.com/blog/1360-introducing-contributions
3.1. Software developer profile

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>Number of Projects forked by the user</em></td>
<td>Zero or Positive Integer</td>
</tr>
<tr>
<td><em>Contribution Coverage</em></td>
<td>[0%-100%]</td>
</tr>
<tr>
<td><em>Average Share of Contributions</em></td>
<td>[0%-100%]</td>
</tr>
<tr>
<td><em>Average Number of contributors to the Forked Projects</em></td>
<td>Zero or Positive Real Number</td>
</tr>
</tbody>
</table>

Table 3.1: Attributes to indicate collaboration skills exhibited by a developer on GitHub.

back to the original project. We considered only forked projects by the user because, for a developer it is easier to have contributions for one’s owned project as the developer would already be aware of the project details. However, in case of forked projects the developer should display concrete collaboration skills to make the contributions count.

2. **Contribution Coverage**: This attribute would indicate the percentage of projects contributed back by the developer, out of all the projects forked by the developer. Marlow et al. [25] mentioned that recruiters wanted the developer to contribute back to projects rather than just forking the projects.

3. **Percentage of Contributions**: The percentage of contributions are used to signify the level of collaboration skills as having more percentage of contributions to by the developer to the forked project involves collaboration with the project community. Thus, the attribute indicating the percentage of contributions made by the developer out of all contributions towards the projects that satisfy the below mentioned two constraints were used:

   • Project was forked by the developer.
   • Developer has some amount of contributions for the project forked by the developer.

4. **Average Number of contributors to the Projects Forked by the user**: There would be many other contributors contributing to the projects. This attribute indicates on average the number of other users who have contributed to the projects that were forked and contributed by the developer. This attribute indicates the significance of the percentage of contributions made by the developer.

### 3.1.3 Project Management Ability

Project Management skills indicate developer ability to manage contributors and code contributions so that the project’s progress is in the direction, where the project’s collaborators wanted it to be. Marlow et al. mentioned that the recruiters were interested to know about soft skills such as project management ability of the developer using developer GitHub data. Since project management skills is a very broad subject, we indicated the project
management skills through the basic elements of project management in the software development context like managing code contributions and managing GitHub users who contributed code. The attributes used for indicating project management skills were based on the pull requests handling for a project. In table 3.2 below we mention the attributes along with the range of values that were used to indicate the project management skills of the developer. The attributes that were used to indicate the project management skills exhibited by the GitHub user are described in detail along with why the mentioned attributes were selected.

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Scope to Manage Projects</td>
<td>Zero or Positive Integer</td>
</tr>
<tr>
<td>Projects Managed Coverage</td>
<td>[0%-100%]</td>
</tr>
<tr>
<td>Percentage of Managed Code Contributions</td>
<td>[0%-100%]</td>
</tr>
<tr>
<td>Average Number of Contribution Managers</td>
<td>Zero or Positive Real Number</td>
</tr>
<tr>
<td>Users Managed</td>
<td>Zero or Positive Integer</td>
</tr>
</tbody>
</table>

Table 3.2: Attributes to indicate project management skills

3.1.3.1 Attributes to indicate project management ability

- **Scope to Manage Projects**: For every project on GitHub there is an owner and a few collaborators who could not only contribute to project, but also have the right to manage different contributions received from other GitHub users. This attribute indicates the number of the projects for which the developer had an opportunity to manage contributions. The developer was considered to had an opportunity to manage projects if below mentioned two constraints are satisfied:
  - Developer must be a collaborator or an owner for the project.
  - The project that is being considered should have received some contributions from the GitHub users after the developer had rights to manage contributions.

- **Projects Handling Coverage**: The attribute indicates the percentage of projects the developer had actually managed contributions, from all of the projects the developer had an opportunity to manage contributions. A pull request was considered to be handled by the user if the user closed a pull request. We did not consider the pull requests in the scenarios where the developer was not assigned a pull request or had participated in a closed pull request through, for example commenting on a pull request. The reason to ignore these cases was, we cannot directly infer what kind of role the developer had while participating in the pull request. However, in a case where the developer was not assigned a pull request, but a pull request was closed by the developer then, we can safely assume that the developer had played some role in handling the pull request because, a user cannot simply close the pull request without knowing properly about what the pull request was about or without proper understanding of
3.1. Software developer profile

the state of the pull request. This attribute would indicate how keen the developer was on handling code contributions.

- **Percentage of Managed Code Contributions**: This attribute indicates, the percentage of contributions managed by the developer, out of all the managed contributions in the projects that were managed by the developer. This attribute was selected, as this attribute would indicate the code management ability of the developer.

- **Average Number of Contribution Managers**: Indicates an average number of users involved in managing contributions to the projects that were managed by the developer. This attribute was aimed to understand how many other GitHub users the developer had to compete to handle code contributions.

- **Users Managed** Indicates the number of users managed by the developer that have contributed to the projects that were managed by the developer. This attribute aimed to indicate the quantity of users the developer had managed.

### 3.1.4 Programming language skill

The programming language expertise indicates how a GitHub user-contributed code in coherence with the project community requirements in the specified language. To indicate the level of expertise of a developer with a programming language, we used the attributes mentioned in Table 3.3 to indicate the quantity and efforts of developer in code contributions. Then to understand the quality of those code contributions we categorised the quantity of contributions into four categories namely **accepted, rejected, under review and not reviewed**. First we describe how the categorization was made and how each category would indicate the quality of code contributions. Then attributes indicating the quantity and efforts of contributions would be described.

#### 3.1.4.1 Categories to indicate the quality of contributions

In GitHub, Code contributions are made through commits. Thus, the categorization made to indicate the quality of contributions was done depending on the following two constraints:

- If the commits were not made through a pull request then the commit would fall in not reviewed category.

- If a commit was made to the repository through a pull request then the status of the pull request under which the commit fall would define which of the other three categories under which the commit would fall.

The categories indicating the quality of contributions are described below:

- **Accepted**: Accepted commits are the commits that are part of accepted pull requests. This category of commits indicates the highest quality of commits.
3. APPROACH

- Rejected: Rejected commits are the commits that are part of rejected pull requests. This category of commits indicates the lowest quality of commits.

- Under Review: Under Review commits are the commits that are part of pull requests whose status was neither accepted or rejected. The under review commits are considered less quality than accepted category of commits as the commits were yet to be scrutinised by the project community.

- Not Reviewed: Not reviewed commits are the commits that were not part of the pull requests. The not reviewed commits are considered more quality than rejected, but less quality than under review commits as they were not put to be scrutinized by the project community.

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Share of Commits</td>
<td>[0%-100%]</td>
</tr>
<tr>
<td>Number of Files</td>
<td>Zero or Positive Integer</td>
</tr>
<tr>
<td>Number of lines of changes</td>
<td>Zero or Positive Integer</td>
</tr>
<tr>
<td>Number of Projects</td>
<td>Zero or Positive Integer</td>
</tr>
<tr>
<td>Average Number Of Stars For The Projects</td>
<td>Zero or Positive Real Number</td>
</tr>
</tbody>
</table>

Table 3.3: Attributes to indicate the quantity of code contributions

3.1.4.2 Attributes to indicate the quantity of contributions

The attributes mentioned in table 3.3 were used to indicate the quantity of code contributions made for the language under consideration:

- Share of commits: This attribute indicates the percentage of commits made by the developer, out of all the commits related to a specified category of commits across all the developer projects.

- Number of files: This attribute indicates the number of unique files coded by a developer for the language under consideration.

- Number of lines of changes: This attribute indicate the number of lines of changes made by a developer in the mentioned number of files.

- Number of projects: This attribute indicate the number of projects across which “Number of Files” and “Number of lines of changes” were computed.

- Average number of stars: The stars for a project indicate, attentiveness a project have among GitHub Users. An average number of stars received for the projects would indicate the significance of the above-described attributes indicating quantity.
We describe how the attributes across each skill were used to compute a score for comparing the skills between the developer in the implementation chapter of the report. The computed scores were used in ranking the recommendation of the developer profiles to a job advertisement.

3.2 Recommending developer profiles to a job advertisement

In this section we first define the recommendation task and mention the approach we took for accomplishing the defined recommendation task. We then mention some limitations in recommending GitHub developer profiles to job advertisements in the approach we selected and how the limitations were handled.

3.2.1 Approach

In coherence with our research objective we defined our recommendation task as, providing best suited developer profiles for a job advertisement with recruiters having to invest minimum efforts and time to find appropriate profiles. For accomplishing the recommendation task, firstly we inferred the recruiters requirements through software developer related job advertisements. We decided to infer recruiter’s requirements from job advertisements based on the conclusions we drew from the investigated research work and further we decided to consider only software related job advertisements because our research objective is to facilitate software developer recruitment. Secondly, we took the content-based recommendation approach for making the recommendations, because we didn’t had any data indicating user preferences to use for taking a collaborative filtering approach. The main idea of content-based recommendation approach was to represent the content in the form of keywords which would indicate what was mentioned in the content. Thus we represented the content i.e job advertisement and the software developer profile described in section 3.1 using the keywords along with a weight for each keyword. Keywords were used in understanding the requirements of job position in the context of job advertisements and the keywords in software developer profile were assumed to indicate skills known to the developer. Furthermore, the weights for a keyword were used to infer the significance of keywords in the job advertisement and in context of the developer profile we inferred the significance of the keywords based on the developer profile we extracted.

In the section below, we indicate the entity type differences in keywords sets and how we dealt with the those differences in the keyword sets of the job advertisements and software developer profiles.

3.2.2 Entity types differences between the keyword sets of a job advertisement and developer profile

The keywords set used to indicate about the software developer profile described in section 3.1 had entity types differences with the set of keywords from job advertisements. For in-
3. APPROACH

Job Advertisement

Web Developer Job in Beverly, MA, USA.

KICVentures is making an impact are you interested in being a part of the next big thing? What is your opportunity? KICVentures and SpineFrontier are positioned to radically change spine surgery and healthcare delivery as we see it today. SpineFrontier has experienced steady growth since inception and is now ready to accelerate its trajectory toward a potential exit in 3-5 years. At KICVentures we are looking for the best people committed to making this a true success story and share in the upside potential. We are looking for someone to join our team and help us build incredible things. You will work closely with all members of our creative team in a collaborative environment.

We are seeking a Web Developer that loves to build incredible user experiences.

Duties include but aren’t limited to:

- Write and debug HTML5, CSS3, and Javascript.
- Experience with PHP, MySQL.
- Strong skills in HTML5, CSS3.
- Cross-browser and cross-device compatibility testing.
- Knowledge of version control tools.
- Must be familiar with responsive design.
- Experience with Javascript libraries and frameworks (Node.js, Backbone.js, etc.)
- Bachelor’s degree in a relevant field.

What would we need from you?

- 3-5+ years of experience in web development.
- Bachelor’s degree in a relevant field.
- Strong skills in HTML5, CSS3.
- Excellent experience with Object-Oriented Javascript and good knowledge of jQuery.
- Must be familiar with responsive design.
- Cross-browser and cross-device compatibility testing.
- Working knowledge of Wordpress CMS.
- Web Standards.
- Experience with PHP, MySQL.
- Knowledge of version control tools.
- Experience with Javascript libraries and frameworks (Node.js, Backbone.js, etc.)

Key Words other than programming language names in the profile


Sample Developer Profile

<table>
<thead>
<tr>
<th>Skill</th>
<th>Forked Repositories</th>
<th>Contribution Coverage</th>
<th>Contribution Mean</th>
<th>Average number of Contributors</th>
</tr>
</thead>
<tbody>
<tr>
<td>SHELL</td>
<td>4</td>
<td>25%</td>
<td>1.22%</td>
<td>4</td>
</tr>
<tr>
<td>Java</td>
<td>5% of Not Reviewed</td>
<td>4,893 files with 68 changes</td>
<td>1</td>
<td>0.0</td>
</tr>
<tr>
<td>Python</td>
<td>10.00% of Not Reviewed</td>
<td>118 files with 1,160 changes</td>
<td>3</td>
<td>8.0</td>
</tr>
</tbody>
</table>

Figure 3.2: Samples of software related job advertisement and a developer profile

Duties include but aren’t limited to:

- Write and debug HTML5, CSS3, and Javascript.
- Experience with PHP, MySQL.
- Strong skills in HTML5, CSS3.
- Cross-browser and cross-device compatibility testing.
- Knowledge of version control tools.
- Must be familiar with responsive design.
- Experience with Javascript libraries and frameworks (Node.js, Backbone.js, etc.)

What would we need from you?

- 3-5+ years of experience in web development.
- Bachelor’s degree in a relevant field.
- Strong skills in HTML5, CSS3.
- Excellent experience with Object-Oriented Javascript and good knowledge of jQuery.
- Must be familiar with responsive design.
- Cross-browser and cross-device compatibility testing.
- Working knowledge of Wordpress CMS.
- Web Standards.
- Experience with PHP, MySQL.
- Knowledge of version control tools.
- Experience with Javascript libraries and frameworks (Node.js, Backbone.js, etc.)

Key Words other than programming language names in the profile


Figure 3.2: Samples of software related job advertisement and a developer profile
Thus we used only the keywords of entity types that are common in software developer profile and the job advertisement. We used keywords belonging to entity types of *programming languages* or *software* in our recommendation task. Our intuition behind using the mentioned entity types keywords was, all the "software developer" related job advertisements we used in our recommendations would have some keywords of entity types programming language or software.

### 3.3 Evaluation Approach

In the this section we first describe our approach for evaluating the generated developer profile and the recommendation strategy we used for recommending developer profiles to suit job advertisement.

#### 3.3.1 Evaluating developer profile

The main aim of generating developer profile was to reduce the effort the recruiters had to go through when using GitHub for their recruitment purposes. Thus, our evaluation approach also aimed to understated if the information provided in the developer profile has the potential to be used in the recruitment process. Thus, for this purpose we decided to ask the recruiters for their opinion on the generated developer profile along the aspects mentioned as follows:

- Ability of the attributes used to indicate the mentioned skill.
- Recruitment steps in which the developer profile can be useful.
- Ease of understanding of the developer profile.
- Recruiters preference to use the developer profile.

Using the aforementioned aspects we inferred, the extent to which our developer profile reduced the recruiters efforts. The reasons for using these four aspects for evaluation developer profile and how the evaluation was done on these four aspects are mentioned in the Evaluation chapter of this thesis.

#### 3.3.2 Evaluating a recommendation task

The recommendation task evaluation requires selecting metrics according to what the recommendation task is aimed to accomplish. Thus to identify metrics we describe our task of recommending developer profile to job advertisement using a recommendation system framework. The general framework in Hernandez et al. [17] described recommendation system as, consisting of two sub-systems, *interactive* and *non-interactive*. The *interactive* sub-system would be responsible for guiding the user, and its aim would be to answer when and how recommendations should be shown to the user. *Non-interactive* sub-system would
be responsible for choosing useful items to recommend and would aim to answer which items should be selected for the recommendation from the available items.

Thus based on the described general framework of a recommendation system we can describe our sub-system as non-interactive. Thus as the sub-system was non-interactive, we would be focusing on evaluating if the github profiles recommended for the job advertisements are relevant to a job advertisement or not.

Shani et al. [30] discussed the metrics that could be used for evaluating a recommendation system. [30] also discussed metrics that are used for the evaluation of non-interactive sub-system under the category of measuring the prediction accuracy of a recommendation system. Shani et al. discussed three kinds of experiments that could be performed to measure the metrics used for prediction accuracy. The first way was to conduct an offline experiment where we do not require direct interaction with users, but we already need to have the ground truth to compute the metrics in an offline experiment. The second way of performing the experiment was to conduct a user study where a small group of users were asked about their experience with the system. Finally, the third way was to perform an online experiment where the actual users experience with the recommendation system is recorded without the users being aware of the experiment.

The online experiment would depict the real scenario and was best suited for a full recommendation system. We did not perform an online experiment because our recommendation task does not represent a full-fledged recommendation system due to few missing elements such as recording the user preferences and providing a good user interface. Also for performing an offline experiment, we do not have ground truth as we do not know in advance which of the recommended developer profiles would be best suited for a given job experiment. Thus, we performed the experiment by user study and measured the metrics used for evaluating the recommendations made.

We provide the metrics selected and the how those metrics were computed for evaluating our recommendation task in the evaluation chapter of this thesis.

In the next chapter, we describe how the sample software developer profile was generated along with how we implemented our recommendation task to achieve our research objective.
Chapter 4

Implementation

In this chapter, we first provide an overview of the workflow involved in the generation of software developer profile and the profile recommendations to a job advertisement. We then mention about the GitHub user dump and how the software developer profile was created using the GitHub user dump. We then describe in detail about the crawled job advertisement data and finally we end the chapter by describing how the GitHub user profiles and crawled job advertisements were represented to accomplish the recommendation task.

4.1 Workflow overview for developer profile and recommendations

The figure [1] provides an overview of the workflow involved for generating developer profile and recommendation of software developer profiles to a job advertisement using the mentioned tools.

The GHtorrent has GitHub users dump that contains the necessary GitHub user’s information along with a few GitHub API links to get the additional information required for generating the developer profile described in chapter [3].

The GitHub user’s data dump downloaded from the GHtorrent was then stored in a MongoDB database as the GitHub user’s dump data from the GHtorrent was in the JSON format. The Software Developer profile generator uses the details from the MongoDB database and makes some requests to the GitHub API for additional information such as, commits made to the repository and, pull requests to the repository, that are required to generate the proposed profile. The GitHub’s Linguist tool was used to identify the programming language associated with each commit. Python NLKT[22] was used to filter stop words from the commits file name details and then DBPedia Spotlight[27] annotation service was used to provide the entity types for the filtered out words. According to our recommendation approach, we only consider the annotated words that are of type software (or) programming language. The generated developer profile was then represented to be indexed in Apache Solr for the recommendation task.
For obtaining software developer related job advertisements, we crawled monster.com using Scrapy. The crawled job advertisements data was processed using DBPedia Spotlight annotation service to extract keywords related to programming languages and software types. Finally, the job advertisements were represented to be indexed in Apache Solr for the recommendation task.

**DBpedia Spotlight** DBpedia Spotlight is a tool that provides solutions for linking unstructured information sources to the Linked Open Data cloud. DBpedia Spotlight can also be used for other tasks such as Named Entity Recognition, Keyphrase Extraction, Tagging in a text. DBpedia Spotlight offers annotating, disambiguating entities in the provided text as a service over the web API.

The service is available to use for free and DBpedia Spotlight has top F1 score while evaluating for annotation task. F1 score is the harmonic mean of precision and recall values, where F1 score reaches its best value at 1 and worst score at 0. In a binary classification task precision is the ratio of classified true-positive cases to total number of cases classified and recall is the ratio of classified true-positive ratio to the total number of true-positive cases. The other alternative that was available was The Wiki Machine, which had relatively less documentation, and it does not have web API access like DBpedia Spotlight. Thus considering the documentation and also the top F1 score we choose to use DBpedia Spotlight. DBpedia uses three major phases for providing the annotated words for the given text. The first phase is called as spotting phase in which the phrases that may indicate a mention of a DBpedia resource (DBpedia Ontology containing over 4 million instances) are spotted. The next phase is called as candidate selection wherein the spotted phrase is
mapped to the resources. These resources are the candidates for indicating the types of the
spotted phrases. Finally in the disambiguation phase, the candidates that could most likely
indicate the type of the spotted phrase are selected from the candidate types, based on the
context, like the paragraph of the phrase.

In the following sections, we describe in detail regarding generating the software devel-
oper profile and then regarding, how profiles were recommended to a job advertisement.

4.2 Software Developer Profile Generator

In this section, we first mention about some basic types of data and the kind of GitHub
API links that were present in the JSON file of the GitHub user's data dump. We also
provide few statistical values computed across extracted developer profiles set. We then
describe how the GitHub users data dump that was stored in MongoDB from GHTorrent
was used to generate the developer profile indicating three skills, namely collaboration,
project management and, programming language skills.

4.2.1 GHTorrent GitHub user’s data

GHTorrent [14] project mirrors the GitHub events and stores those events. The GitHub user’s
data was openly available for the research community to use through GHTorrent website.
We downloaded the users data dump during 2014-07-29 from the GHTorrent project web-
site. The data dump consisted of 326716 GitHub users data of them 3,04,114 GitHub users
were of type "User". Since GitHub also allows to create GitHub user profiles of type "or-
ganisation" and our aim was only to indicate skills exhibited by software developers, we
only considered the user profiles of type "User" for generating the developer profile. Each
JSON file of the GitHub user data dump represented a GitHub user and contained some
essential information like 'GitHub username', 'GitHub user type', 'number of followers'
for the user, etc. The JSON file of a GitHub user also contained some GitHub API links
that could be used to get additional information through GitHub API. The additional infor-
mation that could be extracted was repositories details, user activity. Access to the private
repositories of GitHub users was not possible as users had to provide access to the appli-
cations that wanted to access the private data. Thus, we decided to use public repositories,
whose information was publicly available through GitHub API.

We selected a sample of GitHub user profiles of type "User" from 304114 GitHub users
data dump and generated 5909 GitHub developer profiles and indexed them in Apache
Solr for implementing our recommendation task. We choose to generate developer profiles
only for a sample of users in data dump because we also used GitHub API for requesting
additional information from the GitHub API. The GitHub API has a restriction of only 5000
requests per hour with basic authentication requests. We choose to select a GitHub user
from GitHub user’s data dump randomly because this will provide a chance for every user
to be selected for generating developer profile. Thus making our results from the evaluation
honest. However, one disadvantage of random sampling in the context of our evaluation is
that it induced some noise in data. The noise in our experiment data can be observed in
4. IMPLEMENTATION

Table 4.1, where out of 5909 profiles only 4393 user profiles had at least one commit in a particular programming language.

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total number of GitHub profiles</td>
<td>5909</td>
</tr>
<tr>
<td>Total number of profiles with a commit in at least one programming language</td>
<td>4393</td>
</tr>
<tr>
<td>Average number of distinct programming language (or) software entity keywords per profile</td>
<td>4.51</td>
</tr>
<tr>
<td>Average number of programming languages per profile</td>
<td>1.57</td>
</tr>
<tr>
<td>Average number of repositories per profile</td>
<td>4.20</td>
</tr>
<tr>
<td>Total number of unique programming language or software keywords in data set</td>
<td>1786</td>
</tr>
</tbody>
</table>

Table 4.1: Basic properties of the developer profile set used for recommendation

The statistical properties of the generated developer profile data set, and their programming language distribution are provided in the tables 4.1 and 4.2 respectively.

<table>
<thead>
<tr>
<th>Programming language</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSS</td>
<td>20.6%</td>
</tr>
<tr>
<td>Javascript</td>
<td>20.6%</td>
</tr>
<tr>
<td>Java</td>
<td>12.0%</td>
</tr>
<tr>
<td>Python</td>
<td>8.8%</td>
</tr>
<tr>
<td>PHP</td>
<td>8.0%</td>
</tr>
<tr>
<td>Shell</td>
<td>7.8%</td>
</tr>
<tr>
<td>Ruby</td>
<td>7.3%</td>
</tr>
<tr>
<td>C</td>
<td>5.6%</td>
</tr>
<tr>
<td>C++</td>
<td>5.6%</td>
</tr>
<tr>
<td>R</td>
<td>5.0%</td>
</tr>
</tbody>
</table>

Table 4.2: Top 10 programming language distribution in the generated profile data set

The table indicates the top ten programming languages being used by the developers in our data set. We could not find research about the sample data indicating the popularity of programming languages in terms of the number of developers writing code in a programming language to compare our random sample distribution. Therefore, we decided to compare our list with most active programming languages among the repositories because its the developers who are pushing code to repositories. Furthermore, from the popularity of programming languages among the repositories we can at least roughly infer the popularity of programming languages among the developers. We compared the top ten programming language list in table with the top ten active programming languages among repositories using GitHut. GitHut visualised the active programming languages among repositories from *Quarter 2 of 2012 to Quarter 4 of 2014*. We checked if each of the reported top 10 programming languages among the developers were among the top 10 programming languages on GitHut in every quarter of period indicated by GitHut. We found that 8 out of the

\[\text{http://githut.info/}\]
10 reported programming languages were among top 10 languages in all the quarters. CSS and R were the two programming languages that were not consistently in the top 10 active programming language list of GitHut. CSS was among top 10 programming languages consistently only during *Quarter3 of 2013* to *Quarter4 of 2014*, while R was never in top 10 of the languages list. Thus based on this comparison outcome we can infer that our decision of randomly selected users could roughly indicate the kind of diversity that can be seen in the popularity among the repositories.

In the following sections, we describe how the developer profile indicating each skill was generated.

### 4.2.2 Generating collaboration skill

In this section, we explain how the attributes values used for indicating the collaboration skills were generated:

1. The first attribute value we determined for indicating the collaboration skill was the number of projects that were forked by the developer. For determining the value, we used repositories API link available in a GitHub user JSON file retrieved from MongoDB. The JSON response for the repositories API link from GitHub API contained essential information regarding repositories like, if the repository was a forked by the user or not. Thus, using repositories API link response we counted the number of repositories using the “Forked” field value.

2. The second attribute value we determined was the contribution coverage value. For determining the value, we required the number of the repositories for which the user had contributed back. We used the JSON response for the repositories API link obtained for an earlier attribute. The JSON response for repositories details, API link also contained “contributors” field for each repository having an API link as the value. Thus by using the “contributors” field API link value we obtained the contributors details for each repository. If the username of the developer we are analysing was present in the “contributors” field of the API link response details, then we considered that the developer had contributed to that repository. We used the similar process for all the repositories that were forked by the user and determined the value for the number of projects the developer had contributed back. Further, we computed the ratio value between the number of projects the user had contributed back and the number of projects the user have forked and obtained the value for contribution coverage.

3. We then obtained the percentage of contributions that were made by the user for the repositories in which the user had made contributions by using the equation 4.1:

\[
\text{Percentage of Contributions} = \frac{\text{Developer Contributions}}{\text{All Contributions}} \quad \text{where}, \quad (4.1)
\]

\[\text{Developer Contributions} = \text{Number of Contributions Made By The Developer} \]
\[\text{All Contributions} = \text{Number of Contributions Received From All The Contributors} \]
4. IMPLEMENTATION

The number of contributions made by a contributor for a repository was also available in the JSON response of "contributors" API link response details, that was obtained for determining contribution coverage. Thus using the details from the contributors details we computed the two values required in the formula and determined the percentage of contributions made by the user.

4. We also obtained the average number of contributors by counting the number of contributors for each repository and then averaged the value that resulted in “average number of contributors”. The number of contributors value for each repository was determined by again using the already obtained JSON response that contained contributor details.

4.2.3 Generating project management skill

In this section, we explain how the attributes values used for indicating the project management skills were generated:

1. We first determined the value for the number of projects for which the developer had scope to manage code contributions. We started by obtaining the projects for which the developer was the owner, or the user had the collaborator status. Then, out of them we only considered those projects that received code contributions from other developers. As a first step, we counted the repositories that were not Forked by the developer since they are the repositories for which the developer was the owner. Thus, the repositories that were not forked by the developer was identified using the "Forked" field value, which must be "False". We then determined the collaborator status of the developer across the remaining of the considered repositories by using the "collaborator" field which contained collaborators API link as the value in the repositories JSON response. A process similar used to obtain the value for the "projects contributed back" was followed to obtain the number of projects for which the developer was the collaborator. Therefore, using the two values the number of projects for which the developer had "scope to manage code contributions" was determined. Due to recent changes in the GitHub API to the collaborators API link determining the projects for which the user was collaborator through the strategy we mentioned above would no longer be possible. One have to trace the user events and search for an event where the developer was made collaborator. However, even the GitHub API only provides with the events that occurred in the past one year. Thus, if a user had become a collaborator one year before then the GitHub API would not be useful to determine the projects for which the user was a collaborator. The only alternative that could be used would be using the GHTorrent project. As GHTorrent was mirroring the GitHub events, the number of projects for which the user had the scope to manage projects could be determined precisely.

2. We then determined the number of projects for which the developer had actually managed code contributions by using the "Pull Requests" field available in the JSON response of developer repositories details. The JSON response for the pull request
API link had a field "closed by" for each pull request in the repository. This field consisted of the username that had closed the pull request. Thus using the "closed by" field we determined the number of projects for which the developer had actually managed contributions.

3. We then determined the percentage of code contributions managed by the developer using the equation [4.2]:

\[
\text{Percentage of Contributions handled} = \frac{\text{Handled Contributions}}{\text{All Handled Contributions}} \quad \text{where, (4.2)}
\]

\(\text{Handled Contributions}\) = The Number of pull requests closed by the developer.

\(\text{All Handled Contributions}\) = Number of closed pull requests that were not assigned to other code managers.

Again the above two mentioned values were obtained using the pull requests API link response from GitHub API as the response for the API link also had a field "assigned to" which was used to filter out the closed pull requests that were assigned to other users.

4. We then determined the “number of users managed” by a developer by counting the unique Username’s that made a pull request. We obtained the Usernames through the "Pull request by" field in the JSON response of the pull requests API link. Out of all the unique usernames available, we only counted those pull requests that were closed by the developer.

5. Finally, we determined the “average number of code managers” for the considered projects. We obtained value by averaging the count of unique Username’s present in the “closed by” field of all the closed pull request details across considered projects.

### 4.2.4 Generating level of programming language skill profile

In this section, we explained the steps involved to generate the attributes values used to indicate the level of programming language skills. We first describe how the commits categories that indicate the quality of code contributions were made using pull request status. We then describe how the commits were indicated with the programming languages they belonged. Finally, we describe how the set of attributes that for the programming language skill profile were determined.

#### 4.2.4.1 Categorising commits

For categorising the commits, we first obtained all the pull requests for the developer projects. Then all the commits in the pull requests were categorised first into three categories based on the pull request status. If the pull request was merged then, all the commits under those pull requests were categorised as accepted. Similarly, all the rejected pull request commits were categorised as rejected. Finally, if the pull request was open, then those
commit were categorised as under review commits. The commits that were not part of pull requests were categorised as not reviewed commits. The not reviewed commits do not include merge commits as the merge commits are a byproduct of accepting a pull request.

4.2.4.2 Detect programming languages for a commit

The GitHub uses ”github/linguist” to determine the programming languages used in repositories. Even GitHub API also provides the list of programming languages in a repository. However, GitHub API does not indicate the programming languages associated with each commit. Thus we used “github/linguist” tool available on GitHub and identified the programming languages associated with each commit. The ”github/linguist” tool provides the list of files in a repository that fall under a particular programming language. Thus, we matched the file names of a commit with the files being categorised under different programming languages. In this way, we determined programming language for each commit. We categorised the files that do not belong to any programming language as unknown and extracted some software technologies (or) tools related to keywords using Dbpedia annotation service. The extracted language and technology keywords were further used for the recommendation task.

4.2.4.3 Quantity of contributions in a programming languages

We considered the percentage of commits, the number of files and number of lines of changes and also the number of repositories to indicate the quantity of contributions in a programming language. Below we describe the process with which each attribute value was determined.

- Percentage of commits: While categorising the commits according to the pull requests status we further determined if the commits were made by the developer or by other GitHub users. After the above categorisation was finished, we computed the \( \text{Percentage of Commits} \) by the developer using the equation \( \text{(4.3)} \) across each commit category, categorised according to the pull request.

\[
\text{Percentage of Commits} = \frac{\text{Commits Made}}{\text{All Commits}} \quad \text{where,}
\]

\[
\text{Commits Made}=\text{Total number of the commits made by the developer.}
\]

\[
\text{All Commits}=\text{The total number of commits made.}
\]

- Number of files and Number of lines of changes: The commits categorised contained unique SHA value for each commit. Using the SHA we extracted the files in commit and number of changes that were made on each of those files. Therefore, we counted the unique number of files present in each category of the commit and determined ”Number of Files”. Furthermore, we also added the changes made to the files and determined the ”Number of changes made” in each commit category.
4.3 Recommending Profiles

- Average Number of Stars Projects: Each JSON repository details response from GitHub API also provides information regarding the number of stars received by that repository. Thus, we determined the value by just averaging the stars of the repositories.

- Number of repositories: The number of repositories was determined by counting the number of unique repositories from which the values described above were extracted for each commit category.

4.3 Recommending Profiles

In the approach chapter, we mentioned few domains related to which the keywords used for the recommendation would belong to and also why those domains were chosen for the recommendation. In this section, we first mention some of the essential indexed fields that were used for performing content-based recommendation of developer profiles to a job advertisement. We then explain the details of crawling job advertisement along with how the crawled job advertisement, and generated software developer were represented to fit in the fields that would be indexed (or) stored (or) both for recommendations. Finally, we explain the recommendation strategy that was implemented to accomplish the recommendation task.

4.3.1 Schema used for recommendations

We used Apache Solr as a tool to make the content-based recommendations. In this section, we describe a few essential indexed fields that were used to make appropriate recommendations. The fields could represent some detail of either a job advertisement or a software developer profile or both. The table below shows some basic indexed fields involved during the recommendation process. The screen shot of detailed schema is provided in Appendix A of this thesis report.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Type</th>
<th>IsMultivalued</th>
</tr>
</thead>
<tbody>
<tr>
<td>userName [Unique]</td>
<td>String</td>
<td>No</td>
</tr>
<tr>
<td>keywords</td>
<td>String</td>
<td>Yes</td>
</tr>
<tr>
<td>keywordWeights</td>
<td>Int</td>
<td>Yes</td>
</tr>
<tr>
<td>languageName</td>
<td>String</td>
<td>Yes</td>
</tr>
<tr>
<td>technologies</td>
<td>String</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 4.3: Basic Schema Details for Indexing

4.3.2 Job advertisement data indexing

We explain first how we obtained the job advertisement data and then explain how the collected data was processed to understand the requirements mentioned in a job advertisement and represent the job advertisement in the schema mentioned above.
4. IMPLEMENTATION

4.3.2.1 Crawling and processing job advertisement data

For obtaining software developer related job advertisements, we crawled the monster.com website using Scrapy library. The job advertisements were crawled using the term software developer on 18-02-2015, and we tried to obtain all the job advertisements we could through our crawling job. Using our crawling job we extracted 411 software developer job advertisements. The job advertisements extracted contained two parts job title and job description.

We sent the text mentioned under the job title and the job description to DBPedia Spotlight annotation service, which responded with annotated words in the sent text. The DBpedia Spotlight annotation service also indicated the type of the annotated words through @types field in the JSON response. The type of annotated words was indicated for example as, DBPedia: Software, DBpedia: Programming Language in @types field of JSON response from DBpedia Spotlight. Out of all the annotated keywords response from the DBpedia Spotlight annotation service, we only considered the keywords that were indicated as the programming language (or) software type. We selected only those two types because these were the two major types we found that were related to software under the Things category of DBpedia Spotlight. Furthermore, our recommendation task was to recommend developer profiles, solely based on skills mentioned in the job advertisements.

The aforementioned process was followed for each job advertisement we crawled, for extracting programming languages and software keywords that also form the skill set requirements for the job advertisements that are related to a software developer. The table 4.4 and 4.5 indicates the properties of our job advertisement data set and also language distribution of top 10 languages in the job advertisement data set.

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total number of job advertisements in the data set</td>
<td>411</td>
</tr>
<tr>
<td>Average number of words in job description of a job advertisement</td>
<td>287.33</td>
</tr>
<tr>
<td>Total number of unique programming language or software related keywords found in data set</td>
<td>313</td>
</tr>
<tr>
<td>Total number of keywords of type software that are available on DBpedia Spotlight</td>
<td>25,203</td>
</tr>
<tr>
<td>Total number of keywords of type VideoGame: is a subtype of software; that are available on DBpedia Spotlight</td>
<td>16,933</td>
</tr>
<tr>
<td>Total number of keywords of type Programming Languages: is a subtype of software; that are available on DBpedia Spotlight</td>
<td>453</td>
</tr>
<tr>
<td>Job advertisements containing at least one language or software related keywords in job title</td>
<td>335</td>
</tr>
<tr>
<td>Average number of unique programming language or software related keywords in a job advertisement</td>
<td>6.31</td>
</tr>
<tr>
<td>Average number of unique programming language or software related keywords in the job title</td>
<td>2.84</td>
</tr>
</tbody>
</table>

Table 4.4: Properties of the job advertisement data set used in recommendation task
4.3. Recommending Profiles

<table>
<thead>
<tr>
<th>Programming language</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Javascript</td>
<td>38.4%</td>
</tr>
<tr>
<td>Java</td>
<td>32.1%</td>
</tr>
<tr>
<td>PHP</td>
<td>25.0%</td>
</tr>
<tr>
<td>SQL</td>
<td>22.8%</td>
</tr>
<tr>
<td>C#</td>
<td>14.8%</td>
</tr>
<tr>
<td>Python</td>
<td>11.43%</td>
</tr>
<tr>
<td>Ruby</td>
<td>9.0%</td>
</tr>
<tr>
<td>C++</td>
<td>8.0%</td>
</tr>
<tr>
<td>Perl</td>
<td>4.8%</td>
</tr>
<tr>
<td>Objective-c</td>
<td>4.6%</td>
</tr>
</tbody>
</table>

Table 4.5: Top 10 programming languages distribution in the job advertisement data set

4.3.2.2 Job advertisement processing to accomplish recommendations

Having described how the keywords were extracted from the job advertisements we now describe the approach that was taken to indicate the level of significance for the keywords that would be used in the recommendation task. As mentioned above only programming language (or) software related extracted keywords were used and the approach to indicate the significance of keywords for the programming language (or) software keywords only is described as following:

- Job Title: The job title of a job advertisement provide significant and essential requirements for the job such as the skill set requirements involved in the job position, location. Kaur et al. [19] mentioned Position Weight Algorithm according to which keywords in “Title” are of more importance than paragraph keywords. Thus, we decided to give the software (or) programming language related keywords mentioned in the job title higher priority over all other software or programming language related keywords mentioned in a job description.

- Job Description: The job description provides a detailed description of education, location, and skill set requirements. We used the number of occurrences of a programming language (or) software keywords in the job description as weights for keywords. We used the number of occurrences for providing weights to keywords based on the intuition that in a local content the number of occurrences of keywords would indicate the significance of the keywords.

Having decided to consider the number of occurrences to indicate the weight of the keywords, we also had to reflect the higher significance of a keyword presence in a title, in its keyword weight. Thus, for reflecting the higher significance of a keyword in the title we assumed to count each occurrence twice. We did not assume much higher value to indicate significance because by doing so we could diminish the effect of other keywords in the job advertisement and this will lead to recommending the developer profiles that are title centric rather than being job advertisement centric. Finally, we computed the weights for
a programming language or software related keywords of each job advertisement using the equation

\[ \text{Keyword Weight} = NOJD + (NOJT + NOJT) \]

where,

\[ NOJD = \text{Number of Occurrences of a keyword in job description.} \]

\[ NOJT = \text{Number of occurrences of a keyword in job title.} \]

**4.3.3 Representing software developer profile to accomplish recommendations**

In this section, we describe how the software developer profile described in Section 3.1 was represented so that recommendations could be made to job advertisements. To represent the developer profile skills we used the technologies and the language keywords extracted while generating the programming language skill profile. The set of attributes across three skills were reduced into three scores for ranking the developer profiles during our recommendation task. The way in which the three scores were used for recommendation task is described in the section 4.3.4 of this chapter. We first describe how the programming language skill set of attributes were reduced into a single 'Language Score' and then describe how collaboration and project management scores are computed.

**4.3.3.1 Representing language skills for recommendation**

We used the programming language skill profile attributes along with the commit categories nad obtained a score that we called "Language Score" for ranking the developer profiles in the recommendation task. The approach for the computation of Language score was based on the value of the attributes indicating the quantity of code contributions and, the efforts put in to make code contributions, along with the commit categories the set of attributes belonged to. For computing the 'Language Score' from the set of attributes for each category into a single score, we made few assumptions that are mentioned appropriately in the description of 'Language Score' computation.

Every commit category had a set of attributes and the set of attributes under each commit category were considered to have a different effect on the "Language Score". The kind of effect was dependent on the significance a commit category would bring in to indicate the programming language skills of a developer. Below we describe in detail the computations that were made using the attributes under each commits category to derive "Language Score".

**Computing Initial Score according to the Commit category to indicate the "quantity" and "efforts put in" for code contributions:** We assumed the order of commits category significance as provided in the equation (4.5) for computing 'Language Score’ and commit category was used to indicate the quality of contributions.

\[ \text{Accepted} > \text{UnderReview} > \text{NotReviewed} > \text{Rejected} \] (4.5)
4.3. Recommending Profiles

The Accepted commit category was considered most significant because the code accepted from the pull request would pass through the project community review. Marlow et al. [25] only indicated about accepted code commit as a positive signal. However, what kind of signals one could draw from the rejected code of commits was not indicated. Nevertheless, Bosu et al. [6] indicated that coding as one of the aspects that cause the peers to lose trust on the developer. Therefore, the rejected commit category was considered least significance of all the categories. The under review commit category was given slightly higher significance than the not reviewed commit category because the code submission to other users project’s required more effort than committing code to developer’s own projects. However, in some cases, the attentiveness a project have attained might affect our assumption. For instance, the Not Reviewed commit category code contributions would have bought the repository much attentiveness on the GitHub platform. However, even those cases were automatically taken into consideration since we also considered the average number of stars attained for the code contributed projects while computing the ‘Language Score’.

However, the commits order mentioned above does not indicate the kind of effect a commit category have on 'Language Score'. For instance, the kind of effect a commit category could have on the ‘Language Score’ might be positive (or) negative (or) neutral effect. Although, while ranking the commit categories we might have already indicated the kind of effect each category would have on 'Language Score’. In following paragraphs, we clearly mention the assumptions we made on, the kind of effect and the amount of effect each commit category have in the computation of 'Language Score'.

We first describe how we computed the quantity of code contributions across each commit category using the set of attributes. We then describe the amount of change in the quantity of contributions across each commit category and the kind of effect each commit category would have on computing 'Language Score'.

The attributes that would directly represent the volume of code contributions were the Number of files and the Number of changes across those files. Thus, to indicate the quantity of contributions we multiplied the Number of files and Number of lines of changes as shown in equation 4.6. For this computation, we assumed that different files involved different kinds of changes although this assumption may not be true in all the cases.

\[
\text{Quantity Of Code Contributions} = \text{[Number Of Files} \times \text{Number Of Lines Of Changes] (4.6)}
\]

The Number of repositories were assumed to indicate the efforts put in because every repository could have different kind of community rules, styles to make code contributions and, concepts to deal with. The Average number of stars over the contributed repositories was assumed to indicate the extra efforts that were to be put in to make the code contributions to repositories. We made this assumption because Average number of stars indicates the attentiveness gained by repositories on GitHub. Furthermore, the extra efforts indicated by Average number of stars value would further magnify when made across many repositories. Thus considering all the mentioned factors we used the equation 4.7 to compute
4. Implementation

\textit{Total efforts put in}.


\begin{equation}
\text{Total efforts put in} = \text{Basic Efforts} + \text{Extra Efforts} \quad \text{where,}
\end{equation}

\begin{align*}
\text{Basic Efforts} &= \text{Number Of Repositories} \\
\text{Extra Efforts} &= \text{Number Of Repositories} \times \text{Average Number of Stars}
\end{align*}

The following assumptions were made regarding the kind of effect a commit category would have on the 'Language Score'. Simultaneously, we also mention regarding the amount of changes (or) effects that were made to parameters \textit{Quantity of contributions} and \textit{Total efforts put in} under each commit category on the Language Score. Furthermore, the Initial score for rejected category remained unchanged so that it would have unchanged negative effect on the 'Language Score'.

- **Accepted Commit**: The accepted commit category attributes would have a positive effect on the Language Score. The product of "quantity of contributions" and "efforts put in" would result in a score that indicates both quantity and also the efforts put in. We called this score as "initial score" and we used the score to reflect the amount of positive effect by the accepted commit by changing the "initial score". We assumed to double the "Initial score" under this category because, even a small quantity of code contribution or a small amount of effort made under this category was considered highly significant by recruiters.

- **Under Review Commit**: The product of "quantity of contributions" and "efforts put in" was used for computing "Initial score" for this commit category also. The under review commit category was also assumed to have a positive effect on the Language Score made. The positive effect was given for this commit category solely because, creating a code contribution to others project was seen as a positive signal in [25]. However, since the code was neither accepted or rejected we assumed that the amount of positive effect this commit category would have on the initial score was an additional half of actual initial score. By choosing the addition operation over the multiplication, we allowed the "Accepted Commit" contributions to have a major impact on the final 'Language Score' than any other category.

- **Not Reviewed Commit**: The product of "quantity of contributions" and "efforts put in" was again used for computing "initial score" for not reviewed commit category. The not reviewed commit category would have a neutral effect on the Language Score. The neutral effect was assumed because the code contributions were not reviewed by project community since the contributions were made to their own projects.

- **Rejected Commit**: The rejected commits category was assumed to have a negative effect on the Language Score because the project community rejected the code contributions. The "efforts put in" parameter were not considered to be part of the negative effect on language score. However, having more number of repositories in the
rejected commit category was assumed as a bad sign. Thus, this aspect was also re-
lected in the Language Score by computing the "initial score" in the rejected commit
category as a product of "quantity of contributions" and "number of repositories".

The computation of initial score across each category as described above is indicated in
the table 4.6.

<table>
<thead>
<tr>
<th>Commits Category</th>
<th>Initial Score For The Commit Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accepted</td>
<td>((\text{Quantity Of Code Contributions} \times \text{Total efforts put in}))</td>
</tr>
<tr>
<td>Rejected</td>
<td>((\text{Quantity Of Code Contributions} \times \text{NumberOfRepositories}))</td>
</tr>
<tr>
<td>Under Review</td>
<td>((\text{Quantity Of Code Contributions} \times \text{Total efforts put in}))</td>
</tr>
<tr>
<td>Not Reviewed</td>
<td>((\text{Quantity Of Code Contributions} \times \text{Total efforts put in}))</td>
</tr>
</tbody>
</table>

Table 4.6: Initial Score For Each Commits Category

The changes made to the initial score according to the commit category are indicated in
table 4.7. We formulated these equations mentioned in table 4.7 based on our understanding
of the parameters and the effect they could have on language score. The negative effect
from the rejected category code contributions were reflected on LanguageScore in the final
computation step as the percentage of commits category attribute was not yet taken in into
consideration in the computation of language score.

<table>
<thead>
<tr>
<th>Commits Category</th>
<th>Changes Made to initial Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accepted</td>
<td>((\text{initial Score}))</td>
</tr>
<tr>
<td>Rejected</td>
<td>(\text{initial Score} \ [\text{Not Changed}])</td>
</tr>
<tr>
<td>Under Review</td>
<td>((\text{initial Score}) + (\text{initial Score}))</td>
</tr>
<tr>
<td>Not Reviewed</td>
<td>(\text{initial Score} \ [\text{Not Changed}])</td>
</tr>
</tbody>
</table>

Table 4.7: Changes Made to the initial Score According to Commits Category

Adjusting Percentage of Commits Score According to Commit Category  The value
of the percentage of commits in the respective categories reflects the degree of effects each
commit category was assumed to have on the Language Score. Thus, the adjustments were
also made to the percentage of commits attribute in a similar way as the adjustments that
were made on "initial score" across each commit category. The adjusted percentage of
commits value was called RecomputedPercentageOfCommits\( (\text{RPC}) \). For the rest of this
chapter, we call RecomputedPercentageOfCommits as RPC for naming convenience. Table
4.8 indicates the computation of RPC across each commit category.

Combining the initial score with recomputed percentage of commits Score  Having
computed the initial score and RPC, we now combine these parameter values to get a base
score for each commit category for computing final Language Score. The table 4.9 indicates
how a base score for each commit category was computed.
4. IMPLEMENTATION

<table>
<thead>
<tr>
<th>Commits Category</th>
<th>Recomputed Percentage Of Commits (RPC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accepted</td>
<td>(\text{Percentage Of Commits} + \text{Percentage Of Commits})</td>
</tr>
<tr>
<td>Rejected</td>
<td>(\text{Percentage Of Commits} [\text{Not Changed}])</td>
</tr>
<tr>
<td>Under Review</td>
<td>(\text{Percentage Of Commits} + \left(\frac{\text{Percentage Of Commits}}{2}\right))</td>
</tr>
<tr>
<td>Not Reviewed</td>
<td>(\text{Percentage Of Commits} [\text{Not Changed}])</td>
</tr>
</tbody>
</table>

Table 4.8: Recomputing the Percentage of Commits According to Commits Category

<table>
<thead>
<tr>
<th>Commits Category</th>
<th>Base Score Of Commit Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accepted</td>
<td>(\text{Accepted initial Score} + (\text{Accepted initial Score} \times \text{Accepted RPC}))</td>
</tr>
<tr>
<td>Rejected</td>
<td>(\left(\text{Rejected initial Score} + (\text{Rejected initial Score} \times \text{Rejected RPC})\right)) (\left(\text{Average Number Of Stars For Rejected Commits Repositories} + 1\right))</td>
</tr>
<tr>
<td>Under Review</td>
<td>(\left(\text{Under Review initial Score} + (\text{Under Review initial Score} \times \text{Under Review RPC})\right))</td>
</tr>
<tr>
<td>Not Reviewed</td>
<td>(\left(\text{Not Reviewed initial Score} + (\text{Not Reviewed initial Score} \times \text{Not Reviewed RPC})\right))</td>
</tr>
</tbody>
</table>

Table 4.9: Combining initial score with recomputed percentage of commits

**Final Score to compare programming language skills of developer profiles:** After the base score for each commit category was computed, we computed the final LanguageScore value by adding the neutral or positive effect base score of commit categories and reducing the negative effect base score of commit categories as indicated in the equation 4.8.

\[
\text{Language Score} = \left(\text{Accepted BaseScore} + \text{UnderReview BaseScore} + \text{NotReviewed BaseScore}\right) - \left(\text{Rejected BaseScore}\right)
\] (4.8)

4.3.3.2 Representing collaboration skills for recommendation

In this section, we describe how the collaboration skills set of attributes were represented to compute a score and be used for ranking in the recommendation task.

**Interpretation Of Collaboration Attributes** A developer had exhibited a high level of collaboration skills if the values of the ‘Percentage of contributions’ and the ‘Average size of contributors’ are concurrently high. Moreover, the level of collaboration skills exhibited by a developer increases further depending on the higher values of ‘Contribution coverage’ and ‘Number of forked projects’.

The logic behind assuming the interpretation mentioned above was that if the developer had contributed more while competing with a lot of other contributors then the contributor had collaborated well by understanding project communities goals. Moreover, if a developer was able to contribute back to number of the forked projects then it would be a further plus
4.3. Recommending Profiles

for the developer as the developer was able to collaborate with a wide range of project communities.

The equation 4.9 indicates how the attributes of the collaboration skills were combined according to aforementioned description to provide a collaboration score:

\[
\text{Collaboration Score} = (\text{Contribution Coverage} \times \text{Percentage Of Contributions} \times \text{Contributed Back Projects}) + (\text{Contributed Back Projects} \times \text{Average Contributors Size})
\]

(4.9)

4.3.3.3 Representing project management skills for recommendation

In this section, we describe how the project management skills set of attributes were represented to compute a score and be used for ranking in the recommendation task.

**Interpretation of Project Management Attributes:** A developer had exhibited high level of code management and user management skills if the values of the 'Average share of contributions managed', 'Number of users managed' and 'Average number of contributions managers' are concurrently high. Moreover, the level of code management and user management skills exhibited by a developer further increases depending on the higher values of 'Projects Managed Coverage' and 'Right to Manage Projects'.

The logic behind assuming the above interpretation was that, if the developer was able to manage a lot of code contributions along with lot of code contributors by competing with many other collaborators available, then it could logically be inferred that the developer displayed good code management skills. Furthermore, if a developer is managing lot of code contributions and code contributors for many projects where the developer is a collaborator, then it further add to the project management capabilities of a developer.

The equation 4.10 indicates how the attributes of the project management skills were combined according to aforementioned description to provide a project management score:

\[
\text{Project Management Score} = (\text{Projects Managed Coverage} \times \text{Percentage Of Code Contributions Handled} \times \text{Projects Handled}) + (\text{Handled Projects} \times \text{Average Collaborators Size})
\]

(4.10)

4.3.4 Recommendation strategy

In this section, we explain how the similarity between the job advertisement was computed to make a recommendation, and further we describe the ranking approach that was used to rank the profiles accordingly using the three scores that we computed for each developer profile.
4.3.4.1 Similarity computation function

We used the Apache Solr’s default similarity to compute the similarity between a job advertisement and the generated GitHub user profile. Apache Solr provides the Apache Lucene similarity function for computing the similarity. Apache Lucene similarity function is further based on the vector space model. In this model, the documents between whom the similarity needs to be computed are represented as weighted vectors in a multi-dimensional space. The cosine similarity metric is widely used for computing the similarity between the documents in the vector space model. The documents in our case can be referred to job advertisements or the developer profiles. However, since our task is to recommend the developer profiles that suit job advertisements, we only compute the similarity between the job advertisement and the developer profiles.

The cosine similarity value, when computed using the equation 4.11 between the job advertisement and the two developer profiles using the vectors indicated in the table 4.10 has few drawbacks. The cosine similarity values resulted in were 0.000011 between job advertisement and Developer profile 1 and 0.0030 for the same job advertisement and Developer Profile 2. The cosine similarity for the Developer Profile 1 is lower than the Developer Profile 2 although the terms *jquery*, *eclipse*, *sql* are not present in the Developer Profile 2. The reason for the kind of similarity scores being obtained is due to the non-matching terms influence during the vector length normalisation phase of similarity computation according to the equation 4.11.

\[
\text{Cosine Similarity}(V(a), V(b)) = \frac{V(a) \cdot V(b)}{|V(a)||V(b)|}
\]

Where \(V(a)\cdot V(b)\) is the dot product of the weighted vectors, and \(|V(a)|\) and \(|V(b)|\) are their Euclidean norms.

Thus, in our recommendation task instead of using the cosine similarity equation 4.11 we used Apache Solr default similarity method that is an accustomed form of the equation 4.11 and handles the aforementioned issue with the equation 4.11 in context of our recommendation task.

Apache Solr refined Similarity function

Apache Solr, a search engine, aims to find the appropriate documents for the provided query. This query and document style is apt for our recommendation task because our aim is to find the developers that contain as many keyword terms mentioned in the job advertisement as possible according to the respective weights. In the following paragraph, we indicate regarding what a query and document are in the context of search engine along with how Apache Solr default similarity method looks to facilitate similarity between the query and the document.

A query can be defined as a formal statement indicating the information based on which the documents are to be retrieved. Thus in the context of a search engine the similarity computation moves from the similarity between the two documents to the similarity between the query and the documents. In the context of our recommendation task, where developer profiles are recommended to suite a job advertisement, a query will be a job advertisement and the documents we are trying to find will are the developer profiles. This analogy was drawn
4.3. Recommending Profiles

<table>
<thead>
<tr>
<th>Terms</th>
<th>Job Advertisement ( tf-idf )</th>
<th>Developer Profile 1 ( tf-idf )</th>
<th>Developer Profile 2 ( tf-idf )</th>
</tr>
</thead>
<tbody>
<tr>
<td>tomcat</td>
<td>0.0714</td>
<td>0</td>
<td>0.0714</td>
</tr>
<tr>
<td>eclipse</td>
<td>0.0022</td>
<td>0.0022</td>
<td>0.0022</td>
</tr>
<tr>
<td>java</td>
<td>0.0011</td>
<td>0.0011</td>
<td>0.0011</td>
</tr>
<tr>
<td>jquery</td>
<td>0.0024</td>
<td>0.0024</td>
<td>0</td>
</tr>
<tr>
<td>linux</td>
<td>0.014</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>sql</td>
<td>0.0058</td>
<td>0.0058</td>
<td>0</td>
</tr>
<tr>
<td>weblogic</td>
<td>0.3333</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>css</td>
<td>0</td>
<td>8.18</td>
<td>0</td>
</tr>
<tr>
<td>git</td>
<td>0</td>
<td>4.97</td>
<td>4.97</td>
</tr>
<tr>
<td>glassfish</td>
<td>0</td>
<td>0.25</td>
<td>0</td>
</tr>
<tr>
<td>gradle</td>
<td>0</td>
<td>0.0125</td>
<td>0</td>
</tr>
<tr>
<td>groovy</td>
<td>0</td>
<td>0.0333</td>
<td>0</td>
</tr>
<tr>
<td>javascript</td>
<td>0</td>
<td>7.25</td>
<td>0</td>
</tr>
<tr>
<td>jdbc</td>
<td>0</td>
<td>0.0312</td>
<td>0</td>
</tr>
<tr>
<td>jsp</td>
<td>0</td>
<td>0.0294</td>
<td>0</td>
</tr>
<tr>
<td>log4j</td>
<td>0</td>
<td>0.0121</td>
<td>0</td>
</tr>
<tr>
<td>lucence</td>
<td>0</td>
<td>0.0714</td>
<td>0</td>
</tr>
<tr>
<td>maven</td>
<td>0</td>
<td>0.0188</td>
<td>0.0188</td>
</tr>
<tr>
<td>shell</td>
<td>0</td>
<td>0.0019</td>
<td>0.0019</td>
</tr>
<tr>
<td>velocity</td>
<td>0</td>
<td>0.0588</td>
<td>0</td>
</tr>
<tr>
<td>mvc</td>
<td>0</td>
<td>0.0357</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 4.10:

because, just like a query indicating the information needed to retrieve relevant documents, a job advertisement indicates how a relevant developer profile should be.

Having mentioned what a query \( q \) and document \( d \) is, we describe Apache Solr default similarity method that facilitates computation of relevancy score between the query and the documents.

The relevancy score computation in Apache Solr is computed using the formula mentioned in the equation 4.12 as mentioned in chapter 3 of [15]. The equation 4.12 is formulated in such a way that, different aspects of query and documents that influence the relevancy score are incorporated, so that developers of a search engine can leverage them to provide relevant documents to queries of the search engine users. The provided equation was not evaluated in TREC style due to the open-source nature of the project as mentioned in [2]. The Apache Solr project is widely being used by different commercial websites such as eBay, IBM Websphere Commerce, Netflix [11] which indicates that the Apache Solr is trustworthy to use. However, if the mentioned websites are using default similarity method (or) if they are using their custom made relevancy methods that are allowed in Apache Solr
is not known.

$$Score(q,d) = \sum_{t \in q} (tf(t \in d) \cdot idf(t)^2 \cdot t.getBoost() \cdot norm(t,d)) \cdot coord(q,d) \cdot queryNorm(q)$$  \hspace{1cm} (4.12)$$

Where: 
t = term; d = document; q = query; f = field
tf(t \in d) = numTermOccurrencesInDocument^{1/2}
idf = 1 + log(numDocs/(docFreq+1))
coord(q,d) = numTermsInDocumentFromQuery/numTermsInQuery
queryNorm(q) = 1/(sumOfSquaredWeights^{1/2})
sumOfSquaredWeights = q.getBoost()^2 \cdot \sum_{t \in q} (idf(t) \cdot t.getBoost())^2
norm(t,d) = d.getBoost().lengthNorm(f).f.getBoost()$$

In our recommendation task there was no need for a custom relevancy function, because the resources we have in our hands for computing relevancy such as, the set of keywords and their weights can be leveraged using the equation 4.12 itself. Although we do not use all the aspects included in the formula, we mention about all the elements of the formula along with indicating the elements we used in the formula for our recommendation task. The method indicated in equation 4.12 uses only the terms mentioned in the query in computation rather than using all the terms in both query and the document for the computation.

The purpose of each element in the formula is provided as follows which a developer of search engine can use according to their use case:

- **Term Frequency(tf) and Inverse Document Frequency(idf)** Term frequency is a measure of how often a particular term appears in a matching document and its an indication of how well a document matches the term. The idf score indicates how rare a search term is across the documents set.

- **boost** This aspect allows the users and developers to leverage the domain knowledge known about the documents to be used in generating relevancy score and thus providing better relevant documents for the queries. Solr facilitates users or developers of a search engine to provide boost factors for each query term at query time. Furthermore, it also allows developers of a search engine to boost the document or fields within documents at index time that we describe as part of field norm described in the following sentence.

- **Normalisation Factors** These aspects include three norms namely field norm, query norms and, coord factor into relevancy score. The field norm factor is computed during the index time and is a combination of matching document boost, length normalisation factor to penalise longer documents and the matching field’s boost. The query norm factor is used to make the relevancy score across the documents comparable. Finally, the coord factor is a measure to indicate how much of the query each document matches.

In our recommendation task we used the tf-idf scores, query time boosting of query terms. In the normalisation factors of the formula, we used query norms and the coord fac-
tor for computing the relevancy score between the job advertisement and developer profile. Thus, we created a query from the keywords that were indexed for each job advertisement. We also included the keyword weights in the query by using the keyword weights of the job advertisement. Furthermore, Apache Solr also allows developers of the search engine to mention the fields or a custom score function based on which the ranking of the relevant documents can be done in the query itself. Although we will use the relevance score initially to rank the documents we also need to rank documents when the relevance scores between two documents are equal. The relevancy score in case of our recommendation task will be equal when the same keyword terms of documents match the query terms. For instance, if the query contains three terms and all the three terms are also present in the two documents then the relevancy score will be equal because in our documents set the keywords frequencies are one. Furthermore, the idf scores of terms are equal considering that fact that each keyword terms appears only once in the developer profile document. Therefore, we also mentioned the ranking details in the query created using the job advertisement keywords for finding the relevant developer profiles. The fields that were used in the ranking and the approach we took for ranking are described in following section.

4.3.4.2 Ranking

For ranking the profiles that would be recommended for a job advertisement we used the following approach.

1. The profiles are initially ranked according to the relevance score as indicated previously.

2. If there was a tie with the relevancy score between any two profiles then a custom score was computed for each profile that would be used to break the tie. The custom score was computed using a weighted sum of the Language Score and the weights for the language scores was known from the query.

3. If the custom score was also tied then we used the project management score to break the tie.

4. If the project management score was also tied then we use the collaboration score to break that tie.

In the next chapter, we describe how we evaluated our proposed developer profile and also the profile recommendations that were made to job advertisements.
Chapter 5

Evaluation

In this chapter we describe the evaluation procedure we followed for evaluating our contributions made in the project. The project evaluation covered our two contributions; the first one was the generated GitHub developer profile from the github user’s repository data and the second was the recommendation of generated developer profiles to job advertisements. We first describe the evaluation of the generated profile followed by the evaluation of our recommendation task.

5.1 Evaluating Generated Developer Profile

In this section, we mention our evaluation goal and then describe the aspects across which the generated developer profile was assessed to achieve our evaluation goal. We then describe the evaluation process followed for assessing the project across those aspects, followed by the description of our experiment step up.

5.1.1 Assessment aspects of developer profile

Our first evaluation goal was to assess the extent to which our generated GitHub developer profile had addressed the recruiters issue of investing many efforts to verify signals on GitHub and infer developer skills. Thus, to achieve our evaluation goal we used the following four aspects of assessment.

- **Aspect 1 (ASP1)** The first aspect of the assessment we considered was to know the relevance of attributes that were used to indicate the specified skill exhibited by a developer on GitHub. The developer profile attributes were created based on the signal descriptions in Marlow et al. [25]. However, we still wanted to evaluate the developer profile attributes relevance to indicate the skill because in the skill profiles we also have few statistical attributes that were inferred based on the descriptive signals.

For instance, in the collaboration skill profile attributes such as the *Average percentage of contributions* made by the developer along with the attribute *Average contributors size* was not mentioned directly in [25]. Similarly, the project management profile attributes were not directly mentioned but inferred from the description that,
recruiters see code management as a project management skill. Furthermore, even in case of programming language profile also the attributes were not mentioned but we inferred them based on the description provided on the quality of contributions. Finally, all the three skill profiles were represented using a set of attributes instead of using a single attribute to indicate the skill. However, the set of attributes together were better in indicating the indicated skills than a single attribute mentioned is still an assumption. Therefore, instead of assuming our conversion of descriptive signals into few statistical attributes was correct we evaluated the correctness of statistical attributes derived from the descriptive signals. Furthermore, we also assessed if and the set of attributes we used were more useful to indicate the skill than a single attribute in a skill.

The assessment of correctness of developer profile is useful for another reason also. For instance, if the set of attributes we used for indicating a particular skill were wrong, then the generated developer profile would also never be used by the technical recruiters. Thus, considering all the mentioned reasons this aspect of assessment was basic and important aspect to assess regarding the generated developer profile.

- **Aspect 2 (ASP2)** The second aspect of the assessment was to know during which stages of a traditional recruitment process a technical recruiter preferred to use the developer profile. Since we are trying to provide social recruiting advantage for recruiters, the assessment along this aspect would help us to understand, where does our generated developer profile fit well in a traditional recruitment process.

- **Aspect 3 (ASP3)** The third aspect of the assessment was to know the ease of understanding of the attributes that were mentioned to indicate a skill. This aspect of assessment was also necessary because if the technical recruiter felt hard to understand the developer skill attributes mentioned then the technical recruiters may not use the generated developer profile.

- **Aspect 4 (ASP4)** The fourth aspect of the assessment was recruiters preference over using the information in the generated developer profile to infer the skills exhibited by a developer on GitHub platform.

Having mentioned the aspects across which we would be assessing and why those aspects were relevant for our assessment, we mention below how we evaluated the developer profile along the mentioned aspects.

### 5.1.2 Questions for the developer profile assessment

In this section, we mention the approach we took for assessment along with the questions and the kind of demographic respondents we considered for our assessment.
For evaluating the generated developer profile across the aspects discussed above, we decided to take a questionnaire and interview approach as the mixed approach would provide a better assessment to the generated developer profile. In the questionnaire and interview, we asked the respondents few questions that cover the four aspects of our assessment and the responses received for those questions was used to assess our generated developer profile. We used the following questions in the questionnaire and interview for assessing developer profile across four indicated aspects.

**ASP1 Questions** For ASP1 we asked the following two questions to assess if we have used the correct set of attributes and if all the attributes together can indicate the skills better than a single attribute.

1. **Question 1:** How relevant do you consider each attribute for indicating the described skill exhibited by the developer?

   The respondents provided their responses using an ordinal scale where, the *Highly Relevant* option was used to signify the high relevance of an attribute in indicating the skill. While *Highly Irrelevant* option was used to signify the high irrelevance of an attribute to indicate the skill.

   We divided the subjective scale mentioned into three sides, and we used each side for inferring the validity of the attribute. The first one is the positive side that includes the options of *Highly Relevant, Quite Relevant and, Slightly Relevant*. The second is the negative side of the scale that has *Highly Irrelevant, Quite Irrelevant and, Slightly Irrelevant*. The third side of the scale is the neutral side that has *Neither Relevant or Irrelevant* option.

2. **Question 2:** How much would you agree with the statement: Instead of one attribute, all the attributes together under the described skill would exhibit the indicated skill better?

   The respondents provided their responses using an ordinal scale where, *Completely Agree* was used to signify high acceptance to the statement made while *Completely Disagree* was used to signify the high disagreement with the statement.

   We divided the subjective scale mentioned into three sides, and we used each side for inferring the validity of the set of attributes together for indicating a skill. The first is the agreed (or) positive side that includes the options of *Completely Agree, Mostly Agree, Slightly Agree*. The second side is the disagreed (or) negative side that have the options of *Completely Disagree, Mostly Disagree and, Slightly Disagree*. The third category is the neutral category that has the option of *Neither Agree or Disagree*.

   We did not provide "I do not know" option to the respondents, but we provided the respondents to give their opinion in the comments section for each skill profile. Having described the questions and the scale in the following paragraphs we describe, how we conclude the validity using the responses received for the questions. We numbered the options selected by the respondents from one to seven to compute the statistical values of
median and variance because we used a subjective scale for recording responses. We used the value of one to indicate the options *Highly Irrelevant or Completely Disagree* for the two questions. We used the value of seven to indicate the options *Highly Irrelevant or Completely Disagree* for the two questions.

We consider the validity of an attribute to indicate the developer skill, based on the median value along with the variance. A median value on the positive side of the scale indicates that the attribute is valid and if the attribute is on the negative side of the scale then the attribute is invalid. If the median value is on the neutral value of the scale, then we consider the attribute validity as neither valid or invalid. Furthermore, before concluding a particular attribute as invalid or valid or neither of them using the median value of responses, we also check the variance value. If the variance value in responses for an attribute is high enough to alter the median value position from one side to another side of the scale, then we further analyse the responses for concluding the validity. We determine if the variance value can alter the median value by subtracting or adding the variance value with the median value. For further analysis of responses, we considered the maximum number of respondents selecting options belonging to positive or negative or neutral side of the scale and determined the validity. If the number of responses selecting the options on each side is same, then we consider the attribute validity as cannot be decided based on the received responses. Similar process was used for the second question responses to infer if the set of attributes together indicate a skill.

**ASP2 Question**  For ASP2 we asked the following question to understand during which step in the traditional recruitment process can our developer profile be useful. *Select the stages in a traditional recruitment process, you would prefer to use the proposed developer profile.*

For the above question, we provided few options, from which the respondents were allowed to select any number of the options provided according to their opinion. The options provided were the steps in a traditional recruitment process which were mentioned in [9]. We also provided an ”Other” option for the respondents to record a step which may not be provided in the options. The steps we mentioned in the options were as follows:

1. **Step 1:** Assessing whether a candidate has actually mastered the mentioned skills.
2. **Step 2:** Assessing the impressions that third parties received from a candidate.
3. **Step 3:** While interviewing a candidate.

We considered the maximum number of votes from the respondents for a recruitment step to infer, the step in traditional recruitment process where the generated developer profile is useful.

**ASP3 Question**  For ASP3 we asked the following question to understand the if it is easy for the respondents to understand the attributes used.  *How easy was it for you to understand the attributes used to infer about each skill?*
The respondents provided their responses using an ordinal scale where the "very hard" option is used to indicate the high degree of hardness for understanding. The option "very easy" option is used to indicate the high degree of easiness in understanding the skill profile. Having described the questions and the scale in the following paragraphs we describe, how we conclude the ease of understating of the skill profiles using the responses received for the questions.

We numbered the options selected by the respondents from one to seven to compute the statistical values of median and variance because we used a subjective scale for recording responses. We used the value of one to indicate the option Very Hard and the value of seven to indicate the option of Very Easy.

Just like the ASP1 we divided the subjective scale mentioned into three sides, where we used each side for inferring the validity of the set of attributes used for indicating a skill. The first is the easy side that includes the options of Very Easy, Quite Easy, Slightly Easy. The second side is the hard side that have the options of Very Hard, Quite Hard and, Slightly Hard. The third side of the scale is the neutral side that has the option of Neither Easy or Hard.

Similar to the ASP1 of inferring the validity, here we infer the ease of understanding. The median along with the variance values are used to infer the ease of understanding of the skill profile. If the variance is high enough to effect median value, we further investigate the responses to the questions to infer ease of understanding. We considered the maximum responses for a particular side of the scale namely easy (or) hard (or) the neutral, to infer ease of understanding.

**ASP4 Question** For ASP4 we asked the following question to evaluate if the developer profile is being preferred for use in inferring regarding the skills of the developer.

_How much would you prefer to use the developer profile when compared to your way of inferring about skills exhibited by a developer on GitHub? If you have not tried or do not know other ways then please provide your preference without any comparison._

The responses were asked using an ordinal scale where, Strongly prefer was used to signify high usage while Wouldn’t prefer at all was used to signify the high disagreement of usage. Just like the ASP1 we divided the subjective scale mentioned into three sides, and we used each side for inferring recruiters preference to use the skill profile. The preferred side of the scale consists of options Strongly prefer, Mostly Prefer, Slightly Prefer. The not preferred side of the scale has the options Strongly wouldn’t prefer, Mostly wouldn’t prefer and, Slightly wouldn’t prefer. The neutral side of the scale have options Neither prefer or wouldn’t prefer.

Similar to the ASP1 of inferring the validity, here we infer the preference to use. The median along with the variance values are used to infer the preference to use the skill profile. If the variance is high enough to effect median value, we further investigate the responses to the questions to infer ease of understanding. We considered the maximum responses to a particular side of the scale namely preferred or not preferred or the neutral side to infer recruiters preference to use skill profiles.

Having indicated the questions and their responses interpretation process for evaluating the developer profile across the four aspects, in the following paragraph we describe the
kind of demographics of respondents that will be relevant to our evaluation.

**Respondents for developer profile evaluation** As indicated in the conclusions we drew from the GitHub related work, the recruiters without software knowledge were finding it difficult to understand the information derived from developer traces online. Thus, even in our evaluation a respondent having experience in recruiting software developers along with some software knowledge were considered most relevant. The responses from the respondents with the described demographics weigh higher than any other kind of respondents in our developer profile evaluation. We explicitly mentioned about the significance of the responses from the respondents with the recruitment experience because we also asked the opinions of the respondents who did not had any recruitment experience but had some software knowledge. There were two reasons for this decision, the first reason was that considering the opinions of software developers themselves who would be judged using the generated developer profile would be worth knowing, across the first aspect of our developer profile assessment. The second reason was that there was difficulty in finding respondents with recruiting experience along with some software knowledge. During our evaluation, we did manage to find 13 respondents with software developer recruitment experience along with some software knowledge. The demographic details of the respondents regarding their recruitment experience along with their background are provided in the Results chapter of this thesis.

Although it may appear that the GitHub approach may not be a useful prospect considering the extra requirement of having some software background, but it is not completely true. For instance, in the traditional recruitment process of a software developer, the opinion of people with software knowledge are essential to judge software skills of the potential candidates. Thus, the GitHub developer profile information can reduce the efforts of the people involved in judging the potential candidate’s skills during the recruitment process. Furthermore, during the early days of a start-up software company, generally software experts in the startup act as recruiters. Thus considering these aspects a developer profile indicating skills of a developer using developer traces on GitHub, can still be useful in the software developer recruitment process.

5.1.3 **Questionnaire and Interview setup for evaluation**

**Questionnaire** For assessing the generated developer profile along the four aspects discussed previously, we created a questionnaire that consisted of the profile attributes description along with the questions associated with our profile assessment. The questionnaire took about 20 to 25 minutes for the respondents to complete. We hosted the questionnaire online on social media recruiting groups on LinkedIn along with sending the questionnaire link through e-mails to different people whose demographic details are relevant to our evaluation task. We also provided space to receive comments on the developer skill profiles, and the opinions of the respondents can be used to infer the aspects along which we can improve our generated developer profile. All the respondents during the questionnaire self-evaluated their knowledge about the recruitment experience in recruiting a software developer. During the evaluation through the questionnaire, we also received some feedback personally
on the questionnaire for improvements. The suggestions were for more precise description of the attributes along with the description of the skill being indicated. We considered this feedback to improve the description of the attributes used along with the skills description.

For validating the responses for the questionnaire, we took the following approach.

We considered the respondents who provided comments for the developer skill profiles as valid responses. We took this decision because providing some specific comments requires proper understanding of the profile, and this also shows respondents interest in providing the responses, considering we did not give any incentives for the respondents. If the respondents did not provide any comments, then we compared the responses received for the two similar kinds of questions in our questionnaire. Thus, if respondents who did not provide any comments on the developer profile provided consistent answers for the two similar kinds of questions, then the responses from the respondents were considered valid. This approach will help in validation because if the respondent answered the questions by randomly selecting the options that would lead to inconsistency in the responses of the two similar kinds of questions. The two similar kind of questions we considered for validation are as follows:

1. How much would you prefer to use the developer profile when compared to your way of inferring about skills exhibited by a developer on Github? If you have not tried or do not know other ways then please provide your preference without any comparison.

2. Similar Kind of Question: How relevant does each proposed skill profile have in your opinion, for the recruitment process you would follow for recruiting a software developer. This question was also provided the subjective scale with "Highly Relevant" as the highest level of relevance and "Highly Irrelevant" as least level of relevance.

In context of validating our responses, two subjective options selected by a respondent for the same kind of questions are considered consistent, if the options are same or the options are only one level away on the subjective scale. For instance, for the first question if the respondent have selected that he would "Mostly Prefer" to indicate preference and for the second question if the respondent has provided "Highly Relevant" option then these responses were considered consistent because they are one level away on the subjective scale.

**Interview** We used the same questions of the questionnaire in conducting the interviews with the people having recruitment experience along with some software knowledge. The interviews we conducted were a semi-structured interview. For the evaluation through interviews we represented the generated developer profile in an infographic style instead of providing descriptions regarding the profile, as we would be present in person to explain about the profile attributes if necessary. The structured part of the interview contained the same questions as the questionnaire, except that we used a different way to present the developer profile. The interviewees provided their responses to the questions on the questionnaire, and after the questions were answered we asked interviewees to provide general comments and improvements, they would like to see in the generated developer profile.
5. Evaluation

The interviews conducted in total took approximately half an hour to complete. We used these comments to understand the aspects along which our generated developer profile fell short of our research objective. The comments given after submitting the responses to the questions were noted on a paper and used during analysis of our results.

The questionnaire and the structured part of interviews we conducted consisted of three parts with each part being described in the following sections. The screen shot showing the profile description for the questionnaire and the questions we asked on the profiles are provided in Appendix B of this thesis. We also provide the infographic we used to represent the profile during interviews in Appendix B of this report.

5.1.3.1 Demographic Information of the respondents

In the demographic information section of the questionnaire, we asked the respondents regarding their experience in recruiting software developers along with the job position, experience with GitHub and also regarding if they used GitHub in recruiting software developers.

5.1.3.2 Validity of the generated profile

In this section of the questionnaire we described the attributes used to indicate each described skill exhibited by a software developer and then asked the respondents to indicate their opinion on the relevance level of each attribute towards indicating the skill. We also asked the respondents if they think that the attributes described altogether would better indicate the skill we tried to exhibit than a single attribute. This section of questionnaire covered the first aspect of our assessment.

5.1.3.3 Relevance of the generated developer profile in the recruitment process

In this final part of the questionnaire, we tried to assess how the respondents with recruitment experience and also who had used and didn’t use the GitHub before for recruitment, perceive our generated developer profile. The main questions in this section of the questionnaire included: if the respondents of the questionnaire preferred to use the generated developer profile to know about the skills exhibited by a developer on GitHub and also about how easy it was for the respondents to understand the attributes used to indicate of each skill. This section of questionnaire covered the questions regarding the remaining three aspects of our evaluation.

5.2 Evaluating Recommendations Of Profiles For The Job Advertisements

The second contribution of the project was recommending the GitHub user profiles to a job advertisement. A recommendation system task that recommends GitHub profiles to job advertisements would significantly reduce the efforts a technical recruiter had to put in to find appropriate software developers for the mentioned skill set in job advertisements. However,
5.2. Evaluating Recommendations Of Profiles For The Job Advertisements

the extent to which the efforts of a recruiter are reduced depends on how good the recommendation task performed. Thus in the second aspect of the project evaluation our aim was to evaluate if the recommended user profiles based on the skill set requirements were relevant for a given job advertisement or not. There was already considerable research on the metrics that we could use for the evaluation of a recommendation system that we leveraged for evaluating our developer profile recommendations made to a job advertisement.

We have discussed in the Evaluation Approach section of the Approach chapter, that our recommendation task falls under the non-interactive task, and the approach we took to measure appropriate metrics was through a user study. However, by measuring the metrics through the user study, we cannot precisely know if a profile that’s being judged relevant is the best developer profile for that job advertisement. To precisely know if a developer profile that is judged relevant for a job advertisement is the best developer profile then the person judging the relevance had to see all the developer profiles in the dataset and then judge. Thus, this is a threat to validity for the evaluation metric values of our recommendation task.

In the following sections, we first describe the metrics that were appropriately selected to evaluate our recommendation task. Finally, we describe the experiment setup we used for measuring the selected metrics through which we infer how well our recommendation task performed.

5.2.1 Metrics Selection for Evaluating the recommendations

Having mentioned the kind of experiment we performed, we now describe in this section the metrics that we used for measuring the prediction accuracy. Shani et al. [30] discussed three categories of prediction accuracy measures:

• **Measuring the accuracy of rating predictions**: This category of measures could be used when the task of the recommender was to predict how a user would rate certain item.

• **Measuring the accuracy of usage predictions**: This category of measures could be used when the task of the recommender was to predict items that a user could use. This category of prediction accuracy was similar to our recommendation goal, as we recommend useful developers profiles relevant to a job advertisement's skill set.

• **Measuring the accuracy of rankings of items** This category of metrics were used when the recommended list of items are according to the user preference. In our project, we also ordered the developer profiles so that the best profiles that matched the job advertisement skill set requirements are recommended first.

Now having decided the categories that would be applicable for the evaluation of our recommendation task, we now describe the metrics we used for the evaluation in each category that applied to our recommendation task:
• **Metrics For Accuracy of usage prediction** Precision and Recall were the two basic metrics we computed for our evaluation. Precision indicates the fraction of the number of useful items recommended to the total number of relevant items recommended. The recall is the fraction of total number of useful items recommended to the total number of actual relevant items available. Thus precision and recall at a value \( N \) where \( N \) is the number of recommended developers profiles for a job advertisement were to be computed. However, the computation of the precision and recall values for a fixed length of recommendations say \( N \) would result in equal precision and recall values, although conceptually they indicate different aspects of accuracy. In the Results chapter, we describe how we computed the precision and what precision values did our recommendation strategy yielded.

• **Metrics For Accuracy of ranking of items** We used normalized distance-based performance measure (NDPM) metric for measuring our ranking accuracy based on the description provided in Herlocker et al. [16]. We used NDPM because NDPM is similar in form to the Spearman and Kendall Tau rank correlations, but NDPM metric provides a more accurate interpretation of the effect of tied user ranks. We present the details of how we computed the NDPM metric while presenting the NDPM metric values in the results chapter.

Having indicated the metrics we used for evaluating the accuracy of recommendations we now describe in the section below, the experiment that was performed to measure the selected metrics for evaluation.

### 5.2.2 Experiment setting for evaluation of recommendations

For evaluating our recommendation task through a user study approach, the biggest constraint we had was finding appropriate users to judge the relevance of a developer profile for a job advertisement. This constraint was similar to the difficulty we had for finding the respondents for our questionnaire while evaluating our generated developer profile. Furthermore, the user should be able to understand the generated developer profile also to judge the relevance to job advertisement. Therefore, we requested the software developer respondents who had already answered the questionnaire to solve the task of judging the relevance between the generated software developers and a job advertisement. We choose to ask the respondents for our questionnaire for performing relevance judgement task, because the respondents were already familiar with our generated developer profile and it would be easier for them to judge the relevance between the developer profile and job advertisement. Although all of the previous respondents to the questionnaire did not respond, a very few of them responded to help. Thus, we choose a random sample of 23 job advertisements for our evaluation based on the respondents who were available for solving our relevance tasks.

In the following section we described the experiment setup details on the CrowdFlower platform through which the relevance opinions were received from the respondents.
5.2. Evaluating Recommendations Of Profiles For The Job Advertisements

5.2.2.1 CrowdFlower platform for evaluating recommendations by a user study

CrowdFlower\(^1\) is a platform that helps users to create human computation jobs. The human computation jobs range from detecting objects shape in a picture to judging the relevance of a search query and the resulted items. The platform also consisted of some templates which the users can use to create the jobs according to their task requirements. In context of our evaluation experiment the CrowdFlower job template that we used for our evaluation since our evaluation was also related to relevance judgement. A task in our CrowdFlower job consisted of a job advertisement along with a recommended generated developer profile. We computed the precision for the accuracy with the developer profile list length of five. We choose the shorter length so that recruiters had to invest least amount of efforts to find appropriate developers for their job advertisements. Therefore, the generated developer profile mentioned in the task of our CrowdFlower job were the top five developer profiles that were recommended using our recommendation strategy.

The task of the contributors to our CrowdFlower job was to judge the relevance between job advertisement and the generated developer profile using the attributes of our generated developer profile and job advertisement details. We explicitly asked the contributors to our CrowdFlower job to judge the relevance along the skill set requirements of job advertisement. We asked the CrowdFlower contributors to indicate the relevance between the job advertisement and the generated GitHub profile on a scale of one to four. Where one indicated "irrelevant" on a subjective scale and 4 indicated a "perfect match" between job advertisement and recommended developer profile. The contributors used the subjective description of each option we provided and judged the relevance between a job advertisement and recommended developer profile, which provides some uniformity in the relevance judgement. The subjective descriptions along which the CrowdFlower contributors made the relevance judgements are as following:

1. "Irrelevant: Paired with Job Advertisement by Mistake" It is obvious that the result doesn't match job advertisement’s skill set requirements at all. Even if there are keywords that profile and skill set in the job advertisement have in common, the context and intent of usage is entirely different.

2. "Relevant: Obvious Differences" A part of skill set requirements mentioned in the job advertisement matches with the provided developer profile, but there are also differences(programming languages expertise, Software technologies).

3. "Relevant: Possible Differences" There is a slight mismatch between job advertisement skill set requirements intent and the provided developer profile, but you do not think that it is necessary.

4. "Perfect Match" Profile matches every property specified by job advertisement skill set, such as programming languages expertise and software technologies. Job advertisement intent is clearly satisfied, and this is the kind of developer profile the technical recruiter intended to find with respect to a job advertisement.

\(^1\)http://www.crowdflower.com/
5. Evaluation

Having mentioned the basic structure of our CrowdFlower relevance job we mention in the following paragraphs some essential CrowdFlower job details in our experiment setup.

**What kind of information was provided to CrowdFlower relevance job contributors to make relevance judgements?** We provided the descriptions of the attributes that were used to generate the developer profile using GitHub user data. Thus a job contributor when in doubt regarding the meaning of any attributes used in the recommended developer profile, a job contributor can use the provided developer profile attribute descriptions. The generated developer profile we provided for judging relevance of a job advertisement did not contain any demographic information about that developer profile. We did not provide the demographic information because the relevance was to be judged only across the skill set requirements of a job advertisement. The information we provided regarding the developer profile contained the values of the attributes that indicated the skills exhibited by the developer as described in the implementation chapter.

The details that was provided with respect to job advertisement were the job title and job description that we crawled from the monster.com website. The job contributor were allowed to use the web resources like Google search engine when the contributor required understanding about few concepts in the job advertisement or the generated developer profile of the task.

**Who were the contributors who solved the tasks?** We used the CrowdFlower internal channel to make the CrowdFlower tasks available to our contributors who would judge the relevance between the job advertisement and the recommended developer profile. As mentioned previously the contributors who solved the tasks on CrowdFlower were the respondents of our previously described questionnaire. Some of the contributors to our CrowdFlower tasks were doing masters in computer science and some of the contributors had already completed their masters in different fields but still had reasonable software knowledge to judge the relevance.

The contributors judging the relevance were not given any incentive for solving the tasks. Also the contributors were only given the four relevance judgement options described above to select and there were no additional options provided to the contributors. All the contributors self-evaluated regarding their software knowledge during the evaluation responses taken from the questionnaire. Thus, we did not ask the contributors again to rate regarding their software knowledge. Furthermore, considering the educational background of the contributors, providing them freedom to use any media required to know more about the details in the job advertisement was assumed to be sufficient for the contributors to solve the task.

**When was a task status considered finished and how many contributors were involved?**
A task was considered finished when two contributors registered their opinions regarding the relevance between the recommended developer profile and job advertisement. We choose the number to be two based on the number of the respondents available to solve the tasks. Although more number of people judging the relevance tasks could add concreteness to the
relevance judgement, we took this decision considering the constraints we had. Also each contributor solving the tasks viewed two tasks on each page.

In total we had 9 contributors who solved our 230 tasks considering each relevance judgement task required at-least 2 respondents to be completed. The appendix B of this report provides the screen shots of task instructions and also a screen shot of job advertisement and a generated developer profile that was provided to our relevance judging task contributors.
Chapter 6

Results

In this chapter, we provide the results and interpret the results obtained through questionnaire and the experiment we performed to evaluate our contributions. We first, mention and interpret the results from the assessment of our generated software developer profile. We then present and interpret the results regarding our recommendation task of recommending generated developer profiles to job advertisements.

6.1 Software Developer Profile Assessment Results

The questionnaire and interviews we used for measuring the validity and the relevance of the developer profile in the recruitment process received responses from 31 respondents in total. Out of 31 respondents, 13 respondents had some experience of recruiting a software developer, and the rest of 18 were software developers. Out of 13 respondents with recruiting experience, we collected the responses through semi-structured interviews with six interviewees.

Out of the six respondents, four of them have recruitment experience of 1 to 3 years, and two of them had more than six years of experience in recruiting software developers. The interviews lasted about twenty to twenty-five minutes, and all the interviewees who provided responses were professors with different designations in the field of computer science. Out of the other remaining seven recruiters whose responses we received through the questionnaire, two respondents had more than six years. Out of the remaining 5, two respondents had 4-6 years, and the remaining three respondents had less than a year experience in recruiting software developers. Out of 7 recruiters who provided responses through the questionnaire, 5 of them were working in the software industry in different positions, while the remaining two were doing their masters in computer science at the time of the questionnaire. All the respondents to our questionnaire with and without recruiting experience had some software knowledge. Furthermore, we did not provide any incentives for the interviewees (or) respondents to our questionnaire for their responses.

While validating the responses received from the questionnaire as per the process we described in Evaluation chapter, we found responses from two respondents with recruitment experience as invalid. Thus, we considered only five recruiter’s responses obtained through
6. Results

the questionnaire as valid. In case of the respondents without recruitment experience we found three invalid responses. Thus, we analysed only 15 developer’s responses received through the questionnaire as valid. For the interview responses, all the responses were adequate. Thus, all the six interview responses were considered for assessing the developer skill profiles. Furthermore, we perceived the responses received through interviews as more concrete than the responses obtained from the questionnaire. We perceived the responses in such a way because the questionnaire does not involve a direct conversation with the respondents. Also, the number of valid recruiter’s responses are more from the interview than the valid recruiter responses from the questionnaire. But this does not mean that the responses received through the questionnaire as invalid or irrelevant because we did validate the responses we received through the questionnaire. We only used the assumed concreteness of the interview responses when there was no unanimous conclusion regarding the assessment of the developer profile along any of the four aspects.

In the following sections, we present the results obtained on four aspects of developer profile evaluation. For each of the four aspects, we present the results interpreted from recruiters responses obtained through interviews and questionnaire respectively. For the ASPI, we also present the results from the developers responses.

6.1.1 ASPI: Developer profile and its attributes validity

In this section, we present the responses we received for two questions used to assess developer profile along the ASPI. For each skill profile in our developer profile, we first present the recruiters responses received from interviews and then we present recruiters responses obtained through the questionnaire. Finally, we present the results based on the responses of the developers without recruitment experience. While providing the results, we simultaneously interpret the validity of each attribute used to indicate skill profile from the perspective of both categories of respondents. Furthermore, we infer the validity of each skill profile as a set of attributes based on the responses received from different kinds of respondents obtained in two ways, i.e., interview and questionnaire.

We followed the process described in the evaluation chapter, for validating the attributes of the skill profile along with each skill profile as a set of attributes based on the received responses. We used the value of 1 for the high irrelevance (or) completely disagree options and value 7 for high relevance or completely agree options. We used the values between one and seven to compute the median values for each attribute relevance and statement agreement for inferring the validity of the skill profiles.

6.1.1.1 Collaboration skill profile and its attributes validity

In this section, we describe the validity of collaboration skill profile and its attributes from the perspective of respondents with recruitment experience obtained through interviews and questionnaire. We then present the validity from the developers perspective based on the responses from the questionnaire. Finally, we conclude this section by discussing the validity
of the collaboration skill profile based on the conclusions drawn from different respondents and obtained in various ways.

**Validity based on the recruiters responses through interviews**  
As described in the evaluation chapter, the first question was aimed to assess the relevance attributes used to indicate the skill of a developer. The table 6.1 indicates the median and variance values responses received for the four attributes we used to indicate collaboration skill.

The attribute “number of projects forked by the developer” received a median value of 2 which is on the negative side of the scale. The variance value of 2.90 can influence the median value towards the positive side of the subjective scale used for inferring the validity of attributes. Thus, we cannot conclude yet that the attribute is invalid because the median value falls on the negative side of the scale. We further investigated the responses received and we found that respondents selected three unique options on the subjective scale. Of them, two choices belonged to the negative category and one option belonged to the positive category. Furthermore, 4 out of 6 respondents selected option belonging to the negative side of the scale. Thus, based on the responses received from interviews, we can conclude that "number of projects forked by the developer” attribute is not a valid attribute for indicating collaboration skill.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Median</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number Of Projects Forked by Developer</td>
<td>2</td>
<td>2.90</td>
</tr>
<tr>
<td>Projects Contributed Back Coverage</td>
<td>5.5</td>
<td>0.96</td>
</tr>
<tr>
<td>Percentage Of Contributions Made</td>
<td>5.5</td>
<td>0.96</td>
</tr>
<tr>
<td>Average Number of Contributors</td>
<td>5</td>
<td>0.4</td>
</tr>
</tbody>
</table>

Table 6.1: Median and variance values for each attribute on relevance to indicate collaboration skill based on recruiters responses from the interview.

The attribute "Projects contributed back coverage” received a median value of 5.5 which falls on the positive side of the scale. The variance of 0.96 also indicates that the attribute is already valid and can indicate the collaboration skill. All the six respondents selected an option that belonged to the positive side of the scale. Furthermore, three unique categories were selected, and all of them belonged to the positive side of the scale. Thus "Projects contributed back coverage” can be considered as a valid attribute based on the interview responses we received from respondents with recruitment experience.

The attribute "Percentage Of Contributions Made” also received a median value of 5.5 which falls on the positive side of the scale. The variance value is also 0.96 thus we can already conclude that the attribute "Percentage Of Contributions Made” is a valid attribute to indicate the collaboration skill. All the six respondents selected an option that belonged to the positive side of the scale. Furthermore, three unique categories were selected, and all of them belonged to the positive side of the scale thus further confirming that "Percentage Of Contributions Made” is a valid attribute to indicate collaboration skill.

The attribute "Average Number of Contributors” received a median value of 5 which is on the positive side of the scale. The variance value of 0.4 which is also not influential
6. Results

enough to change the median value from positive to negative. Thus, ”Average Number of Contributors” is also a valid attribute that can be used to indicate the collaboration skill.

The second question aimed to assess if all the attributes together would indicate the collaboration skill better, the table 6.2 indicates the median and variance value inferred from the respondent’s responses having recruitment experience.

| Validity of set of attributes together indicating collaboration skill better |
|-----------------------------|-----------------------------|
| Median                      | Variance                   |
| 5.5                         | 0.96                       |

Table 6.2: Opinion on set of attributes together would better indicate collaboration skill based on recruiters responses from the interview.

The median value received on the degree of agreement of with the statement is 5.5, and the variance value is 0.96. The median value of 5.5 is on the positive side of the scale, and the variance is also not high enough to influence the median value from positive to the negative side of the scale. Furthermore, all the six respondents selected options on the positive side of the scale. Respondents selected three unique options on the scale where; all the three selected options were on the positive side of the subjective scale. Thus, we can conclude that the set of attributes together mentioned in the skill profile can indicate the collaboration skill better than the mentioned single attribute. Therefore, we can also infer that collaboration skill profile provided with the set of attributes is valid based on the recruiters responses received from the interview.

**Validity based upon the recruiter’s responses from the questionnaire**

The table 6.3 indicates the median and variance values based on the recruiter’s responses received for the four attributes we used to indicate collaboration skill from the questionnaire.

The attribute ”number of projects forked by the developer” received a median value of 5 which is on the positive side of the scale. However, the variance value of 2.3 can influence the median value towards the negative side of the subjective scale. Thus, we cannot conclude yet if the attribute is valid or invalid. We further investigated the responses received and found that the respondents selected four unique options on the subjective scale. Out of four, two options belonged to the positive side of the scale and out of the remaining two, one option each belonged to the neutral and negative side of the scale. Furthermore, three out of five respondents selected options that belonged to the positive side of the scale. Therefore, based on the recruiter’s responses received from the questionnaire, we can conclude that ”number of projects forked by the developer” attribute is a valid attribute for indicating collaboration skill.

The attribute ”Projects contributed back coverage” received a median value of 6 which is on the positive side of the scale. The variance of 1.7 could take the median value close to the neutral side of the subjective scale. Thus, we further analysed the responses received and found that 4 out of 5 respondents selected an option that belongs to the positive side of the scale. Furthermore, three unique options on the scale were selected, and two out of the three options belonged to the positive side of the scale. Thus ”Projects contributed back
6.1. Software Developer Profile Assessment Results

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Median</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number Of Projects Forked by Developer</td>
<td>5</td>
<td>2.3</td>
</tr>
<tr>
<td>Projects Contributed Back Coverage</td>
<td>6</td>
<td>1.7</td>
</tr>
<tr>
<td>Percentage Of Contributions Made</td>
<td>6</td>
<td>0.3</td>
</tr>
<tr>
<td>Average Number of Contributors</td>
<td>6</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 6.3: Median and variance values for each attribute on relevance to indicate collaboration skill based on recruiters responses from questionnaire.

"Projects Contributed Back Coverage" can be considered as a valid attribute based on the recruiter’s responses from the questionnaire.

The attribute "Percentage Of Contributions Made" also received a median value of 6 which is on the positive side of the scale. The variance value was only 0.3 based on this we can already conclude that the attribute "Percentage Of Contributions Made" is a valid attribute to indicate the collaboration skill. All the five respondents selected an option that belonged to the positive side of the scale. Furthermore, two unique options on the scale were selected, and all of them belonged to the positive side of the scale. Therefore, "Percentage Of Contributions Made" is a valid attribute to indicate collaboration skill based on recruiter’s responses from the questionnaire.

The attribute "Average Number of Contributors" received a median value of 6 which is on the positive side of the scale. However, the variance value of 3 is influential enough to change the median value from positive to the negative side of the scale. Thus, we further analysed the responses received from the respondents on "Average Number of Contributors". We found that three unique options on the subjective scale were selected. Two out of three selected options were on the positive side of the scale, and one option belonged to the negative side of the scale. Furthermore, 4 out of 5 respondents selected options that are on the positive side of the scale. Therefore, based on the further analysis of the recruiter’s responses from the questionnaire we can conclude that "Average Number of Contributors" is also a valid attribute to indicate the collaboration skill.

For the second question if all the attributes together would indicate the collaboration skill better the table 6.4 indicates the median and variance based on recruiter’s responses from the questionnaire.

<table>
<thead>
<tr>
<th>Validity of set of attributes together indicating collaboration skill better</th>
<th>Median</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>6</td>
<td>0.7</td>
</tr>
</tbody>
</table>

Table 6.4: Opinion on set of attributes together would better indicate collaboration skill based on recruiters responses from the questionnaire.

The median value of the degree of agreement of with the statement is 6, and the variance value is 0.7. The median value of 6 is on the positive side of the scale, and the variance is also not high enough to influence the median value from positive to the negative side of
6. RESULTS

the scale. Furthermore, all the five respondents selected options on the positive side of the scale. Respondents selected three unique options on the scale where; all the three selected options were on the positive side of the subjective scale. Thus, we can conclude that the set of attributes together mentioned in the skill profile can indicate the collaboration skill better than a single attribute. Therefore, we can also infer that collaboration skill profile with the set of attributes is valid, based on the recruiter’s responses from the questionnaire.

Validity based upon the developer’s responses from the questionnaire  The table 6.5 indicates the median and variance values based on the developer’s responses received for the four attributes; we used to indicate collaboration skill from the questionnaire.

The attribute “number of projects forked by the developer” received a median value of 5 which is on the positive side of the scale. However, the variance value of 3.28 could influence the median value to the negative side of the relevance subjective scale. Thus, we cannot conclude yet that the attribute is valid or invalid. We further investigated the responses received and we found that respondents selected five unique options on the subjective scale. Out of 5, two options belonged to the positive side of the scale, and the remaining three options belonged to the negative side of the scale. Furthermore, 9 out of 15 respondents selected option belonging to the positive category and the remaining 6 selected options on the negative side of the scale. Therefore, based on the responses received from developers, we cannot profoundly conclude if “number of projects forked by the developer” attribute is a valid or invalid option. The developer’s responses appear to be divided regarding the validity of this attribute.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Median</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number Of Projects Forked by Developer</td>
<td>5</td>
<td>3.28</td>
</tr>
<tr>
<td>Projects Contributed Back Coverage</td>
<td>6</td>
<td>1.5</td>
</tr>
<tr>
<td>Percentage Of Contributions Made</td>
<td>6</td>
<td>1.12</td>
</tr>
<tr>
<td>Average Number of Contributors</td>
<td>5</td>
<td>3.26</td>
</tr>
</tbody>
</table>

Table 6.5: Median and variance values for each attribute on relevance to indicate collaboration skill based on developers responses from questionnaire

The attribute “Projects contributed back coverage” received a median value of 6 which is on the positive side of the scale. The variance of 1.5 could take the median value close to the neutral side of the subjective scale. Thus, we further analysed the responses received and found that 14 out of 15 respondents selected an option that belong to the positive side of the scale. Furthermore, four unique options on the scale were selected, and three out of four selected options belonged to the positive side of the scale. The remaining one option belonged to the negative side of the scale. Therefore, “Projects contributed back coverage” can be considered as a valid attribute based on the developer’s responses from the questionnaire.

The attribute “Percentage Of Contributions Made” also received a median value of 6 which is on the positive side of the subjective scale. The variance value is only 1.12 and
we can already conclude that the attribute "Percentage Of Contributions Made" is a valid attribute to indicate the collaboration skill. Out of 15 respondents, 12 selected an option that belongs to the positive side of the scale. Out of the remaining three, two selected a neutral option and one selected a negative option. Furthermore, five unique options on the scale were selected and of them three options belonged to the positive side of the scale. Out of the remaining two, one belonged to the neutral side, and one belonged to the negative side of the scale. Therefore, based on developer’s responses we can conclude that "Percentage Of Contributions Made" is a valid attribute to indicate collaboration skill.

The attribute "Average Number of Contributors" received a relevance of 5 which is on the positive side of the scale. The variance value received was 3.26 which is influential enough to change the median value from positive to the negative side of the scale. Thus, we further analysed the responses received from the respondents on "Average Number of Contributors". We found that five unique options on the scale were selected, and three out of the five options are on the positive side of the scale and the remaining two were on the negative side. Furthermore, 12 out of 15 respondents selected options that are on the positive side of the scale and the remaining three selected options belonging to the negative side of the scale. Therefore, based on the further analysis of the developer’s responses, we can conclude that "Average Number of Contributors" is also a valid attribute to indicate the collaboration skill.

For the second question if all the attributes together would indicate the collaboration skill better the table 6.6 indicates the median and variance value based on the developer’s responses from the questionnaire.

Table 6.6: Opinion on set of attributes together would better indicate collaboration skill based on developers responses from the questionnaire.

<table>
<thead>
<tr>
<th>Validity of set of attributes together indicating collaboration skill better</th>
<th>Median</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>6</td>
<td>2.12</td>
</tr>
</tbody>
</table>

The median value of the degree of agreement of with the statement is 6, and the variance value is 2.12. The median value of 6 is on the positive side of the scale, and the variance value can influence the median value from positive to the negative side of the scale. Thus, we further analysed the responses of the developers. Out of the 15 respondents, 13 selected options belonging the positive side and the remaining two selected options belonging to the negative side of the scale. Furthermore, respondents selected five unique options on the scale where, three selected options were on the positive side of the subjective scale, and the remaining two were on the negative side of the scale. Thus, we can conclude that the set of attributes together in the collaboration skill profile can indicate the skill better than a single attribute. Therefore, we can also infer that collaboration skill profile with the set of attributes is valid based on the developer’s responses from the questionnaire.

Conclusions on collaboration skill profile  We now infer final conclusions on the collaboration skill profile and its attributes based on the validity interpretations from the recruiter’s and developers responses through the interview and questionnaire. In all the three consid-
6. **Results**

Refined sets of responses, three out of the four attributes in the skill profile were concluded as valid attributes. The attribute "Number of projects forked by the developer" was considered invalid based on the recruiter’s responses from the interview. However, the attribute "Number of projects forked by the developer" was concluded to be valid based on the recruiter’s responses from the questionnaire. The developers opinion on the validity of this attribute was not conclusive. Considering three different conclusions were drawn from the three sets of analysed responses, we decided to use the most profound set of responses out of the three sets of responses. The recruiter’s responses from the interview were considered more concrete than other sets of responses. Therefore, we conclude that "Number of projects forked by the developer" as an invalid attribute. Regarding the validity of the collaboration skill profile together as a set of attributes, all the three sets of responses concluded the collaboration skill profile as a valid skill profile. Therefore, we can conclude that the collaboration skill profile with all the attributes together can be considered as valid.

6.1.1.2 **Project management skill profile validity**

In this section, we describe the validity of project management skill profile and its attributes from the perspective of respondents with recruitment experience obtained through interviews and questionnaire. We then present the validity from the developers perspective based on the responses from the questionnaire. Finally, we conclude this section by discussing the validity of the project management skill profile based on the conclusions drawn from different respondents and obtained in various ways.

**Validity based on the recruiter’s responses from the interview**

The table 6.7 indicates the median and variance values for the five attributes we used to indicate project management skill, based on the recruiter’s responses from the interview.

The attribute "Scope To Manage Projects" received a median value of 5.5 which is on the positive side of the scale. However, the variance value of 2.5 can influence the median value towards the negative side of the subjective scale. Thus, we cannot conclude yet that the attribute is valid or invalid. We further investigated the responses received and we found that respondents selected four unique options on the subjective scale. Out of four options, two options belonged to the positive side of the scale. Out of the remaining two options, one option belonged to the negative side of the scale and the remaining one belonged to the neutral side of the scale. Furthermore, four out of six respondents selected options belonging to the positive side of the scale. Out of the remaining 2 one respondent selected an option on the neutral side and the other selected an option on the negative side of the scale. Therefore, based on the majority of the positive recruiter’s responses from the interview, we can conclude that "Scope To Manage Projects” attribute is a valid option to indicate the project management skill.

The attribute "Actually Managed Projects Coverage” received a median value of 6.5 which is on the positive side of the scale. The variance value of 3.9 can influence the median value from positive to the negative side on the subjective scale. Thus, we further analyse responses to decide on the validity of the attribute. Out of the six respondents, four selected an option that belongs to the positive side of the scale and the remaining two selected an
6.1. Software Developer Profile Assessment Results

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Median</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scope To Manage Projects</td>
<td>5.5</td>
<td>2.5</td>
</tr>
<tr>
<td>Actually Managed Projects Coverage</td>
<td>6.5</td>
<td>3.9</td>
</tr>
<tr>
<td>Percentage of Managed Code Contributions</td>
<td>6</td>
<td>3.1</td>
</tr>
<tr>
<td>Average Number Of code Contribution Managers</td>
<td>5.5</td>
<td>4.5</td>
</tr>
<tr>
<td>Users Managed</td>
<td>6</td>
<td>5.36</td>
</tr>
</tbody>
</table>

Table 6.7: Median and variance values for each attribute, on relevance to indicate project management skill based on recruiter’s responses from the interview.

option on the negative side of the scale. Furthermore, three unique categories were selected and out of them 2 belonged to the positive side of the scale and one belonged to the negative side of the scale. Therefore, "Actually Managed Projects Coverage" can be considered as a valid attribute based on the majority of the positive recruiter’s responses from the interview.

The attribute "Percentage of Managed Code Contributions" received a median value of 6 which is on the positive side of the scale. The variance value is 3.1 which can influence the median value to the negative side of the subjective scale. Thus, we further analysed the responses to decide on the validity of the attribute. Out of the six respondents, four selected an option that belonged to the positive side of the scale. Out of the remaining two, one selected option belonging to the negative side of the scale. The remaining one respondent selected an option belonging to the neutral side of the scale. Furthermore, four unique categories were selected, out of them two options belonged to the positive side of the scale. Out of the remaining two, one option belonged to the negative side and the other remaining selected option belonged to the neutral side of the scale. Therefore, based on the majority of the positive recruiter’s responses to the attribute, we can infer that "Percentage of Contributions Made" is a valid attribute to indicate project management skill.

The attribute "Average Number Of Code Contribution Managers" received a median value of 5.5 which is on the positive side of the subjective relevance scale. The variance value of 4.5 can influence the median value from positive to the negative side of the scale. Thus, we further analysed the responses to decide on the validity. Out of the six respondents, 4 selected an option that belongs to the positive side of the scale. Out of the two remaining, one selected negative and the other selected option belonged to the neutral side of the scale. Furthermore, respondents selected five options on the scale. Out of the five, three options belonged to the positive side of the scale. Out of the remaining two, one option belonged to the negative side and the other remaining option belonged to the neutral side of the scale. Therefore, based on the majority of respondents selecting positive options on the scale, we can infer that "Average Number Of Code Contribution Managers" is a valid attribute to indicate project management skill.

The attribute "Users Managed" received a relevance of 6 which is on the positive side of the scale. The variance value of 5.36 can influence the median value from positive to the negative side of the relevance scale. Thus, we further analysed the responses to decide on the validity of the attribute. Out of the six respondents, 4 selected an option that belonged to the positive side of the scale. Out of the remaining two, one selected the negative option on the
scale and the other selected option belonging to the neutral side of the scale. Furthermore, four unique categories were selected and out of them two options belonged to the positive side of the scale. Out of the remaining two, one option belonged to the negative side and the other remaining option belonged to the neutral side of the scale. Therefore, based on the majority of the respondents selecting positive options on the scale, we can infer that "Users Managed" is a valid attribute to indicate project management skill.

For the second question if all the attributes together would indicate the project management skill better, the table 6.8 indicates the median and the variance in responses.

| Validity of set of attributes together indicating project management skill better |
|----------------|----------------|
| Median   | Variance   |
| 6.5      | 1.36       |

Table 6.8: Opinion on set of attributes together would better indicate project management skill based on recruiters responses from the interview.

The median value of the degree of agreement of with the statement is 6.5 and the variance value is 1.36. The median value of 6.5 is on the positive side of the scale and the variance can influence the median value from positive to the neutral side of the scale. Thus, we further analysed the responses of the developers. Out of the six respondents, 5 selected options on the positive side and the remaining one selected the option on the neutral side of the scale. Respondents selected three unique options on the scale where, 2 selected options were on the positive side of the subjective scale and the remaining one was on the neutral side of the scale. Therefore, the majority of respondents selecting positive options on the scale. Based on this, we can conclude that the set of attributes together mentioned in the skill profile can indicate the project management skill better than a single attribute. We can also infer that project management skill profile with the set of attributes is valid based on the recruiters responses from the interview.

Validity based on the recruiter’s responses from the questionnaire

The table 6.11 indicates the median and variance values for the five attributes we used to indicate project management skill, based on the recruiter’s responses from the questionnaire.

The attribute "Scope To Manage Projects" received a median value of 5 which is on the positive side of the scale. The variance value of 5.2 can influence the median value from positive to the negative side of the subjective relevance scale. Thus, we cannot conclude yet if the attribute is valid or invalid. Thus, we further investigated the responses received and found that respondents selected four unique options on the relevance subjective scale. Out of four, three options belonged to the positive side of the scale and one option belonged to the negative side of the scale. Furthermore, 4 out of 5 respondents selected option belonging to the positive side of the scale and one selected an option on the negative side of the scale. Therefore, based on the majority of positive responses received, we can conclude that "Scope To Manage Projects" attribute is a valid option to indicate the project management skill.
### 6.1. Software Developer Profile Assessment Results

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Median</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scope To Manage Projects</td>
<td>5</td>
<td>5.2</td>
</tr>
<tr>
<td>Actually Managed Projects Coverage</td>
<td>4</td>
<td>4.5</td>
</tr>
<tr>
<td>Percentage of Managed Code Contributions</td>
<td>5</td>
<td>3.2</td>
</tr>
<tr>
<td>Average Number Of Code Contribution Managers</td>
<td>6</td>
<td>5.0</td>
</tr>
<tr>
<td>Users Managed</td>
<td>6</td>
<td>5.7</td>
</tr>
</tbody>
</table>

Table 6.9: Median and variance values for each attribute, on relevance to indicate project management skill based on recruiter’s responses from the questionnaire

The attribute ”Actually Managed Projects Coverage” received a median value of 4 which is on the neutral side of the scale. The variance of 4.5 can influence the median value from neutral to positive or negative side on the scale. Thus, we further analysed responses to decide on the validity of the attribute. Out of the five respondents, two respondents selected an option that belonged to the positive side of the scale. Out of the remaining three, two selected option on the negative side of the scale and one selected an option on the neutral side of the scale. Out of 4 unique categories that were selected, 2 of them belonged to the negative side of the scale and one belonged to positive and one belong to the neutral side of the scale. There was no majority for either the positive or negative options on the relevance scale. Therefore, ”Actually Managed Projects Coverage” cannot be considered as valid or invalid based on the recruiter’s responses from the questionnaire.

The attribute ”Percentage of Managed Code Contributions” received a median value of 5 which is on the positive side of the scale. The variance value is 3.2 which can influence the median value to the negative side of the subjective relevance scale. Thus, we further analysed the responses to decide on the validity. Out of the five respondents, four selected an option that belonged to the positive side of the scale and one selected option belonging to the negative side of the scale. Furthermore, only two unique categories were selected and out of them one option belonged to the positive side of the scale and the other option belonged to the negative side. Therefore, since the majority of respondents selected an option on the positive side of the relevance scale, we can infer that ”Percentage Of Contributions Made” is a valid attribute to indicate project management skill.

The attribute ”Average Number Of Code Contribution Managers” received a median value of 6 which is on the positive side of the scale. The variance value is 5.0 which can influence the median value from positive to the negative side of the scale. Thus, we further analysed the responses to decide on the validity. Out of the five respondents, 4 selected an option that belongs to the positive side of the scale and the remaining one selected option belonging to the negative side of the scale. Furthermore, only two unique categories were selected and out of them one option belonged to the positive side of the scale and one option belonged to the negative side of the scale. The majority of respondents selected an option on the positive side of the relevance scale. Therefore, we can infer that ”Average Number Of Code Contribution Managers” is a valid attribute to indicate project management skill.

The attribute ”Users Managed” received a median value of 6 which is on the positive side of the scale. The variance value is 5.7 which can influence the median value from
RESULTS

positive to the negative side of the scale. Thus, we further analysed the responses to decide on the validity. Out of the five respondents, 4 selected an option that belongs to the positive side of the scale and the remaining one selected an option belonging to the negative side of the scale. Furthermore, three unique categories were selected and out of them two options belonged to the positive side of the scale and the remaining one option belonged to the negative side of the scale. Therefore, since the majority of respondents selected options on the positive side of the relevance scale, we can infer that "Users Managed" is a valid attribute to indicate project management skill.

For the second question if all the attributes together would indicate the project management skill better [6.10] indicates the median and the variance for the responses.

<table>
<thead>
<tr>
<th>Validity of set of attributes together indicating project management skill better</th>
<th>Median</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5</td>
<td>0.5</td>
</tr>
</tbody>
</table>

Table 6.10: Opinion on set of attributes together would better indicate project management skill based on recruiters responses from the questionnaire.

The median value of the responses on the degree of agreement with the statement is 5 and the variance value is 0.5. The median value of 5 is on the positive side of the scale and the variance is small, but the variance value can influence the median value close to the neutral side of the scale. Thus, we further analysed the responses of the developers. Out of the five respondents, 4 selected options on the positive side and the remaining one selected option on the neutral side of the scale. Respondents selected three unique options on the scale where, the two selected options were on the positive side of the subjective scale and the remaining one was on the neutral side of the scale. Therefore, the set of attributes together mentioned in the skill profile can indicate the project management skill better than a single attribute. We can also infer that project management skill profile with the set of attributes is valid based on the responses received from questionnaire from the recruiters.

Validity based on the developer’s responses from the questionnaire

The table 6.11 indicates the median and variance values based on the responses received for the four attributes we used to indicate project management skill from the recruiters obtained through the questionnaire.

The attribute "Scope To Manage Projects" received a median value of 6 which is on the positive side of the scale. The variance value of 1.23 can influence the median value closer to the neutral side of the relevance subjective scale. Thus, we further investigated the responses received and found that respondents selected five unique options on the subjective scale. Out of five, three options belonged to the positive side of the scale and one option belonged to the negative side of the scale and one option belonged to the neutral side of the scale. Furthermore, 13 out of 15 respondents selected option belonging to the positive side of the scale. Out of the remaining two one selected option on the negative side and another one selected an option on the neutral side of the scale. Thus, the majority of the developer’s responses from questionnaire selected options on the positive side of the scale. Therefore,
we can conclude that "Scope To Manage Projects" attribute is a valid option to indicate the project management skill.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Median</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scope To Manage Projects</td>
<td>6</td>
<td>1.23</td>
</tr>
<tr>
<td>Actually Managed Projects Coverage</td>
<td>6</td>
<td>1.68</td>
</tr>
<tr>
<td>Percentage of Managed Code Contributions</td>
<td>5</td>
<td>1.09</td>
</tr>
<tr>
<td>Average Number Of Code Contribution Managers</td>
<td>5</td>
<td>1.42</td>
</tr>
<tr>
<td>Users Managed</td>
<td>6</td>
<td>2.11</td>
</tr>
</tbody>
</table>

Table 6.11: Median and variance values for each attribute on, relevance to indicate project management skill based on developer’s responses from the questionnaire.

The attribute "Actually Managed Projects Coverage" received a median value of 6 which is on the positive side of the scale. The variance of 1.68 can influence the median value towards the negative side of the relevance scale. Thus, we further investigated the responses. Out of the 15 respondents, 12 selected an option that belonged to the positive side of the scale. Out of the remaining three, two respondents selected an option on the negative side of the scale and one selected an option on the neutral side of the scale. Furthermore, out of 5 unique options that were selected, 3 of them belonged to the positive side of the scale and one belonged to negative and another remaining one belong to the neutral side of the scale. Therefore, since the majority of the developer’s responses from the questionnaire selected positive options on the relevance scale, we can conclude that "Actually Managed Projects Coverage" is valid to indicate project management skill.

The attribute "Percentage of Managed Code Contributions" received a median value of 5 which is on the positive side of the scale. The variance value is 1.09 which can influence the median value to the negative side of the subjective relevance scale. Thus, we further analysed the responses to decide on the validity. Out of the 15 respondents, 13 selected an option that belongs to the positive side of the scale. Out of the remaining two, one selected option belonging to the negative side of the scale and other remaining one selected an option belonging to the neutral side of the scale. Furthermore, five unique options were selected and out of them three options belonged to the positive side of the scale. Out of the remaining two, one option belonged to the negative side and another one option belonged to the neutral side of the scale. Thus, the majority of the developer’s responses from questionnaire selected options on the positive side of the relevance scale. Therefore, we can infer that "Percentage Of Contributions Made” is a valid attribute to indicate project management skill.

The attribute "Average Number Of Code Contribution Managers" received a median value of 5 which is on the positive side of the scale. The variance value is 1.42 which can influence the median value from positive to the negative side of the scale. Thus, we further analysed the responses to decide on the validity. Out of the 15 respondents, 10 selected an option that belongs to the positive side of the scale. Out of the remaining 5, four selected an option belonging to the neutral side of the scale and the remaining 2 selected an option belonging to the negative side of the scale. Furthermore, five unique options were selected
and out of them three options belonged to the positive side of the scale. Out of the remaining two, one option belonged to the negative side of the scale and another one belonged to the neutral side of the scale. Thus, the majority of developer’s respondents selected options on the positive side of the scale. Therefore, we can infer that “Average Number Of Code Contribution Managers” is a valid attribute to indicate project management skill.

The attribute “Users Managed” received a median value of 6 which is on the positive side of the scale. The variance value of 2.11 can influence the median value from positive to the negative side of the scale. Thus, we further analysed the responses to decide on the validity. Out of the 15 respondents, 12 selected an option that belongs to the positive side of the scale. Out of the remaining 3, 2 selected an option belonging to the negative side of the scale and one selected an option belonging to the neutral side of the scale. Furthermore, six unique options were selected and out of them three options belonged to the positive side of the scale. Out of the remaining three, two options belonged to the negative side of the scale and the remaining one option belonged to the neutral side of the scale. Therefore, since the majority of the respondents selected an option that belonged to the positive side of the scale, we can infer that “Users Managed” is a valid attribute to indicate project management skill.

For the second question if all the attributes together would indicate the project management skill better, the table 6.12 indicates the median and the variance in responses.

| Validity of set of attributes together indicating project management skill better |
|--------------------------|------------------|
| Median                  | Variance        |
| 6                       | 1.45            |

Table 6.12: Opinion on set of attributes together would better indicate project management skill based on developers responses from the questionnaire.

The median value of the degree of agreement of with the statement is 6 and the variance value is 1.45. The median value of 6 is on the positive side of the scale and the variance can influence the value towards the neutral side of the scale. Thus, we further analysed the responses of the developers. Out of the 15 respondents, 13 selected options on the positive side of the scale. Out of the remaining two, one selected an option on the neutral side of the scale and another one selected an option on the negative side of the scale. Respondents selected five unique options on the scale where, 3 selected options were on the positive side of the subjective scale. Out of the remaining two, one option was on the neutral side of the scale and the other one on the negative side of the scale. The majority of the respondents selected options related to the positive side the of the degree of agreement scale. Therefore, the set of attributes together mentioned in the skill profile can indicate the project management skill better than a single attribute. We can also infer that project management skill profile with the set of attributes is valid based on the responses received from questionnaire from the developers.

Conclusions on project management skill profile We now draw final conclusions on the project management skill profile and its attributes based on the validity interpretations from the recruiter’s and developers responses through the interview and questionnaire. In all the
three considered sets of responses, four out of the five attributes in the skill profile were concluded as valid attributes. The attribute "Actually Managed Projects Coverage" validity was not conclusive based on the developer’s responses from the questionnaire. However, the attribute "Actually Managed Projects Coverage" was concluded to be valid based on the recruiter’s responses from the questionnaire and interview. Considering two different conclusions were drawn from the three sets of analysed responses, we decided to use the most profound set of responses out of the three sets of responses. The recruiter’s responses from the interview were considered more concrete than other sets of responses. Therefore, we conclude that "Actually Managed Projects Coverage" also as the valid attribute. Regarding the validity of the project management skill profile together as a set of attributes, all the three sets of responses concluded the project skill profile as a valid skill profile. Therefore, we can conclude that the project management skill profile with all the attributes together can be considered as valid.

6.1.1.3 Programming language skill and its attributes validity

In this section, we describe the validity of programming language skill profile and its attributes from the perspective of respondents with recruitment experience obtained through interviews and questionnaire. We then present the validity from the developers perspective based on the responses from the questionnaire. Finally, we conclude this section by discussing the validity of the programming language skill profile based on the conclusions drawn from different respondents, obtained using interview and questionnaire.

Validity based on the recruiters responses through interviews

The table 6.13 indicates the median and variance values responses received for the four attributes we used to indicate programming language skill.

The attribute "Percentage of Commits Made in a commit category" received a median value of 5 which is on the positive side of the scale. The variance value of 2.5 can influence the median towards the negative side of the subjective relevance scale. Thus, we cannot conclude yet that the attribute is valid or invalid. We further investigated the responses received and found that respondents selected four unique options on the subjective scale. Out of the four, two options belonged to the positive category and one option each belonged to neutral and negative side of the scale. Furthermore, 4 out of 6 respondents selected options belonging to the positive side of the scale. Out of the remaining two, one respondent selected an option on the neutral side and another one selected an option on the negative side of the subjective scale. Thus, the majority of the recruiter’s responses from the interview selected positive options on the scale. Therefore, we can conclude that "Percentage of Commits Made in a commit category" attribute is a valid attribute for indicating programming language skill.

The attribute "Number of Files" received a median value of 4 which is on the neutral side of the scale. The variance value of four can influence the median value towards either positive or the negative side of the relevance scale. Thus, we further investigated the responses received to decide on the validity. Three out of the six respondents selected options that belonged to the positive side of the scale and the remaining three selected options
6. Results

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Median</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percentage of Commits Made in a commit category</td>
<td>5</td>
<td>2.5</td>
</tr>
<tr>
<td>Number of Files</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Number of lines of changes</td>
<td>4</td>
<td>4.5</td>
</tr>
<tr>
<td>Number of projects</td>
<td>6</td>
<td>3.1</td>
</tr>
<tr>
<td>Average number of stars for the Projects</td>
<td>5.5</td>
<td>1.36</td>
</tr>
</tbody>
</table>

Table 6.13: Median and variance values for each attribute on, relevance to indicate programming language skill based on recruiters responses from the interview.

belonging to the negative side of the scale. Furthermore, respondents selected four unique options on the scale. Out of the four, two options belonged to the positive side of the scale and the remaining two options belonged to the negative side of the scale. Thus, neither options selected on the positive side of the scale or the options selected on the negative side of the scale got the majority from the respondents. Therefore, the validity of "Number of Files" attribute cannot be decided based on the recruiter’s responses from the interview.

The attribute "Number of lines of changes" also received a median value of 4 which is on the neutral side of the scale. The variance value is 4.5 which can again influence the median value to either positive or negative side of the scale. Thus, we decided to analyse further the responses received for this attribute. Out six respondents, 3 selected options that belonged to the positive side of the scale and the remaining 3 selected options belonging to the negative side of the scale. Furthermore, respondents selected five unique options, out of the five three options belonged to the positive side of the scale and the remaining two belonged to the negative side of the scale. Thus, neither options selected on the positive side of the scale or the options selected on the negative side of the scale got the majority from the respondents. Therefore, the validity of "Number of lines of changes" attribute cannot be decided based on the recruiter’s responses from the interview.

The attribute "Number of projects" received a median value of 6 which is on the positive side of the scale. The variance value of 3.1 can influence the median value from positive to the negative side of the scale. Thus, we further investigated the responses received and found that respondents selected four unique options on the subjective relevance scale. Out of the four, two options belonged to the positive side of the scale and out of the remaining two, one option each belonged to neutral and negative side of the scale. Furthermore, 4 out of 6 respondents selected options belonging to the positive side of the scale. Out of the remaining two, one selected an option belonging to the neutral side and another one selected an option belonging to the negative side of the subjective scale. Thus, the majority of the recruiter’s responses from the interview selected options on the positive side of the scale. Therefore, based on the recruiter’s responses, we can conclude that "Number of projects" attribute is a valid attribute for indicating programming language skill.

The attribute "Average number of stars for the Projects" received a median value of 5.5 which is on the positive side of the scale. The variance value of 1.36 can influence the median value towards the negative side of the scale. Thus, we further investigated the responses received and found that respondents selected three unique options from the sub-
6.1. Software Developer Profile Assessment Results

Objective scale. Out of the three, two options belonged to the positive category and remaining one option belonged to the negative side of the scale. Furthermore, five out of the six respondents selected options belonging to the positive side of the scale and the remaining one selected an option that was on the negative side of the subjective scale. Thus, the majority of the recruiter’s responses from the interview selected options on the positive side of the scale. Therefore, based on the recruiter’s responses from the interview, we can conclude that ”Average number of stars for the Projects” attribute is a valid attribute for indicating programming language skill. The second question aimed to assess if all the attributes together would indicate the project management skill better, the table 6.2 indicates the median and variance value inferred from the recruiter’s responses from the interview.

<table>
<thead>
<tr>
<th>Agreement on attributes altogether with commit category indicate skill better</th>
<th>Median</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>6</td>
<td>3.5</td>
</tr>
</tbody>
</table>

Table 6.14: Opinion on set of attributes together with commit category would better indicate programming language skill based on recruiters responses from the interview.

The median value received on the degree of agreement of with the statement is 6, and the variance value is 3.5. The median value of 6 is on the positive side of the scale, and the variance is high enough to influence the median value from positive to the negative side of the scale. Thus, a further analysis of the responses was required. Five out of the six respondents selected options on the positive side and one respondent selected an option belonging to the negative side of the scale. The respondents selected four unique options on the scale where; three selected options were on the positive side, and one was on the negative side of the subjective scale. Thus, the majority of the recruiter’s responses from interview selected options on the positive side of the scale. Therefore, we can conclude that the set of attributes together with the commits category mentioned in the skill profile can indicate the programming language skill better than a single attribute. We can also infer that programming language skill profile with the set of attributes is valid.

Validity based upon the recruiter’s responses from the questionnaire  The table 6.15 indicates the median and variance values based on the responses received for the four attributes we used to indicate programming language skill based on the recruiter’s responses from the questionnaire.

The attribute ”Percentage of Commits Made in a commit category” received a median value of 6 which is on the positive side of the scale. The variance of 1.7 can influence the median value from the positive side to the neutral side of the scale. Thus, we cannot conclude yet that the attribute is valid or invalid. We further investigated the responses received and found that respondents selected three unique options on the subjective scale. Out of the three, two options belonged to the positive side of the scale, and the remaining one option belonged to the negative side of the scale. Furthermore, four out of five respondents selected options belonging to the positive side of the scale and the remaining respondent selected an option on the negative side of the scale. Thus, the majority of the recruiter’s responses
6. Results

from questionnaire selected options on the positive side of the scale. Therefore, based on the recruiter’s responses from the questionnaire, we can conclude that “Percentage of Commits Made in a commit category” attribute is a valid attribute for indicating programming language skill.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Median</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percentage of Commits Made in a commit category</td>
<td>6</td>
<td>1.7</td>
</tr>
<tr>
<td>Number of Files</td>
<td>5</td>
<td>2.3</td>
</tr>
<tr>
<td>Number of lines of changes</td>
<td>5</td>
<td>0.5</td>
</tr>
<tr>
<td>Number of projects</td>
<td>5</td>
<td>1.7</td>
</tr>
<tr>
<td>Average number of stars for the Projects</td>
<td>6</td>
<td>0.3</td>
</tr>
</tbody>
</table>

Table 6.15: Median and variance values for each attribute on, relevance to indicate programming language skill based on recruiters responses from the questionnaire.

The attribute ”Number of Files” received a median value of 5 which is on the positive side of the scale. The variance of 2.3 could influence the median value to the negative side of the scale. Thus, we further analysed the responses received and found that three out of five respondents selected options that belonged to the positive side of the scale. The remaining two selected an option on the negative side of the scale. Furthermore, respondents selected three unique options on the scale, and two out of the three options belonged to the positive side of the scale and one belonged to the negative side of the scale. Thus, the majority of the recruiter’s respondents from questionnaire selected options on the positive side of the scale. Therefore, ”Number of Files” can be considered as a valid attribute based on the recruiter’s responses from the questionnaire.

The attribute ”Number of lines of changes” also received a median value of 5 which is on the positive side of the scale. The variance value of 0.5 could influence the median value close to the neutral side of the scale. Thus, we further investigated the responses received for this attribute. Four out of the five respondents selected options that belonged to the positive side of the scale and one respondent selected an option that is on the neutral side of the scale. Furthermore, respondents selected three unique options on the scale and of them two options belonged to the positive side of the scale and one option belonged to the neutral side of the scale. Thus, the majority of the recruiter’s respondents from the questionnaire selected options on the positive side of the scale. Therefore, we can conclude that ”Number of lines of changes” is a valid attribute based on the recruiter’s responses from the questionnaire.

The attribute ”Average number of stars for the Projects” received a relevance of 6 which is on the positive side of the scale. The variance value of 0.3 which is not influential enough to change the median value from positive to negative. Therefore without any further investigation, we can conclude that ”Average number of stars for the Projects” is a valid attribute to indicate the programming language skill, based on the recruiter’s responses from the questionnaire.

The second question responses regarding if all the attributes together with the commits category would indicate the programming language skill better was analysed. The table
6.1. Software Developer Profile Assessment Results

6.16 indicates the median and variance value based on the recruiter’s responses from the questionnaire.

<table>
<thead>
<tr>
<th>Agreement on attributes altogether with commit category indicate skill better</th>
<th>Median</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5</td>
<td>0.8</td>
</tr>
</tbody>
</table>

Table 6.16: Opinion on set of attributes together with commit category would better indicate programming language skill based on recruiters responses from the questionnaire.

The median value of the degree of agreement of with the statement is 5, and the variance value is 0.8. The median value of five is on the positive side of the scale, and the variance value can influence the median value close to the neutral side of the scale. All the five respondents selected options on the positive side of the scale. Furthermore, respondents selected three unique options on the scale where, all the three selected options were on the positive side of the subjective scale. Thus, the majority of the recruiter’s responses from the questionnaire selected options on the positive side of the scale. Therefore, we can conclude that the set of attributes together with the commits category mentioned in the skill profile can indicate the programming language skill better than a single attribute. We can also infer that programming language skill profile with the set of attributes is valid based on the recruiter’s responses from the questionnaire.

Validity based upon the developer’s responses from the questionnaire The table indicates the median and variance values based on the responses received for the four attributes we used to indicate programming language skill based on the developer’s responses from the questionnaire.

The attribute "Percentage of Commits Made in a commit category" received a median value of 5 which is on the positive side of the scale. The variance value of 2.4 can influence the value towards the negative side of the subjective relevance scale. Thus, we cannot conclude yet that the attribute is valid or invalid. We further investigated the responses received and found that respondents selected six unique options on the subjective scale. Out of the six unique options selected, three options belonged to the positive side of the scale. Out of the remaining three, two options belonged to the negative side of the scale and one option belonged to the neutral side of the scale. Furthermore, 12 out of 15 respondents selected options belonging to the positive side of the scale. Out of the remaining three, two selected options belonging to the negative side of the scale and one selected option on the neutral side of the scale. Thus, the majority of the developer’s responses from the questionnaire selected options on the positive side of the scale. Therefore, based on developer’s responses from the questionnaire, we can conclude that "Percentage of Commits Made in a commit category" attribute is valid to indicate the programming language skill.

The attribute "Number of Files " received a median value of 5 which is on the positive side of the scale. The variance value of 2.3 could influence the median value to the negative side of the scale. Thus, we further analysed the responses received and found that 8 out of 15 respondents selected options belonging to the positive side of the scale. Out of the remaining
6. Results

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Median</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percentage of Commits Made in a commit category</td>
<td>5</td>
<td>2.4</td>
</tr>
<tr>
<td>Number of Files</td>
<td>5</td>
<td>2.3</td>
</tr>
<tr>
<td>Number of lines of changes</td>
<td>5</td>
<td>2.69</td>
</tr>
<tr>
<td>Number of projects</td>
<td>6</td>
<td>2.26</td>
</tr>
<tr>
<td>Average number of stars for the Projects</td>
<td>5</td>
<td>2.92</td>
</tr>
</tbody>
</table>

Table 6.17: Median and variance values for each attribute on, relevance to indicate programming language skill based on developers responses from the questionnaire.

seven, three selected option belonging to the neutral side of the scale. The remaining four selected options belonging to the negative side of the scale. Furthermore, all seven unique options on the scale were selected. Thus, the majority of the developer’s responses from the questionnaire selected options on the positive side of the scale. Therefore, based on the analysis of the responses ”Number of Files” be considered as a valid attribute based on the questionnaire responses from developers.

The attribute ”Number of lines of changes” also received a median value of 5 which is on the positive side of the scale. The variance value of 2.69 can influence the median value to the negative side of the scale. Thus, we further investigated the responses received for this attribute. Out of the 15 respondents, 12 selected options that belong to the positive side of the scale and the remaining three selected options on the negative side of the scale. Furthermore, six unique options on the scale were selected and of them three options are on the positive side of the scale and the other three on the negative side of the scale. Thus, the majority of the developer’s responses from the questionnaire selected options on the positive side of the scale. Therefore, based on developer’s responses from the questionnaire, we can conclude that ”Number of lines of changes” is a valid attribute to indicate programming language skill.

The attribute ”Number of projects” received a median value of 6 which is on the positive side of the scale. The variance value of 2.26 is influential enough to change the median value from positive to the negative side of the scale. Thus, we further analysed the responses received from the respondents on ”Number of projects”. We found that respondents selected six unique options on the scale. Out of the six, three belonged to the positive side of the scale. Out of the remaining three, two were on the negative side and one was on the neutral side of the scale. Furthermore, 12 out of 15 respondents selected options that are on the positive side of the scale. Out of the remaining three, two selected negative options and one selected options on the neutral side of the scale. Thus, the majority of the developer’s responses from the questionnaire selected options on the positive side of the scale. Therefore, based on developer’s responses from the questionnaire, we can conclude that ”Number of projects” is also a valid attribute that can be used to indicate the programming language skill.

The attribute ”Average number of stars for the Projects” received a median value of 5 which is on the positive side of the scale. The variance value of 2.92 is influential enough to change the median value from positive to the negative side of the relevance scale. Thus, we
further analysed the responses received from the respondents on the attribute. We found that respondents selected five unique options on the scale. Out of the five, three options belonged to the positive side of the scale and two belonged to the negative side. Furthermore, 10 out of 15 respondents selected options that are on the positive side of the scale. The remaining five respondents selected options belonging to the negative side of the scale. Thus, the majority of the developer’s responses from the questionnaire selected options on the positive side of the scale. Therefore, based on developer’s responses from the questionnaire, we can conclude that "Average number of stars for the Projects" is also a valid attribute that can be used to indicate the programming language skill.

The second question responses regarding if all the attributes together with the commits category would indicate the programming language skill better were analysed. The table 6.18 indicates the median and variance value based on the developer’s responses from the questionnaire.

| Validity of set of attributes together indicating programming language skill better |
|-----------------------------------------------|-------------------|
| Median | Variance |
| 6 | 1.52 |

Table 6.18: Opinion on set of attributes together with commit category would better indicate programming language skill based on developers responses from the questionnaire.

The median value of the degree of agreement of with the statement is 6, and the variance value is 1.52. The median value of 6 is on the positive side of the scale, and the variance can influence the median value from positive to the neutral side of the scale. Thus, we further analysed the responses of the developers. Out of the 15 respondents, 12 selected options on the positive side of the scale. Out of the remaining three, two selected options on the negative side of the scale and one selected options on the neutral side of the scale. Respondents selected five unique options on the scale where, three selected options were on the positive side of the subjective scale. Out of the remaining two, one was on the negative side and another one was on the neutral side of the scale. Thus, the majority of the developer’s responses from the questionnaire selected options on the positive side of the scale. Therefore, we can conclude that the set of attributes together with the commits category mentioned in the skill profile can indicate the programming language skill better than a single attribute. We can also infer that programming language skill profile with the set of attributes and commit category is valid based on developer’s responses from the questionnaire.

Conclusions on programming language skill profile We now draw final conclusions on the programming language skill profile and its attributes with the commits category. We draw conclusions based on the validity interpretations from the recruiter’s and developers responses through the interview and questionnaire. In all the three considered sets of responses, three out of the five attributes in the skill profile were concluded as valid attributes. The attributes ’Number of files” and ”Number of lines of changes” validity was not conclusive based on the recruiter’s responses from the interview. However, the attributes ”Number
of files” and “Number of lines of changes” was concluded to be valid based on the recruiter’s and developer’s responses from the questionnaire. Considering the validity was not conclusive from the most profound set of responses out of the three sets of responses. The recruiter’s responses from the questionnaire were considered to draw final conclusions on the validity of the attributes. Therefore, we conclude that “Number of files” and “Number of lines of changes” also as the valid attribute. Regarding the validity of the programming language skill profile together with the commits category and the set of attributes, all the three sets of responses concluded the programming language skill profile as a valid skill profile. Therefore, we can conclude that the programming language skill profile with all the attributes together with commit category as valid.

6.1.2 Developer Profile Assessment in the Recruitment Process

In this section, we present and interpret the responses received for the remaining three aspects and derive conclusions from the results obtained along the three aspects.

6.1.2.1 ASP2: Steps in traditional recruitment process recruiters preferred to use the developer profile

In this section, we provide and interpret the responses regarding the ASP2 of assessing the developer profile. The responses received from the interviews are presented and discussed first followed by the responses received from the questionnaire.

As described in the evaluation chapter we asked the respondents the following question to assess developer profile along ASP2:

- Select the stages in a traditional recruitment process, you would prefer to use the proposed developer profile.

Responses from the interview For the questions asked regarding the ASP2, the figure 6.1 indicates the number of responses received for each step in a traditional recruitment process. One recruiter indicated that he cannot decide whether he can use the profile in the indicated steps. Thus, effectively only five respondents indicated the steps developer profile can be used. Recruiters gave step one the highest number of votes regarding the usage of developer profile in the recruitment process. Therefore, we can infer that most of the respondents preferred to use the generated developer profile in assessing whether the candidate have possessed the mentioned skills when compared to other stages.

Responses from the questionnaire For the questions asked regarding the ASP2, the figure 6.2 indicates the number of responses we received from the questionnaire for each step in the traditional recruitment process. Recruiter’s responses from the questionnaire also gave the highest number of votes to the step one of the recruitment process. Therefore, we can infer that most of the respondents preferred to use the generated developer profile in assessing whether the candidate have possessed the mentioned skills when compared to other stages of the recruitment process.
Conclusions

The responses from the interviews and questionnaire suggest that the recruiters felt that the developer profile is more suitable for assessing whether the candidate have possessed the mentioned skills than other stages of the traditional recruitment process. Therefore, we can also infer that the proposed developer profile is a more suitable option when comparing if the software developer have possessed the mentioned skills.

6.1.2.2 ASP3: Ease of Understanding

In this section, we present and discuss the responses received from respondents with recruitment experience regarding the ease of understanding of the developer skill profiles. The responses received from interviews and questionnaire are provided and discussed in the following paragraphs. Finally, we further analyse the results received from the interview and questionnaire to draw final conclusions on the developer profile along the ASP3.

As mentioned in the Evaluation chapter, the responses for the following question is analysed for assessment. The subjective scale options were converted to numerical values to compute median and variance values. The table 6.1.2.2 indicates the median and variance based on the responses received from interviews and questionnaire.

- How easy was it for you to understand the attributes used to infer about each skill?
6. Results

Figure 6.2: Steps in a Traditional recruitment process where respondents preferred to use the developer profile, obtained from the questionnaire

Responses from the interview The median value received for the collaboration skill was 4.5 with a variance value of 3.2. The median value suggests that the recruiters felt the collaboration skill slightly easy. However, the variance value can influence the median value to "very hard" or "very easy". Thus, we further analysed the responses received from interviews. Out of the six respondents, three respondents selected an option that indicates different degrees of easiness to understand. Out of the remaining three, two selected options that indicated different degrees of hardness to understand. The remaining one respondent provided a neutral option. Thus, most of the developers selected an option that indicated different degrees of easiness regarding collaboration skill attribute. Therefore based on the recruiter’s responses we conclude that the collaboration skill profile was slightly easy to understand.

The median value received for the project management skill profile was 4 with a variance value of 3.86. The median value suggests that the recruiters felt the project management skill profile was neither easy or hard for understanding. However, the variance value can influence the median value to "very hard" or "very easy". Thus, we further analysed the responses received from interviews. Out of the six respondents, three respondents selected an option that indicates different degrees of easiness to understand. Out of the remaining all three selected options that indicated different degrees of hardness for the developer profile understanding. Since respondents selected hard or easiness options equally, we cannot conclude that the collaboration skill profile was easy or hard for understanding.
The median value received for the programming language skill profile was 3 with a variance value of 4. The median value suggests that the recruiters felt the programming language skill profile was slightly hard for understanding the profile. However, the variance value can influence the median value to "very easy" side of the subjective scale. Thus, we further analysed the responses received from interviews. Out of the six respondents, three respondents selected an option that indicates different degrees of hardness to understand the profile. Out of the remaining three, 2 selected option that indicates the profile as neither easy or hard and one selected an option that indicates that understanding of the profile is easy. Thus, most of the respondents selected an option that indicates ease of understanding of the skill profile as hard. Therefore based on these responses, we conclude that the programming language skill profile was slightly hard to understand.

<table>
<thead>
<tr>
<th>Skill Profile</th>
<th>Responses from Interview</th>
<th>Responses from Questionnaire</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collaboration Skill</td>
<td>4.5</td>
<td>5</td>
</tr>
<tr>
<td>Project Management Skill</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>Programming Language Skills</td>
<td>3</td>
<td>5</td>
</tr>
</tbody>
</table>

Table 6.19: Median and variance values for each skill profile on ease of understanding based on recruiters responses from the interview and questionnaire.

Responses from the questionnaire The median value received for the collaboration skill was 5 with a variance value of 0.2. The median value suggests that the recruiters felt the collaboration skill slightly easy. All the five respondents selected an option that indicated different degrees of easiness to understand. Thus, most of the respondents selected options that indicate it was easy to understand the skill profile although the degree of easiness selected was not too high. Therefore, based on responses, we conclude that the collaboration skill profile was slightly easy to understand for the recruiters.

The median value received for the project management skill profile was 3 with a variance value of 4.3. The median value suggests that the recruiters felt the project management skill profile was slightly hard for understanding the profile. However, the variance value can influence the median value to "very easy" to understand. Thus, we further analysed the responses received from questionnaire. Out of the five respondents, three respondents selected an option that indicates different degrees of hardness for understand the profile. The remaining two selected an option that indicates the profile as easy to understand. Thus, most of the respondents selected an option that indicates ease of understanding of the profile as hard. Therefore based on these responses, we conclude that the project management skill profile was slightly hard to understand.

The median value received for the programming language skill was 5 with a variance value of 3.2. The median value suggests that the recruiters felt the project management skill profile slightly easy to understand. However, the variance value can influence the median value to "very hard" or "very easy". Thus, we further analysed the responses received from
6. Results

questionnaire. Out of the five respondents, three respondents selected an option that indicates different degrees of easiness to understand. The remaining two selected options that indicted different degrees of hardness to understand. Thus, the majority of respondents selected an option that belonged to easiness for understanding the programming language skill profile. Therefore, based on the recruiter’s responses, we conclude that the programming language skill profile was slightly easy to understand.

Conclusions on ease of understanding Based on the conclusions drawn from responses collected in both ways, we can further make the following conclusions on the ease of understanding of the three skill profiles. Regarding the collaboration skill profile, it can be concluded that it was slightly easy to understand the skill profile since both sets of responses concluded the same ease of understanding levels. Regarding the project management skill profile the conclusions drawn from the responses are not same. The set of recruiter’s responses from the interview was not conclusive. However, the set of recruiter’s responses from the questionnaire indicate the skill profile as slightly hard to understand the project management profile. Therefore, we conclude that the project management skill profile as slightly hard, since the interview responses that was considered the most concrete set of responses than questionnaire was not conclusive. Regarding the programming language skill profile the conclusions drawn from both the sets of responses were not on the same side of our ease of understanding subjective scale. The responses drawn from questionnaire indicate the skill profile as slightly easy. However, the responses drawn from interviews concluded the ease of understanding as slightly hard. Since we considered responses from the interview as more concrete, we conclude that ease of understanding for our programming language skill profile as slightly hard.

6.1.2.3 ASP4: Preference to use

In this section, we present and discuss the responses received from respondents with recruitment experience regarding the preference to use the developer skill profiles. The responses received from interviews and questionnaire are provided and discussed in the following paragraphs. Finally, we further analyse the results received from the interview and questionnaire to draw final conclusions on the developer skill profiles along the ASP4.

As mentioned in the Evaluation chapter, the responses for the following question is analysed for assessment. The subjective scale options were converted to numerical values to compute median and variance values. The table indicates the median and variance based on the responses received from interviews and questionnaire.

- How much would you prefer to use the developer profile when compared to your way of inferring about skills exhibited by a developer on Github? If you have not tried or do not know other ways then please provide your preference without any comparison.

Responses from the interview The median value received for the collaboration skill was 5 with a variance value of 5.6. The median value suggests that the recruiters slightly preferred to use the collaboration skill profile. However, the variance value can influence the
6.1. Software Developer Profile Assessment Results

median value to ”Strongly Wouldn’t prefer” to ”Strongly prefer”. Thus, we further analysed the responses received from interviews. Out of the six respondents four respondents selected an option that indicates respondents preference to use the collaboration skill profile. All the remaining two respondents selected options that indicate the recruiter’s preference as, ”do not want to use” the skill profile. Thus, most of the respondents selected options that indicated different degrees of preference to use the collaboration skill profile. Therefore, based on recruiter’s responses from the interview we conclude that the collaboration skill profile was slightly preferred to be used.

The median value received for the project management skill profile was also 5 with a variance value of 4.9. The median value suggests that the recruiters slightly preferred to use the project management skill. However, the variance value can influence the median value to ”Strongly Wouldn’t prefer” on the subjective scale. Thus, we further analysed the responses received from interviews. Out of the six respondents, four respondents selected an option that indicates different degrees preferences to use the profile. The remaining two respondents selected options that indicate respondents preference of ”do not want to use” the skill profile. Thus, most of the respondents selected an option that indicated different degrees of preference to use project management skill. Therefore, based on recruiter’s responses from the interview, we conclude that the programming language profile was slightly preferred to be used.

The median value received for the programming language skill profile was 4.5 with a variance value of 7.36. The median value suggests that the recruiters showed little preference to use the programming language skill profile. However, the variance value can influence the median value to ”Strongly Wouldn’t prefer” or ”Strongly prefer”. Thus, we further investigated the responses received from interviews. Out of the six respondents, three respondents selected an option that indicates different degrees of preference to use the skill profile. Out of the remaining three, two selected options that indicate respondents preference of not to use the skill profile. The remaining one respondent selected an option that indicated a neutral opinion on preference to use the skill profile. Thus, most of the respondents selected an option that indicated different degrees of preference to use skill profile. Therefore, based on these responses, we conclude that the programming language skill profile was slightly preferred to be used.

<table>
<thead>
<tr>
<th>Skill Profile</th>
<th>Median</th>
<th>Variance</th>
<th>Median</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collaboration Skill</td>
<td>5</td>
<td>5.6</td>
<td>6</td>
<td>0.7</td>
</tr>
<tr>
<td>Project Management Skill</td>
<td>5</td>
<td>4.9</td>
<td>4</td>
<td>4.2</td>
</tr>
<tr>
<td>Programming Language Skills</td>
<td>4.5</td>
<td>7.36</td>
<td>6</td>
<td>0.5</td>
</tr>
</tbody>
</table>

Table 6.20: Usage preference median values for each skill profile from respondents with recruiting experience from the interview and questionnaire.
6. **RESULTS**

**Responses from the questionnaire** The median value received for the collaboration skill was 6 with a variance value of 0.7. The median value suggests that the recruiters mostly preferred to use the collaboration skill profile. All the five respondents selected options that indicate respondents preference to use collaboration skill profile. Thus, most of the respondents selected options that indicate respondents preference to use skill profile. Therefore, based on responses, we conclude that the collaboration skill profile was mostly preferred to be used by the respondents.

The median value received for the project management skill profile was 4 with a variance value of 4.2. The median value suggests that the recruiters had a neutral opinion on the preference to use the project management skill profile. However, the variance value can influence the median value to "Strongly Wouldn't prefer" or "Strongly prefer" on the subjective scale. Thus, we further analysed the responses received from questionnaire. Out of the five respondents, two respondents selected an option that indicates different degrees of preference to use the skill profile. Out of the remaining three, two selected an option that indicates the neutral opinion of the to use the skill profile. The remaining one respondent did not prefer to use the skill profile. Thus, there is no clear majority of respondents preferring to use the skill profile. Therefore, based on these responses, we conclude that the project management skill profile was neither preferred nor wasn’t preferred to use by the respondents.

The median value received for the programming language skill was 6 with a variance value of 0.5. The median value suggests that the recruiters mostly preferred to use the programming language skill profile. All the five respondents selected options that indicate respondents preference to use programming language skill profile. Thus, most of the developers selected options that indicate respondents preference to use skill profile. Therefore, based on these responses, we conclude that the programming language skill profile was mostly preferred to be used by the respondents.

**Conclusions on preference to use** Based on the conclusions drawn from responses collected from the interview and the questionnaire we can further draw the following conclusions on the preference of use for the developer skill profiles. Regarding the collaboration skill profile, the interview responses concluded slight preference to use profile while the questionnaire responses concluded that recruiters mostly preferred to use the skill profile. Since we considered that interview responses are more concrete than questionnaire responses and have the higher number of valid responses, we conclude that the collaboration skill profile could only attain slight preference of usage from the recruiters. Regarding the project management skill profile, the questionnaire set of responses concluded neutral preference of use for the skill profile. However, the interview set of responses concluded slight preference to use the skill profile. Thus, since we consider the interview responses as more concrete and has one more valid responses than the questionnaire we conclude that the project management skill profile was also slightly preferred by the respondents. Regarding the programming language skill profile the conclusions drawn from both the responses were on the same side of preference to use on the subjective scale. The responses drawn from questionnaire indicate the programming language skill profile was mostly preferred to be used. The responses drawn from interviews concluded slight preference of usage by the
6.1. Software Developer Profile Assessment Results

respondents towards the skill profile. Since we considered the interview responses as more concrete, we conclude that the programming language skill profile as slightly preferred to be used by the recruiters.

6.1.3 Conclusions on the Proposed Developer Profile

In this section, we discuss the proposed developer profile in the context of our research objective using the results obtained through our assessment the four aspects.

Firstly, we could identify a few set of attributes that are valid for inferring three skills from a developer’s GitHub public data. However, during our evaluation we could also identify some improvements that can be made to the existing attributes of each skill profile to provide much better skill profile than the currently proposed. Regarding the collaboration skill profile a recruiter suggested the following improvement.

"I would like to look at the performance of a contributor in relative to the average within the project."

Thus, the recruiter is also interested to see if the developer has more contributions "or" fewer contributions when compared with the average as a baseline.

Regarding the programming language skill one of the recruiter in the interview said that

"I would also like to see the timeline during which the indicated language commits were made."

Furthermore, another recruiter during the interview indicated that

"I would like to see the details of the size of the pull request, it should be short and understandable."

Finally, one common improvement was suggested by the recruiters across all the skill profiles. The recruiter wanted to know what kind of comments are being made by the developer while making the contributions, handling pull requests and discussing about their pull requests.

The developer profile assessment along ASP4 did not result in positive results. According to our conclusions along ASP4, the recruiters selected either slight preference to use (or) neutral preference to use the developer skill profiles. The conclusions from the ASP3 indicates that two of the three developer profile skills were difficult to understand. The remaining one profile was considered slightly easy. Furthermore, the comments we received from the recruiters indicate the incompleteness in the developer skill profiles. Therefore, based on responses received, we could infer that lack of ease of understand and incompleteness in some aspects as the reasons for the recruiters low preference towards the skill profile.

In context of our research objective of reducing the recruiters effort to know about a developer profile, we can derive the following conclusion:

The developer profiles we proposed could only provide a set of attributes that together can indicate three skills of a developer profile. However, the developer profile fell short of providing easy to understand and a complete developer profile, that can completely reduce the recruiters efforts in using GitHub user data for analysing a potential candidate on GitHub.
6. Results

6.2 Accuracy of the recommendation task

In this section, we provide the precision and ranking metric values for our recommendation task, and simultaneously we discuss the accuracy of the recommendation task using the precision and ranking metric values inferred from the CrowdFlower experiment.

6.2.1 Profile Recommendation Precision

We mentioned the conditions under which the relevance values were received, in the evaluation chapter where we described the recommendation task evaluation setup. In this section, we describe the computation process of the precision metric value from the CrowdFlower experiment data. Finally, we interpret the accuracy of the recommendations, based on the precision values obtained from the CrowdFlower experiment.

The results from the experiment setup contained the job advertisement and developer profile pairs. Each pair had an average relevance value ranging from one to four. One represents that lowest level of relevance between the job advertisement and developer profile and four represents the perfect match between job advertisement and recommended developer profile. Precision value was computed using the equation 6.1.

\[
\text{Precision} = \frac{\text{Number Of True Positive}}{\text{Number Of True Positive} + \text{Number Of False Positive}} \quad \text{where,} \quad (6.1)
\]

TruePositive = The items that are recommended and relevant to the user.
FalsePositive = The recommended items, but are not relevant to the user.

The experiment setup we conducted did not ask the contributors for CrowdFlower task for a binary response on relevance. However, the equation 6.1 indicates the need for a binary decision to compute the precision value. Therefore, we assumed and used the following two conditions for categorising the developer profile as a true positive or false positive:

- **Condition 1:** A developer profile was considered relevant or true positive when the average relevance value received through the experiment was greater than 3.

- **Condition 2:** Furthermore, when the developer profile average relevance value was equal to 3 we also considered the variance between the two responses received for categorising the developer profile as a true positive. When the average relevance value was equal to three, we categorised a developer profile as relevant if the variance value was zero for the two responses. Otherwise, the developer profile was categorised as irrelevant or false positive.

By using the two described conditions for each job advertisement, we computed the number of true positive and false positive developer profiles. Since, our recommendation task involved recommending, five developer profiles for each job advertisement, the denominator value in the equation 6.1 would be 5 for all our precision values computation.
6.2. Accuracy of the recommendation task

The figure 6.3 and table 6.21 indicates the precision with which our recommendation strategy recommended developer profiles to job advertisements under the above described true positive conditions. The figure 6.3 also indicates that there was no job advertisement in our 23 job advertisements that received a precision value between 0.46 and 0.59. The mean value for the precision metric was 0.39 which indicates that approximately on average only two developer profiles of the 5 recommended developer profiles were relevant. Out of the 23 job advertisements, four received a precision value of zero. The term "relevant" in this context, is labeled as: At least both of the respondents judging the relevance indicated that the recommended developer had mostly matched the skill requirements of a job advertisement. Furthermore, the gap between the developer profile skills and the required skills in the job advertisement should not be significantly big. Thus, based on the discussed precision values it could be interpreted that our content-based recommendation strategy could not produce effective results.
6.2.2 Ranking Accuracy

In this section, we describe the normalised distance-based performance measure (NDPM) metric computation from the crowd flower experiment data and then interpret the ranking accuracy of the recommendations based on the NDPM metric values. Finally, we also mention how the profile skills related scores we used altogether for ranking performed in the instances when the similarity score was equal. We then used NDPM metric to compared the developer profile rankings generated by our recommendation strategy against the actual rankings generated. We used the average relevance values obtained through our crowd flower experiment that was described in evaluation chapter for each job advertisement. NDPM was computed using the equation 6.2 as mentioned in Herlocker et al. [16].

\[
NDPM = \frac{2C^- + C^u}{2C^i}
\]  

(6.2)

\(C^-\) is the number of contradictory preference relations between the system ranking and the actual ranking. A contradictory preference relation is when the system indicates that item A is more relevant than item B, and the actual ranking is the opposite. \(C^u\) is the number of compatible preference relations, where the actual ranking of item A is higher than item B, but the system ranking has items A and B ranked at equal relevance levels. \(C^i\) is the total number of preferred relationships in the ranking. In other words, pairs of actually ranked items for which one item is rated higher than the other.

NDPM measure value results in 0 if the ranking of developer profiles recommended to a job advertisement are perfect as actual ranking and if the ranking of all the developer profiles are wrong then the NDPM value results in 1. The table 6.22 indicates the kind of NDPM values that we obtained based on our crowd flower experiment over the 23 job advertisements.

<table>
<thead>
<tr>
<th>Normalised Distance-Based Performance Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
</tr>
<tr>
<td>Variance</td>
</tr>
<tr>
<td>Median</td>
</tr>
<tr>
<td>Mode</td>
</tr>
</tbody>
</table>

Table 6.22: Normalised Distance-Based Performance Measure Statistical Values

The table 6.22 indicates the ranking accuracy of the developer profiles when compared with actual ranking based on the average relevance from the crowd flower experiment. The mean value for the NDPM was 0.43 which indicates that the generated rankings of the developer profile for a job advertisement were not accurate enough. The same can be derived based on median and mode values also. Moreover, we found 126 instances in our experiment where instead of the similarity score either the custom score (or) the project management score (or) the collaboration score were used for ranking. Out of 126 instances, there were 101 instances where the contributors clearly indicated the relevance of one profile more over another. Out of those 101 instances, in 62 instances the other three skill
scores together ordered the profile, according to the actual ranking. Each instance could be defined as a pair of recommended developer profiles whose similarity scores were similar.

### 6.2.3 Draw Backs of our recommendation Strategy

In this section we mention the drawbacks of our recommendation strategy along the precision accuracy and the ranking accuracy aspects.

#### 6.2.3.1 Reasons for Low precision Accuracy

Some of the main drawbacks of the content-based recommendation system could also be attributed to our recommendation strategy also. For instance, the Apache Solr default similarity function uses the exact keywords of the job advertisements to compute the relevancy score between the job advertisement and the developer profile by ignoring the meaning of the keywords. Furthermore, before selecting the job advertisements for the experiment, we did not check if the selected job advertisement had at least five relevant developer profiles in our experiment data set. For example, one job advertisement skill requirement was that a developer relevant to job advertisement should know "Web Drupal" technology. However, in our data there was no developer profile that had a keyword named "Web Drupal". Thus, our content-based recommendation strategy could not handle these cases. However, a content-based recommendation strategy that also used the ontologies for the keywords would have handled these cases.

#### 6.2.3.2 Reasons for Low Ranking Accuracy

In our ranking strategy, the first preference was given to similarity value between the job advertisement and the developer profile. Although giving higher preference to the similarity values was a right decision, but the condition of waiting for the Apache Solr relevance scores to tie and then using the custom scores for ranking is the drawback. For instance, a job advertisement contained 15 keywords. Now let us consider a developer profile A contained ten keywords from the job advertisement and another developer profile B contained nine keywords that are contained in the job advertisement and developer profile A. Thus, the one keyword that is not present in the developer profile B, in the job advertisement is only one. In this scenario, developer profile A will receive a higher relevancy score than the developer profile B and our ranking strategy will place the developer A above the developer B. However, in this same context if the developer profile B had a higher custom score than developer profile A and since the missing keyword weight is only one, a user would rank developer profile B above the developer profile A. Thus, this clearly shows our ranking strategy lacks competency to handle these scenarios and resulting in a lower ranking accuracy.

93
Chapter 7

Conclusions and Future Work

In this chapter, we give an overview of the project’s contributions. After the overview, we will reflect on the results in the context of our two research questions. Finally, some ideas for the future work are discussed.

7.1 Contributions

We tried to achieve research objective that resulted in two contributions. The two contributions are as follows:

1. We investigated regarding a valid set of attributes that can be used by the recruiters for inferring the skills exhibited by a developer on GitHub.

2. We investigated the usage of content-based recommendation technique using keywords, for recommending GitHub developer profiles to suit job advertisements, solely based on the skill set requirements mentioned in the job advertisement.

7.2 Discussion

In this section, we discuss the results we obtained in the context of our two research questions along with the threats to validity for our project and results.

7.2.1 GitHub Developer Skill Profiles

The profile we generated to indicate the skills of a developer profile contained an appropriate set of attributes that can together indicate the mentioned skill. All the three sets of responses indicated their agreement that, the set of attributes together can indicate the mentioned skill better than a single attribute. However, the responses we received indicated two major drawbacks in the generated developer profile. The first one was the complexity in understanding the developer profile. In our assessment, two out of the three skill profiles received “Slightly hard” opinion from the respondents along ease of understanding. The other drawback was the incompleteness in the developer profile. The
comments we received from the respondents show that some more aspects should be added to the developer profile to make it complete. For instance, the respondents wanted to see the timeline of the code commits made by the developer in a specific language. Therefore, from the context of our RQ1 we can conclude that we have identified few valid set of attributes to indicate the developer skills. However, the developer skill profile to become usable requires few improvements along the ease of understanding and completeness aspect of the developer profile.

7.2.1 Threats to Validity

The threats to the validity for our generated developer profile and to our assessment results of the developer profile skills.

**Threats to Validity of generated developer profile** The threats to the validity of the developer profile comes from the fact that we used GitHub data for generating the developer profile. Kalliamvakou et al. [18] discussed some perils and promises of mining GitHub data. In case of our generated developer profile, the main perils that apply to our project are as following:

1. If the commits in a pull-request are reworked (in response to comments) GitHub records only the commits that are the result of the peer-review, not the original commits.
2. Most pull requests appear as non-merged even if they are actually merged.

GitHub API is continuously evolving and some of the details that are accessible currently may not be possible to access in the future. The best example for this happened during our thesis project. If a user was a collaborator or not for a repository, was available for public access. However, during changes on GitHub API they removed this information from public API access.

**Threats to Validity for the results of developer profile** The results we obtained during the assessment of our generated developer profile cannot be generalised as the opinion of all the recruiter views or opinions. However, we believe that the insights that we obtained through our assessment would be the opinion of most of the recruiters who wanted to infer about developer skills using GitHub.

7.2.2 Recommending developer profiles to job advertisements

The content-based recommendation strategy we used for recommending developer profiles to job advertisement did not yield good precision because of the semantics missing in the keywords. Furthermore, the developer profile does not provide any details regarding developer’s expertise on software technologies or tools. Even our ranking strategy also did not indicate good results. The main reason for this can be attributed to waiting for the relevance scores to tie and then using a next ranking score in priority for further ranking of
the profiles. Therefore, in the context of our RQ2 we can conclude that the content-based recommendation technique using the keywords to recommend developer profiles to suit job advertisements have not yielded significant accuracy.

**Threats to Validity of recommendations made**  The main threat to the validity of the recommendation contributions was that we made few assumptions for converting the set of attributes values into a single score for a programming language or skill. The assumptions made for computing the scores cannot be true in all the cases because people viewing the profile may have different opinions on the relation between the attributes mentioned. Another threat to validity was that a commit containing a software technology (or) tool keyword was assumed that the developer knows about that technology (or) tool that may not be true in all the cases. Finally another threat to the validity of our recommendation contribution was that we used annotation service of DBpedia for knowing about programming languages or technologies present in a text. Thus, the efficiency of the service may have the effect on the annotations in some cases.

**Threats to validity for the results of recommendations**  The main threat to validity for our results on the recommendation was that we inferred the relevance of the recommendations based on the contributors to the crowd flower tasks. Thus the precision and the ranking accuracy may varied a bit depending on the crowd flower contributors. However, still the effects may not be significant enough to affect the conclusions we drew from the results.

### 7.3 Future work

In this section, we mention some ideas for the future work.

The major future work on the developer profile could be indicating the timeline of code contributions along with commits category, for each programming language a developer have made code contributions. Furthermore, providing a profile to indicate the developer skills on software technologies (or) tools could be useful. Finally, an appropriate visualisation of the skill profile attributes would help the developer profile to understand easily the skill exhibited by a developer.

The future work on recommending the developer profiles to a job advertisement could be about the usage of the ontology for the recommendation. Most of the technical respondents during the questionnaire and interviews indicated that the profile would best be suited to check if developer on GitHub exhibited the skills mentioned or not. Thus, exploring the comparison of the CV or LinkedIn profiles with our generated developer profile could also be useful.
Bibliography


[33] Yuri Takhteyev and Andrew Hilts. Investigating the geography of open source software through github.

Appendix A

Implementation

In this appendix we provide the screen shot of the schema used for indexing developer profiles and job advertisement.

Figure A.1: Schema used for indexing documents
Appendix B

Evaluation

In this appendix we provide the screen shots related to the questionnaire and interviews we used for assessing our generated developer profile.

Figure B.1: Attributes descriptions and questions asked For the collaboration skills
### Attributes to infer about Project Management skills exhibited by the Developer

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Attributes Description</th>
<th>Range</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Right to Manage Projects</td>
<td>Indicates the number of projects for which the developer have right to manage code contributions.</td>
<td>Zero or Positive Integer</td>
<td>1</td>
</tr>
<tr>
<td>Projects Covered</td>
<td>Indicates the percentage of projects the developer have utilized the right to manage contributions from all of the projects the developer have rights to manage.</td>
<td>0% - 100%</td>
<td>50%</td>
</tr>
<tr>
<td>Share of Managed Contributions</td>
<td>Indicates the share of contributions managed by the developer out of all the managed code contributions in the projects that were managed by the developer.</td>
<td>0% - 100%</td>
<td>20%</td>
</tr>
<tr>
<td>Average Number of Contribution Managers</td>
<td>Indicates an average number of users involved in managing contributions to the projects that were managed by the developer.</td>
<td>Zero or Positive Integer</td>
<td>5</td>
</tr>
<tr>
<td>Users Managed</td>
<td>Indicates the number of users managed by the developer that have contributed to the projects that were managed by the developer.</td>
<td>Zero or Positive Integer</td>
<td>10</td>
</tr>
</tbody>
</table>

Each attribute Relevance towards Inferring about Project Management Skills

Select a relevance level in your opinion, regarding each attribute indicating about Project Management skills exhibited by the developer.

<table>
<thead>
<tr>
<th>Scope To Manage Projects</th>
<th>Highly Relevant</th>
<th>Quite Relevant</th>
<th>Slightly Relevant</th>
<th>Neither Relevant or Irrelevant</th>
<th>Slightly Irrelevant</th>
<th>Quite Irrelevant</th>
<th>Highly Irrelevant</th>
</tr>
</thead>
</table>

Figure B.2: Attributes descriptions and questions asked for the project management skills
<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accepted</td>
<td>Accepted commits are the commits that were scrutinized and accepted by the project contributions manager.</td>
</tr>
<tr>
<td>Rejected</td>
<td>Rejected commits are the commits that were scrutinized and rejected by the project contributions manager.</td>
</tr>
<tr>
<td>Under Scrutiny</td>
<td>Under scrutiny commits are the commits that are ready to be either accepted or rejected by project contributions managers.</td>
</tr>
<tr>
<td>Not Released</td>
<td>Not released commits are the commits that were not scrutinized as they were from project contributions managers.</td>
</tr>
</tbody>
</table>

**Attributes used under each category of commits described above**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Attribute Description</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average of Commits Done</td>
<td>Indicates an average the number of commits a developer had, out of all the commits related to a specific category of commits across all the developer projects.</td>
<td>[0% - 100%]</td>
</tr>
<tr>
<td>Number of Files</td>
<td>Indicates the number of unique files edited by a developer for the language under consideration.</td>
<td>Zero or Positive Integer</td>
</tr>
<tr>
<td>Number of Lines of Changes</td>
<td>Includes the number of lines of changes made by a developer in the mentioned number of files.</td>
<td>Zero or Positive Integer</td>
</tr>
<tr>
<td>Average or Minimum of Projects</td>
<td>The data for a project indicates, on average or minimum, among GitHub Users, an average of lines removed for the projects is indicated using the attribute.</td>
<td>Zero or Positive Integer</td>
</tr>
</tbody>
</table>

**Attribute relevance for inferring about the level of programming language skills exhibited by developer.**

Select a relevance level in your opinion, regarding each attribute indicating programming language skills exhibited by a developer.

- **Percentage of Commits Made in a specific category**
- **Number of Files**
- **Number of lines of changes**
- **Number of projects**
- **Average number of stars for the Projects**

*Figure B.3: Attributes descriptions and questions asked for the programming language skills*
Figure B.4: Infographic used for the collaboration skills
Figure B.5: Infographic used for the project management skills
Figure B.6: Infographic used to indicate the programming language skills
Figure B.7: Instructions used to describe the task of the crowdflower evaluation job
B. Evaluation

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Attribute Description</th>
<th>Range</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number Of Projects Forked By Developer</td>
<td>A forked project is a personal copy of another user's project, which allows the developer to freely experiment and contribute back to the original project.</td>
<td>Zero or Positive Integer</td>
<td>6</td>
</tr>
<tr>
<td>Contribution Coverage</td>
<td>Indicates the <strong>percentage</strong> of contributed projects by the developer, out of all forked projects by the developer.</td>
<td>[0%-100%]</td>
<td>60%</td>
</tr>
<tr>
<td>Share of Contributions</td>
<td><strong>Share of contributions</strong> indicate the percentage of contributions by the developer towards contributed forked projects, out of all the contributions to the contributed forked projects.</td>
<td>[0%-100%]</td>
<td>5%</td>
</tr>
<tr>
<td>Average Size of contributors to Forked Projects</td>
<td>Indicates an <strong>average number</strong> of users who have contributed to the projects that were forked and contributed by the developer.</td>
<td>Zero or Positive Integer</td>
<td>20</td>
</tr>
</tbody>
</table>

**Figure B.8:** Instructions used to describe the task of the crowdflower evaluation job
Figure B.9: Instructions used to describe the task of the crowdflower evaluation job

<table>
<thead>
<tr>
<th>Attributes for Project Management Skills of Developer.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Attributes Description</td>
</tr>
<tr>
<td>Right to Manage Projects</td>
</tr>
<tr>
<td>Range</td>
</tr>
<tr>
<td>Zero or Positive Integer</td>
</tr>
<tr>
<td>Projects Managed Coverage</td>
</tr>
<tr>
<td>Range</td>
</tr>
<tr>
<td>0%-100%</td>
</tr>
<tr>
<td>Share of Managed Contributions</td>
</tr>
<tr>
<td>Range</td>
</tr>
<tr>
<td>0%-100%</td>
</tr>
<tr>
<td>Average Number of Contribution Managers</td>
</tr>
<tr>
<td>Range</td>
</tr>
<tr>
<td>Zero or Positive Integer</td>
</tr>
<tr>
<td>Users Managed</td>
</tr>
<tr>
<td>Range</td>
</tr>
<tr>
<td>Zero or Positive Integer</td>
</tr>
</tbody>
</table>

4. Constraints while Judging The Relevance
   - Consider only the skill set requirements from the job advertisement while making relevance judgment.
   - You should ignore the education and location requirements while making relevance judgment.

5. Select a Relevance Level Under the above provided constraints:
   "Inrelevant: Poorly Worded Job Advertisement"
   - It's obvious that the job description doesn't match the job advertisement's skill set requirements at all.
   - Even if there are keywords that profile and skill set is in the job advertisement, there is no common, the contrast and intent of usage is entirely different.
   "Relevant: Obvious Differences"
   - A part of skill set requirements mentioned in the job advertisement matches with the provided developer profile, but there are also differences (e.g., programming languages expertise, Software technologies).
   "Relevant: Possible Differences"
   - There is a slight mismatch between job advertisement skill set requirements and the provided developer profile, but you don't think that it's necessary.
   "Perfect Match"  
   - Profile matches every property specified by the job advertisement skill set, such as programming languages expertise, Software technologies, etc.  
   - Job advertisement is very clearly stated and this is the kind of developer profile the technical recruiter intended to find with respect to job advertisement.

Figure B.10: Job advertisement presented in the task of crowdflower evaluation job
### B. Evaluation

#### Figure B.11: Developer Profile presented in the task of crowdflower evaluation job

<table>
<thead>
<tr>
<th>Collaboration In Public Projects</th>
<th>Others Repositories</th>
<th>Contribution Coverage</th>
<th>Contribution Mean</th>
<th>Contributed Projects Team Mean Size</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2</td>
<td>100.0%</td>
<td>11.490864%</td>
<td>3.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Project Management Skills</th>
<th>Managed Repositories</th>
<th>Managed Coverage</th>
<th>Managed Contribution Mean</th>
<th>Management Team Mean Size</th>
<th>Number Of Users Managed</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0.0%</td>
<td>0.0%</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Job Related Language Skills Of The User</th>
</tr>
</thead>
<tbody>
<tr>
<td>Language Name: objective-c</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Commit Style Category</th>
<th>Average of Commit Share Across Number of Files</th>
<th>With Number Of Changes On Number of Repositories</th>
<th>Average Number Of Stars</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accepted</td>
<td>66.666667%</td>
<td>15.0</td>
<td>65.0</td>
</tr>
<tr>
<td>Rejected</td>
<td>0.0%</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Under Review</td>
<td>0.0%</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Not Reviewed</td>
<td>97.67442%</td>
<td>118.0</td>
<td>1.875</td>
</tr>
</tbody>
</table>

Technologies:
- Other Languages Known By The User: shell, java, ruby, javascript, c, swift, css
- Other Keywords Associated With The User: Interface builder, xcode, git, sinatra, android, eclipse, lint, mercury, source code, c++, markdown, mpi, cat