Project Proposal: Study the knowledge flow into *Stack Overflow* and from *Stack Overflow* to *GitHub*

Wenhan Zhu (Cosmos)
w65zhu@uwaterloo.ca
University of Waterloo
Waterloo, Canada

ABSTRACT

Stack Overflow is the most popular Q&A website about programming. *Stack Overflow* contains large amount of public knowledge that should not be ignored. Many code snippets from answers on *Stack Overflow* have been found in used by projects hosted on *GitHub*. *Stack Overflow* posts consists links to many other websites that can be considered as its sources. In this project we intend to find the knowledge flow of *Stack Overflow* posts.

In this project we explored what are the most popular source of references on *Stack Overflow* and looked at how are *Stack Overflow* code snippets reused on *GitHub*. TODO: Results for RQ 2, 3, 4, 5

KEYWORDS

Stack Overflow, *GitHub*

1 INTRODUCTION

Stack Overflow, since its introduction in 2011, has became the largest Q&A website that focuses on programming. Due to the massive about of answer and questions on the website, there are large amount of public knowledge. Extracting and understanding these information on *Stack Overflow* has been quite popular recently in the Software Engineering community. There has been numerous studies on *Stack Overflow* focusing on different aspects. People are trying to understand how *Stack Overflow* works and are trying to predict the quality answers on *Stack Overflow*. Recently, a new database *SOTorrent* based on the official *Stack Overflow* datadump has been created.[1] This dataset creates new opportunities to learn about *Stack Overflow*

GitHub is the largest code sharing website that hosts *git* repositories. There are many code segments on *GitHub* that originates from *Stack Overflow* posts. There has been multiply efforts trying to study the effects of using *Stack Overflow* code snippets on *GitHub*. Previous studies has shown that code reuse on *GitHub* from *Stack Overflow* could cause security issues. [5]

In this project, we want to look at how information is shared to *Stack Overflow* and how *GitHub* projects reuses code snippets from *Stack Overflow*. We will be trying to answer the following research questions.

(1) RQ1: What information are shared to *Stack Overflow*
(2) RQ2: What projects reuses code from *Stack Overflow*
(3) RQ3: Are *Stack Overflow* code modified when used on *GitHub* projects?
(4) RQ4: Does code snippets introduced from *Stack Overflow* on *GitHub* evolve over time?
(5) RQ5: Does updates to code snippets originated from *Stack Overflow* on *GitHub* triggers updates to the original post?

The rest of this report is formatted as follows, Section 2 will talk about the background information and the dataset; Section 3 will answer the research questions; Section 4 is the conclusion; Section 5 is related work; Section 6 is threats to validity and Section 7 covers future work. Section 8 is the summary.

2 BACKGROUND

To understand the scope of this project, more details about the infrastructure of *Stack Overflow* and *SOTorrent* is needed. And some information about code clone is stated here.

2.1 *Stack Overflow*

2.1.1 Stack Overflow official data dump. In *Stack Overflow*'s data dump, it contains information about all the posts and their history on the website. Other information such as comments in each post and the upvotes and downvotes to a post is also recorded.

In *Stack Overflow* both the question and answer of considered as posts. Each post have a unique *PostId* and their *PostTypeId* determines whether they are a question post or an answer post. An answer post have a *ParentId* that identifies the question they are answering.

2.1.2 SOTorrent. *SOTorrent* is build on top of *Stack Overflow* official data dump. In addition to the contents on *Stack Overflow*'s official data dump. They analyzed the body of each *Stack Overflow* post history and separated them into text and code blocks. Then they created a relationship between post versions so that studying the evolution of each block can be done easily. They also extracted every link in *Stack Overflow* posts and comments and also queried the *Google GitHub Dataset* to look for references of *Stack Overflow* posts on *GitHub*. These information made the study of this project possible. In this project, we are using the Dec. 2018 version of this dataset.

2.2 *GitHub dataset*

Google GitHub Dataset is public dataset that contains repositories from *GitHub*. The documentation of this datasets seems to be lost on the internet, however, earlier blog posts on the subject suggests that this dataset mirrors code that have a clear open source license, ignores unchanged forks and un-notable projects such as empty projects. And it only contain files that are less then 10M. Some basic queries people ran suggests that it contains around 5% of all the repositories. And as of June 2017, around 54% of project with more

than 15 stars are mirrors on the dataset. In the rest of the reports, data from *GitHub* will mean data of *GitHub* in this public dataset.

2.3 Code Clones

When a code snippets is reused on *GitHub* projects, we want to understand how much it was modified. Research in code cloning has identified the type of code clone in 4 different ways. The definition are from *Roy*'s work on comparing code clone tools. [7]

TYPE-1 clone is a exact copy of the code besides white spaces, layout and comments.

TYPE-2 clone is syntactically identical fragments except for variations in identifiers, literals, types and everything above.

TYPE-3 clone is copied fragments with further modifications in addition to everything above.

TYPE-4 clone is two piece of code with same functionality but implemented differently.

Modern code clone tools are very good at TYPE-1, TYPE-2 clones but suffer to succeed in the later too. In this report, we will focus on TYPE-1 clones.

There are 3 major types of clone detection tool, text-based, token-based and tree-based. In this project we will be using basic token-based comparison on *Python* code snippets.

3 CODE REUSE ON *GITHUB*

For RQ3, Are *Stack Overflow* code modified when reused on *GitHub* projects, we want to study when *Stack Overflow* code is referenced in *GitHub* files, how much is modified. We limited our study on *Python* files that have references to *Stack Overflow* in this project. *Python* has a built in lexical tokenize that can tokenize valid *Python* code.

3.1 Method

There is a table in *SOTorrent* that records all references to *Stack Overflow* posts in files on *Google GitHub Dataset*. The table is extracted by matching a regular expression line by line on each file to check whether it matches a *Stack Overflow url*. This dataset as of Dec. 2018 is built soly from the data extracted from the files. The *PostId* and *PostTYpeId* are extracted from the *url* extracted from individual files. Due to this reason, there contains many conflicting information on this dataset. Using this dataset, we extracted all references of *Stack Overflow* posts of *GitHub* files that have an extension of *.py* which is a file extension for *Python* files. This resulted in 126,244 unique references of *Stack Overflow* posts for *Python* files on *GitHub*. We then filtered the results by its *PostTypeId* to only have answer posts. By doing so, we only limit the references to exact answer posts so there is no ambiguity here about which exact post the file is referencing to. Due to the information was not verified with *Stack Overflow* official dump, we cross verified the *PostId* remaining with the official *Stack Overflow* data dump to remove noise from the references. We then retrieved all the files from *GitHub* and ignored files that were not retrievable. This resulted in 42,895 unique references to *Stack Overflow* answer posts in *Python GitHub* files. By manually looking at the files there we were not able to retrieve, we realized that these repositories were either deleted or turned in to private repositories between the version of *SOTorrent* we are using, Dec. 2018, and the time of this project, Mar. 2019.

In order to measure how much code was changed when introducing *Stack Overflow* code segments to *GitHub*, we need to define a metric for measuring the similarity. In this study we are using the token similarity defined as follows. Here, S_{SO}, F_{GH} refers to the code segment on *Stack Overflow* and the file on *GitHub* that contains the reference to the post containing the code segment. T is the tokenizing function that removes all formatting and style tokens such as line break, white spaces and comments. *MaxSubTokens* calculates the maximum matching sub-token of 2 token lists and return the maximum matching sub-token. When dealing with tokens for *Python* code, we matches the exact string from the token and not anything else.

$$Sim(S_{SO}, F_{GH}) = \frac{|MaxSubTokens(T(S_{SO}), T(F_{GH}))|}{|T(S_{SO})|}$$

For each reference we have, we extracted the code segments of the *Stack Overflow* post by using a *HTML parser*. This step was very straight forward due to posts in *Stack Overflow* are written in *Markdown* and every code segment are surrounded by `<code> some code segment </code>`. We then run the *Python* tokenizer on each of the code segments of the post and compare it to the result of tokenizing the *GitHub* file referencing this post. We ignored all reference if either the post does not contain any code segments or if the tokenizer fails on any of the code segments. To combat general matched tokens, we limited the minimum matched token length to be 40. Also, to make sure we are not double counting files in the same repository but different branches, we limited ourselves to the `master` branches.

This gave us a total of 3,490 references that matches our requirements. Some references have more then one code segment in the post that satisfies our constraints, we only consider the one with maximum similarity of all the code segments. This happens only to about 100 of the references. The method we are looking at code reuse from *Stack Overflow* on *GitHub* is by definition looking at TYPE-1 clones of code.

3.2 Results

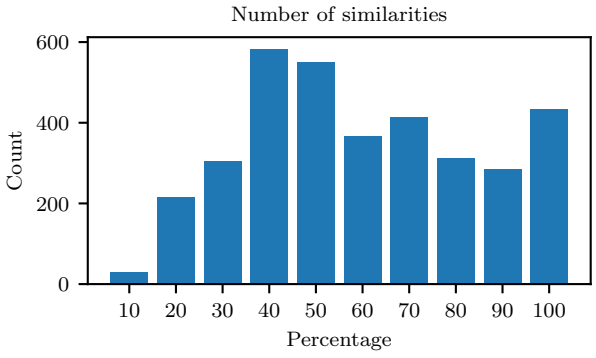

Figure 1: Distribution of similarity percentage

Figure 2 shows the distribution of percentage of similarities between *Stack Overflow* posts and their references on *GitHub*. The mean and median of the distribution are 55.00% and 50.70%.

The results show that on average, when a *Stack Overflow* code snippet is reused on *GitHub* projects, the maximum length of matching sub-token is a little more than the length of the tokens extracted from code segments. So when a reuse happens, on average at least half of the content remains unchanged. To further explored what are the popular changes, we manually looked at some of the references to have a better understanding.

On *Stack Overflow* many *Python* code segments contain additional information about the modules they are importing. These import statements are often separated from the code segment in practice. So when matching the tokens between the two, we will lose similarity percentage due to not having these matched. Some other common reasons are adding statements and removing statements to better fit needs and changing the variable names to match the actual use. Modifying variable names can be taken care of if we move to a TYPE-2 clone detection techniques. Detecting adding, removing statements are much harder. And taking care of the special cases of imports is very *Python* specific, we think if the focus is only *Python* snippets these can be treated, but in general it's very hard to deal with those language specific things.

So for RQ3, Are *Stack Overflow* code modified when used on *GitHub* projects, on average a specific code segments from *Stack Overflow* reused on a *GitHub* file contains a little more than half identical parts.

4 EVOLUTION OF REUSED *STACK OVERFLOW* CODE SNIPPETS

In RQ4, Does code snippets introduced from *Stack Overflow* on *GitHub* evolve over time, we want to see whether code snippets from *Stack Overflow* on *GitHub* are updated over time.

4.1 Method

We took the final 3,490 reference that we find from the previous section and cloned the bare repository from *GitHub*. There were 1 repository that was deleted between when we retrieved the files for the previous step and for this analysis, so we have 3,489 references to work with for this research question.

We first cloned the bare repository for *GitHub* for each of the references. Then using the information from the maximum length sub-tokens of *Stack Overflow* code segments and *GitHub* files, we retrieved the number of lines that contained those tokens. Using this information, we were able to retrieve the history of these lines from the *git* repository we retrieved from *GitHub* by using `git log`.

The `git log` information we retrieved from the repositories gives the information of every commit that touches the aforementioned lines of code. It is also able to trace through the lines modified dynamically so that we will get all the information. The other popular command `git blame` can also do this, but it lacks the ability to recursively trace every commit that modified the lines specified.

4.2 Results

Figure 3 shows the distribution of number of commits for the 3,489 references we have looked at. The majority of the code segments

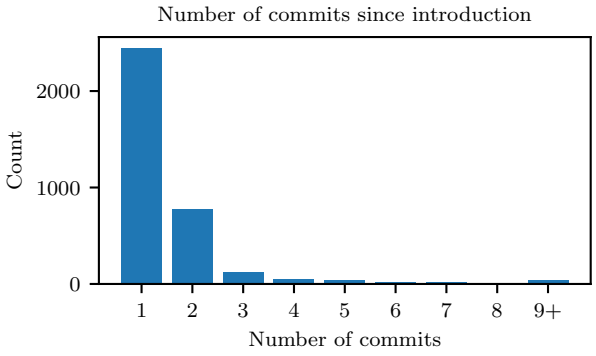

Figure 2: Distribution of number of commits

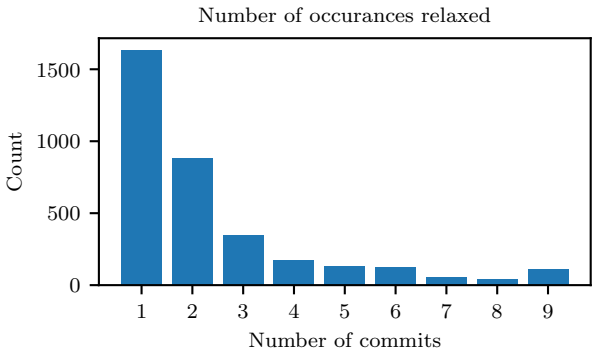

Figure 3: Distribution of number of commits relaxed

never received an update after its introduction. Considering that the results from from the code segments that were matched with files of present and looked back for changes, it is likely that the modified code segments were not captured during token matching. So we retrieved another log with relaxed line numbers both 5 more lines before and after the starting and ending lines from the tokens. Figure 4 captures the distribution of number of commits regarding the relaxed lines.

So for RQ4, does code snippets introduced from *Stack Overflow* on *GitHub* evolve over time? Our results showed that code snippets reused on *GitHub* does not get updated the majority of the time.

5 CO-EVOLUTION OF REUSED CODE SNIPPETS

We investigate RQ5, does updates to code snippets originated from *Stack Overflow* on *GitHub* triggers updates to the original post, in this section. We want to understand whether changes to the code segments reused on *GitHub* co-evolve with the original code snippet from *Stack Overflow*.

5.1 Method

We took the references of *Python* code snippets from the previous section and determined the date of the first commit that introduced the code segment from *Stack Overflow*. We then extracted the number of edits for a post from *Stack Overflow* data dump. Then we compared the time of each edit on *Stack Overflow* and the earliest commit date of the code snippets on *GitHub*.

5.2 Results

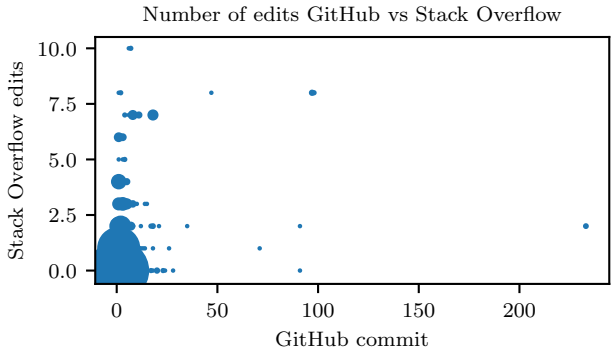

Figure 4: Scatter plot for commits and edits

Figure 5 is the scatter plot of the number of commits since introduction of code snippets on *GitHub* and the number of edits to the *Stack Overflow* post. Each data point is scaled based on the number of points at this location. Besides a few outliers, we can see that most of the code snippets does not get updates either on *Stack Overflow* or *GitHub*. The *Pearson* correlation between the number of commits and edits is 0.2187 with $p = 4.410^{-38}$. This shows that there exist only a weak to very weak correlation between commits on *GitHub* code snippets from *Stack Overflow* and the edits to the original post.

So for RQ5, updates to code snippets reused from *Stack Overflow* on *GitHub* generally does not trigger an edit to the original post.

6 RELATED RESEARCH

Since *Stack Overflow*'s introduction, it has attracted many researchers to work on the subject. Some research wants to understand the model of *Stack Overflow* [2] and why it works with its gamification. [3] Interaction between *Stack Overflow* and other platform has also been looked at. *Vasilescu et al.* looked into common users between *Stack Overflow* and *GitHub*. They discovered that novice and experienced developer from *GitHub* have different patterns of activity on *Stack Overflow*. [10]

There has also been research done to study how to improve *Stack Overflow*. *Duijn et al.* looked into prediction the quality of questions by looking at how many answers the question attracted. [4] Previous study has shown that the number of answers of a question is highly correlated to the number of upvotes. Therefore, *Dujin et al.* decided to use 2 answers (the mean of answers to *Stack Overflow* posts) as a line for quality questions. *Saha et al.* created a model to better predict tags for *Stack Overflow* questions. [8]

Researchers also use *Stack Overflow* data to improve the development process of developers. *Ponzanelli et al.* tried to mine *Stack Overflow* for better IDE suggestions. [6]

Mining *Stack Overflow* has been really popular in recent years. The public knowledge from *Stack Overflow* is very rich. *Treude et al.* mined information from *Stack Overflow* to augment API documentation. [9]

7 VULNERABILITIES

7.1 Threats to Validity

Both the reference to and from *Stack Overflow* in *SOTorrent* are extracted by looking at *HTML* links. So there could be referenced information but not present in the dataset. Also not every code reuse is referenced, for code reuse from *Stack Overflow* to *GitHub* we could look at site wise code clone detection but the scale will be enormous and unlikely to be possible due to the size of the corpus. There won't be any effective way off my mind how we would detect unreferenced sources to *Stack Overflow*.

The source clone tool that is landed on could lead to problems. So I'll try my best to find the tool that serves the purpose will.

7.2 Potential Problems

In this case, the source code is required from *GitHub*. Due to the API limits of *GitHub* it's unsure whether acquiring the source code will be smooth.

As a novice to databased and previous problems I've encountered with them. Having the dataset setup could also be more effort than expected.

Having not used a source clone tool before, experimenting and setting up could have problems.

Time could also be a problem due to the scale of the project. However, I would like to answer most of the questions.

REFERENCES

[1] Sebastian Baltes, Lorik Dumani, Christoph Treude, and Stephan Diehl. 2018. SOTorrent: Reconstructing and Analyzing the Evolution of Stack Overflow Posts. In *Proceedings of the 15th International Conference on Mining Software Repositories (MSR '18)*. ACM, New York, NY, USA, 319–330. https://doi.org/10.1145/3196398.3196430

[2] Anton Barua, Stephen W. Thomas, and Ahmed E. Hassan. 2014. What are developers talking about? An analysis of topics and trends in Stack Overflow. *Empirical Software Engineering* 19, 3 (01 Jun 2014), 619–654. https://doi.org/10.1007/s10664-012-9231-y

[3] Huseyin Cavusoglu, Zhuolun Li, and Ke-Wei Huang. 2015. Can Gamification Motivate Voluntary Contributions?: The Case of StackOverflow Q&A Community. In *Proceedings of the 18th ACM Conference Companion on Computer Supported Cooperative Work & Social Computing (CSCW'15 Companion)*. ACM, New York, NY, USA, 171–174. https://doi.org/10.1145/2685553.2698999

[4] Maarten Duijn, Adam Kučera, and Alberto Bacchelli. 2015. Quality Questions Need Quality Code: Classifying Code Fragments on Stack Overflow. In *Proceedings of the 12th Working Conference on Mining Software Repositories (MSR '15)*. IEEE Press, Piscataway, NJ, USA, 410–413. http://dl.acm.org/citation.cfm?id=2820518.2820574

[5] F. Fischer, K. Böttinger, H. Xiao, C. Stransky, Y. Acar, M. Backes, and S. Fahl. 2017. Stack Overflow Considered Harmful? The Impact of Copy amp;Paste on Android Application Security. In *2017 IEEE Symposium on Security and Privacy (SP)*. 121–136. https://doi.org/10.1109/SP.2017.31

[6] Luca Ponzanelli, Gabriele Bavota, Massimiliano Di Penta, Rocco Oliveto, and Michele Lanza. 2014. Mining StackOverflow to Turn the IDE into a Self-confident Programming Prompter. In *Proceedings of the 11th Working Conference on Mining Software Repositories (MSR 2014)*. ACM, New York, NY, USA, 102–111. https://doi.org/10.1145/2597073.2597077

[7] Chanchal K. Roy, James R. Cordy, and Rainer Koschke. 2009. Comparison and Evaluation of Code Clone Detection Techniques and Tools: A Qualitative Approach. *Sci. Comput. Program.* 74, 7 (May 2009), 470–495. https://doi.org/10.1016/j.scico.2009.02.007

[8] Avigit K. Saha, Ripon K. Saha, and Kevin A. Schneider. 2013. A Discriminative Model Approach for Suggesting Tags Automatically for Stack Overflow Questions. In *Proceedings of the 10th Working Conference on Mining Software Repositories (MSR '13)*. IEEE Press, Piscataway, NJ, USA, 73–76. http://dl.acm.org/citation.cfm?id=2487085.2487103

[9] C. Treude and M. P. Robillard. 2016. Augmenting API Documentation with Insights from Stack Overflow. In *2016 IEEE/ACM 38th International Conference on Software Engineering (ICSE)*. 392–403. https://doi.org/10.1145/2884781.2884800

[10] B. Vasilescu, V. Filkov, and A. Serebrenik. 2013. StackOverflow and GitHub: Associations between Software Development and Crowdsourced Knowledge. In *2013 International Conference on Social Computing*. 188–195. https://doi.org/10.1109/SocialCom.2013.35