An Exploratory Study of Analyzing JavaScript Online Code Clones

Md Rakib Hossain Misu
University of California Irvine
Irvine, California, USA
mdrh@uci.edu

Abdus Satter
University of Dhaka
Dhaka, Bangladesh
abdus.satter@iit.du.ac.bd

ABSTRACT

Online code clones occur due to reusing code snippets in software repositories from online resources such as GitHub and Stack Overflow. Previous works have shown that snippets from Stack Overflow are reused in other open-source projects and vice versa. Analysis of online code reusing patterns could identify outdated code, understand developers' practices, and help to design new code search engines. This study analyzed JavaScript online code clones between Stack Overflow and GitHub repositories. We first developed a JavaScript code corpus to search online clones. The clone search results reported 12,579 online clones between 276,547 non-trivial syntactically validated Stack Overflow snippets and 292 GitHub repositories. We manually classified the top 10% (1257) pairs of clones in seven online clone patterns. We observed that around 70% of JavaScript snippets in Stack Overflow posts are copied from GitHub repositories or from other external sources. Moreover, only 30.59% of JavaScript Snippets in Stack Overflow accepted answers could be considered as reusable snippets.

CCS CONCEPTS

• **Software and its engineering → Software evolution**.

KEYWORDS

Code Clone, JavaScript, Stack Overflow, GitHub

ACM Reference Format:
Md Rakib Hossain Misu and Abdus Satter. 2022. An Exploratory Study of Analyzing JavaScript Online Code Clones. In *30th International Conference on Program Comprehension (ICPC '22), May 16–17, 2022, Virtual Event, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3524610.3528390

1 INTRODUCTION

Code cloning is a widespread practice to reuse code snippet through copying and pasting. In a previous study, it is observed that typical software repositories may contain 7% to 23% cloned code [5]. It is still controversial whether code clone is beneficial or harmful for software repositories. In a few ways, it can be useful for software development; for instance; a well tested modularized code snippet is easy to reuse. It may reduce the costs of software development and maintenance. However, in many situations, code cloning is detrimental as it propagates bugs, violates license, increases software size and raises design-related smells [11]. Similarly, code clones occurred between online platforms and software repositories are known as ***"online code clones"*** [23]. These clones can occur in both directions - online platform to software repositories and vice-versa. Like traditional code clones, online code clones are also responsible for bug propagation, license violation, reusing of outdated snippets and introducing software vulnerabilities. Analysis of code reusing pattern in the online platforms could help researchers understand developers practices, identify toxic code snippets and build code searching tools [8]. Previous studies [23] have been performed to investigate toxic code snippets where researchers analyzed some patterns of Java online clone snippets between Stack Overflow and a small set of open-source software repositories. To the best of our knowledge, in the literature, no further studies had been reported on online clones pattern analysis in other programming languages. To alleviate the lack of work in online code clone analysis, we aim to conduct a series of exploratory studies to investigate the online clone patterns in multiple languages such as JavaScript, C, C++, C#, Python and Java. Our ultimate goal is to compare the trend of online clone patterns in these languages and detect the outdated snippets [28]. As a part of our long-term vision, in this study, we reported an exploratory study of analyzing JavaScript online code clones. To do so, we first built a code corpus after extracting JavaScript snippets from Stack Overflow and GitHub repositories. We executed Siamese, a clone searcher, on this corpus and identified the online clone pairs. We then performed a manual classification of these clone pairs to understand their patterns and usages. **Figure 1** depicts the four phases of how we conducted the study and in the following sections we provide a brief description of these phases.

ICPC '22, May 16–17, 2022, Virtual Event, USA

ACM ISBN 978-1-4503-9298-3/22/05.
https://doi.org/10.1145/3524610.3528390

2 STUDY DESIGN

Specifically, we conducted this JavaScript online clone analysis study between Stack Overflow and GitHub repositories to answer the following research questions.

- **RQ1 [Reusable JavaScript Snippets in Stack Overflow]:** *What percentage of JavaScript snippets in Stack-overflow are considered to be reusable?*
- **RQ2 [JavaScript Online Code Clones]:** *To what extent JavaScript online clones occur between Stack Overflow and GitHub repositories?*
- **RQ3 [Patterns of JavaScript Online Code Clones]:** *How do JavaScript online code clones appear?*

2.1 JavaScript Corpus Building

We first built a corpus that contains extracted JavaScript snippets from Stack Overflow and GitHub repositories to conduct the study.

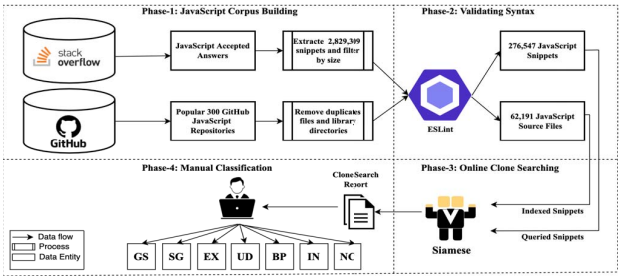

Figure 1: Four Phases of JavaScript Online Clone Pattern Analysis

2.1.1 ***Stack Overflow JavaScript Snippets.*** As a part of Stack Exchange Network [19], a regular data dump of Stack Overflow is created in a large *xml* file. The data dump contains the whole collection of Stack Overflow posts and their answers. To collect the snippets, we first downloaded the latest data dump of Stack Overflow posts [21]. In the downloaded **xml** file, code snippets are found inside $< code >...< /code >$ tag and snippets can be extracted from the value of this tag. To do so, we identified all the Stack Overflow posts that are tagged with the keyword **"javascript"**. However, we were only interested in extracting the reusable JavaScript snippets. Therefore, we decided to extract the code snippets that were only found in the accepted answer of any post. We did this because the accepted answer of a post more likely contains useful code snippets, and most of the time, these accepted snippets are voted and reused by other developers. Researchers considered the accepted answers snippets in many empirical studies[27] [23] and they also emphasized the reusability of the accepted answers [22] [15]. Hence, we located the accepted answer and then extracted the JavaScript snippets. Finally, we collected in total **2,829,309** JavaScript snippets from Stack Overflow accepted answers.

2.1.2 ***GitHub JavaScript Repositories.*** GitHub produces some metrics to represent how a repository evolves and how other developers interact with it. For instance, **star** represents the number of positive feedback a repository gains from other developers, and it also refers to the popularity of a repository. In this part, we aimed to create a collection of popular open-source JavaScript repositories from GitHub. Before including any repository in our corpus, we set a popularity criterion. For instance, we only considered those JavaScript repositories with the highest number of stars. According to our selection process, we utilized the GitHub Search API to query the JavaScript repository. We also ranked the search result by the **forks** count, which exhibits how many other developers used it. We have selected the top 300 repositories from the ranked query result and crawled all the default branches (master or main) for those repositories. However, we found that in 8 repositories, the default branch was *gh-pages* that hosts repository documentation site. Therefore, we discarded them and used the downloaded 292 repositories in our corpus. These repositories represent a statistically sound sample for our study. However, a repository might contain many duplicate **"*.js"** files [13] and additionally it might contain **"node_modules"** or **"lib"** directory which includes the dependent library files. These files are not the actual part of a repository, and considering these files during the analysis can lead our findings in another direction. We identified all the ***"node_modules"*** directories and the duplicate files in the repositories and removed them using ***rdfind*** [24]. Overall we removed **416 "node_modules"** and we deleted in total **51,016** duplicate files. These numbers are very significant as considering these duplicate files might alter our findings. In the next step, we validated all the snippets and files.

Table 1: Statistics of JavaScript Code Corpus

Corpus	Language	Files	Blank	Comments	LOC
SO Snippets	JavaScript	276,547	771,508	376,809	5,476,143
GitHub Repositories	JavaScript	62,191	2,127,778	2,888,517	12,253,701
Total		339,023	7,976,477	11,199,368	61,828,665

2.2 Validating Syntax

In the Stack Overflow JavaScript snippets and GitHub repository collections, many snippets might not be syntactically correct and not even purely written in JavaScript. JavaScript is widely used in web development as a scripting language, and consequently, JavaScript snippets can be embedded inside XML or HTML tags. During clone searching, the presence of XML and HTML in any snippet might alter the search result. Hence, we had to keep only the syntactically validated snippets and JavaScript source files in the corpus to have a correct clone search result. We used a popular JavaScript linter named **ESLint**[9] [26] to validate JavaScript syntax. By utilizing ESLint, we were able to collect **276,547** syntactically valid Stack-Overflow snippets and **62,191** JavaScript source files in 292 GitHub repositories. To have an overview of collected snippets and files, we computed the LOC of the whole JavaScript corpus with *cloc*. **Table 1** shows the overall statistic of our developed corpus.

2.3 Online Clone Searching

To search online clones in our developed corpus, we utilized Siamese [22], a highly scalable clone searcher. We incorporated with Siamese because it outperformed in clone searching compared to other state-of-the-arts clone detectors such as SourcererCC [25], NiCad [6]. Additionally, in the literature, it is the matured clone searching tool that's infrastructure allowed us to develop JavaScript language extensions to search clones in the JavaScript corpus. Utilizing the extended version of Siamese, we searched cloned snippets with the configuration of method level granularity having a size of greater or equal to ten lines [12]. The clone searching phase includes two steps. (i) First, we performed indexing in GitHub repositories source files to build an inverted index of method-level snippets. (ii) We used Stack Overflow snippets as queries and searched them in the inverted index with the same method level granularity. Upon completion of searching, Siamese reported **12579** clone pairs.

2.4 Manual Classification

In this phase, we conducted a manual investigation on Siamese's top **10%** reported clone pairs (**1257**). Our goal was to classify the clones pairs in various online clone patterns. For this purpose, we were inspired by the process as Ragkhitwetsagul et al. [23] followed in their toxic code analysis study. They proposed seven patterns of online code clones based on how the clones were spread in the various online platforms. A summary of these seven online clones patterns [23] is listed in **Table 2** . Here, **GitHub to Stack Overflow** ($GH \Rightarrow SO$) represents the clone pairs where a snippet is copied from GitHub to Stack Overflow. The evidence of such copying can be found in the description/comments of the Stack Overflow snippet. Similarly, **Stack Overflow to GitHub** ($SO \Rightarrow GH$) represents the opposite direction of copying a snippet from Stack Overflow to

Table 2: Online Code Clones Pattern [23]

Patterns	Description
GS	Copied from GitHub to Stack Overflow ($GH \Rightarrow SO$)
SG	Copied from Stack Overflow to GitHub ($SO \Rightarrow GH$)
EX	Copied from an external source to Stack Overflow/GitHub
UD	Copied from an external source outside the project (unknown)
BP	Boiler-plate or IDE auto-generated, framework scaffolding
IN	Inherited methods or similar interface implementation
NC	Not Clones

GitHub. The evidence is found in GitHub snippet with commenting the link of Stack Overflow question. Pattern **EX (External Sources)** refers to copying a snippet from other external sources such as blogs, forums, tutorials and using them in Stack Overflow or GitHub repositories. Pattern **UD (Unknown Direction)** represents a clone pair for which we could not have sufficient information to identify its origin. The major difference between EX and UD is that in UD, we could not identify the direction and any external source of its origin. However, EX contains the external direction of the snippet's origin. Then, **BP (Boiler-Plate)** refers to clone pair containing boiler-plate snippets which were generated by IDE, external frameworks, or scaffolding. Pattern **IN (Inheritance/Interface)** exposes the clone pair containing identical code snippets due to overriding the same interfaces or inherited methods. The last pattern **NC (Not Clones)** represents the false-positive clone. To accelerate the classification process, we developed a simple web application [10] that helped us to view the snippets side by side and browsed the original snippet if required. We also stored the classification results and the feedback for further analysis and validation.

3 RESULTS

To answer **RQ1**, we utilized only the extracted Stack-Overflow snippets. Then to answer **RQ2**, we considered the number of reported and manually validated true-positive clones pairs. The findings of online clone pattern analysis are reported in the answer of **RQ3**.

3.1 RQ1: [Reusable JavaScript Snippet in Stack Overflow]

We extracted in total **2,829,309** JavaScript snippets from the Stack Overflow accepted answers. Then we computed the Line of Code (LOC) of each snippet (without any comments or blank lines) by using a popular tool *cloc* [7]. Following an empirical evidence [5], we then categorized all the snippets into three different classes based on their LOC. The classes are **1) Trivial** (LOC<4) **2) Fair** (4<= LOC<= 6) and **3) Usable** (6<LOC). Although, six lines of snippets are well-accepted minimum clone size in clone benchmark [5], snippets of less than six lines contain a large number of boiler-plate code. After the categorization, we identified that near to three quarters of JavaScript snippets in Stack Overflow accepted answers are actually ***Trivial*** snippets, about **69.41%**. Besides, the presence of ***Fair*** class snippets are close to **8.42%** and the ***Usable*** class snippets are **22.17%**. Considering the re-usability (Fair & Usable) only 30.59% of JavaScript snippets are actually **reusable** snippets.

Summary: *In Stack Overflow, around **(69.41%)** of JavaScript snippets derived from accepted answers are **Trivial**. Only **(30.59%)** of snippiest can be considered as **Reusable**.*

3.2 RQ2: [JavaScript Online Code Clones]

Initially, Siamese reported 12,579 clone snippets containing approximately 0.40% (1109) of syntactically validated Stack Overflow snippets. These snippets are associated with 138 GitHub repositories. The average ratio of the cloned line in Stack Overflow is **21.20%**. During the manual investigation on the top 10% (1257) of reported clone pairs, we recognized that 802 (0.29%) Stack Overflow snippets are cloned, associated with 131 GitHub repositories. However, we identified **140** false-positive clones, which are categorized as NC (Not Clones). Therefore, after removing those NC clone pairs, the actual number of true-positive clones we found is **1117**. For **1117** pairs of clones, the average ratio of the cloned line in Stack Overflow snippets is **27.11%.**

Summary: *Our manual investigation found **1,117** true-positive JavaScript online clones that occurred between **276,547** syntactically validated Stack Overflow snippets, and **292** GitHub JavaScript repositories.*

3.3 RQ3: [Patterns of JavaScript Online Code Clones]

We conducted a manual classification of the **1257 clone pairs** and the classification results are discussed in the following.

3.3.1 ***GitHub ⇒ Stack Overflow (GS).*** We investigated **615** clone pairs as **GS (48.93%)**. It reveals that almost half of the total classified snippets were copied from GitHub repositories to Stack Overflow. It happens because of the wide usage and popularity of some JavaScript libraries. Most web applications contain JavaScript libraries such as jQuery, Bootstrap, AnularJS, d3 into their front-end or web-view. The highest number of snippets is copied from the jQuery repository. The ratio of **GS** clone pairs exhibits a trend that developers frequently copied JavaScript snippets from GitHub to Stack Overflow.

3.3.2 ***Stack Overflow ⇒ GitHub(SG).*** The number of clone pairs we classified as **SG** is **26** that amounts to **2.07%** of the total 1257 clone pairs. The GitHub source files explicitly contain the Stack Overflow post link as comments in these pairs. However, compared to **GS** clones pairs, the amount of **SG** pairs are significantly small. One possible reason is that developers were not interested in referring the copied code snippets into their source files. Therefore, many **SG** clone pairs were unrecognized or even recognized as *External Sources (**EX**)* or *Unknown Direction (**UD**)*.

3.3.3 ***External Sources (EX).*** We identified that **270 (21.48%)** pairs of snippets were copied from External Sources. We classified the pairs **EX** clone pair when we could not find the origin of the Stack Overflow snippet in our GitHub repository collection. However, we located them in another online blog, tutorials, and QA sites. It is expected that both the Stack Overflow post and the GitHub source file should attribute the origin of the source. However, we would not find evidence of that. In the majority of the clone pairs, we observed that in the Stack Overflow post, developers usually added the references of the origin. However, in GitHub, this practice was rarely followed. After analyzing the code snippets (especially the source path) in many GitHub repositories, we realized that source files were derived from the *lib/, library/, external/ or vendor/*

external directories. Although we deleted the **"node_modules"** directory still, there are some directories that contain external library files. Moreover, in many cases, developers directly import the source flies of various external libraries into their projects. Clone pairs associated with these external files produced many **EX** pairs.

3.3.4 ***Unknown Direction (UD).*** We classified **35 (2.78%)** online clone pairs as **UD** because we could not find their origin from where they were copied to Stack Overflow or in the GitHub repositories. Although they are highly similar, the origin of these clone pairs could not be found even in the comment of the Stack Overflow post or GitHub source documentation. These snippets could be the possible ***original*** snippets written by the developers while answering a post in Stack Overflow.

3.3.5 ***Boiler-Plate (BP):.*** We tagged **159** number of boiler-plate clone pairs during the manual investigation, and it amounts to **12.65%** of all clones we classified. We observed that most of the boiler-plate clone pairs are occurred due to the language internationalization and localization. Besides, frameworks such as AngularJS, React native are responsible for producing many BP clone pairs.

3.3.6 ***Inheritance/Interface (IN).*** We realized that only a few clone pairs happen due to inheritance and interfaces. In our observation, we found only **12** snippets **(0.95%)** that were occurred because of inheritance. The reasons are JavaScript's dynamic nature, and the JavaScript does not directly support interface implementation. It is also rare to implement interfaces in JavaScript since the language design itself does not impose developers to implement an interface or override extended methods. Moreover, in JavaScript, inheritance is achieved with the function prototyping, and it mostly treats objects as functions. Because of such language features, only a few IN clone pairs were found.

3.3.7 ***Not Clones (NC).*** We identified **140 (11.14%)** clone pairs as non-clone pairs. More specifically, they are false-positive clones caused by the normalization of source code identifiers. We analyzed that the uglification of JavaScript sources represents source code with parametrized tokens. Thus, it appears to be similar to normalized Stack Overflow snippets, and because of such uglification, many NC clones occurred between Stack Overflow snippets and uglified JavaScript source files in GitHub repositories.

Summary: *It appears that **615** JavaScript snippets pairs were copied from GitHub to Stack Overflow. In contrast, only **26** pairs were cloned from Stack Overflow to GitHub. Besides, **270** pairs of snippets are copied from external sources to Stack Overflow. No origin was found for **35** pairs, which remained as unknown. Also, **159** number of clone pairs are investigated as the boiler-plate.*

4 THREATS TO VALIDITY

4.1 Internal Validity

We only utilized Siamese for code searching purposes and its reported clone pairs in the manual classification. Hence, adopting any other clone searchers instead of Siamese may alter the findings of this study. It would be better to apply other clone detectors on our JavaScript corpus and perform clone merging to have a list of clone pairs. Unfortunately, to the best of our knowledge, other clone searchers are not compatible with search clones in JavaScript corpus except Siamese. Besides, our corpus only contains the code snippets extracted from Stack Overflow accepted answers. We considered these snippets since these are the highly visible and functional snippets that can solve the problem. Therefore, these accepted snippets have better chances to be reused in open source projects. However, in many Stack Overflow posts, the accepted answer is not the most popular if we consider the number of votes as a popularity indicator. Therefore, considering the most voted answers might improve and alter our findings. Five graduate students have performed the manual classification of the clone pairs individually and blindly. This manual approach causes the classification to have subjective biases and human errors. However, we tried our best to reduce the errors rates during the classification by discussing controversial issues. For instance, if a pair falls into multiple patterns, we finalize it based on its votes to fall into a specific pattern.

4.2 External Validity

We downloaded Stack Overflow data dump released on ***02-June-2020***. The Stack Overflow data dump version may have minimal effect on the study since the posts are rarely updated. Users occasionally do minor modifications, for instance, a few line addition or deletion. However, our overall findings may slightly differ in newer or old releases of data dump as it may contain more or fewer posts with accepted answers. In contrast, the findings will significantly vary for GitHub repositories versions since GitHub repositories are updated frequently by the developer commits. We crawled all the GitHub repositories snapshot on **25th-July-2020**. Therefore, the snippets we considered from GitHub repositories may be updated or removed in newer commits. As a result, our clone search might not recognize those snippets as cloned if any changes occurred in a particular snippet. Besides, we only consider the default branch of the repository, and in some repositories, the **master or main** branch is not the default or latest branch. Considering other branches except for the **master or main** branch might have significant effects on our findings. **Replication Package:** We provide all artifacts and sources to reproduce and validate the findings of our study [1].

5 RELATED WORK

Stack Overflow helps software engineers and researchers to get quick and practical insights into a problem and how to solve it. In recent and past few years, many studies have been conducted to represent Stack Overflow's usability, practices, and knowledge for better understanding. Stack Overflow contains the most reusable code solution hence some studies focused on efficient code search through incorporating Stack Overflow snippets [4, 14, 27]. Many of the studies described the evolution and changed histories of Stack Overflow posts and answers [3, 16]. A few studies also raised the concern that using snippets from Stack Overflow would violate the licensing and make it a code laundering platform [2]. Research also proposed approaches to improve the unanswered question and issues related to a post [17, 18]. Studies have also been conducted to explore the code duplication within the Stack Overflow posts and GitHub Repositories [20]. Our study completely differs from all of these works since we focused more on analyzing the patterns of online clones in JavaScript. Additionally, we are working on extending the scope of our study to other popular languages.

6 CONCLUSION & FUTURE WORK

Reusing of code snippets from online Q&A sites to software repositories introduces online code clones. This study analyzed JavaScript online code clones between Stack Overflow and open source GitHub repositories. Our corpus contains 276,547 non-trivial syntactically valid snippets from Stack Overflow and 292 GitHub repositories. We identified a collection of 12579 pairs of online clones within the corpus and filtered the top 10% (1257) of clone pairs from the collection to investigate their patterns manually. It is found that most of the snippets in the Stack Overflow are not reusable code snippets and that around 70% of accepted JavaScript snippets were copied from GitHub or external sources. Our future goal is to extend the scope of this study to analyze the online clone patterns in other languages such as Python, C, C++, and C# and to conduct a comparative study across the languages.

7 ACKNOWLEDGEMENT

We want to thank Dr. Jens Krinke and Dr. Chaiyong Ragkhitwetsagul for their directions & support during the initiation of this study.

REFERENCES

[1] 2022. *JavaScript Online Code Clones*. Zenodo. https://doi.org/10.5281/zenodo.6395244
[2] Le An, Ons Mlouki, Foutse Khomh, and Giuliano Antoniol. 2017. Stack Overflow: A code laundering platform?. In *IEEE 24th International Conference on Software Analysis, Evolution and Reengineering, SANER 2017, Klagenfurt, Austria, February 20-24, 2017*, Martin Pinzger, Gabriele Bavota, and Andrian Marcus (Eds.). IEEE Computer Society, 283–293.
[3] 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 2018, Gothenburg, Sweden, May 28-29, 2018*, Andy Zaidman, Yasutaka Kamei, and Emily Hill (Eds.). ACM, 319–330.
[4] Sebastian Baltes and Christoph Treude. 2020. Code duplication on stack overflow. In *ICSE-NIER 2020: 42nd International Conference on Software Engineering, New Ideas and Emerging Results, Seoul, South Korea, 27 June - 19 July, 2020*. ACM, 13–16.
[5] Stefan Bellon, Rainer Koschke, Giuliano Antoniol, Jens Krinke, and Ettore Merlo. 2007. Comparison and Evaluation of Clone Detection Tools. *IEEE Trans. Software Eng.* 33, 9 (2007), 577–591.
[6] James R. Cordy and Chanchal K. Roy. 2011. The NiCad Clone Detector. In *The 19th IEEE International Conference on Program Comprehension, ICPC 2011, Kingston, ON, Canada, June 22-24, 2011*. IEEE Computer Society, 219–220.
[7] Al Danial. 2022. CLOC: Count Lines Of Code. https://github.com/AlDanial/cloc. [Online], [Accessed: 2022-02-02].
[8] Themistoklis G. Diamantopoulos and Andreas L. Symeonidis. 2015. Employing Source Code Information to Improve Question-Answering in Stack Overflow. In *12th IEEE/ACM Working Conference on Mining Software Repositories, MSR 2015, Florence, Italy, May 16-17, 2015*, Massimiliano Di Penta, Martin Pinzger, and Romain Robbes (Eds.). IEEE Computer Society, 454–457.
[9] ESLint. July 2020. Users. https://eslint.org/docs/rules/.
[10] js-online clone.web.app. March 2022. js-online-clone.web.app. https://js-online-clone.web.app/.
[11] Cory Kapser and Michael W. Godfrey. 2008. "Cloning considered harmful" considered harmful: patterns of cloning in software. *Empir. Softw. Eng.* 13, 6 (2008), 645–692.
[12] Rainer Koschke, Raimar Falke, and Pierre Frenzel. 2006. Clone Detection Using Abstract Syntax Suffix Trees. In *13th Working Conference on Reverse Engineering (WCRE 2006), 23-27 October 2006, Benevento, Italy*. IEEE Computer Society, 253–262.
[13] Cristina V. Lopes, Petr Maj, Pedro Martins, Vaibhav Saini, Di Yang, Jakub Zitny, Hitesh Sajnani, and Jan Vitek. 2017. DéjàVu: a map of code duplicates on GitHub. *Proc. ACM Program. Lang.* 1, OOPSLA (2017), 84:1–84:28.
[14] Adriaan Lotter, Sherlock A. Licorish, Bastin Tony Roy Savarimuthu, and Sarah Meldrum. 2018. Code Reuse in Stack Overflow and Popular Open Source Java Projects. In *25th Australasian Software Engineering Conference, ASWEC 2018, Adelaide, Australia, November 26-30, 2018*. IEEE Computer Society, 141–150.
[15] Saraj Singh Manes and Olga Baysal. 2019. How often and what StackOverflow posts do developers reference in their GitHub projects?. In *Proceedings of the 16th International Conference on Mining Software Repositories, MSR 2019, 26-27 May 2019, Montreal, Canada*, Margaret-Anne D. Storey, Bram Adams, and Sonia Haiduc (Eds.). IEEE / ACM, 235–239.
[16] Saraj Singh Manes and Olga Baysal. 2021. Studying the Change Histories of Stack Overflow and GitHub Snippets. In *18th IEEE/ACM International Conference on Mining Software Repositories, MSR 2021, Madrid, Spain, May 17-19, 2021*. IEEE, 283–294.
[17] Saikat Mondal, Mohammad Masudur Rahman, and Chanchal K. Roy. 2019. Can issues reported at stack overflow questions be reproduced?: an exploratory study. In *Proceedings of the 16th International Conference on Mining Software Repositories, MSR 2019, 26-27 May 2019, Montreal, Canada*, Margaret-Anne D. Storey, Bram Adams, and Sonia Haiduc (Eds.). IEEE / ACM, 479–489.
[18] Saikat Mondal, C. M. Khaled Saifullah, Avijit Bhattacharjee, Mohammad Masudur Rahman, and Chanchal K. Roy. 2021. Early Detection and Guidelines to Improve Unanswered Questions on Stack Overflow. In *ISEC 2021: 14th Innovations in Software Engineering Conference, Bhubaneswar, Odisha, India, February 25-27, 2021*, Durga Prasad Mohapatra, Samaresh Mishra, Tony Clark, Alpana Dubey, Richa Sharma, and Lov Kumar (Eds.). ACM, 9:1–9:11.
[19] Stack Exchange Network. 2022. Stack Exchange Network. https://archive.org/download/stackexchange. [Online], [Accessed: 2021-12-05].
[20] Manziba Akanda Nishi, Agnieszka Ciborowska, and Kostadin Damevski. 2019. Characterizing duplicate code snippets between stack overflow and tutorials. In *Proceedings of the 16th International Conference on Mining Software Repositories, MSR 2019, 26-27 May 2019, Montreal, Canada*, Margaret-Anne D. Storey, Bram Adams, and Sonia Haiduc (Eds.). IEEE / ACM, 240–244.
[21] Stack Overflow. 2022. Stack Overflow Post. https://archive.org/download/stackexchange/stackoverflow.com-Posts.7z. [Online], [Accessed: 2021-12-05].
[22] Chaiyong Ragkhitwetsagul and Jens Krinke. 2019. Siamese: scalable and incremental code clone search via multiple code representations. *Empir. Softw. Eng.* 24, 4 (2019), 2236–2284.
[23] Chaiyong Ragkhitwetsagul, Jens Krinke, Matheus Paixão, Giuseppe Bianco, and Rocco Oliveto. 2021. Toxic Code Snippets on Stack Overflow. *IEEE Trans. Software Eng.* 47, 3 (2021), 560–581.
[24] rdfind. July 2020. Users. https://rdfind.pauldreik.se/.
[25] Hitesh Sajnani, Vaibhav Saini, Jeffrey Svajlenko, Chanchal K. Roy, and Cristina V. Lopes. 2016. SourcererCC: scaling code clone detection to big-code. In *Proceedings of the 38th International Conference on Software Engineering, ICSE 2016, Austin, TX, USA, May 14-22, 2016*, Laura K. Dillon, Willem Visser, and Laurie Williams (Eds.). ACM, 1157–1168.
[26] Kristín Fjóla Tómasdóttir, Mauricio Finavaro Aniche, and Arie van Deursen. 2020. The Adoption of JavaScript Linters in Practice: A Case Study on ESLint. *IEEE Trans. Software Eng.* 46, 8 (2020), 863–891.
[27] Di Yang, Pedro Martins, Vaibhav Saini, and Cristina V. Lopes. 2017. Stack overflow in github: any snippets there?. In *Proceedings of the 14th International Conference on Mining Software Repositories, MSR 2017, Buenos Aires, Argentina, May 20-28, 2017*, Jesús M. González-Barahona, Abram Hindle, and Lin Tan (Eds.). IEEE Computer Society, 280–290.
[28] Haoxiang Zhang, Shaowei Wang, Tse-Hsun Peter Chen, Ying Zou, and Ahmed E Hassan. 2019. An empirical study of obsolete answers on stack overflow. *IEEE Transactions on Software Engineering* (2019).